Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 389749, 8 pages
http://dx.doi.org/10.1155/2014/389749

Research Article

Share-of-Surplus Product Line Optimisation with Price Levels

X. G. Luo,[1,2] Z. C. Zhang,[1] C. K. Kwong,[2] and J. F. Tang[1]

[1] *Department of Systems Engineering, State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110004, China*
[2] *Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon 999077, Hong Kong*

Correspondence should be addressed to X. G. Luo; xgluo@mail.neu.edu.cn

Received 5 November 2013; Revised 23 January 2014; Accepted 17 February 2014; Published 1 April 2014

Academic Editor: J. J. Judice

Kraus and Yano (2003) established the share-of-surplus product line optimisation model and developed a heuristic procedure for this nonlinear mixed-integer optimisation model. In their model, price of a product is defined as a continuous decision variable. However, because product line optimisation is a planning process in the early stage of product development, pricing decisions usually are not very precise. In this research, a nonlinear integer programming share-of-surplus product line optimization model that allows the selection of candidate price levels for products is established. The model is further transformed into an equivalent linear mixed-integer optimisation model by applying linearisation techniques. Experimental results in different market scenarios show that the computation time of the transformed model is much less than that of the original model.

1. Introduction

Today, many firms adopt the strategy of product line optimisation to satisfy diverse customer requirements and gain competitive advantages. There are already many product line optimisation models proposed by scholars, for example, buyers' welfare [1], seller's welfare [2], share-of-choice [3], and share-of-surplus [4]. The comprehensive descriptions on the classifications and limitations of product line optimisation can be found in the survey papers by Kaul and Rao [5] and Belloni and Freund [6].

Kraus and Yano [4] established the share-of-surplus product line optimisation model. This model is nonconcave, which makes it very difficult to obtain the global optimal solution. To solve the model, they developed a heuristic algorithm based on simulated annealing to obtain the near-optimal solutions.

In the Kraus and Yano [4] model, product prices are defined as continuous decision variables. However, product line optimisation is a decision process in the early stages of product development; in addition, although product pricing is involved in this process, it tends to be strategic and relatively rough. Precise product pricing is determined only after a product is available for sale. For example, when planning a product line, a company may price a product approximately (e.g., $12,000 or $13,000) rather than precisely (e.g., $12,399). Therefore, taking product price as a continuous decision variable and obtaining the precise optimal product price may not be necessary for most firms; an alternative approach is to select prices from the candidate price levels. This approach has been widely applied in many research papers related to other types of product line optimisation [7–11].

In this research, the Kraus and Yano [4] model is extended as a nonlinear integer programming model that allows the selection of candidate price levels for products. The model is further transformed into an equivalent linear mixed-integer optimisation model by applying linearisation techniques. Because the transformed model can be solved by many well-developed algorithms, such as the simplex-based branch-and-bound algorithm, it can be handled by existing commercial optimisation software packages, for example, IBM ILOG and LINGO. In the numeric experiment section, the computation times of the original model and transformed model are compared in various market scenarios, and the results show that the transformed model is very effective.

The rest of this paper is organised as follows: in Section 2, an extended model for the share-of-surplus product line optimisation is established and further transformed into an equivalent linear model. In Section 3, simulation cases of product line optimisation in three market scenarios are generated, and numeric experiments are performed to empirically verify the effectiveness of the proposed approach. Characteristics of the linearised model are discussed and conclusions are drawn in Section 4.

2. Optimisation Model and Linearisation

2.1. The Extended Share-of-Surplus Product Line Optimisation Model. The optimisation problem is described as follows. A firm is going to develop a product line. There are I market segments for the product, each of which contains customers with homogeneous preferences. The size (or the number of expected consumers) of the ith segment is denoted as D_i. A reference set containing J candidate products has been generated for further selection. The utility (measured in dollars) of the customers in the ith market segment towards the jth product is denoted as U_{ij}. The variable cost of the jth product is denoted as c_j. There are K_j product price levels for the jth product, and the kth price level of the jth product is denoted by p_{jk}. The purpose of the problem is to determine which products from the reference set should be included in the product line and which price level to choose for each product of the product line to maximise the company's total profit.

Assumptions of the optimisation problem include [4] the following. (1) Customers choose product by following the share-of-surplus choice rule. Kraus and Yano [4] explained the rule as follows. "The share-of-surplus choice rule defines the probability that a customer in a segment selects a certain product as the ratio of the segment's surplus from this particular product to the segment's total surplus across all products with positive surplus (for that customer or segment)." (2) competitive companies do not respond to the company's moves (i.e., not a game). (3) The prices of the competitive products are given constants. (4) Price discrimination is not allowed. (5) Customers have complete information regarding the available products and their prices.

