Ma *SpringerPlus (2016)5:727*
DOI 10.1186/s40064-016-2421-0

SpringerPlus

RESEARCH Open Access

A nonlinear bi-level programming approach for product portfolio management

Shuang Ma*

*Correspondence:
mashuang@tju.edu.cn
School of Management
and Economics, Beijing
Institute of Technology,
Beijing 100081, China

Abstract

Product portfolio management (PPM) is a critical decision-making for companies across various industries in today's competitive environment. Traditional studies on PPM problem have been motivated toward engineering feasibilities and marketing which relatively pay less attention to other competitors' actions and the competitive relations, especially in mathematical optimization domain. The key challenge lies in that how to construct a mathematical optimization model to describe this Stackelberg game-based leader–follower PPM problem and the competitive relations between them. The primary work of this paper is the representation of a decision framework and the optimization model to leverage the PPM problem of leader and follower. A nonlinear, integer bi-level programming model is developed based on the decision framework. Furthermore, a bi-level nested genetic algorithm is put forward to solve this nonlinear bi-level programming model for leader–follower PPM problem. A case study of notebook computer product portfolio optimization is reported. Results and analyses reveal that the leader–follower bi-level optimization model is robust and can empower product portfolio optimization.

Keywords: Product portfolio management, Nonlinear bi-level programming, Leader–follower joint optimization, Bi-level nested genetic algorithm

Background

In the highly competitive environment, an important strategic decision for an organization is to optimize its product portfolio to stimulate sales and increase revenues (Ho and Tang 1998). Therefore, optimal product portfolios are interesting for many people over last decades. PPM is a dynamic decision process that analyzes production ability and market potential, and hence determines the optimal portfolio of products with consideration of profit maximization (McNally et al. 2009). PPM is used to develop a set of products with their diversity such as attribute, attribute level, also simultaneously considering product performance, competitive environment information, engineering requirement, manufacturing procedure, and etc. (Cooper et al. 1999). Thus, PPM problem is not a single-objective optimization but a multi-objective combinatorial optimization problem.

In recent years, with the development of product portfolio design, researches in different perspectives of product portfolio optimization have been raised. Such as, considering product portfolio planning with the view of customer–engineering interaction (Jiao and Zhang 2005); using a data mining method to manage a portfolio of products in the

Springer Open

cross-domain (Pachidi et al. 2014), but with little concerns in competitive environment or competitive relationship in the product portfolio optimization research.

According to Stackelberg, each firm in a competitive market can be a leader or a follower. The leader is a decision-maker and the follower is characterized as a firm that behaves according to a Nash reaction function (Colson et al. 2007). Therefore, the leader's PPM optimization results could be very different from the follower's because of their disparate decision-making positions. Optimization for PPM problem with competitive environment considerations is not an optional issue, but a targeted one. In this paper, a hierarchical joint optimization model is developed in line with bi-level programming, in which focal manufacturer plays as a leader while others act as followers.

The rest of the paper proceeds as follows. Next section reviews related work regarding PPM and bi-level programming. PPM with competitive considerations is formulated as a leader–follower joint optimization problem in "Problem description" section. "Bi-level joint optimization" section elaborates the formulation of a bi-level optimization model for PPM. "Model solution" section develops a bi-level nested genetic algorithm for efficient solution of the hierarchical joint optimization model. A case study of notebook computer is reported in "Case study" section and paper concludes in "Conclusions" section.

Related work

PPM

PPM problem is defined as a dynamic decision-making process in which some optimization criteria should be finished, such as share-of-choice (Sadeghi and Zandieh 2011). Four main goals of PPM are summarized by Cooper et al. (2002): (1) Value maximization, (2) Strategic choice, (3) New product and technology choices and (4) Balancing resources. First, value maximization is to optimize resource allocation to gain the maximal value with this product portfolio. Strategic choice aims to ensure a right decision on the business strategy. And new product and technology choices link to what kind of active products, R&D projects and new technology are updated and revised. Balancing resources asks to ensure right number of projects (products) with limited resources and allocate resources between projects or products.

