Manufacturing & Service Operations Management

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Pricing Personalized Bundles: A New Approach and An Empirical Study

Zhengliang Xue, Zizhuo Wang, Markus Ettl

To cite this article:
Zhengliang Xue, Zizhuo Wang, Markus Ettl (2016) Pricing Personalized Bundles: A New Approach and An Empirical Study. Manufacturing & Service Operations Management 18(1):51-68. https://doi.org/10.1287/msom.2015.0563

M&SOM MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Vol. 18, No. 1, Winter 2016, pp. 51–68
ISSN 1523-4614 (print) | ISSN 1526-5498 (online)

informs

http://dx.doi.org/10.1287/msom.2015.0563

Pricing Personalized Bundles: A New Approach and an Empirical Study

Zhengliang Xue
IBM T. J. Watson Research Center, Yorktown Heights, New York 10598, zxue@us.ibm.com

Zizhuo Wang
Department of Industrial and Systems Engineering, University of Minnesota, Minneapolis, Minnesota 55414, zwang@umn.edu

Markus Ettl
IBM T. J. Watson Research Center, Yorktown Heights, New York 10598, msettl@us.ibm.com

This paper studies the pricing strategies for personalized product bundles. In such problems, a seller provides a variety of products for which customers can construct a personalized bundle and send a request for quote (RFQ) to the seller. The seller, after reviewing the RFQ, has to determine a price based on which the customer either purchases the whole bundle or nothing. Such problems are faced by many companies in practice, and they are very difficult because of the potential unlimited possible configurations of the bundle and the correlations among the individual products. In this paper, we propose a novel top-down and bottom-up approach to solve this problem. In the top-down step, we decompose the bundle into each component and calibrate a value score for each component. In the bottom-up step, we aggregate the components back to the bundle, define important features of the bundle, and segment different RFQs by those bundle features as well as customer attributes. Then we estimate a utility function for each segment based on historical sales data and derive an optimal price for each incoming RFQ. We show that such a model overcomes the aforementioned difficulties and can be implemented efficiently. We test our approach using empirical data from a major information technology service provider and the test result shows that the proposed approach can improve the effectiveness of pricing significantly.

Keywords: pricing; personalized bundle; utility model; data analysis
History: Received: February 28, 2014; accepted: September 9, 2015. Published online in *Articles in Advance* November 13, 2015.

1. Introduction

In this paper, we consider pricing strategies for personalized product bundles with distinctive configurations. This research is motivated by a practical problem faced by a leading information technology (IT) service provider. In this problem, a seller provides a large number of different products to its customers. Each customer, after reviewing the products the seller provides, constructs a personalized bundle of products according to her need,[1] and submits a request for quote (RFQ) to the seller. The seller, after receiving the RFQ, has to offer the customer a price for the bundle of products she requests. The customer then decides whether to purchase the entire bundle at the seller's price.

The above process exists in many of today's business settings. For example, when an IT company (e.g., Google) purchases a variety of computers and servers from a hardware company (e.g., Dell), or a construction company (e.g., Emerson Construction Company) purchases a fleet of construction vehicles and equipment from a manufacturer (e.g., Caterpillar), an RFQ process could take place.[2] From the seller's point of view, designing a pricing solution for such a process is extremely important because the company's revenue and profit hinge on the success of the pricing mechanism.

In the company that motivated our research, the pricing decisions were made by humans before this study. That is, the company employs a group of professionals (called pricers), who review incoming RFQs and decide the prices based on their experiences and some business guidelines. However, with the fast development of information technology accompanied by the growing number of quantitative tools for analyzing data, such methods are often ineffective. Indeed, it is extremely difficult for a human to process large sets of data to obtain useful information

[1] To reduce confusion, in this paper, we use *she* or *her* to refer to the customer, and *he* or *his* to refer to the seller.

[2] In practice, there are other purchase (or procurement) formats, such as negotiation or auctions. In this paper, we focus on the RFQ process.

or to compute an optimal price quantitatively. Thus the traditional method is very inefficient, especially when considering the overhead incurred in hiring the pricers to perform this task.

In this paper, we propose a new approach to achieve the pricing task for personalized product bundles when abundant historical sales data are available. We develop a procedure that utilizes and analyzes historical data, and finally outputs an *optimal price* for each incoming RFQ. Our procedure takes into account useful characteristics of each RFQ, predicts the behavior of each customer, and can be fully automated by computers. We show using actual business data that our approach indeed helps companies increase revenue and profit in a significant manner.

Compared to the extensive literature on customized pricing, the problem we study is unique. In the traditional customized pricing literature, the focus is typically on pricing for different customer segments for a generic product or a set of products, in which customer segmentation is mainly based on customer characteristics and/or the product the customer chooses (among a few alternatives). In our problem, the seller faces bid requests of distinctive configurations, and literally there could be an infinite number of possible configurations for a bundle. Therefore, we have to first figure out how to segment incoming RFQs based on the bundle configurations, which is a unique task in our setting. To make things more difficult, in our problem, different components in a bundle may correlate in certain ways, e.g., some of them may be complementary (or substitutable) to each other, which creates a further hurdle for estimating the customer valuation for each configuration. Finally, as most customer's configurations have never been seen before, it is not viable to compute the optimal prices for each bundle offline. Instead, all the pricing decisions have to be made in an online fashion. However, the decisions have to be made fast because customers frequently demand a prompt response from the seller.

To address these difficulties, we propose a novel two-step method to estimate the customer's purchase behavior toward personalized bundles. The method consists of a top-down step and a bottom-up step. In the top-down step, we fully decompose the bundle to each single component, group them based on the similarity of component features, and evaluate their intrinsic values by giving each component a "value score." To carry out this step, we adapt the regression tree model in the statistical learning literature to iteratively perform the analysis and group the components. Then in the bottom-up step, we aggregate the component value scores back to the bundle level, creating several useful bundle characteristics. Then we segment the RFQs based on the bundle and customer characteristics. Within each segment, we estimate a unique utility function for each RFQ using available historical data. Finally, we adopt a logit choice model to translate the utility function into a purchase probability and compute the optimal bundle price.

The proposed approach has several advantages. First, by performing the top-down step and computing the component value scores, the method captures the intrinsic value of each component, which is the foundation for estimating the bundle value. Second, by utilizing the component value scores, we define some characteristics for the bundle. These characteristics reveal important features of a bundle, including possible interactions between different components. Combining these characteristics with customer information, we could segment the incoming RFQs and apply customized pricing for each segment. Lastly, the proposed approach can be accomplished efficiently in practice. In particular, the top-down step can be done offline (with occasional updates) and the bottom-up step can be carried out in real time immediately after receiving each RFQ, and an optimal price can be computed instantly. The inputs required for our approach are historical RFQ data including both win and loss cases, which we believe are available in many large companies. Therefore, the proposed strategy can be easily implemented in practice.

In this paper, we demonstrate the strength of our proposed approach using real-world data from the company that motivated our research. In actual implementations, we use both historical data and those obtained from the operational use of the method. The results show significant improvement of revenue and profit after adopting the new pricing approach. Therefore, we believe the proposed method is promising for helping companies that face similar problems improve their pricing efficiency in the era of big data.

1.1. Literature Review

There are two streams of literature that are related to our work: the literature on bundle pricing and the literature on customized pricing. In this section, we review the relevant works and position our work in the literature.

Bundling, the strategy of selling products in particular combinations, has long been a focus of research interest in the economics, marketing, and operations management communities. Broadly speaking, there are two types of bundling strategies: seller-determined bundles and customer-designed bundles (personalized bundles). Most of the prior literature studies the seller-determined bundles. We refer to Venkatesh and Mahajan (2009) and Hanson and Martin (1990) for comprehensive reviews on the literature of seller-determined bundles. There are two main decisions in such a scheme: (1) what bundles to offer and (2) how to price the bundles and the individual

products. To analyze such decisions, the main issues to consider are the nature of the heterogeneity in consumers' valuations of products, the correlation of component valuations, the degree of complementarity or substitutability of products, etc. Compared to these works, we consider a setting where the bundle is determined by the customer, and the customer's valuation of each product (and the correlation between these valuations) is unknown to the seller. Therefore, our problem is different and in some sense more challenging than those studied in this literature.

On the personalized bundle side, the literature is relatively scarce. In Hitt and Chen (2005), the authors propose a pricing scheme in which the customers may select a fixed number of goods for a certain fixed price. They name such a scheme "customized bundle pricing" and study the properties of such a pricing scheme and compare it to other traditional bundling schemes. However, in their setting, the price is not really personalized to each bundle configuration, and the main theme is still bundle design rather than customized pricing. Several subsequent works take a similar approach, e.g., Wu et al. (2008), Jiang et al. (2011), Basu and Vitharana (2011). In contrast, in our setting, instead of listing the price for each combination, the seller sets the price after reviewing the requested bundle from the customer. And the price should utilize the information revealed from the customer's request, including both bundle characteristics and customer's attributes. In addition, our problem is purely data driven and does not assume any particular valuation structure of the customers. Such features distinguish our problem from those in this literature.

Our work is also closely related to the literature of customized pricing. For comprehensive reviews of this literature, we refer the readers to Phillips (2005), Bodea and Ferguson (2014), and Murthi and Sarkar (2003). In the customized pricing literature, the focus is mainly on pricing for different customer segments for a generic product (or a small set of products). The key in such problems is to find the right segmentation of the customers and to determine the price sensitivities in each segment. For example, airlines or hotels segment customers based on the time of purchase and the preference among different cabins or different types of rooms, among many other characteristics. Then based on these characteristics, different customers are offered with different prices. Similar pricing strategies are widely adopted in B2B settings, in which customers are segmented based on whether they are large or small customers, whether they are existing or new customers, etc. In a high level, our work shares a similar spirit with this literature: We also segment the customers and offer them customized prices. However, the way we perform the segmentation is unique. Instead of focusing on the customer characteristics, we segment mainly based on the configuration of the bundle requested by the customer. In our case, there are literally an infinite number of possible configurations a customer could request, therefore, how to find a proper segmentation is nontrivial. Moreover, within each segment, there are still many possible bundle configurations, and how to estimate the customer's valuations on different bundles also presents a challenging task.