Let y_{ij} be a binary decision variable such that $y_{ij} = 1$ if all or a portion of customers in the ith segment choose the jth product and $y_{ij} = 0$ otherwise, and let x_{jk} be a binary decision variable such that $x_{jk} = 1$ if the kth price level is selected for the jth product and $x_{jk} = 0$ otherwise; the optimisation problem can be formulated as the following nonlinear integer programming model (Model A):

$$\text{Max} \quad \sum_{i=1}^{I}\sum_{j=1}^{J} D_i \left(\sum_{k=1}^{K_j} x_{jk}p_{jk} - c_j\right) \times \left[\frac{\left(U_{ij} - \sum_{k=1}^{K_j} x_{jk}p_{jk}\right) y_{ij}}{\sum_{j'=1}^{J}\left[\left(U_{ij'} - \sum_{k=1}^{K_{j'}} x_{j'k}p_{j'k}\right) y_{ij'}\right]}\right] \tag{1}$$

$$\text{s.t.} \quad \sum_{j'=1}^{J}\left[\left(U_{ij'} - \sum_{k=1}^{K_{j'}} x_{j'k}p_{j'k}\right) y_{ij'}\right] > 0, \quad i = 1,2,\ldots,I \tag{2}$$

$$\left(U_{ij} - \sum_{k=1}^{K_j} x_{jk}p_{jk}\right) y_{ij} \ge 0, \quad i = 1,2,\ldots,I; \quad j = 1,2,\ldots,J \tag{3}$$

$$\left(U_{ij} - \sum_{k=1}^{K_j} x_{jk}p_{jk}\right)\left(1 - y_{ij}\right) \le 0, \quad i = 1,2,\ldots,I; \quad j = 1,2,\ldots,J \tag{4}$$

$$\sum_{k=1}^{K_j} x_{jk} = 1, \quad j = 1,2,\ldots,J \tag{5}$$

$$y_{ij}, x_{jk} \in \{0,1\}, \quad i = 1,2,\ldots,I; \quad j = 1,2,\ldots,J; \quad k = 1,2,\ldots,K_j. \tag{6}$$

The objective function of the model is to maximise the total profit (revenue minus cost) of the product line in all segments. The expression $\sum_{k=1}^{K_j} x_{jk}p_{jk} - c_j$ in the objective function represents the marginal profit of the jth product. The expression $(U_{ij} - \sum_{k=1}^{K_j} x_{jk}p_{jk})y_{ij}/\sum_{j'=1}^{J}[(U_{ij'} - \sum_{k=1}^{K_j} x_{j'k}p_{j'k})y_{ij'}]$ represents the choice probability of customers in the ith market segment towards the jth product by following the share-of-surplus purchase choice rule. Constraint (2) can avoid the case that the denominator part is equal to zero. Constraint (3) ensures that a product's utility surplus must be nonnegative if the product is selected by customers. Constraint (4) guarantees that a product's utility surplus must be nonpositive if the product is not selected by customers. Constraint (5) ensures that only one price level is selected for a specific product. Constraint (6) confines the decision variables y_{ij} and x_{jk} as binary ones.

2.2. Linearisation Approach. Consider the following linear mixed-integer programming model (Model B):

$$\text{Max} \quad \sum_{i=1}^{I}\sum_{j=1}^{J} D_i \left[\left(U_{ij} + c_j\right)\sum_{k=1}^{K_j} p_{jk}\, z2_{ijk} - U_{ij}c_j z1_{ij} - \sum_{k=1}^{K_j}\sum_{k'=1}^{K_j} p_{jk}p_{jk'} z3_{ijkk'}\right] \tag{7}$$

$$\text{s.t.} \quad \sum_{j'=1}^{J}\left(U_{ij'} z1_{ij'} - \sum_{k=1}^{K_{j'}} p_{j'k} z2_{ij'k}\right) = 1, \quad i = 1,2,\ldots,I \tag{8}$$

$$z1_{ij} \le M y_{ij}, \quad i = 1,2,\ldots,I; \; j = 1,2,\ldots,J \tag{9}$$

$$z1_{ij} \le z_i^{\text{deno}} + M\left(1 - y_{ij}\right), \quad i = 1,2,\ldots,I; \quad j = 1,2,\ldots,J \tag{10}$$

$$z1_{ij} \geq z_i^{\text{deno}} + M\left(y_{ij} - 1\right), \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J \tag{11}$$

$$z2_{ijk} \leq Mz4_{ijk}, \quad i = 1, 2, \ldots, I; \; j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{12}$$

$$z2_{ijk} \leq z_i^{\text{deno}} + M\left(1 - z4_{ijk}\right), \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{13}$$

$$z2_{ijk} \geq z_i^{\text{deno}} + M\left(z4_{ijk} - 1\right), \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{14}$$

