Product Selection Problem: Improve Market Share by Learning Consumer Behavior

Silei Xu
slxu@cse.cuhk.edu.hk
Department of Computer Science and Engineering
The Chinese University of Hong Kong
New Territories, Hong Kong

John C.S. Lui
cslui@cse.cuhk.edu.hk

ABSTRACT
It is often crucial for manufacturers to decide what products to produce so that they can increase their market share in an increasingly fierce market. To decide which products to produce, manufacturers need to analyze the consumers' requirements and how consumers make their purchase decisions so that the new products will be competitive in the market. In this paper, we first present a general distance-based product adoption model to capture consumers' purchase behavior. Using this model, various distance metrics can be used to describe different real life purchase behavior. We then provide a learning algorithm to decide which set of distance metrics one should use when we are given some historical purchase data. Based on the product adoption model, we formalize the $k$-most marketable products (or $k$-MMP) selection problem and formally prove that the problem is NP-hard. To tackle this problem, we propose an efficient greedy-based approximation algorithm with a provable solution guarantee. Using submodularity analysis, we prove that our approximation algorithm can achieve at least 63% of the optimal solution. We apply our algorithm on both synthetic datasets and real-world datasets (TripAdvisor.com), and show that our algorithm can easily achieve five or more orders of speedup over the exhaustive search and achieve about 96% of the optimal solution on average. Our experiments also show the significant impact of different distance metrics on the results, and how proper distance metrics can improve the accuracy of product selection.

Categories and Subject Descriptors
F2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

Keywords
Product selection; consumer behavior; model learning; submodular set function; approximation algorithm

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD ’14, August 24–27, 2014, New York, NY, USA.
Copyright 2014 ACM 978-1-4503-2956-9/14/08 ...$15.00.
http://dx.doi.org/10.1145/2623330.2623692.

1. INTRODUCTION
Product competition in the current digital age is becoming increasingly fierce. Consumers can easily access the information about a given product via the Internet. Moreover, consumers can share their opinions on products in the form of ratings or reviews via various web services, e.g., Amazon. Therefore, instead of relying on the sales pitch by salesmen or traditional TV advertisements, consumers can now review many competing products before they make their final purchase decision. Manufacturers, on the other hand, can use the web information, such as ratings and reviews, to gain a better understanding of consumers’ requirements on various products. This leads to a new challenge on how to discover consumers’ preferences, and how these preferences may help manufacturer to select appropriate new products so to compete with other manufacturers in the market.

To introduce new products into a market, a manufacturer usually has a set of candidate products to consider. However, due to budget constraints, the manufacturer can only produce a small subset of these candidate products. The objective of a manufacturer is to select a subset of products which can maximize its profit or market share. In this study, we consider the following scenario: In a market consisting of a set of existing products from various manufacturers and a set of consumers, a manufacturer wants to select the “$k$ most marketable products” from a set of candidate products so as to maximize the market share of all products from this manufacturer (this includes the possibility that some existing products in the market are from the same manufacturer).

One of the major challenges of the “$k$ most marketable products” problem is how to model various consumers’ adoption behavior, i.e., how consumers make their purchase decisions. Different adoption behavior may lead to different product selection results. However, there is a lack of formal work of how to model these behaviors using available data. Furthermore, finding the optimal solution to the “$k$ most marketable products” problem can be shown to be NP-hard in general.

In this paper, we first model the consumers’ adoption behavior with a generalized distance-based model where different distance metrics can be used to describe many different consumers behaviors. We then propose a method to learn which set of distance metrics one should use when we are given some historical purchase data. We also present a computationally efficient approximation algorithm to solve the $k$ most marketable products problem. To the best of our knowledge, this is the first paper that provides the formal
consumers’ adoption model and the analysis of product selection. The contributions of this paper are:

- We formulate the problem of finding the k most marketable products (k-MMP) for a manufacturer.
- We model the adoption behavior of consumers using a general distance-based product adoption model which can take on various different distance metrics.
- We provide a learning method to determine the appropriate set of distance metrics using the historical purchase data on market share of a subset of the existing products.
- We prove that the k-MMP problem is NP-hard and propose a computationally efficient approximation algorithm. By proving the monotonicity and submodularity properties of the objective function, we show that our approximation algorithm provides a $(1-1/e)$-approximation as compared with the optimal solution.
- We carry out experiments on synthetic and real-world datasets to demonstrate the computational efficiency and quality of its solutions. We also illustrate how one can select the appropriate distance metrics by learning from the historical purchase data so as to improve the market share.

The outline of the paper is as follows. In Section 2, we propose a general product adoption model which can accommodate different distance metrics to describe the consumers’ adoption behavior, and we formulate the k-MMP problem. In Section 3, we present a learning method to select the appropriate set of distance metrics according to the historical market share of existing products. In Section 4, we propose an exact algorithm for the case of $k = 1$ and prove that finding the exact solution for $k > 1$ is NP-hard. To tackle the computational challenge, we present an approximation algorithm in Section 5. We show that this algorithm is computationally efficient and also provides a high quality solution guarantee. In Section 6, we perform experiments on both the synthetic data and the real-world data. Related work is shown in Section 7, and Section 8 concludes.

2. MATHEMATICAL MODELS AND PROBLEM FORMULATION

In this section, we first present a model of a market by considering both products and consumers. Then we present a distance-based product adoption model to describe various consumers’ product adoption behaviors. Based on these models, we formulate the k-MMP problem.

2.1 Market Model

Let us consider a market which consists of a set of l consumers $\mathcal{C} = \{c_1, c_2, \ldots, c_l\}$ and a set of m existing products $\mathcal{P}_E = \{p_1, p_2, \ldots, p_m\}$. Let $M$ represent a manufacturer in the market, and $\mathcal{P}_M$ denote the set of existing products produced by $M$, where $\mathcal{P}_M \subseteq \mathcal{P}_E$ and $|\mathcal{P}_M| = m_M$. The remaining products in $\mathcal{P}_E$ are from other manufacturers who are the competitors of $M$. These competing products are denoted by $\mathcal{P}_C$, where $\mathcal{P}_C \subseteq \mathcal{P}_E$ and $|\mathcal{P}_C| = m_C$. According to these definitions, we have $m = m_M + m_C$, $\mathcal{P}_E = \mathcal{P}_M \cup \mathcal{P}_C$, and $\mathcal{P}_M \cap \mathcal{P}_C = \emptyset$.