There have been some studies of customized pricing in a bundle context. Such studies often originate from the travel industries, where airlines or hotels respond to travelers who request multidestination itineraries or multiday stays in a hotel. A review of these literature can be found in Talluri and van Ryzin (2004). Two main methods have been studied in the literature. One is called the bid-price strategy, in which a bid price is computed for each resource and the price of a bundle is simply set to be the sum of the bid prices of the components involved, see, e.g., Talluri and van Ryzin (1998). The other method formulates the problem as a dynamic program and uses the solution to determine the control method, see, e.g., Bertsimas and Popescu (2003). However, there are two main differences between those two methods and the problem we study. First, in those studies, the arrival rates of each type of customers (with certain requests) are assumed to be known, and the analyses are based on that information. However, in our work, no distributional information about the request types are known, thus those results are not applicable in our setting. Second, both the bid-price control and the dynamic programming control are still locally linear controls (meaning the price for a bundle is the sum of prices for each component, given the bundle size is small). Such control methods do not explicitly take into account the possible interactions (e.g., complementarity or substitutability) of the components that exist in our application. Therefore, our research is quite different from theirs.

1.2. Organization of the Paper

The rest of the paper is organized as follows: In §2, we propose a general framework to deal with the personalized bundle pricing problem. In §3, we customize our framework to a concrete example. In §4, we demonstrate the strength of our approach using an empirical study from a major IT company, and the financial impact of our method is assessed. Section 5 summarizes our main results.

2. Estimation Framework

We consider a seller who offers I components from his product line. We index the components by $i \in \mathcal{I} = \{1, \ldots, I\}$. By the nature of the components, we can classify them into one of M product families,

$\mathcal{J}_1, \ldots, \mathcal{J}_M$ and each component i belongs to a certain product family $\mathcal{J}_m$. For example, a typical IT company can classify its products into hardware, software, maintenance, etc. Sometimes the components can be grouped to even more detailed product families based on their functionality.

Besides belonging to a certain product family, each component i is endowed with some attributes, which we denote by a vector $\mathbf{a}_i = (a_{i1}, a_{i2}, \ldots, a_{iK})$. These attributes are intrinsic to the component and are viewed as known parameters in our study. For example, the attributes could include the cost, list price, certain level of delegation prices,[3] etc. In particular, those attributes could take continuous or discrete (or binary) values.

In our model, customers make purchases through the RFQ process. In such a process, the customers can submit to the seller an RFQ for a bundle of components. Any nonnegative integer quantity of each component is allowed in a bundle. Mathematically, a request of a customer can be specified by an I-dimensional vector

$$\mathbf{d} = (d_1, \ldots, d_I),$$

where $d_i \geq 0, d_i \in \mathbb{Z}$ denotes the amount of component i this customer is interested in purchasing.

For each RFQ from a customer, there is customer type information $\mathbf{z}$ associated with it, where $\mathbf{z} \in \mathcal{Z}$. Here we assume $\mathcal{Z}$ contains all the relevant customer information, such as region, channel, industry sector, loyalty, purchase history, etc.

After receiving each RFQ, the seller must provide a quoted price to the customer for her requested bundle. The customer, after receiving the quoted price, can decide either to make the purchase of the whole bundle she requested or leave the system without purchasing. We assume the customer who decides not to purchase will not come back to the system and we do not allow partially accepting an order. Also, we assume that there are sufficient supplies for each component, so inventory is not a consideration in our model.

Assume the seller has J historical RFQ data in his database indexed by j. In each historical sales data j, we have the following information:

- The corresponding customer type information $\mathbf{z}_j$.
- The RFQ specification $\mathbf{d}_j = (d_{1j}, \ldots, d_{Ij})$.
- The quoted price p_j.
- The customer's response to RFQ j: $\xi_j \in \{0, 1\}$.

That is, whether she purchased or not at price p_j.

Our objective is to use the historical data to calibrate a model to characterize the purchase behaviors of the customers and develop a pricing tool for this process.

To this end, we will establish a utility model for any bundle configuration, customer attribute, and quoted price. That is, we will establish a utility function:

$$u(\mathbf{d}, \mathbf{z}, p), \tag{1}$$

where $\mathbf{d}$ is the bundle configuration, $\mathbf{z} \in \mathcal{Z}$ is the customer type, and p is the quoted price. With such a utility function, we use a discrete choice model, e.g., the logit model, to estimate the purchase probability of an RFQ:

$$q(\mathbf{d}, \mathbf{z}, p) = \frac{\exp(u(\mathbf{d}, \mathbf{z}, p))}{1 + \exp(u(\mathbf{d}, \mathbf{z}, p))}. \tag{2}$$

Then we can choose an optimal price p to maximize the expected profit (or revenue) of the seller:

$$\max_p (p - c(\mathbf{d})) \cdot q(\mathbf{d}, \mathbf{z}, p), \tag{3}$$

where $c(\mathbf{d})$ is the cost to fulfill an order $\mathbf{d}$. It is easy to see that given the form of $q(\mathbf{d}, \mathbf{z}, p)$, the above optimization problem is a single variable one and thus can be solved efficiently (we refer to Phillips 2005 for discussions of such optimization problems). Therefore, the key in this process is how to make use of the historical data to calibrate a proper utility function $u(\mathbf{d}, \mathbf{z}, p)$. In the traditional customized pricing literature, there are many works that estimate a utility function for a fixed product given customer information $\mathbf{z}$ and price p. Those works typically use the customer information to divide the customers into different segments and calculate a unique utility function for each segment. Compared to the prior literature, the difficulty in our problem lies in how to incorporate the bundle configuration information $\mathbf{d}$ into the utility function. Intuitively, there are two roles that $\mathbf{d}$ could play in the utility function:

1. The bundle configuration $\mathbf{d}$ directly affects the customer's valuation of the bundle, because it changes the actual products included in the bundle.
2. The bundle configuration $\mathbf{d}$ may reveal important type information of the corresponding customer, therefore, providing important references for segmentation.

A proper utility function should capture both effects. In the following subsections, we propose an approach to achieve this goal.

2.1. Estimating the Utility Function: A Bottom-Up Step

In this section, we discuss how to estimate a utility function for each RFQ. As mentioned earlier, this is the key step toward estimating the purchase probability for each RFQ and further computing the optimal

[3] A delegation price is a price that the firm expects to sell under certain market conditions.

price. Remember, in each RFQ, we have the following information:

- The bundle configuration $\mathbf{d}$.
- The attribute of the customer $\mathbf{z} \in \mathcal{Z}$.

As we said in the end of last section, the main challenge in designing a proper utility function lies in incorporating the two roles that the bundle configuration $\mathbf{d}$ can play in the utility function. To capture the two roles, we start with the following utility function:

$$\tilde{u}(\mathbf{d}, \mathbf{z}, p) = \alpha_0^{\mathbf{d},\mathbf{z}} + \alpha^{\mathbf{d},\mathbf{z}} h(\mathbf{d}, \mathbf{a}) - \beta^{\mathbf{d},\mathbf{z}} p, \tag{4}$$

where $h(\cdot,\cdot)$ is a real-valued function, capturing the "value" of the bundle, $\mathbf{a}$ is the attribute vector of all components, i.e., $\mathbf{a} = (\mathbf{a_1}, \ldots, \mathbf{a_I})$. Note that in (4), $\mathbf{d}$ plays the two roles as suggested in the end of last section:

1. It directly affects the customer's valuation of the bundle. This is through the term $h(\mathbf{d}, \mathbf{a})$. In particular, different composition of the bundle will lead to different value $h(\mathbf{d}, \mathbf{a})$ of the bundle, because of the different components included in a bundle.

2. It reveals the characteristics of the customer by affecting the coefficients in the utility function, including the price sensitivity coefficient. This is through the coefficient terms $\alpha_0^{\mathbf{d},\mathbf{z}}$, $\alpha^{\mathbf{d},\mathbf{z}}$ and $\beta^{\mathbf{d},\mathbf{z}}$.

Now the question seems to be how to choose a proper $h(\mathbf{d}, \mathbf{a})$ as well as to calculate the coefficients in (4). However, there is one hurdle that prevents us from applying such a model. In practice, the number of components a seller offers is often large, thus the number of possible bundle configurations is often huge, if not infinite. As a result, for most of the bundle configurations, there are either no historical data or the data are very scarce. Therefore, directly calibrating a model as in (4) is unrealistic and could lead to a very noisy result. To solve this issue, we need to apply some level of aggregation: Instead of using the bundle configuration vector $\mathbf{d}$ to capture the bundle characteristics, we choose some aggregate information regarding $\mathbf{d}$ to perform such tasks. In this way, we can maintain the two roles $\mathbf{d}$ plays in the utility function while keeping the model realistic and tractable.

In our approach, we define some aggregate bundle features, $\mathbf{b} = \{\mathrm{b}_1, \ldots, \mathrm{b}_\mathrm{L}\}$ and use these aggregate bundle features in the utility function. Obviously, these bundle features depend on the bundle configuration $\mathbf{d}$. In the meantime, these features also often depend on the attributes of the components included in the bundle. For example, one bundle feature could be the bundle total cost, or it could be the proportion (based on costs) of a certain family of components in the bundle. In either case, the bundle feature depends on both $\mathbf{d}$ and $\mathbf{a}$. Therefore, we write the aggregate features as $b_l = b_l(\mathbf{d}, \mathbf{a})$. And the utility function can be rewritten as follows:

$$u(\mathbf{b}, \mathbf{z}, p) = \alpha_0^{\mathbf{b},\mathbf{z}} + \sum_{\mathrm{l}=1}^{\mathbf{L}} \alpha_\mathrm{l}^{\mathbf{b},\mathbf{z}} \mathrm{b}_\mathrm{l}(\mathbf{d}, \mathbf{a}) - \beta^{\mathbf{b},\mathbf{z}} p. \tag{5}$$

In practice, choosing which bundle features to consider often requires deep thoughts, personal experiences, and numerical tests. We will give a concrete example in §3.

Finally, we want to calibrate the coefficients in the utility function (5). Even with the aggregate bundle features, it is still not realistic to have one set of coefficients for each combination of values of $b_1, \ldots, b_L$. Therefore, we choose to implement a segmentation model. That is, we first segment the RFQs using a hierarchical segmentation tree, in which each level of the tree is segmented by using either a customer information $\mathbf{z}$ or a bundle feature b_l. Then for each leaf in the segmentation tree, we have a unique set of coefficients thus a unique utility function. In practice, the hierarchical segmentation can be performed automatically by software, or manually based on experiences of a manager, or a combination thereof. A concrete example of such a segmentation model will be shown in §3.