A sizeable body of research on PPM has been reported over the last decade, generally can be divided into four categories with different optimization goals (Otten et al. 2015). With value maximization goal, the fiscal optimization should be performed, and methods are much various, for example, Expected Commercial Value (ECV) method, Productivity Index (PI) and Return on Investment (ROI) method (Cooper et al. 2001; Dickinson et al. 2001). As for the second goal, Bai and Sarkis (2013) pointed out that strategic alignment is the most important success factor for developing business processes and performance management. And a method for achieving strategic alignment with its product portfolio is called the strategic bucket method which is defined as a financial support for a new product development in line with an extraordinary strategy (Chao and Kavadias 2007). The third perspective is new product and technology choices. According to Barczak et al. (2009), performance of a company is coupled with the rate of innovative new product and technology practice. Meanwhile, current literature on PPM covers new product and technology choices with lifecycle consideration, and it's just like a project in new product design process (Tolonen et al. 2015). Balancing resources can be clarified by terms and parameters such as time, risk, markets and technologies (Tolonen et al. 2015). This process can be accomplished by developing a resource capacity analysis with the quantification of product's demand for resources versus the availability of them (Otten et al. 2015).

Bi-level programming

The theoretic ground form of bi-level programming formulation originates from the Stackelberg games (von Stackelberg 1952). Bi-level optimization refers to a mathematical programming which contains a sub-optimization problem as its constraint, first studied and proposed by Bracken and McGill (1973). Bi-level programming theory rapidly becomes an important branch in mathematical programming field because of its abstract of an essential class of hierarchical decision-making problems including Stackelberg Game. Although as a generalized form of mathematical programming, bi-level programming is very different from ordinary mathematical programming. Bi-level optimization problems are complex and often belong to a higher complexity class than their corresponding single-level relaxations (Sakawa et al. 2011). For example, Jeroslow shows that even a linear bi-level programming problem is NP-hard (Jeroslow 1985). Du and Wang (2009) point out that when upper-level constraints contain the optimization solution of the lower-level problem the feasible region may become discontinuous. Bi-level programming is also much more complicated due to difficulties in solving the problems (Christiansen et al. 2001). Traditional solution approaches include vertex enumeration algorithms and methods based on Kuhn–Tucker (KKT) conditions or penalty functions.

Recent trends of bi-level programming mainly revolve around computing and solving this critical problem. One focus of researches is on the theoretical solutions. For example, Ye and Zhu (2010) investigate for bi-level programming solutions based on the value functions. While another one focuses on the approximation solutions, for instance, Sakawa et al. (2011) work on fuzzy interaction-based satisfaction solutions for bi-level programming. The leader–follower Stackelberg and bi-level optimization have been applied in a number of fields, including networks (van Hoesel 2008), advertising (Aust and Buscher 2012), retailer supply chains (Xiao et al. 2014; Esmaeili et al. 2009), and supply chain contracts (Chen et al. 2012).

Problem description

Consistent with a Stackelberg game, product portfolio management can be formulated as a hierarchical optimization problem consisting of one upper-level and one or more lower-level decision-making agents. The former plays a leader's role, denoted as *F*, whilst the latter acts as a follower, denoted as *f*. The leader's decision-making results will serve as the constraints of followers' optimization problem, whereas the followers feed their optimal solutions back to the leader for adjustment of the leader's decision. For the leader–follower joint optimization problem, the leader and followers can be the real designers or design teams, or any virtual decision makers.

Figure 1 illustrates the bi-level decision model for product portfolio optimization. The leader *F* determines its selections firstly, and the follower *f* adjusts its product portfolio subjecting to the feasible regions given by the leader.