Suppose the manufacturer $M$ wants to produce some new products to maximize its utility, i.e., the market share. $M$ has a set of n candidate new products to choose from, which we denote by $\mathcal{P}_N = \{p_{m+1}, p_{m+2}, \ldots, p_{m+n}\}$. Note that all the products in $\mathcal{P}_N$ are new to the market, in other words, $\mathcal{P}_N \cap \mathcal{P}_E = \emptyset$. Due to the budget, technological and manufacturing constraints, the manufacturer $M$ can only produce $k \leq n$ of these candidate products in $\mathcal{P}_N$.

Each product in $\mathcal{P}_E \cup \mathcal{P}_N$ is associated with $d$ attributes denoted by $\mathcal{A} = \{a_1, a_2, \ldots, a_d\}$. Each attribute $a_i$ is represented by a non-negative real number, and higher value implies higher quality. One can use $a_i$ to represent various attributes of a given product, e.g., durability, ratings, inverse of price. Hence, the quality of a product can be described by a $d$-dimensional vector. Specially, the quality of product $p_i$ is described by the vector $q_i = (q_{i1}, q_{i2}, \ldots, q_{id})$, where $q_{ij} \in [0, \infty)$, $\forall j \in \{1, 2, \ldots, d\}$ indicates $p_i$’s quality on attribute $a_j$. Similarly, each consumer in $\mathcal{C}$ is also associated with $\mathcal{A}$ to describe his requirements on different attributes. Let $r_i = (r_{i1}, r_{i2}, \ldots, r_{id})$ be the requirement vector of consumer $c_i$, where $r_{ij} \in [0, \infty)$, $\forall j \in \{1, 2, \ldots, d\}$ indicates $c_i$’s minimum requirement on attribute $a_j$, i.e., $c_i$ requires that the product’s quality on attribute $a_j$ is at least $r_{ij}$, or he will not adopt (or purchase) that product.

**Example 1.** To illustrate the notations, we present an example in Figure 1. Consider a market of smart phones where we have two existing products $\mathcal{P}_E = \{p_1, p_2\}$ and three consumers $\mathcal{C} = \{c_1, c_2, c_3\}$. Manufacturer $M$ is considering two candidate products $\mathcal{P}_N = \{p_3, p_4\}$. Let say each product is described by two attributes: $a_1$ is the inverse of price (units per thousand dollars, UPM for short) and $a_2$ is durability (years), and they are represented in the horizontal and the vertical axis respectively. The quality vectors of products and the requirement vectors of consumers are shown in the figure (with $\mathcal{P}_E \triangleleft \mathcal{P}_N \triangleright \mathcal{C} \circ \circ$). For instance, the quality vector of $p_1$ is $(2, 6)$, so we can purchase two units of $p_1$ with one thousand dollars (or the price of $p_1$ is $5000$), and the durability of $p_1$ is six years. Similarly, the requirement vector of $c_1$ is $(1, 5)$, so consumer $c_1$ wants a product which is at most $1000$ and can last for at least five years.

![Figure 1: An illustration of the market model](image-url)

2.2 Product Adoption Model

We assume that a consumer may adopt a product if the product satisfies his requirement. We say that a product satisfies a consumer’s requirements if and only if the product meets the requirements of that consumer on all attributes. Formally, we define the product satisfiability condition.

**Definition 1. (Product satisfiability)** Consider a consumer $c_i$ and a product $p_j$. We say the product $p_j$ satisfies...
the consumer $c_i$ if and only if $q_j[t] \geq r_i[t], \forall t = 1, \ldots, d$. We denote this relationship as $p_j \succ c_i$, and $p_j$ is said to be a satisfactory product of $c_i$, while $c_i$ is a potential consumer of $p_j$ in other words.

For example, consider the products and consumers depicted in Figure 1. One can observe that the quality vector of $p_1$ is $(2, 6)$ and the requirement vector of $c_1$ is $(1, 5)$. Since $2 > 1$ and $6 > 5$, so $p_1$ satisfies $c_1$, or $p_1 \succ c_1$. Similarly, we have $p_2 \succ c_2$ and $p_3 \succ c_3$.

We assume that if a consumer has some satisfactory products, then he will adopt one unit of product from any of these feasible products. When a consumer $c_i$ has only one satisfactory product, say $p_j$, then $c_i$ will adopt $p_j$ for sure. However, it becomes complicated when there are multiple satisfactory products. All previous works [7, 12, 13, 16] assume that the consumer will randomly adopt one of the satisfactory products, but this is not realistic in many situations. In the following, we present the distance-based adoption model to describe some realistic and representative product adoption behavior when consumers make their purchase decisions. Our model is very general to model various product adoption behaviors in the real world scenarios.

In a real world market, products with higher quality usually attract more consumers. Therefore, we use a distance measure between a product’s quality and a consumer’s requirement to decide which product the consumer may adopt. Note that consumers will only consider their satisfactory products. Furthermore, larger distance implies better quality. Let $d_{i,j}$ be the distance between the consumer $c_i$’s requirement vector ($r_i$) and the product $p_j$’s quality vector ($q_j$). We assume that $c_i$ will adopt the product $p_j$ which has the largest distance among all his satisfactory products. If there are multiple satisfactory products which have the same largest distance measure with $c_i$, then $c_i$ will randomly select one of these products. Mathematically, we define the distance-based adoption model as follows.

**Definition 2. (Distance-based adoption model)** Given a consumer $c_i$ and a set $P$ of products available in the market, let $FP(c_i|P)$ be the set of products which have the largest distance between their quality vectors and $c_i$’s requirement vector among all $c_i$’s satisfactory products. The probability that $c_i$ adopts a product $p_j \in P$ is

$$\Pr(i,j|P) = \begin{cases} \frac{1}{|FP(c_i|P)|} & \text{if } p_j \in FP(c_i|P), \\ 0 & \text{otherwise}. \end{cases}$$  \(1\)

Note that we can use many distance metrics, e.g., $l_1$, $l_2$, $l_\infty$ norms. For instance, if $l_1$ norm (or the Manhattan distance) is used, then consumers will choose the satisfactory products which have the largest sum of all components’ values in the quality vectors. To describe different adoption behaviors of different consumers in a real world market, we also take into account the weighted distance metrics. Let $w_i$ be the weight of attribute $a_t$, $w_i \geq 0, \forall a_t \in A$, then under the $l_1$ norm, the distance $d_{i,j}$ can be expressed as:

$$d_{i,j} = \sum_{a_t \in A} w_t \cdot (q_j[t] - r_i[t]).$$  \(2\)

It is important to point out that the algorithms we present in this paper are general to all distance metrics. Readers can use other distance metrics when appropriate. In here, we present four representative distance metrics which we use as examples for illustrations and experiments.