To summarize, in the bottom-up step, we

- choose a set of bundle features $\mathbf{b} = \{\mathrm{b}_1, \ldots, \mathrm{b}_\mathrm{L}\}$ that are used to identify important bundle characteristics, and
- use $\mathbf{b}$ together with $\mathbf{z}$ to segment incoming RFQs according to bundle and customer characteristics by means of a hierarchical tree.

As a result, we will have a unique utility function $u(\mathbf{b}, \mathbf{z}, p)$ for each segment of customers. To estimate the coefficients in each of those unique utility functions, we can use historical data $\mathbf{d}, \mathbf{z}, p, \xi$ to perform a logistic regression. Then, when a new RFQ comes in, we can parse it to find the right segment it belongs to and use the corresponding utility function to calculate the utility and purchase probability at any given price.

2.2. Computing the Component Value Scores: A Top-Down Step

In the last subsection, we proposed a framework to estimate the utility function for each RFQ, which in principle could constitute, in itself, the main engine of our analysis and can be applied on its own. However, there is often an extra step that can be taken in practice to make the framework work even better. As one can see, one important step in the above procedure is to define some aggregate features for a bundle, which depend on the bundle configuration and the attributes of each component. However, sometimes, the information contained in the component attribute vector $\mathbf{a}$ may not provide sufficient information for directly calculating the bundle features. In particular, and often the case, the component attribute vector $\mathbf{a}$ contains rather raw information of a component, such as the cost, list price, release date, etc. Although this information is useful and forms the basis for estimating certain features of the component, it does not

provide a direct measure for perhaps the most important information about a component: how a customer values the component. Apparently, such a measure is very important in estimating the utility of a bundle that involves the component. In this section, we propose a method to compute such a measure, which we will refer to as the *value score* of the component. As we later show in empirical tests, computing the component value scores and using them in our estimation framework are essential to achieve a good performance.

At first glance, estimating the value of each component is not hard. An intuitive thought might be to define a value variable for each component and perform a regression using them as independent variables and the quoted prices in each historical data as dependent variables, i.e., each row of the regression corresponds to a requested bundle in the past data, with the quoted price being the dependent variable, and the component values being the independent variables. However, this method is infeasible in practice, because the data on each individual component may be very sparse: In many applications where the number of different components is large, it is likely that many components have not appeared or only appeared a few times in the data. If we perform the above regression, the result would be very noisy. Because of the above issue, we need to perform some level of aggregation. Instead of using the value of each component as independent variables, we compute a value score for each component as a function of its attributes, such as cost, list price, etc. However, for different types of components, the way a component's perceived value, thus its value score, depends on its attributes may be very different. We illustrate this point using an example of IT products:

- For most hardware components, the component cost provides a meaningful reference point for its value or at least should take a decent weight when estimating its value. Other attributes such as the list price may play a less dominant role.
- For software components, the variable cost is typically 0 or very close to 0. Therefore, the variable cost is not useful when assessing the component's value. Other attributes may play a more important role.

Even within the same product family, the perceived values of the components may be very different. For example, some newly released high-end components may have perceived values close to their list prices, which could be much higher than their costs; whereas some outdated low-end components may have perceived values close to their costs, which are well below their list prices. Therefore, finding a good way to calculate the component value scores presents a nontrivial task.

To handle this task, we adopt the *regression tree* model in statistical learning. In a regression tree model, a tree structure is built to classify the data (in our case, the components), and in each leaf of the tree, a regression model is constructed to predict a target value based on the attributes of the data. The idea of the regression tree was first proposed by Karalic (1992) and Quinlan (1992) and was further developed in many subsequent works. We refer to Loh (2011) for a recent review of this subject. In our approach, we adopt this method to estimate the value for each component and compute the value scores.[4] For the ease of notation, we call the regression tree in our context the *valuation-regression tree* (VRT). A sample VRT with different types of IT components is shown in Figure 1.

In Figure 1, each leaf corresponds to a class of components. In our VRT model, for each leaf, we use a regression model to obtain a fit for the component value scores as a function of its attributes **a**. Specifically, the component value score (denoted by v_i) in each leaf can be represented by

$$v_i = f(a_{i1}, \ldots, a_{iK}). \tag{6}$$

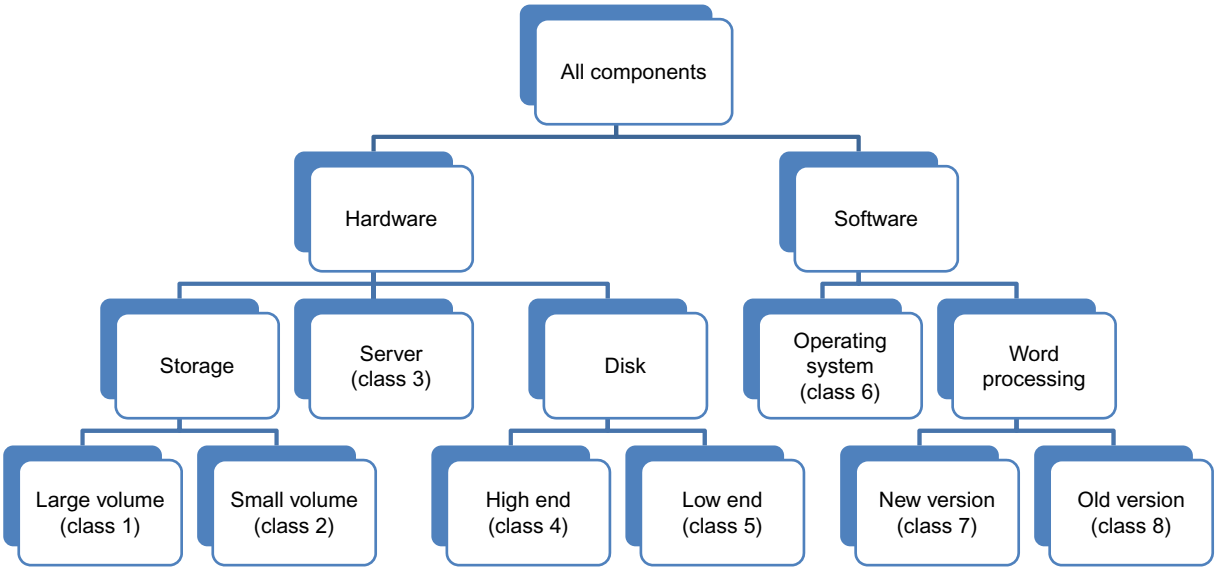
Here the function f often involves several coefficients that need to be determined through a regression analysis (such as through the least squares method) using data that belong to that leaf. Note that in such regression analysis, the independent variables are simply the attributes **a** of each component, however, the dependent variables v's are typically not well specified in the data set. For our regression model to work, we use the following ways to get the dependent variables:

- In the case that there are historical data for selling the single component, we use the sold price as the dependent variable.
- If there are not sufficient historical sales data for the single component, then we consider all the bundles containing that component and define some approximate value for each component. For example, one can decompose the sold price of bigger bundles into each component according to weights such as cost or delegation prices, and use decomposed component price as dependent variables. One concrete case of such an approach is exemplified in §3.1.2.

Note that in the above described procedure, it is possible that some component appears multiple times in the data, and the dependent variable is different for each case. In the corresponding regression model,

[4] It is worth noting that such regression trees have been used to construct consumer segmentations in the pricing literature. In those cases, the segmentation is based on customer attributes and the goal is to fit a response function. In our case, the segmentation is based on component attributes and the goal is to fit the component value scores.

Figure 1 **(Color online) Sample Valuation-Regression Tree**



we will have multiple rows that have the same independent variables but different dependent variables. Then the regression model can be viewed as to find a set of parameters that give the most "consistent" prediction for those historical data.[5]

Finally, in order to build the VRT, we implement a recursive procedure. In the following, we use $\bar{R}^2$ to denote the adjusted R^2 of a regression analysis. It is defined as $\bar{R}^2 = R^2 - (1-R^2)(\rho/(n-\rho-1))$, where n is the number of samples, ρ is the number of independent variables, and R^2 is the regular R-squared value of the regression. A review of $\bar{R}^2$ is referred to in Theil (1958). The procedure to build the VRT is given as follows:

1. Start from the entire set of components. Let $\mathcal{S} = \mathcal{I}$.

2. Run a regression (6) on $\mathcal{S}$ using the independent and dependent variables suggested in the above discussion. Compute the $\bar{R}^2$ of the regression model.

3. Divide $\mathcal{S}$ according to a classification of the components based on an unexploited attribute $k \in \{1, \ldots, K\}$ into several subclasses $\mathcal{S}_1, \ldots, \mathcal{S}_t$. Do regressions and compute the $\bar{R}^2$ for each subclass $\mathcal{S}_t$. Choose a classification (based on a certain attribute) that will result in the highest average $\bar{R}^2$ among the subclasses.

4. If the average $\bar{R}^2$ is higher than the $\bar{R}^2$ obtained at node $\mathcal{S}$ (i.e., the $\bar{R}^2$ obtained at step 2), then apply the classification $\mathcal{S}_1, \ldots, \mathcal{S}_t$ and go back to step 2 with $\mathcal{S}$ equals to each of the $\mathcal{S}_t$'s, respectively.

5. Otherwise, stop and output $\mathcal{S}$ as a leaf node with the regression model specified in step 2.

[5] This idea of using historical data to fit a more smooth prediction model is somewhat similar to the concept of expert systems. We refer the readers to Bowman (1963) for a review on this subject.

In this procedure, we use the adjusted R^2 instead of the plain R^2 because we want to compensate for the number of samples in each data set. It is well known that a small sample size may lead to the problem of overfitting, and the adjusted R^2 can balance the number of data in each subclass and the residual errors.

After the VRT is built, we can obtain a value score for each component: We first go down the tree according to its attribute. Then at the leaf level, we plug in the corresponding regression model. The resulting value will be the corresponding value score for that component. Finally, we note that this process can be done off line, therefore computational time is not a constraint for this step.