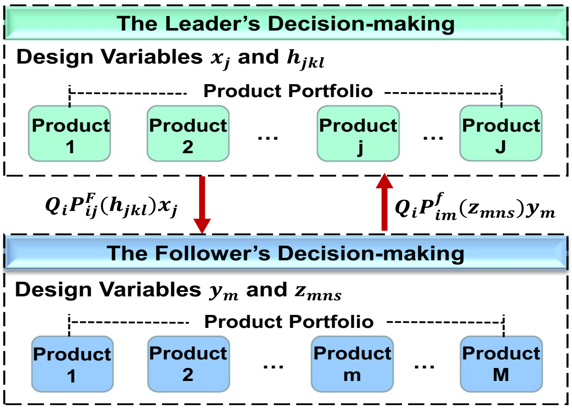
The Leader's Decision-making
Design Variables x_j and h_{jkl}
Product Portfolio
Product 1
Product 2
…
Product j
…
Product J
$Q_i P_{ij}^F(h_{jkl})x_j$
$Q_i P_{im}^f(z_{mns})y_m$
The Follower's Decision-making
Design Variables y_m and z_{mns}
Product Portfolio
Product 1
Product 2
…
Product m
…
Product M

Fig. 1 Bi-level decision-making between leader and follower

Assuming that there is one leader and only one follower in the market place to simplify this bi-level optimization problem. The total number of products in leader's product family is J and the follower's is M, which are separately defined as a set $\{R_j\}$ (j = 1, 2, …, J) and $\{r_m\}$ (m = 1, 2, …, M). With variant attributes and attribute levels, utility and cost of each product is very different. The optimal solutions will be sent to the follower as vectors X and H after the leader's decision-making according to marketing and engineering considerations, then the follower optimizes its PPM problem and sends solutions to the leader as a feedback. I construct a criterion of market share constraint as the coordinated mechanism for this bi-level optimization problem to develop a more satisfactory product portfolio scheme for both leader and follower.

Bi-level joint optimization

As described in Chapter 3, optimization for PPM problem aims to provide an optimal set of products with mixture of attributes into the target marketplace for both leader and follower (Jiao and Zhang 2005). In the following, I address how to formulate the bi-level programming model containing objective functions and constraints for this optimization.

Upper-level PPM optimization

The upper-level deals with F-PPM problem by selecting appropriate combination from a number of J products. The decision variable of upper-level is x_j (j = 1, 2, …, J) and $\{h_{jkl}$ (k = 1, 2, …, J; l = 1, 2, …, L), both are binary variables. Such that $x_j = 1$ indicates that product R_j is selected, and $x_j = 0$ not. And $h_{jkl} = 1$ means that the lth attribute level of the kth attribute is contained in the jth product and $h_{jkl} = 0$ not. According to Jiao and Zhang (2005), economics surplus should be leveraged from both marketing and engineering. This paper proposes to use an objective of shared surplus to optimize the PPM problem, as shown in Eq. (1).

$$\max_{x_j,h_{jkl}} F = \sum_{i=1}^{I}\sum_{j=1}^{J}\frac{U_{ij}^F}{C_j^F}P_{ij}^F Q_i x_j \tag{1}$$

Assume U_{ij}^{F} is the utility of the ith segment for the jth product, which is a linear function of the part-worth utilities of the attribute levels of product R_j, as shown in Eq. (2).

$$U_{ij}^{F} = \sum_{k=1}^{K}\sum_{l=1}^{L_k} w_{jk}^{F}u_{ikl}^{F}h_{jkl} + \pi_{ij}^{F} + \varepsilon_{ij}^{F}, \quad \forall i, j \tag{2}$$

where u_{ikl}^{F} is the customer perceived utility of the ith segment for the lth level of the kth attribute; w_{jk}^{F} is the weight of the kth attribute; π_{ij}^{F} is a constant related to the composite utility of product variant R_j by the ith market; and ε_{ij}^{F} is an error term for each segment–product pair.

I adopt the SIMOPT model for cost calculation, which is proposed by Kwong et al. (2011). The cost function C_j^F can be formulated as follows:

$$C_j^F = \sum_{k=1}^{K}\sum_{l=1}^{L_k} c_{kl}^{F}h_{jkl}, \quad \forall j \tag{3}$$

where C_j^F is the cost of the jth product; c_{kl}^{F} is the variable unit cost for the lth attribute level of the kth attribute.