- **Discrete metric (DM).** We define $d_{i,j} = 1$ for consumers $c_i$ and $c_j$'s satisfactory product $p_j$ in the discrete metric. This distance metric simplifies the adoption model that consumers will randomly select one from all his satisfactory products. Using this distance metric, our work subsumes the adoption models of previous works [7, 12, 13, 16].

- **Norm metric (NM).** In this distance metric, we set the weight $w_i = 1.0, \forall a_t \in A$ based on the $l_1$ norm metric as defined in Equation (2). Note that in general, one can use other norm as distance metric and our algorithms still apply.

- **Price metric (PM).** In a real world market, one common situation is that if a consumer’s requirements are satisfied, then he will select the cheapest product, i.e., the one with the highest quality on the attribute of “price”. In this case, we can set the weight of all attributes to zero except the “price” based on the $l_1$ norm metric as defined in Equation (2).

- **Richman metric (RM).** Unlike the price metric, some consumers may be rich and they are insensitive to the price but only want the best product. In this case, we can set the weight of “price” attribute to zero while setting the weight of other attributes to one.

**Example 2.** To illustrate, let us consider the products and consumers depicted in Figure 1. Suppose that manufacturer $M$ decides to produce $p_1$, then the set of available products in the market is $P = P_E \cup \{p_1\} = \{p_1, p_2, p_3\}$. Let us consider the probability $c_2$ will adopt $p_3$, i.e., $\Pr(2,3|P)$, when $c_2$ uses the above four distance metrics. From Figure 1, one can observe that $c_2$ is satisfied by $p_1$, $p_2$, and $p_3$. If $c_2$ uses the discrete metric, then $d_{2,1} = d_{2,2} = d_{2,3} = 1$, so $\Pr(2,3|P) = 1/3$. If $c_2$ uses the norm metric, then we have $d_{2,1} = 3, d_{2,2} = d_{2,3} = 5$. Hence, $c_2$ will select $p_2$ and $p_3$ with probability $\Pr(2,2|P) = \Pr(2,3|P) = 1/2$. If $c_2$ uses the price metric, then we only need to consider the attribute inverse of price. We have $d_{2,1} = 0, d_{2,2} = 4, d_{2,3} = 5$, so $\Pr(2,3|P) = 1, c_2$ will adopt $p_3$. If $c_2$ uses the richman metric, we have $d_{2,1} = 3, d_{2,2} = 1, d_{2,3} = 0$. Thus, $c_2$ will adopt $p_1$ only, or $\Pr(2,1|P) = 1$ and $\Pr(2,3|P) = 0$.

**2.3 Problem Formulation**

To find the $k$ most marketable products, we first need to define the expected market share of a set of products under the distance-based adoption model. Given the market condition, i.e., the consumers $C$ and the existing products $P_E$, let $P$ be the set of products we consider, then the expected market share of $P$ is defined as

$$MS(P) = \frac{1}{|T|} \sum_{p_j \in P} \sum_{c_i \in TC(p_j)} \Pr(i,j|P_E \cup P),$$  \(3\)

where $t = |C|$, $TC(p_j)$ denotes the set of potential consumers of product $p_j$, and $\Pr(i,j|P_E \cup P)$ is defined in Equation (1).

**Example 3.** Let us illustrate the expected market share of $p_3$, or $MS(\{p_3\})$, by considering the scenario depicted in Figure 1. There are two existing products $(P_E = \{p_1, p_2\})$ and three consumers $(C = \{c_1, c_2, c_3\})$ in the market. By adding product $p_3$ into the market, $p_3$ satisfies consumers $c_2$ and $c_3$. Assume that $c_2$ uses the norm metric, then according to Example 2, we have $\Pr(2,3|P) = 1/2$. Now consider consumer $c_3$. Since we have not added $p_3$ into the market,
3. DISTANCE METRIC LEARNING

As discussed in Section 2, there are various distance metrics one can use and the product selection results can vary significantly depending on the distance metrics according to the results shown in Section 6. Hence, it is important to “learn” about the proper distance metrics (in other words, consumers’ product adoption behavior) from the available data. In this work, we propose a learning method based on the market share of a set of products in the market so to discover the appropriate distance metrics.

Note that in real life, some manufacturers may not release full information about their market share. Therefore, we assume that we only know the market share of a subset of existing products. Formally, let $P_E$ be the $n'$ products that we know the market share data, where $P_E \subseteq P_E$. Let $m_{sj}$ be the market share of $P_j \in P_E$.

Assume that we have a model set consisting of distance-based product adoption models using $m'$ different potential distance metrics, which are numbered from 1 to $m'$. Let $e_{ji}$ be the expected market share of product $p_j$ under the product adoption model using the $i$-th potential distance metric. Let $\theta_i$ be the probability that consumers use the $i$-th distance metric, and $\Theta = (\theta_1, \theta_2, \ldots, \theta_{m'})^T$. Then we can forecast the market share for each product $p_j \in P_E$ as:

$$f_j(\Theta) = \Theta \cdot (e_{j1}, e_{j2}, \ldots, e_{jm'})$$

$$= \theta_1 e_{j1} + \theta_2 e_{j2} + \ldots + \theta_{m'} e_{jm'},$$

where $f_j(\Theta)$ is the forecast market share of product $p_j$.

We can find the best fit for $\Theta$ by minimizing the squared difference between the forecast market share $f_j$ and the real-world market share $m_{sj}$. Let $\Delta_i$ be the difference between $f_j$ and $m_{sj}$, or mathematically, $\Delta_i(\Theta) = |f_j(\Theta) - m_{sj}|$. We can formalize the model selection problem as follows.

Minimize $\sum_{p_j \in P_E} \Delta_i^2(\Theta),$ subject to $\Theta \geq 0, \theta_1 + \theta_2 + \ldots + \theta_{m'} = 1,$

where $\Theta \geq 0$ means that $\theta_i \geq 0, \forall i \in \{1, \ldots, m'\}$. Thus, the problem is reduced to a linear regression problem with constrained least squares approach, which can be solved using the technique in [3]. Once we solve this linear regression problem, we can forecast the market share $f_j(\Theta)$ based on the probability vector $\Theta$.