To summarize, the top-down step consists of building a tree to classify each component (this is in contrast to the tree built in the bottom-up step, which is used to segment the RFQs), and computes a value score for each component. Ultimately, these value scores will be used in the utility function in the bottom-up step together with the bundle configuration **d** and the component attributes **a**.

2.3. Implementing Our Approach

To implement our approach in practice, one first uses the historical data to establish the VRT, and computes the value score for each component (the top-down step). Then with the value scores, one constructs the bundle features and uses them to segment the incoming bundles and estimate a utility function for each segment (the bottom-up step). All of these can be done offline (with occasional updates). Finally, when an RFQ arrives in practice, one first determines the corresponding segment it belongs to, and then computes the optimal price in (3) using the corresponding utility function. In the next section, we use an

industrial application example to demonstrate how to apply the above steps in a concrete case.

3. An Industrial Application

In this section, we customize the general framework to a specific industrial application. The goal is to illustrate how each step in §2 can be carried out in a concrete case. Consider a major IT service provider who offers complicated systems composed of hardware and software. Among the products offered, each component either belongs to a server system or a storage system. The former offers large-scale computational capability for commercial and scientific solutions, and the latter installs and processes the data as a data hub. Thus, the component set $\mathcal{I}$ can be divided into two main product families: $\mathcal{R}$ (server components) and $\mathcal{T}$ (storage components). Furthermore, the server components could be further classified into $\mathcal{R}_1$ (high-end server), $\mathcal{R}_2$ (middle-end server), $\mathcal{R}_3$ (low-end server), etc. Similarly, the storage components could be further categorized by $\mathcal{T}_1$ (high-end storage), $\mathcal{T}_2$ (middle-end storage), $\mathcal{T}_3$ (low-end storage), etc. Here, the notations of $\mathcal{R}$s and $\mathcal{T}$s are just specific instances of the general notations of $\mathcal{I}_m$. We also use $\mathcal{H}$ and $\mathcal{S}$ to denote hardware and software, which are the two commodity types among the components. A typical structure of a bundle in this application is shown in Figure 2. Note that in this application, because of the large number of components offered, the configuration of each personalized bundle is typically unique.

Recall we use d_{ij} to denote the amount of component i in bundle j. We call a bundle a pure hardware one if $\sum_{i\in\mathcal{S}} d_{ij} = 0$ and a pure software one if $\sum_{i\in\mathcal{H}} d_{ij} = 0$. We further define the following parameters for each bundle:

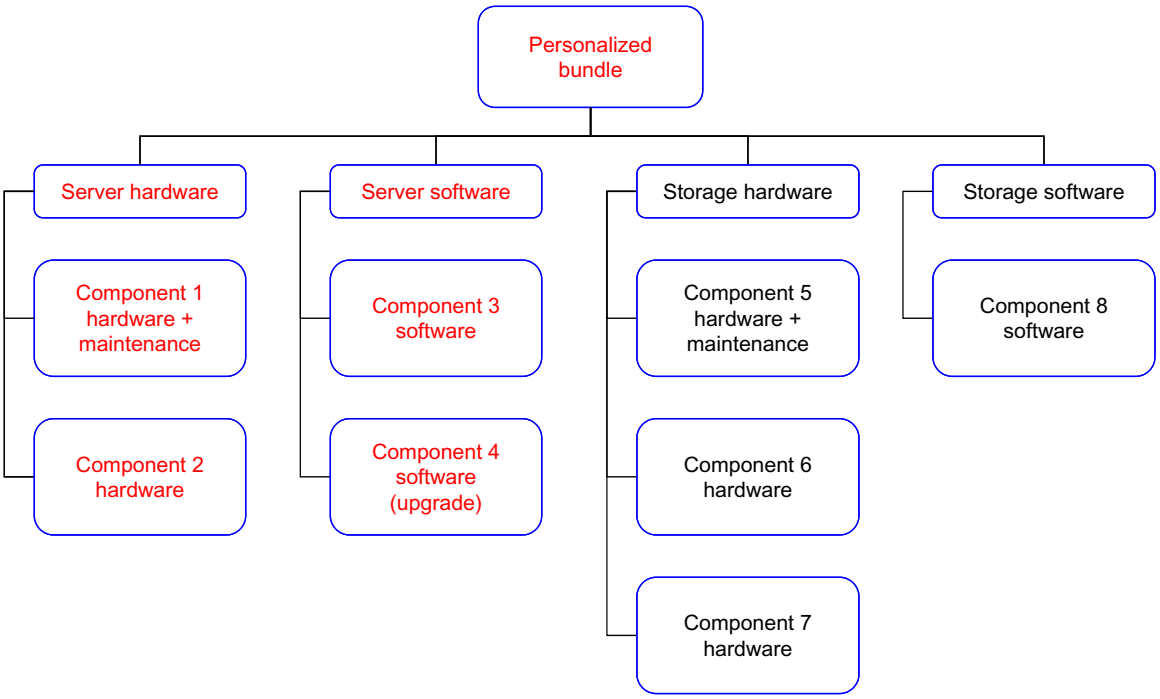
- The bundle cost $c_j = \sum_i d_{ij}c_i$. We have $c_j = c_{\mathcal{H}j} + c_{\mathcal{S}j}$, where $c_{\mathcal{H}j} = \sum_{i\in\mathcal{H}} d_{ij}c_i$ is the hardware cost, and $c_{\mathcal{S}j} = \sum_{i\in\mathcal{S}} d_{ij}c_i$ is the software cost. In this application, the software cost is zero.
- Bundle list price $\bar{p}_j = \sum_i d_{ij}\bar{p}_i$, with $\bar{p}_j = \bar{p}_{\mathcal{H}j} + \bar{p}_{\mathcal{S}j}$. Here $\bar{p}_{\mathcal{H}j}$ and $\bar{p}_{\mathcal{S}j}$ are the sum of list prices of the hardware and software components in the bundle, respectively.
- Bundle value score $v_j = v_{\mathcal{H}j} + v_{\mathcal{S}j}$, with $v_{\mathcal{H}j} = \sum_{i\in\mathcal{H}} d_{ij}v_i$ and $v_{\mathcal{S}j} = \sum_{i\in\mathcal{S}} d_{ij}v_i$. Here v_i is the value score of each component, which is going to be calibrated in the top-down step.

Our objective is to choose an optimal bundle price for each incoming RFQ. In the following, we analyze the component value scores (v_is) in §3.1 and then the utility model in §3.2.

3.1. Top-Down Step

In this section, we adopt the valuation-regression tree model introduced in §2.2 to compute the value score for each component. In the following, we first describe the main attributes of each component in this application, which are used to serve as the independent variables in the regression model. Then we address the issue of how to determine the dependent variables in the VRT model.

Figure 2 **(Color online) A Sample of Personalized Bundles**



3.1.1. Component Attributes. To obtain a good model for the component value scores, we first need to know what attributes a component has that may affect its perceived value. In this application, we consider the following attributes:

- Commodity type: software or hardware.
- Product family: group of components with similar functionality.
- Shelf life information: newly released, matured, upgraded items, obsolete.
- Market environment: monopoly, duopoly, full competition.
- Cost parameter: cost, delegation price, and list price.

In the following, we build a model for the component value score based on these attributes. We adapt the valuation model in §2.2 to the following:

$$v_i = f(c_i, \bar{p}_i, \hat{p}_i, t_i, \mathbf{1}_i). \tag{7}$$

In (7), c_i, $\bar{p}_i$ and $\hat{p}_i$ are the cost, the list price and the delegation price for the component at the time the RFQ is recorded. They are all specified in the data set for each component. The fourth term, $0 \le t_i \le 1$, is the age of component i at the time the RFQ is recorded, which can be calculated as the days after release divided by its entire shelf life. This term can also be viewed as accounting for the obsolescence of a component in its perceived value. The last term $\mathbf{1}_i$ is a binary variable indicating the market environment. In particular, it is equal to 1, if the provider is the market leader for component i, and 0 otherwise. In this application, a team in the sales department specifies an obsolescence score and a binary market environment variable in the data set to each component at the time of sale. Therefore, both t_i and $\mathbf{1}_i$ are immediately available from the data set. (In our application, we choose to include the obsolescence score t_i and the market condition $\mathbf{1}_i$ as independent variables in the regression model rather than using them to discount the prices.)

3.1.2. Computing the Component Value Scores. Now we apply the VRT model introduced in §2.2 to compute the value scores for each component. Remember that the VRT is a tree-type structure in which each leaf node corresponds to a certain regression model. To build the VRT, we run the procedures described in §2.2, i.e., performing regressions and classifications recursively until satisfactory regression models are found at each leaf node. Here the classification is done based on the attributes described in the last subsection. At each leaf node, we use the cost parameters (cost, list price, delegation price), the age parameter, and the market environment factor as the independent variables for the regression model. In the following, we discuss the choice of the dependent variables in the regression model and some implementation issues in the construction of the VRT in this specific application.

To find reasonable dependent variables for the regression model used to build the VRT, we need to estimate the perceived value of each component from the historical data. To this end, we utilize the sold prices in the historical RFQs in this application. Although those prices were not necessarily optimal, they reflect to some degree the customer's valuation of the bundle, which is consistent with the intended meaning of the component value score. For the sales data for bundle j, we have the information of the sold price p_j. We then break it down to obtain an estimation of the valuation of each component. To break it down, we use the delegation price as the weight, as the delegation price reflects the seller's knowledge on the perceived market value of each component. Therefore, the dependent variables v_i^* of component i for historical data j can be written as

$$v_i^* = \hat{w}_i p_j, \text{ with } \hat{w}_i = \hat{p}_i / \hat{p}_j,$$

where $\hat{p}_j$ is the sum of the delegation prices of all components in the bundle j. (As we have mentioned earlier, it is common that there are multiple data that correspond to one component. In that case, we include all of them in our regression analysis, which then computes the regression model that gives the most consistent component value scores.)

With the dependent variables specified, we can build the VRT. When building the VRT, we need to choose a regression model at each leaf node. A natural choice would be a linear model, which can be written as

$$v_i = \theta_0 + \theta_1 c_i + \theta_2 \bar{p}_i + \theta_3 \hat{p}_i + \theta_4 t_i + \theta_5 \mathbf{1}_i + \epsilon_i, \tag{8}$$

where the random variable ϵ_i describes the random errors. However, the issue of *multicollinearity* (Chatterjee et al. 2000) will arise in the linear regression model since there are linear correlations among the independent variables, i.e., the cost, list price, and delegation price. Therefore, special cautions should be applied if one chooses to use the linear regression model.