And according to the MNL model, the choice probability P_{ij}^{F} can be defined as Eq. (4), where μ is a scaling parameter. And Q_i is the size of the ith market segment.

$$P_{ij}^{F} = \frac{\exp\left(\mu U_{ij}^{F}\right)}{\sum_{t=1}^{T}\exp(\mu U_{it})} \tag{4}$$

One of the most important mechanism for this bi-level programming is the market share constraint, which is the interaction between leader and follower, as shown in Eq. (5).

$$\sum_{j=1}^{J} P_{ij}^{F}Q_i x_j \le Q_i - \sum_{m=1}^{M} P_{im}^{f}Q_i y_m, \quad \forall i = 1, 2, \ldots, I \tag{5}$$

The upper-level constraints are mainly related to choice compatibility. For example, one attribute only can be chosen one corresponding level in one product. Other special constraints may be introduced as well. For example, a differentiation condition constraint limits each product in the product line should be different from others.

Lower-level PPM optimization

The lower-level is optimization for the follower's PPM problem. The decision variables of the lower-level are also two binary choice variables y_m (m = 1, 2, …, M) and z_{mns} (n = 1, 2, …, N; s = 1, 2, …, S$_n$). Such that $y_m = 1$ means that product r_m is selected, and $y_m = 0$ not. While $z_{mns} = 1$ means that the sth attribute level of the nth attribute is contained in the mth product and $z_{mns} = 0$ not. Similarly to the leader, the follower's objective is the construction of shared surplus, as shown in Eq. (6).

$$\max_{y_m,z_{mns}} f = \sum_{i=1}^{I}\sum_{m=1}^{M}\frac{U_{im}^{f}}{C_m^{f}}P_{im}^{f}Q_i y_m \tag{6}$$

where U_{im}^{f} is the utility of the ith segment for the mth product; C_m^{f} is the cost of the mth product; P_{im}^{f} is the choice probability; and Q_i is the size of the ith market segment. The constitution of these terms are as same as shown in "Upper-level PPM optimization" section.

The lower-level constraints are mainly about choice compatibility. For example, one attribute only can be chosen one corresponding level in one product. Other special constraints may be introduced as well. For example, a differentiation condition constraint limits each product in the product line should be different from others. The most important choice constraint is from the upper-level, it forms the mechanism of bi-level programming, defined as Eq. (5).

Hierarchical joint optimization model

Compiling Eqs. (1)–(6), I can obtain the general form of hierarchical joint optimization for leader–follower PPM problem, as the following:

$$\max_{x_j,h_{jkl}} F = \sum_{i=1}^{I}\sum_{j=1}^{J}\frac{U_{ij}^{F}}{C_j^{F}}P_{ij}^{F}Q_i x_j \tag{7a}$$

$$\text{s.t.}\quad U_{ij}^{F} = \sum_{k=1}^{K}\sum_{l=1}^{L_k} w_{jk}^{F}u_{ikl}^{F}h_{jkl} + \pi_{ij}^{F} + \varepsilon_{ij}^{F}, \quad \forall i,j \tag{7b}$$

$$C_j^{F} = \sum_{k=1}^{K}\sum_{l=1}^{L_k} c_{kl}^{F}h_{jkl}, \quad \forall j \tag{7c}$$

$$P_{ij}^{F} = \frac{\exp\left(\mu U_{ij}^{F}\right)}{\sum_{t=1}^{T}\exp(\mu U_{it})} \tag{7d}$$

$$\sum_{l=1}^{L_k} h_{jkl} = 1, \quad \forall j,k \tag{7e}$$

$$\sum_{k=1}^{K}\sum_{l=1}^{L_k}(h_{jkl} - h_{j'kl}) \neq 0, \quad \forall j,j',j \neq j' \tag{7f}$$