$$z4_{ijk} \leq x_{jk}, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{15}$$

$$z4_{ijk} \leq y_{ij}, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{16}$$

$$z4_{ijk} \geq 1 + M\left(x_{jk} + y_{ij} - 2\right), \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{17}$$

$$z3_{ijkk'} \geq z_i^{\text{deno}} + M\left(x_{jk} + x_{jk'} + y_{ij} - 3\right), \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j; \quad k' = 1, 2, \ldots, K_j \tag{18}$$

$$\sum_{j'=1}^{J}\left(U_{ij'} y_{ij'} - \sum_{k=1}^{K_{j'}} p_{j'k} z4_{ij'k}\right) > 0, \quad i = 1, 2, \ldots, I \tag{19}$$

$$U_{ij} y_{ij} - \sum_{k=1}^{K_j} p_{jk} z4_{ijk} \geq 0, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J \tag{20}$$

$$U_{ij} - \sum_{k=1}^{K_j} p_{jk} x_{jk} - U_{ij} y_{ij} + \sum_{k=1}^{K_j} p_{jk} z4_{ijk} \leq 0, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J \tag{21}$$

$$\sum_{k=1}^{K_j} x_{jk} = 1, \quad j = 1, 2, \ldots, J \tag{22}$$

$$y_{ij}, x_{jk} \in \{0, 1\}, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{23}$$

$$z1_{ij}, z2_{ijk}, z3_{ijkk'}, z4_{ijk}, z_i^{\text{deno}} \geq 0, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j; \quad k' = 1, 2, \ldots, K_j, \tag{24}$$

where M is a large positive number.

Theorem 1. *Model B has the same optimisation result as that of Model A.*

Proof of Theorem 1 is given in Appendix.

Compared with Model A, Model B is a mixed-linear model and thus can be solved by many well-developed algorithms integrating mature linear programming methods (e.g., simplex, barrier). Most of the commercial optimisation software packages, such as IBM ILOG and LINGO, provide modules for solving this type of problem.

The cost of the transformation lies in the extra intermediate variables and constraints appended to the model. Model B has $IJK^2 + 2IJK + IJ$ more continuous variables and $IJK^2 + 6IJK + 3IJ + I$ more constraints than Model A. However, the results of the experiments show that the transformation is still quite efficient.

3. Numeric Experiments

All the experiments in this section were run on a personal computer (4 GB RAM, 3.30 GHz CPU, Windows 7). The IBM ILOG software package is version 12.4. The LINGO software package is version 11.0. All the generated cases, modelling files, and computational results can be found at: http://faculty.neu.edu.cn/ise/luoxinggang/paper16/data&code.htm.

3.1. Case Generation. Three types of product line design problems, that is, "random," "rich-poor," and "quality", applied in Kraus and Yano [4], were generated for numeric experiments. The first type, "random," corresponds to the practical scenario where consumers in different market segments have idiosyncratic preferences towards a product and there is no obvious relationship between the preferences. The second type, "rich-poor," simulates the scenario where the number of rich consumers with higher utility is small while the number of poor consumers with lower utility is large. The third type, "quality," is designed for representing the scenario where all consumers prefer products with high production cost and tend to have higher utilities for products with higher variable costs.

For the first scenario "random," as the name implies, all the parameters of the model, including U_{ij}, c_j, and D_i, are generated randomly with a uniform distribution. However, because c_j is the product variable cost, which is smaller than the utility of the product, it is generated according to the formula $c_j = U[0.05, 0.2] * \max_{i=1}^{I} U_{ij}$, which implies that the value of cost is a small percentage of $\max_i\{U_{ij}\}$.

For the second scenario "rich-poor," it shows the situation where there are a few "rich" consumers who prefer high-price products and a lot of "poor" consumers who prefer

Table 1: Parameter settings for case generation.

Type	Parameter generation
Random	$U_{ij} = U[0, 100]$ $c_j = U[0.05, 0.2] * \max_{i=1}^{I} U_{ij}$ $D_i = U[50, 100]$
Rich-poor	$U_{ij} = (b_{ij} + 100) * i/I, b_{ij} = U[0, 100]$ $c_j = U[0.05, 0.2] * \max_{i=1}^{I} U_{ij}$ $D_i = [f_{ij} * (I - i)/I] + 50, f_{ij} = U[0, 100]$
Quality	$U_{ij} = b_{ij} * c_j, b_{ij} = U[1.0, 1.5]$ $c_j = U[5, 20]$ $D_i = U[50, 100]$

Note: $U[a, b]$ represents a stochastic variable following a uniform distribution on $[a, b]$.

low-price products. The formula $U_{ij} = (b_{ij} + 100) * i/I$ indicates that a market segment with small i represents "poor" consumers and a market with large i represents "rich" consumers. Correspondently, $D_i = [f_{ij} * (I - i)/I] + 50$ indicates that a market segment with small i has a large number of consumers and a market with large i has a small number of consumers.