**Example 4.** Consider a model set consisting of adoption models using the norm metric (NM), the price metric (PM) and the richman metric (RM). Assume we obtain the real-world market share of three products $p_1, p_2,$ and $p_3$ and we want to forecast the market share of $p_4$. The real-world market share and the expected market share under three different models of these products are shown in Table 4.

<table>
<thead>
<tr>
<th></th>
<th>NM</th>
<th>PM</th>
<th>RM</th>
<th>real-world</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1$</td>
<td>20%</td>
<td>1%</td>
<td>50%</td>
<td>5%</td>
</tr>
<tr>
<td>$p_2$</td>
<td>30%</td>
<td>10%</td>
<td>15%</td>
<td>20%</td>
</tr>
<tr>
<td>$p_3$</td>
<td>7%</td>
<td>30%</td>
<td>0%</td>
<td>20%</td>
</tr>
<tr>
<td>$p_4$</td>
<td>10%</td>
<td>40%</td>
<td>5%</td>
<td>unknown</td>
</tr>
</tbody>
</table>

**Table 1: An example of model selection**

Let $\theta_1, \theta_2,$ and $\theta_3$ be the probability of consumers using the norm metric, the price metric, and the richman metric, respectively. Then we can formalize the problem as follows.

Minimize $\frac{1}{2} \begin{bmatrix} 20\% \cdot \theta_1 & 1\% \cdot \theta_2 & 50\% \cdot \theta_3 & -5\% \\ 30\% \cdot \theta_1 & 10\% \cdot \theta_2 & 10\% \cdot \theta_3 & -15\% \\ 5\% \cdot \theta_1 & 30\% \cdot \theta_2 & 0\% \cdot \theta_3 & -20\% \end{bmatrix}^2,$ subject to $\Theta \geq 0, \theta_1 + \theta_2 + \theta_3 = 1.$

We obtain $\Theta = (0.3074, 0.6026, 0)^T$ by solving the above optimization problem. Thus, we can forecast that the real-world market share of $p_4$ as (10%, 40%, 5%) $\Theta = 0.978%$.

In Section 6, we will show that we can estimate the probability vector $\Theta$ with high accuracy if we know the model set and the market share of a small number of products, based on which, we can find products with higher market share.

4. EXACT ALGORITHM AND HARDNESS

Let us first present the exact algorithm for solving a special case of the $k$-MMP problem when $k = 1$. This will serve as the foundation of our approximation algorithm in Section 5. Then we prove the $NP$-hardness of the $k$-MMP problem when $k \geq 2$.  

4.1 Exact Top-1 Algorithm

One way to find the exact solution of the 1-MMP problem is via exhaustive search. Calculate the expected market share for all candidate products in $P_E$ and select the product with the largest market share. To calculate the expected
market share of a product, we need to check the requirement vectors of all $l$ consumers and the quality vectors of their satisfactory products with time complexity $O(m \cdot l d)$, where $m$ is the number of existing products and $d$ is the dimension of the attribute vector $A$. Assume that we consider a model set $S$ consisting of $m'$ potential product adoption models. Since there are $n$ candidate products, the computational complexity of the exhaustive search is $O(m' \cdot m l d)$.

In the following, we present an enhanced algorithm for the 1-MMP problem based on precomputation. This enhanced algorithm has a lower computational complexity, or $O(m' \cdot (m+n) l d)$. The pseudo code of this algorithm is shown in Algorithm 1.

Algorithm 1: Exact top-1 algorithm

**Input:** $\mathcal{P}_E, \mathcal{P}_M, \mathcal{P}_N, C, S, \Theta$

**Output:** 1-MMP

for all model $t$ in $S$

$\Delta_{\text{sales}}(p_j) \leftarrow 0$;

if $d(i, j) > d_{\text{max}}[i]$ then

$\Delta_{\text{sales}}(p_j) \leftarrow \Delta_{\text{sales}}(p_j) + (1 - m_i[i])$;

else if $d(i, j) = d_{\text{max}}[i]$ then

$\Delta_{\text{sales}}(p_j) \leftarrow \Delta_{\text{sales}}(p_j) + \left(\frac{m_i[i]+1}{m_i[i+1]} - \frac{m_i[i]}{m_i[i]}\right)$;

end

$\Delta_{\text{Sales}}(p_j) \leftarrow \theta_{1} \Delta_{\text{sales}}(p_j) + \ldots + \theta_{m'} \Delta_{\text{sales}}(p_j)$

if $\Delta_{\text{Sales}}(p_j) > \text{max}_\text{increase}$ then

$\text{res} \leftarrow p_j$;

$\text{max}_\text{increase} \leftarrow \Delta_{\text{Sales}}(p_j)$;

end

end

return $\text{res}$

**Lemma 1.** The computational complexity of Algorithm 1 is $O(m' \cdot (m+n)l d)$, where $m' = |S|$, $m = |\mathcal{P}_E|$, $n = |\mathcal{P}_N|$, $l = |C|$, $d = |A|$.

**Proof.** Firstly, it takes $O(d)$ time to calculate the distance for each pair of consumer and product under each adoption model, while there are $l$ consumers, $m$ existing products, and $m'$ product adoption models, so it takes $O(m' l d)$ in total. Then, for each product $p_j \in \mathcal{P}_N$, we calculate the increase of sales caused by adding $p_j$, which takes $O(m' l d)$ time. Since there are $n$ candidate new products, the complexity of these steps is $O(m' l d)$. Therefore, the total computational complexity of Algorithm 1 is $O(m' \cdot (m+n)l d)$.

4.2 Top-k Exact Algorithm

Similarly, exhaustive search is a direct approach to find the exact solution of the k-MMP problem. By enumerating all possible subsets of size $k$ from $\mathcal{P}_N$, and calculating the expected market share of each subset, one can find the set of product with size $k$ which achieves the largest market share. However, the exhaustive approach is not scalable since there exist exponentially many possible subsets. In the following theorem, we formally show that finding the exact solution of the k-MMP problem is NP-hard.

**Theorem 1.** Finding the exact solution for the $k$-MMP selection problem is NP-hard when $k \geq 2$ and the number of attributes is $d \geq 3$.

**Proof.** Please refer to the appendix.

5. APPROXIMATION ALGORITHM

In this section, we extend the top-1 algorithm for the $k$-MMP problem using a greedy-based approximation algorithm. The algorithm is not only computationally efficient, but also provide at least $(1−1/e)$-approximation by exploiting that the market share function is monotone and submodular. In the following, let us first present our approximation algorithm. Then we formally prove its performance guarantee, and finally prove that the market share function we consider is indeed monotone and submodular.