$$\sum_{j=1}^{J} x_j \leq J^{+} \tag{7g}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J} P_{ij}^{F} Q_i x_j \le \sum_{i=1}^{I}\left(Q_i - \sum_{m=1}^{M} P_{im}^{f} Q_i y_m\right) \tag{7h}$$

$$x_j \in (0,1) \tag{7i}$$

$$h_{jkl} \in (0,1) \tag{7j}$$

$$\max_{y_m, z_{mns}} f = \sum_{i=1}^{I}\sum_{m=1}^{M} \frac{U_{im}^{f}}{C_m^{f}} P_{im}^{f} Q_i y_m \tag{7k}$$

$$\text{s.t.} \quad U_{im}^{f} = \sum_{n=1}^{N}\sum_{s=1}^{S_n} w_{mn}^{f} u_{mns}^{f} z_{mns} + \pi_{im}^{f} + \varepsilon_{im}^{f}, \quad \forall i, m \tag{7l}$$

$$C_m^{f} = \sum_{n=1}^{N}\sum_{s=1}^{S_n} c_{ns}^{f} z_{mns}, \quad \forall m \tag{7m}$$

$$P_{im}^{f} = \frac{\exp\left(\mu U_{im}^{f}\right)}{\sum_{t=1}^{T} \exp(\mu U_{it})} \tag{7n}$$

$$\sum_{s=1}^{S_n} z_{mns} = 1, \quad \forall m, n \tag{7o}$$

$$\sum_{n=1}^{N}\sum_{s=1}^{S_n} (z_{mns} - z_{m'ns}) \neq 0, \quad \forall m, m', m \neq m' \tag{7p}$$

$$\sum_{m=1}^{M} y_m \le M^{+} \tag{7q}$$

$$\sum_{i=1}^{I}\sum_{m=1}^{M} P_{im}^{f} Q_i y_m \le \sum_{i=1}^{I}(Q_i - \sum_{j=1}^{J} P_{ij}^{F} Q_i x_j) \tag{7r}$$

$$y_m \in (0,1) \tag{7s}$$

$$z_{mns} \in (0,1) \tag{7t}$$

The upper-level controls the lower-level through X and H, indicating a priority for determining PPM. The lower-level entails a nonlinear parametric programming problem, in which (*X, H*) are parametric variables, suggesting that parametric optimization must be compatible with its own PPM problem. The lower-level returns an optimum function $P^{f}_{im}(z_{mns})$ as the feedback to the upper-level. Optimal solution (X^*, H^*, Y^*, Z^*) describes the design schemes for leader and follower, where X^* and H^* indicate the leader's solutions and Y^*, Z^* are follower's.

Model solution

Solution of bi-level programming tends to be much more complex than single-level optimization problems (Christiansen et al. 2001; Bard 1998). Bi-level programming solutions are generally categorized as direct and indirect methods: solving the bi-level programming directly or converting the bi-level programming to a single-level programming. Direct methods comply with the bi-level decision mechanism, starting with the upper-level and find one solution first. Then the lower-level uses the corresponding upper-level variables as parameters to solve the lower-level problem.

Model (7) is a nonlinear programming with 0–1 variables, which is proved to be a NP-hard problem. And essentially this hierarchical product portfolio management problems entail combinational optimization that can be solved by genetic algorithms (Oliveto et al. 2007). Comparing with traditional calculus-based or approximation-based optimization techniques, GA is excellent in solving combinatorial optimization problems (Kreng and Lee 2004). Due to the distinctive characteristics of the upper- and lower-level optimization problems, I propose two separate GAs for the respective upper and lower levels, while the upper- and lower-level GAs are nested through a constraint handling mechanism embedded through GA encoding and GA operators. Specially, in the bi-level nested GA operations, I address two fitness functions which are consistent with their objective functions of upper- and lower-level, respectively. This kind of fitness function can identify the quality of different potential solutions.