For the third scenario "quality," it assumes that the cost is the most important influential factor for product quality and higher costs ensure better quality. According to the formula $U_{ij} = b_{ij} * c_j$ and the range of b_{ij} (b_{ij} is generated randomly with the range of [1.0, 1.5]), it can be inferred that $c_j = (1/b_{ij}) * U_{ij}$ and the range of $1/b_{ij}$ is [2/3, 1]. In other words, the ratio of cost to utility in this scenario is much higher than those of the previous two scenarios, and the cost has great influence on consumer's utility towards a product.

Table 1 shows the settings of the model parameters for generating the cases.

3.2. Experiment Results. Because all parameters of the cases are randomly generated, for a specific type of product line design problem at given size of market segment (I), number of candidate products (J), and numbers of price levels (K), 100 cases are randomly generated. The presented computational time for each type is the average of computation times of the 100 homogeneous cases. The number of price levels is set as a fixed value for all products in a product line in the experiments for simplicity. Experiments with different values of I, J, and K are performed to show the influence of problem size on average computation time. Table 2 shows the average computation time of Models A and B under the three different types of scenarios in three sets of different values of parameters I, J, and K. Some of the cells in the table are marked as "N/A", indicating that the computation time is not available because the computation time of a case group (100 cases) is over 24 hours. In the following three experiments, IBM ILOG software package was applied to solve Models A and B.

In the first experiment, Models A and B under the three scenarios were solved when the size of market segment (I) is set as different values. A comparison of Models A and B under the three scenarios is shown in Figure 1. The computation time of Model B is much less than that of Model A. From the trend line of data points, it can be inferred that the computation time is almost linearly related to the size of market segment, although the shapes of data points under the three scenarios are slightly different. A triangle marker in Figure 1 represents the difference of Model B's computation time and Model A's computation time. With the increase of the size of market segment, the value of the difference is also enhanced rapidly, indicating that Model B can save more computation time.

In the second experiment, the average computation times of Models A and B under the three scenarios were compared when the number of products (J) is changed. Figure 2 shows that (1) the computation time of Model B is much less than that of Model A, (2) computation times of Models A and B are enhanced nonlinearly (probably exponentially) with the increase of the number of products, and (3) the number of products heavily influences the computational efficiency of Models A and B.

In the third experiment, Models A and B under the three scenarios are solved when the number of price levels (K) is increased. The results are depicted in Figure 3. The results also show that Model B has increasingly better computation efficiency over Model A as the number of price levels increases.

The value of M in Model B should be positive and very large in order to keep the constraints effective. In the above-mentioned three experiments, M is set as $(\min_{i,j,k} U_{ij} - p_k)^{-1} + 1$ because it can ensure that constraint (9)–(14), (17), and (18) work properly. For example, this setting of M of constraint (11) can ensure that the constraint always holds (i.e., no effect) when $y_{ij} = 0$ since M is larger than z_i^{deno}.

The above-mentioned three experiments were also performed by LINGO. The comparison between computation time results of Models A and B has similar trends and conclusions as those by IBM ILOG, except that the computation time of LINGO is much longer than IBM ILOG. The computation results using LINGO can be found at the URL mentioned in the first paragraph of this section.