5.1 Greedy-based Approximation Algorithm

Our approximation algorithm is based on the exact top-1 algorithm to solve the top-k problem. The main idea is as follows. We select $k$ products in $k$ steps. In each step, we select the product which is the solution of the exact top-1 algorithm. Furthermore, instead of building the farthest product tables at each step, we only build them in the first step, and then update the tables in the remaining steps. The pseudo code of this algorithm is depicted in Algorithm 2.

Algorithm 2: Approximation top-k algorithm

**Input:** $\mathcal{P}_E, \mathcal{P}_M, \mathcal{P}_N, C, S, \Theta, k$

**Output:** k-MMP

$\mathcal{P}_{res} \leftarrow \emptyset$;

while $|\mathcal{P}_{res}| < k$ do

$\mathcal{P}_{new} \leftarrow \emptyset$;

for $p_{new} \in \mathcal{P}_{new}$ under each model $t$ in $S$

if $d(i, new) > d_{\text{max}}[i]$ then

$\Delta_{\text{sales}}(p_{new}) \leftarrow \Delta_{\text{sales}}(p_{new}) + (1 - m_i[i])$;

else if $d(i, new) = d_{\text{max}}[i]$ then

$\Delta_{\text{sales}}(p_{new}) \leftarrow \Delta_{\text{sales}}(p_{new}) + \left(\frac{m_i[i]+1}{m_i[i+1]} - \frac{m_i[i]}{m_i[i]}\right)$;

end

$\Delta_{\text{Sales}}(p_{new}) \leftarrow \theta_{1} \Delta_{\text{sales}}(p_{new}) + \ldots + \theta_{m'} \Delta_{\text{sales}}(p_{new})$

if $\Delta_{\text{Sales}}(p_{new}) > \text{max}_\text{increase}$ then

$\mathcal{P}_{res} \leftarrow \mathcal{P}_{res} \cap \{p_{new}\}$;

$\mathcal{P}_M \leftarrow \mathcal{P}_M \cup \{p_{new}\}$;

$\mathcal{P}_N \leftarrow \mathcal{P}_N \setminus \{p_{new}\}$;

end

return $\mathcal{P}_{res}$

**Theorem 2.** (Computational complexity) The computational complexity of Algorithm 2 is $O(m' \cdot (m+n)l d)$, where $m' = |S|$, $m = |\mathcal{P}_E|$, $n = |\mathcal{P}_N|$, $l = |C|$, $d = |A|$.

**Proof.** Based on Lemma 1, it takes $O(d)$ time to build these farthest product tables and $O(m' l d)$ time to find the exact solution of 1-MMP. The complexity of updating tables is only $O(l d)$. Since we only build the tables once and find the 1-MMP $k$ times in Algorithm 2, the computational complexity of Algorithm 2 is $(m' \cdot (m+n)l d)$. 


5.2 Guarantee on Solution Quality

To prove the performance guarantee of our approximation algorithm, let us first introduce the notion of “submodular set function” [11].

**Definition 4. (Submodular set function)** Given a finite ground set \( U \), a function \( f \) that maps subsets of \( U \) to real numbers is called submodular if
\[
f(S \cup \{u\}) - f(S) \geq f(T \cup \{u\}) - f(T), \quad \forall S \subseteq T \subseteq U, u \in U. \quad (7)
\]

Next, we show one interesting property of submodular set functions [5], based on which we design our approximation algorithm with theoretical performance guarantee.

**Theorem 3.** For a non-negative monotone submodular function \( f : 2^U \rightarrow \mathbb{R} \), let \( S \subseteq U \) be the set of size \( k \) obtained by selecting elements from \( U \) one at a time, each time choosing the element that provides the largest marginal increase in the function value. Let \( S' \subseteq S \) be the set that maximizes the value of \( f \) over all \( k \)-element sets. Then we have \( f(S) \geq (1 - 1/e) \cdot f(S') \). In other words, \( S \) provides a \((1-1/e)\)-approximation, or guarantees a lower bound on the quality of solution as compared to the optimal solution.

Applying to the \( k\)-MMP problem, the ground set is \( \mathcal{P}_M \cup \mathcal{P}_N \), the market share function \( MS(\cdot) \) defined in Section 2 maps subsets of \( \mathcal{P}_M \cup \mathcal{P}_N \) to real numbers, i.e., the expected market share of products. According to Theorem 3, if we can prove that \( MS(\cdot) \) is a non-negative monotone submodular set function, then our approximation Algorithm 2 can provide a \((1-1/e)\)-approximation. We leave the proof of these properties in the next subsection, and once we prove them, we have the following theorem.

**Theorem 4. (Performance guarantee)** The approximation algorithm stated in Algorithm 2 provides at least \((1-1/e)\)-approximate solutions compared with the optimal ones, where \( e \) is the base of the natural logarithm.

**Proof.** According to Theorem 5, which will be proved in the following subsection, the market share function \( MS(\cdot) \) in Equation (3) is non-negative, monotone submodular. So, according to Theorem 3, Algorithm 2 provides \((1-1/e)\)-approximate solutions.

5.3 Submodular Market Share Function

Let us consider the market share function \( MS(\cdot) \) defined in Section 2. According to the definition of \( MS(\cdot) \), it is obviously non-negative, so we seek to prove the monotonicity and submodularity properties. For the ease of presentation, we define the following notations. For any set \( S \subseteq \mathcal{P}_M \cup \mathcal{P}_N \) of products, let \( \mathcal{P}_S = \mathcal{P}_M \cup S \) and \( S_j = S \cup \{p_j\} \), let \( pr_i(S) = \sum_{p_i \in S} \Pr(i, j|\mathcal{P}_S) \) denote the probability of the consumer \( c_i \) adopting products in \( S \) when a set \( \mathcal{P}_S \) of products is available in the market. Furthermore, when a set \( \mathcal{P} \) of products is available in the market, we define \( RC(p_j|\mathcal{P}) \) as the set of consumers that \( p_j \) is their farthest product, and recall that \( FP(c_i|\mathcal{P}) \) is the set of farthest products from \( c_i \).