To mitigate the effect of multicollinearity, we can alternatively consider a power regression model:

$$\log(v_i) = \theta_0 + \theta_1 \log(c_i) + \theta_2 \log(\bar{p}_i) + \theta_3 \log(\hat{p}_i) + \theta_4 t_i + \theta_5 \mathbf{1}_i + \epsilon_i. \tag{9}$$

In an empirical study using available data (the detailed background and scope of this study is specified in §4), we find that the power regression model fits the data very well. The resulting goodness of fit is shown in Table 1, which shows the average adjusted

Table 1 Adjusted R^2 for Component Value Scores

Family	$\bar{R}^2$ (No VRT)	$\bar{R}^2$ (VRT)	Family	$\bar{R}^2$ (No VRT)	$\bar{R}^2$ (VRT)
Server SW	0.938	0.953	Storage SW	0.989	0.990
Server HW			Storage HW		
—High end	0.904	0.917	—High end	0.941	0.946
—Middle end	0.934	0.943	—Middle end	0.880	0.915
—Low end	0.942	0.961	—Low end	0.990	0.992

R^2s for the regression models.[6] In particular, the left column of Table 1 shows the adjusted R^2 if we just run a regression for each product family, and the right column of Table 1 shows the average adjusted R^2 for each family when we further classify them using the valuation-regression tree method. We can see that indeed, by using the valuation-regression tree, the goodness of fit can be improved. In particular, all the product families have a decent goodness of fit. Therefore we believe this approach yields a decent assessment for the value of each component.

3.2. Bottom-Up Step

Next we apply the bottom-up step to establish the utility function based on the component value scores computed in §3.1. First, we propose several features to characterize and segment the bundles. Then we define utility models based on the segment and bundle features.

3.2.1. Bundle Features. As discussed in §2.1, to keep the model tractable, we choose some aggregate features b_l to characterize a bundle instead of using the entire configuration vector $\mathbf{d}$. In this application, through interviewing experienced pricing personnel and conducting tests, we find that the following features are of particular importance:

- $\mathbf{w}_c = \{w_{\mathscr{H}}, w_{\mathscr{S}}\}$, with the weight $w_{\mathscr{H}} = \sum_{i\in\mathscr{H}} d_{ij}v_i/(\sum_i d_{ij}v_i)$ and $w_{\mathscr{S}} = \sum_{i\in\mathscr{S}} d_{ij}v_i/(\sum_i d_{ij}v_i)$ indicating the weight (in terms of component value scores) of hardware and software in this bundle.
- $\mathbf{w}_f = \{w_1, \ldots, w_M\}$, with the weight $w_m = \sum_{i\in\mathscr{F}_m} d_{ij}v_i/(\sum_i d_{ij}v_i)$ indicating the weight of product family m.
- $m^* = \arg\max_{m=1,\ldots,M} w_m$, which represents the leading family in the bundle.
- $\mathbf{r} = \{r_c, r_p\} = \{\sum_i d_{ij}v_i/c_j, \sum_i d_{ij}v_i/\bar{p}_j\}$, which indicates the overall grade of the bundle.

Here $\mathbf{w}_c$, $\mathbf{w}_f$, m^* and $\mathbf{r}$ can all be viewed as some terms among the b_l's. Among them, $\mathbf{w}_c$, $\mathbf{w}_f$, and m^* are features about the composition of the bundle. For example, a bundle is hardware based if $w_{\mathscr{H}}$ is close to 1; it is software based, if $w_{\mathscr{H}}$ is close to 0. These features are fundamental for a bundle and should be one of the first factors to consider when building the segmentation model.

Besides the compositional information, another important feature of a bundle is its grade, which is characterized by $\mathbf{r}$. In our definition, $\mathbf{r}$ has two terms. If the bundle is hardware based, the first term will be the main factor, and the overall grade of this bundle is given by $\sum_i d_{ij}v_i/c_j$. Recall that c_j is the total cost of this bundle and $\sum_i d_{ij}v_i$ is the bundle value score. Therefore, the ratio $\sum_i d_{ij}v_i/c_j$ is larger if the bundle contains more high-end hardware since high-end hardware usually has a higher margin. For software-based bundles, $c_j = 0$, and thus the first term is meaningless. Instead, the overall grade of the bundle is measured by $\sum_i d_{ij}v_i/\bar{p}_j$. It is higher when the bundle contains more high-end software since high-end software typically has less discount from the list price.

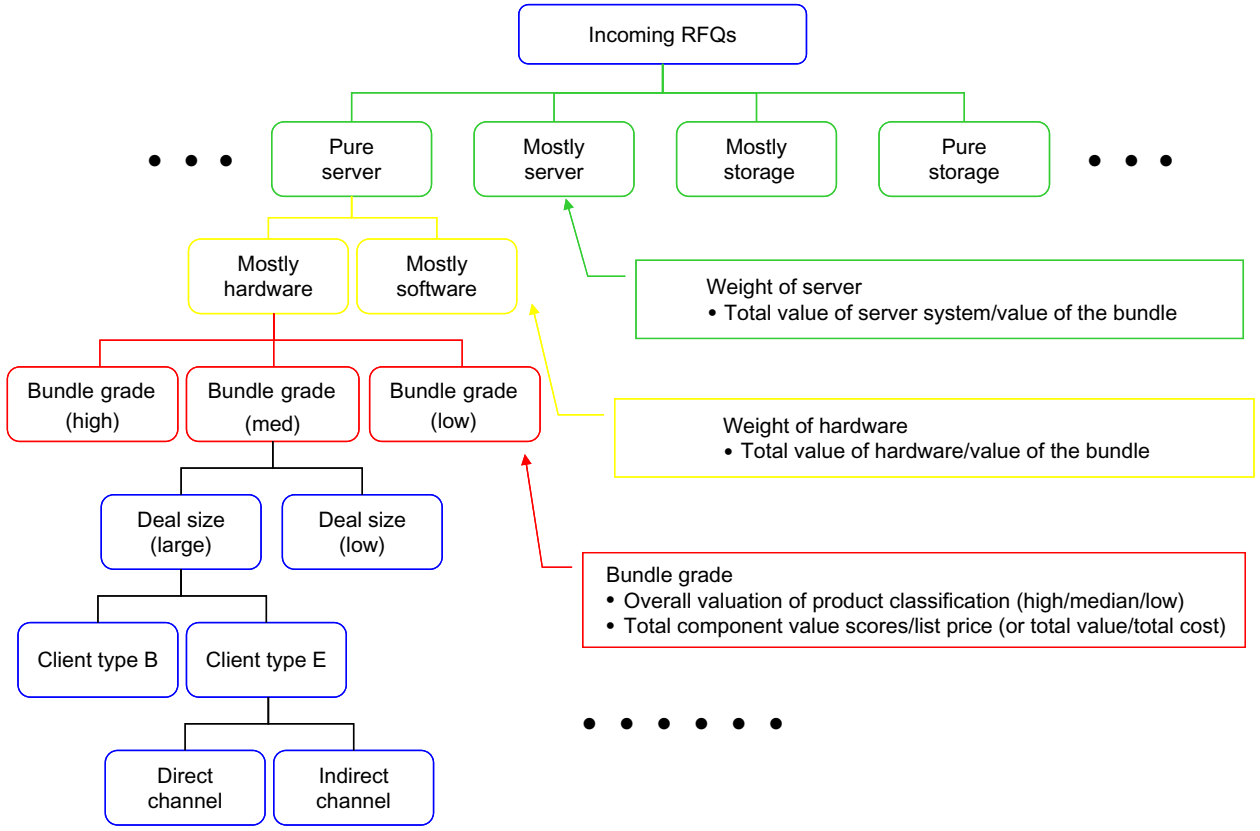
Now we have defined some main features for a bundle. In the next subsection, we segment the incoming RFQs using these features together with the customer attribute information. Then we derive the utility functions in §3.2.3.

3.2.2. Segmentation Model. Now we build a segmentation model for incoming RFQs based on the bundle features and the customer attributes. By consulting experienced salespeople, we know that in this application, some attributes are more important than others. (In the following paragraphs, we will discuss why this is preferred to having the segmentation entirely automated by software.) Thus we propose a hierarchical segmentation model, in which all the attributes (both the bundle ones and the customer ones) are put in a hierarchical order to capture the various features of each RFQ.

In practice, the leading family (m^*) is the first attribute to consider for an RFQ, because it shows the main product type in the bundle: different product types usually have very different price sensitivities. Second, we consider the weights of hardware and software in the bundle ($\mathbf{w}_c$). Typically hardware and software have very different pricing methods, thus it is important to make this distinction. Third, we consider the grade of the bundle ($\mathbf{r}$). Bundles with higher (lower, respectively) grades usually have a smaller (larger, respectively) price sensitivity and therefore it is necessary to have different utility models for them.

[6] Because of confidentiality concerns, Table 1 only shows a portion of product families.

Figure 3 **(Color online) Hierarchical Segmentation of Personalized Bundles**



Besides these factors, another common segmentation criterion is the deal size. Normally, a customer looks for a discount based on the quantity. The four factors addressed above are mainly about the bundle features. Moreover, the segmentation model also considers the customer attributes, such as acquisition or retention, company or government, indirect or direct channel, etc. Figure 3 summarizes a sample hierarchical model in this application.

In each level of the hierarchy, we use some established software for the segmentation. In our implementation, we use the "party" software for that purpose. Party is a software package that can be found in the library of R, which is one of the most popular software for statistics and data mining. In this package, a recursive binary partitioning method *c-tree* is used to perform the partitioning. In *c-tree*, a regression model is used to describe the conditional distribution of a response variable Y given the independent variables X by means of tree-structured recursive partitioning. Details of this method are referred to in Hothorn et al. (2006).