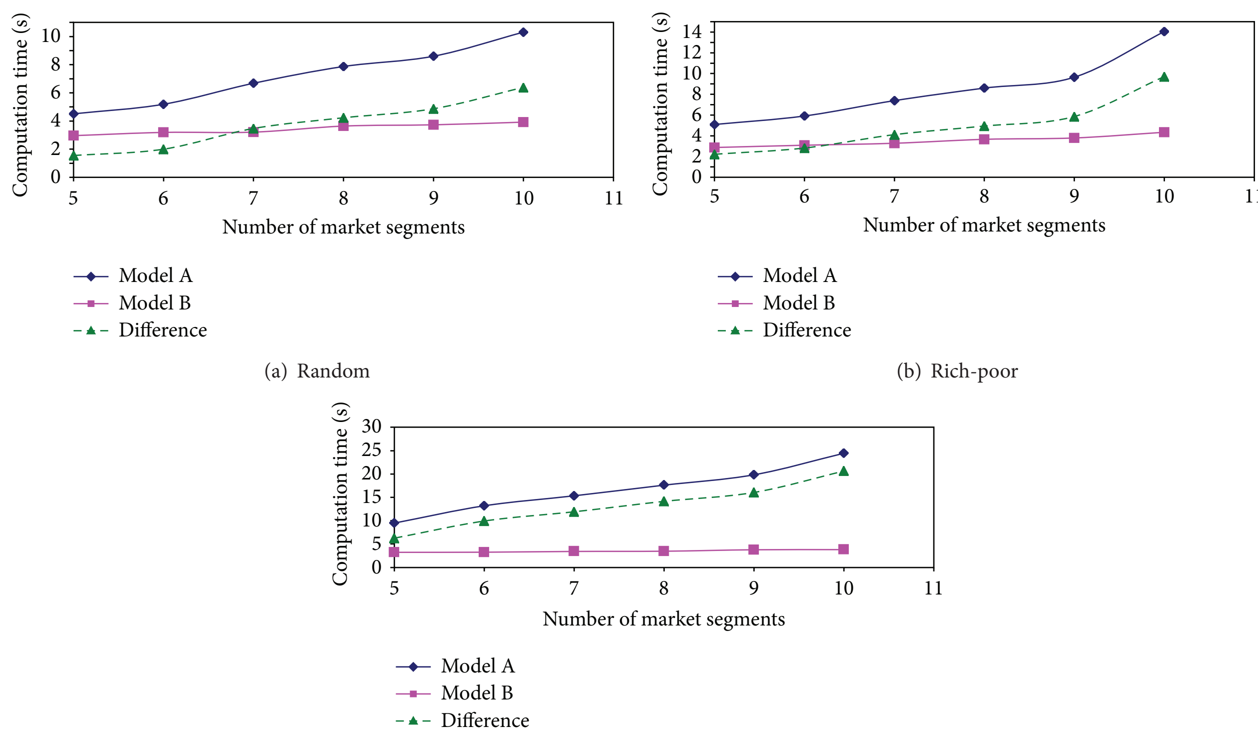
4. Discussion and Conclusions

In this study, the share-of-surplus product line optimisation model by Kraus and Yano [4] is extended to allow the selection of candidate price levels for products, which is more realistic and has been used in other product line models. The proposed model is further transformed into an equivalent linear mixed-integer programming model.

The transformed linear model adds a number of extra auxiliary continuous variables and linear constraints to the original model. However, the numeric experiments show that the computation time of the transformed model is much less than that of the original one in different market scenarios,

Table 2: Average computation time (second) of Models A and B.

	Random		Rich-poor		Quality	
	Model A	Model B	Model A	Model B	Model A	Model B
$I = 5, J = 5, K = 5$	4.51	2.96	5.09	2.87	9.50	3.23
$I = 6, J = 5, K = 5$	5.19	3.19	5.91	3.09	13.20	3.26
$I = 7, J = 5, K = 5$	6.68	3.21	7.39	3.28	15.34	3.42
$I = 8, J = 5, K = 5$	7.87	3.64	8.60	3.66	17.62	3.47
$I = 9, J = 5, K = 5$	8.60	3.73	9.65	3.79	19.84	3.78
$I = 10, J = 5, K = 5$	10.31	3.92	14.05	4.34	24.45	3.81
$I = 5, J = 5, K = 5$	4.51	2.96	5.09	2.87	9.50	3.23
$I = 5, J = 6, K = 5$	33.78	4.25	49.93	7.26	63.47	3.74
$I = 5, J = 7, K = 5$	173.91	9.72	282.09	16.22	340.62	7.37
$I = 5, J = 8, K = 5$	838.31	19.06	N/A	50.32	N/A	14.97
$I = 5, J = 9, K = 5$	N/A	77.33	N/A	215.81	N/A	35.42
$I = 5, J = 10, K = 5$	N/A	291.77	N/A	951.90	N/A	132
$I = 5, J = 5, K = 5$	4.51	2.96	5.09	2.87	9.50	3.23
$I = 5, J = 5, K = 6$	13.86	3.38	17.37	3.83	28.52	4.43
$I = 5, J = 5, K = 7$	35.14	5.35	43.29	7.10	66.62	5.62
$I = 5, J = 5, K = 8$	81.09	6.97	76.66	9.39	129.07	7.13
$I = 5, J = 5, K = 9$	146.08	13.86	125.8	11.64	226.33	9.73
$I = 5, J = 5, K = 10$	244.59	20.61	189.3	16.69	354.34	14.85



(a) Random

(b) Rich-poor

(c) Quality

Figure 1: Comparison of Models A and B when I is changed.

and the efficiency of the transformed model increases with the scale of the problem.