One key fact we use in our proof is that by adding a new product, say \( p_u \), only those consumers in \( RC(p_u|\mathcal{P}_u) \) will change their product adoption decisions. Therefore, to calculate the change of market share caused by adding \( p_u \), we only need to consider the consumers in \( RC(p_u|\mathcal{P}_u) \). Mathematically, we have the following proposition.

**Proposition 1.** Let \( \mathcal{P}_S \) be the set of products in the market, by adding a new product \( p_u \) into the market, \( p_u \in \mathcal{P}_N \setminus \mathcal{P}_S \), the increase of the market share of products in \( S_u \) is
\[
MS(S_u) - MS(S) = \sum_{c_i \in RC(p_u|\mathcal{P}_S)} \frac{1}{1} \left[ pr_i(S_u) - pr_i(S) \right]. \quad (8)
\]

Based on Proposition 1, we now proceed to prove the monotonicity and submodularity of the market share function \( MS(\cdot) \). First, we prove two lemmas (Lemma 2 and 3). Based on these two lemmas, we prove the monotonicity and submodularity properties in Theorem 5.

**Lemma 2.** Let \( S \subseteq \mathcal{P}_M \cup \mathcal{P}_N \) be a set of products, and \( p_u \) be another product in \( \mathcal{P}_N \setminus S \). For a consumer \( c_i \in C \), if \( c_i \in RC(p_u|\mathcal{P}_N \setminus S) \), then we have
\[
pr_i(S_u) - pr_i(S) \geq pr_i(T_u) - pr_i(T). \quad (9)
\]

**Proof.** Please refer to the appendix.

**Lemma 3.** Let \( S \) and \( T \) be two sets of products, \( S \subseteq T \subseteq \mathcal{P}_M \cup \mathcal{P}_N \), and \( p_u \) be another product in \( \mathcal{P}_N \setminus T \). For a consumer \( c_i \in C \), if \( c_i \in RC(p_u|\mathcal{P}_T) \), then we have
\[
pr_i(S_u) - pr_i(S) \geq pr_i(T_u) - pr_i(T). \quad (10)
\]

**Proof.** Please refer to the appendix.

**Theorem 5.** Suppose consumers adopt products following the distance-based adoption model, then the market share function \( MS(\cdot) \) defined in Equation (3) is monotone submodular for the \( k\)-MMP problem.

**Proof.** We prove the monotonicity property first. To prove the monotonicity property, we need to show
\[
MS(S_u) - MS(S) \geq 0 \quad \forall S \subseteq \mathcal{P}_N \cup \mathcal{P}_M, p_u \in \mathcal{P}_N \quad (11)
\]
holds, which can be proved by combining the results of Proposition 1 and Lemma 2.

To prove the submodularity property, according to Definition 4, we need to show
\[
MS(S_u) - MS(S) \geq MS(T_u) - MS(T) \quad (12)
\]
holds \( \forall S \subseteq T \subseteq \mathcal{P}_N \cup \mathcal{P}_M \) and \( p_u \in \mathcal{P}_N \).

In the case of \( p_u \in S \), Inequality (12) holds since both sides are equal to 0. In the case of \( p_u \in T \setminus S \), the right side of the inequality equals 0, while according to the monotonicity, which has been proved, the left side is non-negative. Hence Inequality (12) also holds. In the case of \( p_u \in T \setminus S \), Inequality (12) can be easily proved by combining the results of Proposition 1 and Lemma 3. Thus, Inequality (12) holds\( \forall S \subseteq T \subseteq \mathcal{P}_N \cup \mathcal{P}_M \) and \( p_u \in \mathcal{P}_N \).

6. EXPERIMENTS

We perform experiments on both synthetic datasets and real-world web datasets. We implement our approximation algorithm and the exhaustive search algorithm in C++ and perform experiments on a PC with a 16-core 2.4GHz CPU, 30 GB of main memory under the 64-bit Debian 6.0. First, we use synthetic datasets to evaluate the computational efficiency and accuracy of our approximation algorithm. Then we apply our algorithm on the real-world web datasets to show the impact of different distance metrics, and how to learn distance metrics from some historical sales data and to perform product selection.
6.1 Speedup and Accuracy

We generate the synthetic datasets using the generator provide by [1]. In a real-world market, products usually do not have high quality on all attributes. Instead, they have high quality on some subset of attributes only. For example, a smart phone with a large screen will have high quality on display but low quality on portability. Furthermore, if a product has high quality on most attributes, then the price of this product will be high in general, which indicates low quality on the price attribute. We generate the datasets of products with negative correlation on attributes: Products which have high quality in one attribute tends to have low quality on at least one other attribute. On the other hand, we generate the consumers’ requirement of each attribute independently using a uniform distribution.

We compare the running time and the market share between our approximation algorithm (or greedy) and the exhaustive search algorithm (or exh). We examine the impact of various factors, including the size of datasets (n, m, l, d), the number of new products we need to select (k), and models using different distance metrics (four distance metrics as introduced in Section 2). The default settings of these parameters are: m = 100, n = 20, l = 10, 000, d = 10, k = 2. The computational efficiency and accuracy our experiments are similar under all distance models, so we only show the results for the norm distance metric.

Note that both the running time of our approximation algorithm and the exhaustive algorithm increases linearly with m, l, and d, due to the page limit, we only show the results of varying k and n while keeping other parameters as default values. Table 2 shows the speedup of our approximation algorithm over the exhaustive algorithm. Figure 2 shows the running time of these two algorithms, where the horizontal axis depicts the variation on parameters n (number of candidate products we need to consider) and k (number of products we need to select), while the vertical axis depicts the log scale of the running time, in seconds.

From the table and the figure, one can observe that our approximation algorithm is significantly faster than the exhaustive algorithm: O(nk) times faster when selecting k products from n candidate products. The speedup is around 285,000 even for a small dataset (i.e., select k = 5 product-s from n = 20 candidates). In this case, the running time of exhaustive algorithm is around 40 hours. In the case of selecting five products from n = 80 candidates, our conservative estimate on the running time of the exhaustive algorithm is about 10 years. In contrast, the running time of our approximation algorithm for all cases remain in less than one second. We also test our approximation algorithm on a larger dataset where m = 1, 000, n = 100, l = 1, 000, 000. We select k = 8 new products from the 100 candidates. Our approximation algorithm still only takes about 7 minutes.

Figure 3 depicts the expected market share of the two algorithms. One can observe that our approximation algorithm provides high accuracy: about 0.96 approximation on average as compared with the optimal solution obtained using the exhaustive algorithm. This shows that our algorithm generates results which is much better than the theoretical lower bound guarantee. In fact, the results of the two algorithms are exactly the same for over 80% of all experiments we performed and our approximation algorithm still provides a 0.82 approximation even under the worst case scenario among all experiments.