One may wonder whether we can directly use the *c-tree* to partition the entire data set without doing the hierarchial segmentation first. There are two main disadvantages of doing that. First, the segmentation results do not appear to be robust. By testing different sets of data, we find quite different segmentation results, even though the market condition does not seem to differ by much. Such results may be confusing and people will question the rationality of it. Second, it is difficult to control the number of data in each segment. We observe some segment might have less than 10 data, while some others might have more than 400 data, which is undesirable in practice.

Therefore, in our application, we choose the order of the hierarchical model before we apply the *c-tree* to find the proper split in each level. Using this method, it is also easier to set up the stopping rule for the segmentation. Typically, if we find the number of historical quotes in a segment is below 50 after considering the top three attributes, then we just stop further segmentation and ignore the other attributes lower in the hierarchy.

After the hierarchical segmentation, we are able to classify the bundles into N segments: $\mathcal{J}_1, \ldots, \mathcal{J}_N$. In each segment, we will establish a unique utility model as a function of bundle attributes.

3.2.3. Utility Model. After the segmentation model is built, we construct a utility function for the RFQs in each segment. In the following discussion, we fix a segment $\mathcal{J}_n$. We construct the following utility

function: (the coefficients αs and βs will be different if different $\mathcal{F}_n$ is considered)

$$u_j = \alpha_c \frac{c_j}{v_j} + \alpha_l \frac{\bar{p}_j}{v_j} - \beta \frac{p_j}{v_j} + \alpha_h + (\alpha_s - \alpha_h) \frac{v_{\mathcal{F}j}}{v_j} + \varepsilon_j. \quad (10)$$

Note that in (10), each term has an explicit meaning. The term c_j/v_j rates the grade of a hardware-based bundle (it is the reciprocal of r_c). As we have discussed earlier, a high-end bundle tends to have a higher r_c thus a lower c_j/v_j, therefore we expect α_c to be negative. The term $\bar{p}_j/v_j$ shows the grade of a software-based bundle (it is the reciprocal of r_p). Again, if a bundle is of high grade, then this term will be smaller. Hence, we expect α_l to be negative as well. The term $(\alpha_s - \alpha_h)v_{\mathcal{F}j}/v_j$ shows the effect of having more weight of software attached to the bundle on the overall utility. For example, if $(\alpha_s - \alpha_h) > 0$, then it means that a higher weight of software increases the utility of the bundle, and vice versa. Note that (10) is a special case of the general utility model (5) with each b_l belonging to the features defined in §3.2.1.

To calibrate the coefficients in the utility model (10), we use the logistic regression. In the logistic regression, the independent variables are the bundle configuration, customer type, and quoted prices in the historical data, and the dependent variables are whether the customer bought the bundle or not. The regression can be easily carried out with standard software and the detail is omitted.

4. Business Impact

In this section, we use an empirical study to show the strength of our model. First, we introduce some background of this empirical study. Then we provide numerical results to illustrate the benefit of implementing our pricing solution and discuss the business impact.

4.1. Project Background

Our empirical study is based on a pricing project (we call it the *smart pricing project* and the corresponding pricing solution the *smart pricing tool* in the following discussions) that was launched by a major IT service provider in 2011. The project aimed to develop a real-time support system for the firm's pricing decisions when selling personalized bundles. Before this project was launched, the personalized bundle pricing decisions were mainly determined by personal experience of the sales personnel. That is, for each customer's request, a salesperson reviews the configuration of the bundle along with its cost, list price, delegation price, etc., and then offers a quoted price based on his insight of the market.

The system based on human intelligence was in place for many years. However, several disadvantages emerged during the years. First, it is difficult to share a salesperson's knowledge and experience in a systematic fashion. Second, it is difficult to use all the available information at the time of a quote request. Third, different salespeople often have different opinions toward the same bundle, and the resulting price could be dependent on the particular salesperson that is in charge, resulting in potential inconsistencies in the pricing decisions. Overall, these disadvantages emphasized the need for a new pricing tool that is automated and can utilize all available information in real time.

In §§4.2 and 4.3, we show empirical results that are based on a proof of concept (POC) phase for this project. In particular, we show the test performance of our proposed approach and compare it to a simpler approach that only uses bid-level information. Then in §4.4, we show the results from real implementation of our pricing solution. In the POC, we focus on two brands: a server computer brand and a storage solution brand. The components provided by this company in these two brands were introduced in §3. The data set used in the POC contains 18 months sales data from Q1 of 2012 to Q2 of 2013 with 3,253 RFQs in total. In the POC, we conduct *out-of-sample tests*. More precisely, we pick a 12-month subset of the sales data to train the model, and then apply the fitted model to the RFQs in the remaining 6-month data. In particular, we conducted three tests: (1) use 2012 Q1–Q4 data for training and 2013 Q1–Q2 data for testing, (2) use 2012 Q1–Q2 and 2013 Q1–Q2 data for training and 2012 Q3–Q4 for testing, and (3) use 2012 Q3–Q4 and 2013 Q1–Q2 data for training and 2012 Q1–Q2 data for testing. Although this is not an extensive random out-of-sample test, the results are quite consistent. In addition, people tend to use data from consecutive periods for training in practice. Therefore, the results are representative for the performance of our approach.

4.2. POC Business Value

In this section, we show the results of the POC tests. The setup of the test has been described in the end of last section. To better illustrate the result, we will focus on test 1 in this section, that is, we use the data from Q1–Q4 of 2012 as training data to calibrate the pricing model (which we will call *period 1 data*) and then test the pricing model using the data from Q1–Q2 of 2013 (which we will call *period 2 data*). The results for the other two tests are given in the appendix (see Tables A.1–A.4).

To build our pricing model, we use the period 1 data to calibrate a utility function for each segment of RFQs. In particular, we first apply the top-down step in §3.1 to obtain a value score for each component, then we apply the bottom-up step in §3.2 to construct a utility function (10) for each segment of

Table 2 A Sample Set of Coefficients

Family	α_h	β	$\alpha_s - \alpha_h$	α_l	α_c
High-end server	1.359	−1.596	2.102	−0.767	0.000
Middle-end server	4.577	−2.066	5.518	−5.518	0.000
Low-end server	0.487	−2.769	1.198	−1.387	0.000
High-end storage	5.218	−3.089	6.547	−0.501	0.000
Middle-end storage	10.004	−4.525	7.370	−0.471	0.000
Low-end storage	0.000	−7.431	1.736	−0.719	0.000

the RFQs. We show in Table 2 a sample set of coefficients obtained from one specific segment. Although it is only a small set of coefficients obtained for a specific segment, we highlight some observations about the regression coefficients. And the signs of the coefficients in Table 2 are representative of the other segments.

The zero terms in Table 2 mean that the corresponding coefficients are statistically insignificant. In our regression analysis, we adopt a backward elimination method, thus insignificant coefficients are removed and a new regression only using the remaining independent variables are performed. We first focus on the coefficients β, which can be viewed as the price sensitivity coefficient for the corresponding segment of bundles. In Table 2, we observe that the price sensitivity coefficient of bundles with high-end servers/storage systems is less than (in absolute value) that of the bundles with low-end ones. This implies that the customers who look for low-end systems are more price sensitive than those who are interested in high-end ones, which is consistent with our intuition. Second, the coefficients for the weight of software ($\alpha_s - \alpha_h$) have positive signs, which means that the weight of software in a bundle positively affects the bundle utility. Third, the coefficients α_l are negative, which means a bundle with a higher bundle value score (i.e., higher $v_j/\bar{p}_j$) has a higher utility. Also, in this segment, the coefficients α_c are all insignificant, this is because the selected segment is for bundles with a significant amount of software, thus the cost to value ratio terms are relatively small. Lastly, we observe that the coefficients are very different in different segments, which indicates that customers do have different price sensitivities when buying different bundles. Therefore, it is necessary to segment the incoming RFQs based on the bundle configuration, as we did in our approach.

Now we have our utility model calibrated from period 1 data. We apply it to the period 2 data to assess its performance. To do that, we need to define a way to evaluate the performance, for which we consider what would have happened if we used our pricing strategy (the *optimal price*) in the test data and compare it to the outcome of the actual price (the *approved price*). In the following, we propose an assessment method, which we call the *conditional probability approach*. In the conditional probability approach, we consider four scenarios, which are shown in Figure 4.

In Figure 4, the two columns correspond to two cases: (1) the optimal price is higher than the approved price and (2) the optimal price is lower than the approved price. The two rows correspond to whether the outcome was a *win* (i.e., the customer purchased the bundle at the approved price) or a *loss* (i.e., the customer did not purchase at the approved price). We study each of the four scenarios in Figure 4 as follows:

1. The optimal price p^* is higher than the approved price p and the outcome was a win. In this case, if we had used the optimal price, there is some chance that the customer would not make the purchase since the optimal price is higher. To estimate this probability, we use the win probability model we established using the training data set. Let $q(p)$ be the logit function defined in (2) when p is used. Then the probability that the customer would buy at the optimal price p^* given that she bought at price p can be estimated by the following conditional probability:

$$\frac{q(p^*)}{q(p)}.$$

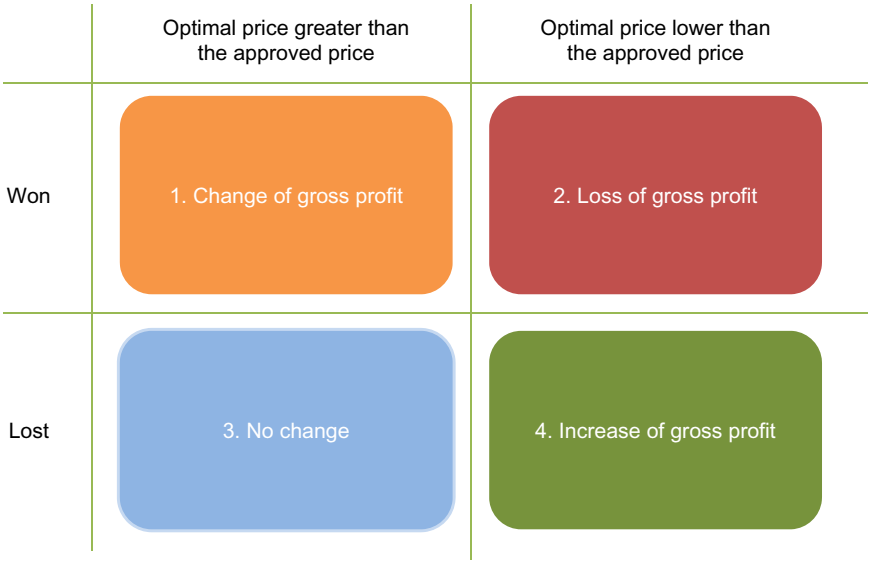
When the bundle is sold at the optimal price (approved price, resp.), the profit is $p^* - c$ ($p - c$, resp.). Therefore, the expected change of revenue (profit, resp.) would be $\Delta G_1 = p^*(q(p^*)/q(p)) - p((p^* - c) \cdot (q(p^*)/q(p)) - (p - c)$, resp.). Note that there are two countervailing forces in this case, i.e., the reduced chance of making a sale and the increased revenue/profit of each sale. In general, either force could be dominant, i.e., ΔG_1 could be either positive or negative.