It must be noted that for large-scale problems, the computation time of the transformed linear model is still too long due to the integer variables, and a metaheuristic algorithm such as evolutionary computation is probably a better choice. However, exact approaches can achieve the global optimal solutions and thus are attractive to small-scale practical problems (e.g., service products, some products of SME). In addition, global optimal solutions can also be used to evaluate

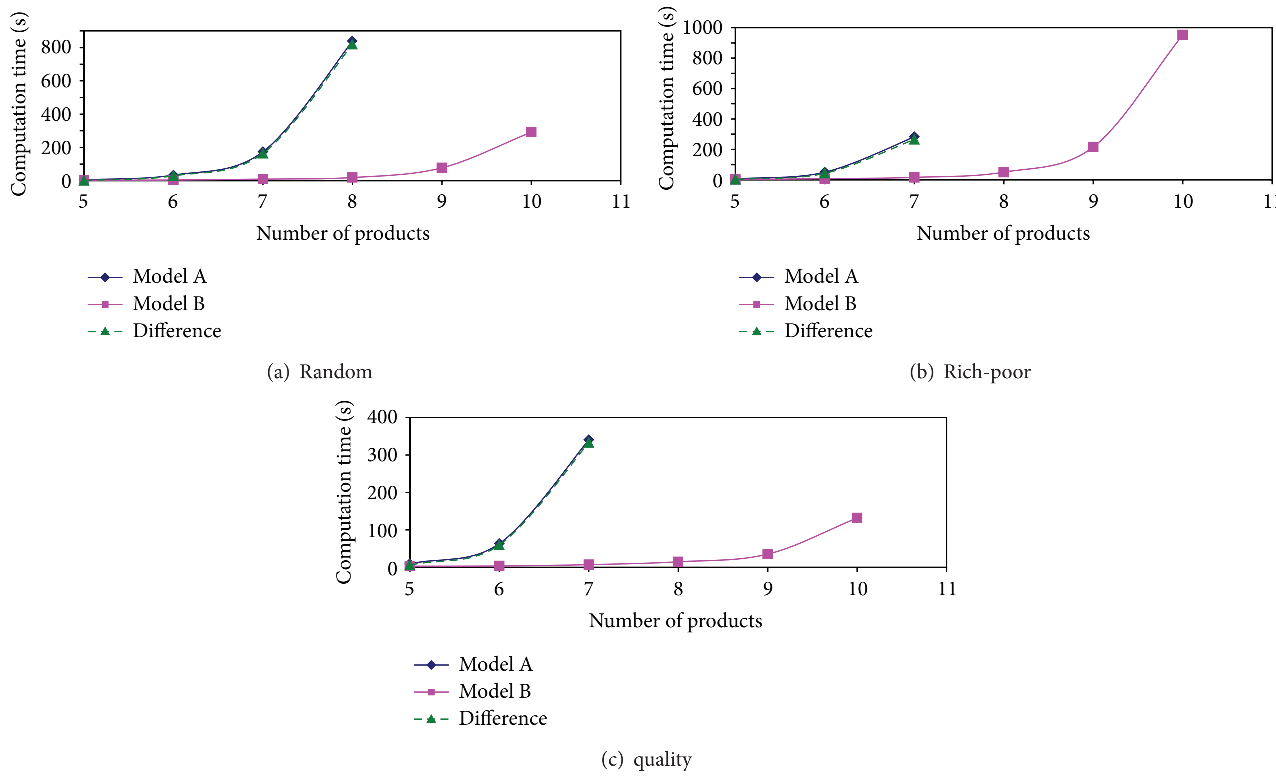


Figure 2: Comparison of Models A and B when J is changed.