<table>
<thead>
<tr>
<th>n</th>
<th>k = 2</th>
<th>k = 3</th>
<th>k = 4</th>
<th>k = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>65.89</td>
<td>1160.37</td>
<td>18799.17</td>
<td>285804.08</td>
</tr>
<tr>
<td>40</td>
<td>256.62</td>
<td>8111.76</td>
<td>287626.53</td>
<td>915.38</td>
</tr>
<tr>
<td>60</td>
<td>535.29</td>
<td>26511.67</td>
<td>535.29</td>
<td>18799.17</td>
</tr>
<tr>
<td>80</td>
<td>915.38</td>
<td>57812.33</td>
<td>1160.37</td>
<td>256.62</td>
</tr>
</tbody>
</table>

Table 2: Speedup: varying k and n
6.2 Impact of Distance Metrics

In this subsection, we perform experiments on a real-world web dataset and aim to show the influence of using different distance metrics.

We extract the TripAdvisor dataset from [15]. Hotels and reviewers of these hotels are considered as products and consumers respectively in this dataset. The reviewers rated hotels on seven attributes: value, room, location, cleanliness, front desk, service, and business service. We use the average rating of an attribute as the quality of that attribute for each hotel. We also add the inverse of the average price of the hotel as the eighth attribute, which is normalized in the range of (1, 5). For each consumer, we extract requirement vector as follows. Let $\bar{r}$ be the average rating of a hotel’s attribute and $r_i$ be the rating from the consumer $c_i$. If $r_i$ is lower than $\bar{r}$, it means that $c_i$ has a higher requirement than average, and if $r_i$ is higher than $\bar{r}$, $c_i$ may have a lower requirement than the average. Thus, we set the requirement of $c_i$ as $\bar{r} + (r - \bar{r})$. For example, if $\bar{r} = 3.5$ and $r_i = 4$, then the requirement of $c_i$ will be $3.5 + (3.5 - 4) = 3$. Table 3 shows the overall statistics of the dataset.

<table>
<thead>
<tr>
<th># of products</th>
<th># of consumers</th>
<th># of attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,605</td>
<td>186,249</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 3: Parameters of our web datasets

We select the first 605 hotels as the candidate products and set the remaining 1000 hotels as the existing products. We apply our approximation algorithm to solve the 2-MMP problem using the four distance metrics introduced in Section 2: discrete metric (DM), norm metric (NM), price metric (PM), and richman metric (RM). The results are shown in the first four rows of the second column in Table 4. One can observe that the results vary greatly when we use different distance metrics. This implies the importance of inferring and understanding consumers’ adoption behavior.

<table>
<thead>
<tr>
<th>distance metrics</th>
<th>ID of selected products</th>
<th>market share of selected products</th>
</tr>
</thead>
<tbody>
<tr>
<td>DM</td>
<td>214, 566</td>
<td>32.33%</td>
</tr>
<tr>
<td>NM</td>
<td>284, 214</td>
<td>11.20%</td>
</tr>
<tr>
<td>PM</td>
<td>566, 350</td>
<td>35.43%</td>
</tr>
<tr>
<td>RM</td>
<td>284, 214</td>
<td>11.20%</td>
</tr>
<tr>
<td>$\Theta$</td>
<td>566, 284</td>
<td>38.09%</td>
</tr>
</tbody>
</table>

Table 4: Results of the 2-MMP problem

6.3 Learning Distance Metrics

In the following, we evaluate the accuracy of our learning method using the same dataset in the last subsection. Since we do not have the information about products’ real-world market share and consumers’ adoption models, we manually set the probability $\Theta$ that consumers use the above four distance metrics. Then we randomly set the distance metric for each consumer according to $\Theta$ and estimate the “real-world market share” by enumerating each consumer’s choice. We estimate the probability as $\Theta$ using the learning method in Section 3 and compare the normalized root-mean-square error (NRMSE) between $\Theta$ and $\hat{\Theta}$ to evaluate the accuracy of our learning method. Note that NRMSE ranges in (0, 1) and lower value implies higher accuracy.

We present the experimental results in the case that $\Theta = (0.1, 0.2, 0.6, 0.1)^T$ and the “real-world market share” of a set $P'_E$ of five products are known. Firstly, we calculate the expected market share of these products under all the four potential models. The results are shown in Table 5 along with the “real-world market share”.

<table>
<thead>
<tr>
<th>ID</th>
<th>DM</th>
<th>NM</th>
<th>PM</th>
<th>RM</th>
<th>real-world</th>
</tr>
</thead>
<tbody>
<tr>
<td>91</td>
<td>0.21</td>
<td>0.13</td>
<td>4.81</td>
<td>0.13</td>
<td>2.93</td>
</tr>
<tr>
<td>500</td>
<td>0.40</td>
<td>0.14</td>
<td>3.33</td>
<td>0.07</td>
<td>0.27</td>
</tr>
<tr>
<td>517</td>
<td>1.30</td>
<td>49.83</td>
<td>2.10</td>
<td>25.38</td>
<td>13.75</td>
</tr>
<tr>
<td>746</td>
<td>1.07</td>
<td>0.79</td>
<td>0.81</td>
<td>11.40</td>
<td>1.87</td>
</tr>
<tr>
<td>350</td>
<td>0.68</td>
<td>1.09</td>
<td>3.80</td>
<td>1.09</td>
<td>2.64</td>
</tr>
</tbody>
</table>

Table 5: Market share (%)
the concept of dominance, which is used as skyline operators [1] to analyze various forms of relationships between products and consumers. A manufacturer can position popular products effectively while remaining profitable by analyzing the dominance relationships. The works in [16, 14, 13, 12] considered the situation that there exist multiple manufacturers. The authors of [16] derived the Nash Equilibrium when each manufacturer modifies its product in a round robin manner to maximize the market share. Wan et al. [14] aimed to find the most competitive products which are not dominated by any competitors without taking into account the consumers. They extended their work in [13, 12] by considering the consumers’ preferences. However, the above papers all aimed to maximize the number of potential consumers, which is not equivalent to the market share derived in this paper. In fact, potential consumers may not lead to higher market share because different consumers have different probability to adopt new products. Authors in [8] aimed to find the products with the maximum expected number of total adopters, which is similar with the market share in our paper. But their algorithm could not provide any theoretical performance guarantee. Furthermore, none of the previous works consider the complicated product adoption behavior of consumers. Instead, they assumed that consumers will make randomly product adoption decisions, which corresponds to a special case of our product adoption model using the discrete norm.