2. The optimal price p^* is less than the approved price p and the outcome was a win. In this case, the optimal price would certainly be accepted by the customer too and the net change of revenue (and profit) is $\Delta G_2 = p^* - p$. Note that ΔG_2 is always negative since the bundle would be sold at a lower price.

3. The optimal price p^* is higher than the approved price p and the outcome was a loss. In this case, the optimal price would certainly result in a loss too. And the net change of profit ΔG_3 is zero.

4. The optimal price p^* is less than the approved price p and the outcome was a loss. When the optimal price is lower, there is some chance that the customer would purchase at the lower price. Similarly, we use a conditional probability formula based on the win probability function calibrated by our regression model to estimate the conditional win probability of the optimal price. In this case, the conditional probability that a customer would purchase at price p^*

Figure 4 **(Color online) Four Different Scenarios in Business Value Assessment**



given that she did not purchase at price p can be estimated by

$$\frac{q(p^*) - q(p)}{1 - q(p)},$$

and the expected gain of revenue (profit, respectively) is $\Delta G_4 = p^*((q(p^*) - q(p))/(1 - q(p)))$ $((p^* - c)((q(p^*) - q(p))/(1 - q(p))))$, resp.).

One criticism people may have is that we have used our own calibrated win probability function to compute the conditional win probability when the optimal price is used. Of course, this is not a perfect approach. However, as one can imagine, no method can perfectly predict what a customer would have done if offered another price in the past, and the win probability model that we estimated has already utilized all the available data to estimate the customer purchase probability. Thus, it is a reasonable approximation. Furthermore, remember that we are studying the performance using an *out-of-sample* approach, i.e., we learn the coefficients of our model using period 1 data and test the performance of our model using period 2 data. Therefore, even if we use the win probability function we calibrated to compute the conditional probability, there is *no guarantee* that the optimal price must perform better. In other words, the comparison between the revenue achieved by the optimal prices and the approved prices should mainly reflect the relative advantages of the pricing strategies, rather than the way we chose to evaluate them. The test results are shown in Table 3.[7]

[7] For confidentiality reasons, the numbers (both the revenue and the profit) in Table 3 are disguised by multiplying a certain constant. However, all the relative relationships of the numbers are maintained (i.e., the % change stays the same). The same comment applies to the later tables too.

In Table 3, each block of rows shows the (expected) revenue and profit earned by the approved prices and the optimal prices, respectively, under each of the four scenarios discussed above (we did not include scenario 3 since we have argued that everything is 0 in that case). The absolute changes as well as the relative changes (in parentheses) are given in the last column. The second from the last set of rows sums up the values in different scenarios and shows the overall performance of our method. The last row shows the gross profit (GP) margin of each pricing mechanism, defined as the profit divided by the revenue. We can observe the following from Table 3:

1. In the first scenario (i.e., win and the optimal price is greater than the proved price), by pricing higher, we overall sell slightly less because of the reduction of purchase probability. Meanwhile, in the optimal pricing, we increase the prices of those bundles and achieve a higher margin if the bundle is sold. From the perspective of the expected profit, we are able to obtain higher gross profits in this scenario.

2. In the second scenario (i.e., win and the optimal price is lower than the approved price), as we have discussed, the optimal pricing method will lose revenue/profit because of the lower price.

3. In the last scenario (i.e., loss and the optimal price is lower than the approved price), the optimal prices recapture some of the lost sales. Our model predicts that reducing the prices in exchange for a higher purchase probability would overall increase the profit, which is signified in the results.

4. Overall, our pricing method achieves a higher revenue at a reduced cost, which results in a significant improvement in the profit. We also observe that the average profit margin of the optimal prices is higher.

Table 3 Financial Assessment for Test 1 (Conditional Probability Approach)

	Actual pricing	Optimal pricing	Change
Scenario 1: win and opt > actual			
Revenue ($)	55.3 M	54.7 M	−0.5 M (−1.0%)
Profit ($)	23.8 M	26.9 M	3.1 M (13.2%)
Scenario 2: win and opt < actual			
Revenue ($)	27.1 M	22.9 M	−4.2 M (−15.6%)
Profit ($)	14.2 M	11.4 M	−2.8 M (−19.7%)
Scenario 4: loss and opt < actual			
Revenue ($)	0.0 M	8.5 M	8.5 M (N/A)
Profit ($)	0.0 M	4.0 M	4.0 M (N/A)
Total			
Revenue ($)	82.3 M	86.1 M	3.8 M (4.6%)
Profit ($)	38.0 M	42.3 M	4.3 M (12.1%)
Overall GP margin (%)	46.1	49.1	3.0

In addition to the above assessment method, we also compare the expected revenue and profit of the optimal prices with the expected profit of the approved prices (under our calibrated model) and the actual revenue. The results are shown in Table 4. As one can see, the expected revenue/profit of the optimal prices also performs well under these measures.

In the appendix, we present the results for the other two tests (the other two choices of hold-out data). Although there are some fluctuations in the results, the revenue/profit improvement in general are quite consistent with the results in Tables 3 and 4.

4.3. Comparison to Other Approaches

In this section, we further validate the value of the smart pricing tool by comparing it to another approach that does not use the component value scores described in our approach. The goal is to demonstrate the benefits of involving this step. In the approach we compare to (which we will refer to as the bid-level approach), we only use aggregate measures for each bundle to construct the utility functions as well as to perform segmentations. More precisely, in the bid-level approach, for each bundle j, one first computes the aggregate attributes of the bundle. To make the factors consistent with those considered in our proposed utility (10), we consider the following bundle attributes: the total cost c_j, the total list price $\bar{p}_j$, the total software delegation prices $\hat{p}_{S_j}$, and the total delegation prices $\hat{p}_j$. Since we no longer have the value scores available, we replace the bundle value score in (10) by the total delegation price (we could also use the total cost or the total list price, the results will be very similar). That is, the bundle utility in the bid-level approach is

$$u_j = \tilde{\alpha}_c \frac{c_j}{\hat{p}_j} + \tilde{\alpha}_l \frac{\bar{p}_j}{\hat{p}_j} - \tilde{\beta} \frac{p_j}{\hat{p}_j} + \tilde{\alpha}_h + (\tilde{\alpha}_s - \tilde{\alpha}_h) \frac{\hat{p}_{S_j}}{\hat{p}_j} + \tilde{\epsilon}_j. \quad (11)$$

In the bid-level approach, we still segment the incoming bundles in the same way as we did in §3.2.2, except that we are no longer able to segment the bundles by the bundle value scores, which depend on the component value scores computed in the top-down step. Instead, we replace the bundle value score by the total delegation price of the bundle. We consider the same data set as we used in §4. In the following, we show the results when using the data from Q1–Q4 of 2012 to calibrate the model, i.e., to create the segmentation of bundles as well as to calibrate the coefficients for each segment, and using the data from Q1-Q2 of 2013 to evaluate the performance. We also test the cases when using the other two half years' data as training data and applying to the remaining half year (see §4.1 for the setup), the results (especially the magnitude of the advantage of the smart pricing tool) are very similar.

First we compare how well the smart pricing tool and the bid-level approach can predict whether customers will purchase using the period 2 data. We find that the average prediction rate (the average of the prediction probability for the true outcome) for the smart pricing tool is 67%, and for the bid-level approach the rate is at 64%. Therefore, the smart pricing tool predicts the purchase results better, which means that by doing the top-down and bottom-up steps and utilizing the component value scores for both bid segmentations and utility constructions, it could fit the data better than an approach that only uses bid-level information.

Next we compare the expected revenue and profit of the two methods. We adopt the same conditional probability method introduced in §4.2 and the results are shown in Table 5. Note that in Table 5, the numbers are computed assuming the true model is the one

Table 4 Financial Assessment for Test 1 (Expected Values)

	Expected (opt. price) ($)	Actual ($)	Expected (appr. price) ($)
Revenue (M)	96.0	82.3	95.9
Profit (M)	44.0	37.9	39.1

Table 5 Results of the Bid-level Approach (Conditional Probability Approach)

	Actual pricing	Bid-level approach	Change
Scenario 1: win and opt > Actual			
Revenue ($)	63.3 M	62.3 M	−1.0 M (−1.6%)
Profit ($)	28.2 M	30.7 M	2.5 M (8.9%)
Scenario 2: win and opt < Actual			
Revenue ($)	19.1 M	16.2 M	−2.9 M (−15.2%)
Profit ($)	10.1 M	8.1 M	−2.0 M (−19.8%)
Scenario 4: loss and opt < Actual			
Revenue ($)	0.0 M	5.1 M	5.1 M (N/A)
Profit ($)	0.0 M	3.9 M	3.9 M (N/A)
Total			
Revenue ($)	82.3 M	83.7 M	1.4 M (1.7%)
Profit ($)	38.3 M	41.4 M	3.1 M (8.1%)
Overall GP margin (%)	46.5	49.5	3.0

Table 6 Comparison Between Our Approach and the Bid-Level Approach

Total	Optimal pricing	Bid-level pricing	Change
Revenue ($)	86.1 M	81.3 M	−4.3 M (−5.0%)
Profit ($)	42.3 M	39.8 M	−2.5 M (−6.0%)

calibrated by the bid-level approach, therefore, one can view that we have given the bid-level approach the "benefit of doubts."