GA encoding

The very basis of GA implementation is about representation of the problem to be solved with a finite-length string called a chromosome. Consistent with the bi-level decision-making of leader–follower PPM problem, two kinds of chromosomes are composed in this bi-level nested GA: the upper-level chromosomes (ULCs) and the lower-level chromosome (LLCs), While ULCs and LLCs represent solution *X*, *H* and *Y, Z*, respectively. As for the ULCs, the length of chromosome *X* is the total number of products *J*, and each gene in this chromosome represents a product; and the length of chromosome *H* is $J^*K^*L_k$, and each gene in this chromosome means an attribute level. Similarly, the length of chromosome *Y* is *M* and each gene means the choice of product; and the length of chromosome *Z* is $M^*N^*S_n$, and each gene represents an attribute level in lower-level. Figure 2 shows the bi-level nested GA encoding.

GA crossover and mutation

To obtain feasible solutions, chromosomes crossover and mutation should be occurred on the basis of satisfying certain design constraints. Crossover in GA operations means that optional two parent chromosomes are randomly chosen to exchange the segments of their genes with a probability. Figure 3a illustrates the crossover mechanism for the

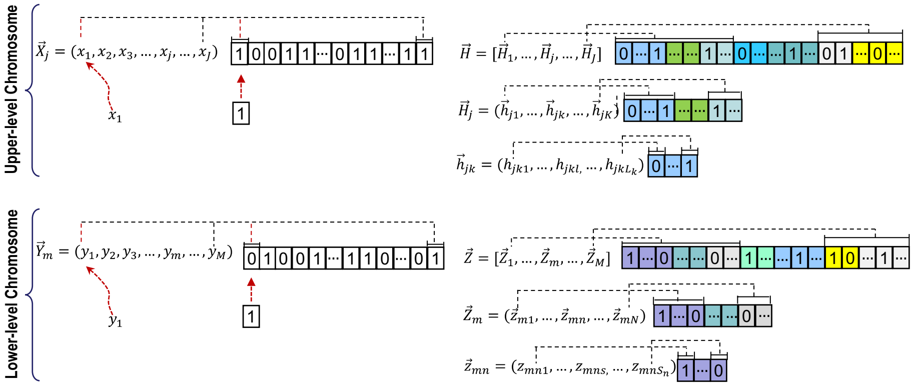
Upper-level Chromosome
$\vec{X}_j = (x_1, x_2, x_3, \ldots, x_j, \ldots, x_J)$
1 0 0 1 1 ··· 0 1 1 ··· 1 1
x_1
1
$\vec{H} = [\vec{H}_1, \ldots, \vec{H}_j, \ldots, \vec{H}_J]$
0 ··· 1 ··· ··· 1 ··· 0 ··· ··· 1 ··· 0 1 ··· 0 ···
$\vec{H}_j = (\vec{h}_{j1}, \ldots, \vec{h}_{jk}, \ldots, \vec{h}_{jK})$
0 ··· 1 ··· ··· 1 ···
$\vec{h}_{jk} = (h_{jk1}, \ldots, h_{jkl}, \ldots, h_{jkL_k})$
0 ··· 1
Lower-level Chromosome
$\vec{Y}_m = (y_1, y_2, y_3, \ldots, y_m, \ldots, y_M)$
0 1 0 0 1 ··· 1 1 0 ··· 0 1
y_1
1
$\vec{Z} = [\vec{Z}_1, \ldots, \vec{Z}_m, \ldots, \vec{Z}_M]$
1 ··· 0 ··· ··· 0 ··· 1 ··· ··· 1 ··· 1 0 ··· 1 ···
$\vec{Z}_m = (\vec{z}_{m1}, \ldots, \vec{z}_{mn}, \ldots, \vec{z}_{mN})$
1 ··· 0 ··· ··· 0 ···
$\vec{z}_{mn} = (z_{mn1}, \ldots, z_{mns}, \ldots, z_{mnS_n})$
1 ··· 0