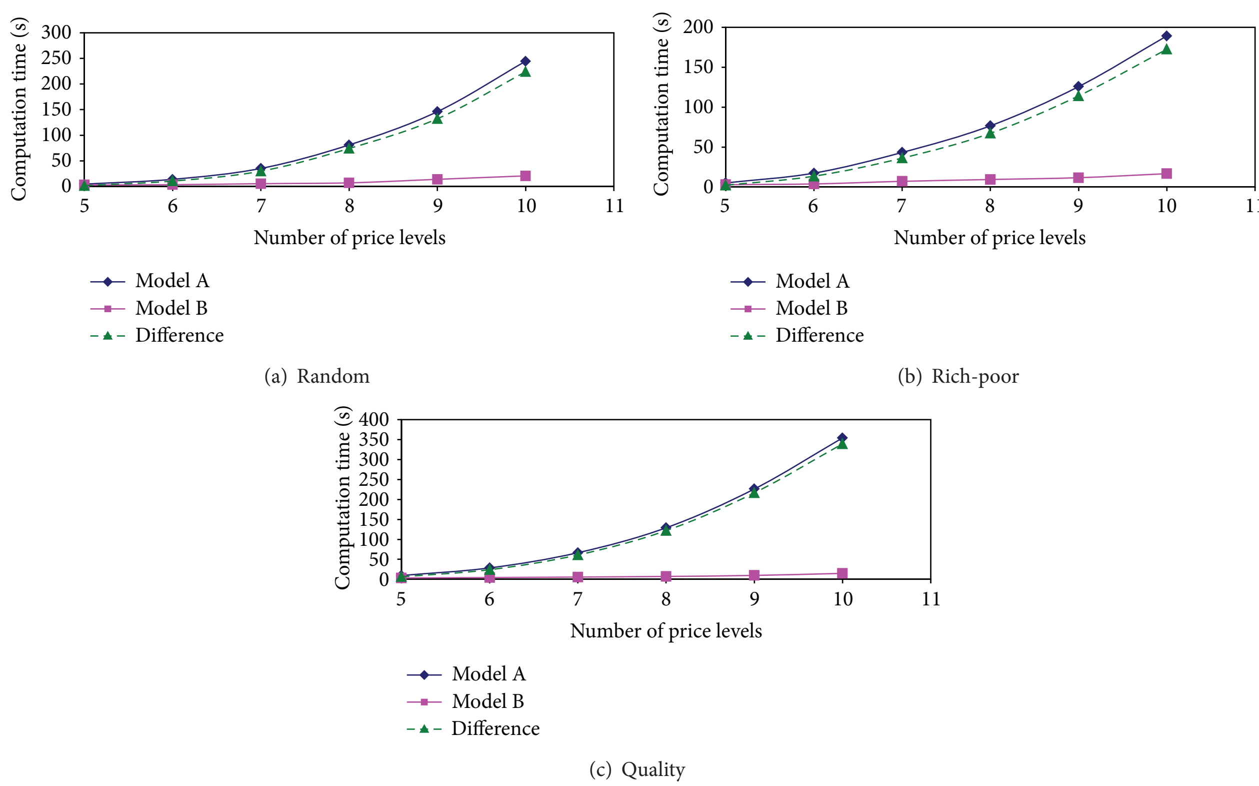


Figure 3: Comparison of Models A and B when K is changed.

the performance of heuristic or metaheuristic algorithms because they can provide a base line for comparison.

Appendix

Proof of Theorem 1

Let $z_i^{\text{deno}} = 1/\sum_{j'=1}^{J}[(U_{ij'} - \sum_{k=1}^{K_{j'}} x_{j'k}p_{j'k})y_{ij'}]$, replace $\sum_{j'=1}^{J}[(U_{ij'} - \sum_{k=1}^{K_{j'}} x_{j'k}p_{j'k})y_{ij'}]$ in the objective function with z_i^{deno}, and transform $z_i^{\text{deno}} = 1/\sum_{j'=1}^{J}[(U_{ij'} - \sum_{k=1}^{K_{j'}} x_{j'k}p_{j'k})y_{ij'}]$ into a constraint $\sum_{j'=1}^{J}(U_{ij'}y_{ij'}z_i^{\text{deno}} - \sum_{k=1}^{K_{j'}} p_{j'k}x_{j'k}y_{ij'}z_i^{\text{deno}}) = 1$. Model A can be reformulated as the following model (Model C):

$$\text{Max} \quad \sum_{i=1}^{I}\sum_{j=1}^{J} D_i \left[\left(U_{ij} + c_j\right) \sum_{k=1}^{K_j} p_{jk} x_{jk} y_{ij} z_i^{\text{deno}} - U_{ij} c_j y_{ij} z_i^{\text{deno}} - \sum_{k=1}^{K_j} \sum_{k'=1}^{K_j} p_{jk} p_{jk'} x_{jk} x_{jk'} y_{ij} z_i^{\text{deno}} \right] \tag{A.1}$$

$$\text{s.t.} \quad \sum_{j'=1}^{J} \left(U_{ij'} y_{ij'} z_i^{\text{deno}} - \sum_{k=1}^{K_{j'}} p_{j'k} x_{j'k} y_{ij'} z_i^{\text{deno}} \right) = 1, \quad i = 1, 2, \ldots, I \tag{A.2}$$

$$\sum_{j'=1}^{J} \left(U_{ij'} y_{ij'} - \sum_{k=1}^{K_{j'}} p_{j'k} x_{j'k} y_{ij'} \right) > 0, \quad i = 1, 2, \ldots, I \tag{A.3}$$

$$\left(U_{ij} y_{ij} - \sum_{k=1}^{K_j} p_{jk} x_{jk} y_{ij} \right) \geq 0, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J \tag{A.4}$$

$$U_{ij} - \sum_{k=1}^{K_j} p_{jk} x_{jk} - U_{ij} y_{ij} + \sum_{k=1}^{K_j} p_{jk} x_{jk} y_{ij} \leq 0, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J \tag{A.5}$$

$$\sum_{k=1}^{K_j} x_{jk} = 1, \quad j = 1, 2, \ldots, J \tag{A.6}$$

$$y_{ij}, x_{jk} \in \{0, 1\}, \quad i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K_j \tag{A.7}$$

$$z_i^{\text{deno}} \geq 0, \quad i = 1, 2, \ldots, I. \tag{A.8}$$

In Model B, there are three constraints, (9), (10), and (11), which directly relate to the continuous intermediate variable $z1_{ij}$. If the binary decision variable $y_{ij} = 0$, then constraints (9) and (24) force $z1_{ij} = 0$; if $y_{ij} = 1$, then constraints (10) and (11) force $z1_{ij} = z_i^{\text{deno}}$. Therefore, if $z1_{ij}$ is replaced by $y_{ij}z_i^{\text{deno}}$ and constraints (9), (10), and (11) are removed in Model B, the modified model has the same optimisation result as that of Model B.