Maximization of submodular functions: Submodular functions have properties which are very similar to the convex and concave functions. The authors of [2, 11] showed that a natural greedy hill-climbing strategy can achieve a provable performance guarantee for a problem of maximizing a non-negative monotone submodular function: at least 63% of optimal. Due to the generality of this performance guarantee, this results has found applications in a number of areas, e.g., discrete optimization [10], materialized view [4], and influence maximization [5].

8. CONCLUSIONS

In this work, we present the problem of finding the k most marketable products (k-MMP) under a distance-based adoption model. Our adoption model is general in that we can use different distance metrics to describe various consumers’ adoption behaviors. Given some historical data sets on market share, we propose a learning method to select the appropriate distance metrics to describe consumers’ production adoption behavior. We prove that the k-MMP problem is \( NP-hard \) when \( k \geq 2 \) and the number of products’ attributes, \( d \), is three or more. We propose a polynomial time approximation algorithm to solve the k-MMP problem. Using the submodularity analysis, we formally prove that our approximation algorithm can guarantee a \((1-1/e)\)-approximation as compared to the optimal solution. We compared our approximation algorithm with the exhaustive search algorithm on the synthetic datasets. The results showed that our approximation algorithm can achieve \( O(n^k) \) times speedup when selecting \( k \) products from \( n \) candidates. Furthermore, the solution quality of our algorithm is about 96% on average, which is much higher than the theoretical lower bound. We also perform experiments on the real-world web datasets to show the crucial impact of different distance metrics and how we can improve the accuracy of product selection using our distance metric selection method.

9. ACKNOWLEDGMENTS

The work of John C.S. Lui is supported in part by the GRF Grant 415112.

10. REFERENCES

APPENDIX

- **Proof of Theorem 1:**

  **Proof.** The NP-hardness proof can be achieved by transforming an NP-hard problem, called the top-
k Representa-
tive Skyline Product (top-k RSP) [9], to a special case of the k-MMP problem.

  Let us state the top-k RSP [9]. Given a set U of points and a positive integer k, compute a set of k skyline points such that the number of points dominated by these k points is maximized. A point p = (p[1], p[2], ..., p[d]) dominates another point q = (q[1], q[2], ..., q[d]) if p[i] ≥ q[i] for 1 ≤ i ≤ d and there exists at least one dimension k such that p[k] > q[k], and we denote this as p ≻ q. Consequently, the skyline point is defined as follows. Given a set U of points, the skyline points of U are the set of U points which are not dominated by any points in U.

  Given an instance of top-k RSP problem, we construct an instance of k-MMP problem, which can be carried out as follows. Let BE = ∅, i.e., m = 0. Let P be the set of skyline points in U and C be the rest, i.e., C = U \ P. Note that in general, the concept of dominance is different from product satisfiability as stated in Definition 1. Formally, we have p > c, ⇒ p j > c, but p j > c, ⇒ p > c. However, if p j > c, then the quality vector p is exactly the same with the requirement vector of c, i.e., p j and c have the same location in the d-dimensional space. But in our construction, the product points are skyline points while the consumer points are not, so there does not exist such kind of c and p j pairing in our construct. Therefore, we can treat dominance and product satisfiability to be the same in this instance.

  Let P be the set of k products we select from P. In this case, since there is no existing product, so if a consumer has any satisfactory product in P, the consumer will adopt one unit of products in P, otherwise, 0. As a result, the expected number of adopters is equal to the number of consumers who have satisfactory products in P. In another word, MS(P) is equal to the total number of points dominated by the skyline points in P divide by the total number of non-skyline points. Since the total number of non-skyline points is fixed, the result of the corresponding top-k RSP problem is also the result of this instance of the k-MMP problem.

  Therefore, any instance of the top-k RSP problem can be transformed to an instance of the k-MMP problem. Since the top-k RSP problem has been proved to be an NP-hard problem, the k-MMP problem is also NP-hard.

- **Proof of Lemma 2:**

  **Proof.** To simplify the proof, we define the following notations. Let σ = |FP(c[S]|PS]| and s = |FP(c[S]|PS)]∩S|, where c ≥ s ≥ 1. Let d = d, where p ∈ FC(c[P]). Since c ∈ FR(pS|PS), we have d > d. If d > d, then c will adopt p with probability 1, i.e., pr(S) = 1. While pr(S) ≤ s, so Inequality (9) holds. If d = d, then FC(c[S]|PS) = FP(c[S]|PS)∪{p}, i.e., |FP(c[S]|PS)|= s−1. Similarly, we have |FP(c[S]|PS)|= s−1. When s = 1, FC(c[S]|PS) is an empty set, which means p is c’s only choice, the situation is the same with the case of d = d. So we only need to consider the case when s−1. Bring the notations into Inequality (9), we have

  \[ pr(S) - pr(S) = s \frac{\sigma - 1}{\sigma - 1} = \frac{\sigma - s}{\sigma(\sigma - 1)} \geq 0, \]

  which can be proved by observing that s ≥ σ ≥ 1.

- **Proof of Lemma 3:**

  **Proof.** Because S ⊆ T, p has more competitors when a set T of products is available in the market. As a result, FC(pT|T) ⊆ FC(pS|S). Since c ∈ FC(pT|T), we have c ∈ FC(pS|S). Follow the same notations in the proof of Lemma 2, let us consider the case of s = 1. In this case, pr(S) = 1, pr(S) = 0, so the left side of Inequality (10) equals to 1. While the right side of the inequality is obviously no larger than 1, so the Inequality (10) holds. Now let us consider the case when s > 1. According to the proof of Lemma 2, we have

  \[ pr(S) - pr(S) = s \frac{\sigma + 1}{\sigma + 1} - s \frac{\delta}{\sigma + 1} = \frac{\sigma - s}{\sigma + 1} \geq 0 \]

  According to Equation (14) and (15), we can derive Inequality (10) as follows.

  Inequality (10) holds

  \[ \frac{\sigma - s}{\sigma - 1} \geq \frac{\sigma + \delta}{\sigma + \delta - 1} \]

  Hence, we only need to show that Inequality (16) holds, which can be proved by observing that s ≥ 1, δ ≥ 0 and s ≥ σ. □