In Table 5, we can see that the performance of the bid-level approach is reasonable. It also increases the revenue and profit by a decent amount. However, the increments are less than that when we use our approach, even if we have used the bid-level model as the underlying model. In Table 6, we further test the bid-level approach using our pricing model as the true demand model. Since we have shown that our model has a higher prediction rate, this comparison is reasonable. Apparently, the result indicates that it is indeed valuable to use the top-down and bottom-up steps. Overall, we observe that by involving those steps, we can substantially improve both the prediction rate and the test results. In the next section, we will further demonstrate the strength of the smart pricing tool by showing the results from real-world implementations.

4.4. Real-World Implementations

After the success of the POC, we developed a Web-based pricing tool that implements the smart pricing method, together with advanced programming interfaces that enable automated RFQ processing and price quoting in real time. Before implementing the pricing tool for full production use, we assess the business impact in a controlled experiment for two selected sales markets based on actual RFQ data.

The experiment was conducted for a period of six months in 2013 in two European sales regions. During that period, there were 774 and 620 RFQs placed to the IT company in regions 1 and 2, respectively. For each incoming RFQ, we randomly select it with probability 30% to be priced by the smart pricing tool. (As an outcome, 237 (30.62%) RFQs in region 1 and 192 (30.96%) RFQs in region 2 were selected.) For those RFQs that were not selected, we used the existing baseline pricing method to price it. The model calibration for the smart pricing tool for each region is done using the one year's data prior to the experiment. During the experiment period, we record the prices used and the outcomes for each RFQ. At the end of the test period, we calculate the resulting revenue and profit in each group. The results are summarized in Table 7.

First, we can see from the bottom lines of Table 7 that in both sales regions, the smart pricing tool clearly leads to higher revenue/profit per RFQ. Combining the two regions, we observe that the smart pricing tool is able to achieve a 30% per quote revenue increase and a 25% per quote profit increase. This is an immediate indication that our pricing tool works very well in this test. Next we take a closer look at the results and explain why the smart pricing tool did well.

In Table 7, we can see that the win rate in region 1 was significantly lower than that in region 2, regardless of the pricing method used. This could be interpreted as the customers in region 1 overall had lower valuations of the products and/or higher price sensitivities. In response to that, the smart pricing tool recommended a lower GP% for the RFQs in that region. Even though the quoted price was lower, the conversion rate in region 1 was greatly improved, which compensated for the lower price and eventually led to increases of the per quote revenue and per quote profit. In contrast, in region 2, the win rate was relatively high. Then the smart pricing tool recommended an averagely higher GP%. In particular, the smart pricing tool was able to identify the attractiveness of each bundle based on the historical market responses, therefore, even with a higher average margin, it can still achieve higher win and profit

Table 7 Results from Real Implementation

	Sales region 1		Sales region 2	
	Smart pricing	Baseline pricing	Smart pricing	Baseline pricing
Mean GP% (%)	31.5	39.7	42.1	38.2
Med. GP% (%)	28.9	38.8	44.4	39.7
Std. dev. GP% (%)	13.8	13.9	14.3	19.2
Quote no.	237	537	192	428
Ship no.	24	32	56	62
Win rate (%)	10.1	6.0	29.2	14.5
Quote revenue ($)	34.40 M	99.12 M	36.54 M	107.85 M
Ship revenue ($)	5.86 M	10.98 M	11.22 M	17.91 M
Revenue conversion rate (%)	17.0	11.1	30.7	16.6
Quote profit ($)	13.40 M	45.62 M	17.50 M	44.36 M
Ship profit ($)	2.69 M	5.43 M	5.31 M	8.88 M
Profit conversion rate (%)	20.1	11.9	30.3	20.0
Revenue per 100 RFQ ($)	2.47 M	2.04 M	5.84 M	4.18 M
Profit per 100 RFQ ($)	1.14 M	1.01 M	2.80 M	2.07 M

conversion rates, as well as significantly higher revenue and profit per RFQ.

From the implementation result, it is clear that the smart pricing tool successfully improved the business of the IT service provider. Several million dollars of gross profit gains are obtained as an outcome. Based on these results, the brand sales executives have decided to deploy the smart pricing solution to all European sales markets at the time this paper was written.

5. Conclusion

This paper studied the pricing strategy for personalized bundles with distinctive configurations. In this problem, most bundles have unique configurations that were never seen before. The seller lacks knowledge about the distribution of component reservation prices and the correlation among components. These features make it difficult to use the traditional bundling models based on the complete information of reservation price and component correlation.

To address these difficulties, we proposed a novel top-down and bottom-up approach based on the historical sales data to segment complicated bundles, calibrate their price sensitivities, and compute the optimal prices. In our proposed approach, we first decompose all the bundles to components, classify them into different families based on their similarity, and assess their value scores. Next, we reassemble them to bundles, characterize the bundles by several attributes, and build up a utility model that takes into account the bundle features and customer attributes. Finally, we compute the purchase probabilities and obtain the optimal prices.

To validate our model, we show the implementation results of the proposed pricing strategy in a major IT company. The results show great promise for our pricing strategy's use in broader applications.

Appendix. Additional Numerical Experiments

Table A.1 Financial Assessment for Test 2 (Conditional Probability Approach)

	Actual pricing	Optimal pricing	Change
Scenario 1: win and opt > Actual			
Revenue ($)	42.1 M	41.0 M	−1.1 M (−2.6%)
Profit ($)	12.8 M	17.8 M	5.0 M (39.0%)
Scenario 2: win and opt < Actual			
Revenue ($)	31.9 M	24.7 M	−7.2 M (−22.5%)
Profit ($)	18.4 M	13.6 M	−4.8 M (−26.1%)
Scenario 4: loss and opt < Actual			
Revenue ($)	0.0 M	10.0 M	10.0 M (N/A)
Profit ($)	0.0 M	4.2 M	4.2 M (N/A)
Total			
Revenue ($)	74.0 M	75.7 M	1.7 M (2.2%)
Profit ($)	31.2 M	35.6 M	4.4 M (14.2%)
Overall GP margin (%)	42.1	47.0	4.9

Table A.2 Financial Assessment for Test 2 (Expected Values)

	Expected (opt. price) ($)	Actual ($)	Expected (appr. price) ($)
Revenue (M)	95.4	74.0	86.8
Profit (M)	43.9	31.2	35.2

Table A.3 Financial Assessment for Test 3 (Conditional Probability Approach)

	Actual pricing	Optimal pricing	Change
Scenario 1: win and opt > Actual			
Revenue ($)	69.2 M	68.9 M	−0.3 M (−0.4%)
Profit ($)	28.6 M	31.7 M	3.1 M (10.8%)
Scenario 2: win and opt < Actual			
Revenue ($)	17.9 M	13.7 M	−4.2 M (−23.5%)
Profit ($)	7.2 M	4.4 M	−2.8 M (−38.9%)
Scenario 4: loss and opt < Actual			
Revenue ($)	0.0 M	9.0 M	9.0 M (N/A)
Profit ($)	0.0 M	4.0 M	4.0 M (N/A)
Total			
Revenue ($)	87.1 M	88.7 M	1.6 M (1.8%)
Profit ($)	35.9 M	40.2 M	4.3 M (12.0%)
Overall GP margin (%)	41.2	45.3	4.1

Table A.4 Financial Assessment for Test 3 (Expected Values)

	Expected (opt. price) ($)	Actual ($)	Expected (appr. price) ($)
Revenue (M)	109.1	87.1	108.8
Profit (M)	49.3	35.9	41.8

References

Basu A, Vitharana P (2011) Impact of customer knowledge heterogeneity on bundling strategy. *Marketing Sci.* 28(4):792–801.

Bertsimas D, Popescu I (2003) Revenue management in a dynamic network environment. *Transportation Sci.* 37(3):257–277.

Bodea T, Ferguson M (2014) *Segmentation, Revenue Management and Pricing Analytics* (Routledge, New York).

Bowman EH (1963) Consistency and optimality in managerial decision making. *Management Sci.* 9(2):310–321.

Chatterjee S, Hadi AS, Price B (2000) *Regression Analysis by Example*, 3rd ed. (John Wiley & Sons, Hoboken, NJ).

Hanson W, Martin RK (1990) Optimal bundling pricing. *Management Sci.* 36(2):155–174.

Hitt L, Chen P (2005) Bundling with customer self-selection: A simple approach to bundling low-marginal-cost goods. *Management Sci.* 51(10):1481–1493.

Hothorn T, Hornik K, Zeileis A (2006) Party: A laboratory for recursive partytioning. Accessed January 1, 2014, http://CRAN.R-project.org/package=party.

Jiang Y, Shang J, Kemerer C, Liu Y (2011) Optimizing e-tailer profits and customer savings: Pricing multistage customized online bundles. *Marketing Sci.* 30(4):735–752.

Karalic A (1992) Employing linear regression in regression tree leaves. Neumann B, ed. *ECAI'92: Proc. 10th Eur. Conf. Artificial Intelligence* (John Wiley & Sons, New York), 440–441.

Loh W-Y (2011) Classification and regression trees. *Wiley Interdisciplinary Rev.: Data Mining and Knowledge Discovery* 1(1):14–23.

Murthi B, Sarkar S (2003) The role of the management sciences in research on personalization. *Management Sci.* 49(10):1344–1362.

Phillips R (2005) *Pricing and Revenue Optimization* (Stanford Business Books, Stanford, CA).

Quinlan JR (1992) Learning with continuous classes. Sterling L, ed. *AI'92: Proc. Australian Joint Conf. Artificial Intelligence* (World Scientific, Singapore), 343–348.

Talluri K, van Ryzin G (1998) An analysis of bid-price controls for network revenue management. *Management Sci.* 44(11):1577–1593.

Talluri KT, van Ryzin GJ (2004) *The Theory and Practice of Revenue Management*, Internat. Ser. Oper. Res. and Management Sci., Vol. 68 (Springer, New York).

Theil H (1958) *Economic Forecasts and Policy* (North-Holland Publishing, Amsterdam).

Venkatesh R, Mahajan V (2009) *Design and Pricing of Product Bundles: A Review of Normative Guidelines and Practical Approaches*, Rao VR, ed. Handbook of Pricing Research in Marketing (Edward Elgar Publishing Company, Northampton, MA), 232–257.

Wu S, Hitt L, Chen P, Anandalingam G (2008) Customized bundle pricing for information goods: A nonlinear mixed-integer programming approach. *Management Sci.* 54(3):608–622.