Similarly, $z2_{ijk}$ can be replaced by $z4_{ijk}z_i^{\text{deno}}$, and $z4_{ijk}$ can be replaced by $x_{jk}y_{ij}$.

Note that $z3_{ijkk'}$ appears only in the objective function besides constraint (18) of Model B, and the nonlinear element containing $z3_{ijkk'}$ is negative, and hence the maximisation objective function minimises $z3_{ijkk'}$. If $x_{jk}, x_{jk'}, y_{ij} = 1$, the effect of constraint (18) and the objective function forces $z3_{ijkk'} = z_i^{\text{deno}}$; if one of $x_{jk}, x_{jk'}, y_{ij} = 0$, the effect of constraint (24) and the objective function forces $z3_{ijkk'} = 0$. Therefore, $z3_{ijkk'}$ can be replaced by $x_{jk}x_{jk'}y_{ij}z_i^{\text{deno}}$.

For Model B, if $z1_{ij}$ is replaced by $y_{ij}z_i^{\text{deno}}$, $z2_{ijk}$ is replaced by $z4_{ijk}z_i^{\text{deno}}$, $z4_{ijk}$ is replaced by $x_{jk}y_{ij}$, $z3_{ijkk'}$ is replaced by $x_{jk}x_{jk'}y_{ij}z_i^{\text{deno}}$, and constraints (9)–(18) are removed, then the optimal solutions of the modified model remain unchanged from the original model (Model B). However, the modified model actually is exactly Model C. Therefore, Model B has the same optimisation result as that of Model C or Model A.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research is financially supported by the National Science Foundation of China (NSFC) (Projects nos. 71171039, 61273204, and 71021061) and the Fundamental Research Funds for Central Universities (Project no. N110204005). The work described in this paper was supported by a grant from The Hong Kong Polytechnic University (Project no. G-YK06). The authors would like to thank the two anonymous reviewers for their valuable comments and suggestions.

References

[1] P. E. Green and A. M. Krieger, "Models and heuristics for product line selection," *Marketing Science*, vol. 4, pp. 1–19, 1985.

[2] R. Kohli and R. Sukumar, "Heuristics for product-line design using conjoint analysis," *Management Science*, vol. 36, no. 12, pp. 1464–1478, 1990.

[3] J. D. Camm, J. J. Cochran, D. J. Curry, and S. Kannan, "Conjoint optimization: an exact branch-and-bound algorithm for the share-of-choice problem," *Management Science*, vol. 52, no. 3, pp. 435–447, 2006.

[4] U. G. Kraus and C. A. Yano, "Product line selection and pricing under a share-of-surplus choice model," *European Journal of Operational Research*, vol. 150, no. 3, pp. 653–671, 2003.

[5] A. Kaul and V. R. Rao, "Research for product positioning and design decisions: an integrative review," *International Journal of Research in Marketing*, vol. 12, no. 4, pp. 293–320, 1995.

[6] A. Belloni, R. Freund, M. Selove, and D. Simester, "Optimizing product line designs: efficient methods and comparisons," *Management Science*, vol. 54, no. 9, pp. 1544–1552, 2008.

[7] S. K. Nair, L. S. Thakur, and K. Wen, "Near optimal solutions for product line design and selection: beam search heuristics," *Management Science*, vol. 41, no. 5, pp. 767–785, 1995.

[8] K. D. Chen and W. H. Hausman, "Mathematical properties of the optimal product line selection problem using choice-based conjoint analysis," *Management Science*, vol. 46, no. 2, pp. 327–332, 2000.

[9] H. Li and S. Azarm, "An approach for product line design selection under uncertainty and competition," *Journal of Mechanical Design, Transactions of the ASME*, vol. 124, no. 3, pp. 385–392, 2002.

[10] J. Jiao and Y. Zhang, "Product portfolio planning with customer-engineering interaction," *IIE Transactions*, vol. 37, no. 9, pp. 801–814, 2005.

[11] X. G. Luo, C. K. Kwong, J. F. Tang, S. F. Deng, and J. Gong, "Integrating supplier selection in optimal product family design," *International Journal of Production Research*, vol. 49, no. 14, pp. 4195–4222, 2011.