Fig. 2 Bi-level nested GA encoding for this optimization

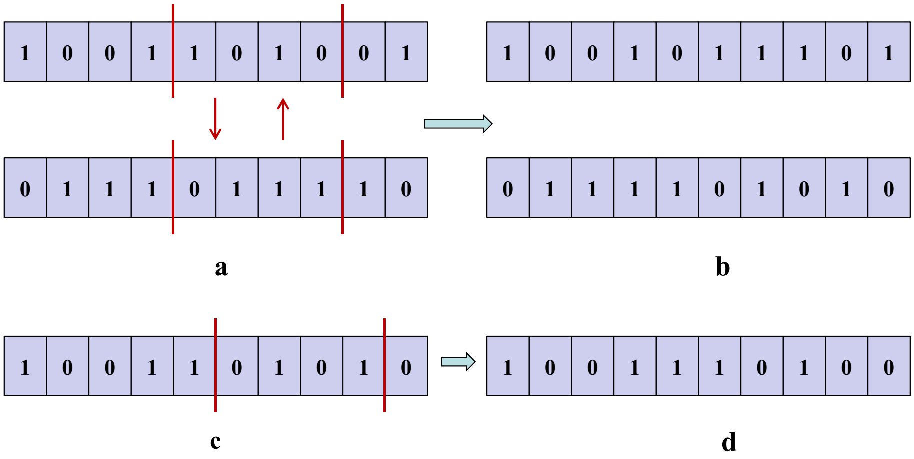

Fig. 3 Crossover and mutation for the ULC. **a** Choice of crossover points, **b** children by crossover, **c** choice of mutation points, **d** child by mutation

ULC X. Once a pair of ULC X chromosomes is chosen, two crossover points are generated randomly for each of them, and the exchange range is specified in the meantime. As shown in Fig. 3b, the children chromosomes are acquired by exchanging the parts of the selected chromosomes.

Mutation sockets in each offspring individually after crossover. It randomly selects a gene with a smaller probability and alters the selection or de-selection of corresponding module instances in a chromosome. Figure 3c shows that how the mutation points are generated randomly. Finally the selected chromosome segment is changed by a random number of modules, as shown in Fig. 3d.

GA nested solution flow

Solving nonlinear bi-level programming of PPM joint optimization relies on principles of evolutionary computation, videlicet the nested genetic evolutions between upper- and lower-level optimization. The bi-level GA with the aforementioned encoding schemas and operators is carried out in a nested way, as shown in Fig. 4. A step-by-step procedure for the bi-level nested algorithm is described as follows:

Step 1 A population of size *N* is generated with the required number of upper-level variables, randomly. Constraint checking is carried out to ensure that the initial population satisfies all design rules specified in the model. Subsequently, the lower-level GA procedure is launched to search the corresponding optimal values of the lower-level variables. If constraint checking fails, set the fitness values to be zero then turn to the next step.

Step 2 Judge whether the population reaches the maximum number of generation. If the GA runs for the maximal number of generations, record the optimal value and then go to the next step. Otherwise, GA operations of selection, crossover and mutation are carried out; and then go back to Step 1.

Step 3 Take the upper-level solution *X* and *H* into the lower-level GA. Verify feasibility of the lower-level population. The fitness is evaluated when the feasibility is satisfied. Otherwise, the fitness value is set to zero.

Step 4 Estimate whether lower-level population achieves the maximum number of generations. If it reaches the upper bound, the optimal variables and optimal value are recorded. Otherwise, procedures of selection, crossover and mutation are invoked and step 3 is repeated till the maximal number of generations is achieved.

Step 5 Echo the lower-level solution *Y* and *Z* back to the upper-level GA. The upper-level GA is re-calculated and the fitness values are re-evaluated. Repeat Step 4 until the lower-level population reaches the maximum number of generations.

Step 6 Check the termination condition of the lower-level GA. If a feasible number is determined for the entire modules, the upper- and lower-level optimal values are recorded and then the iterative procedure ends. Otherwise, proceed to Step 1 and repeat the process.

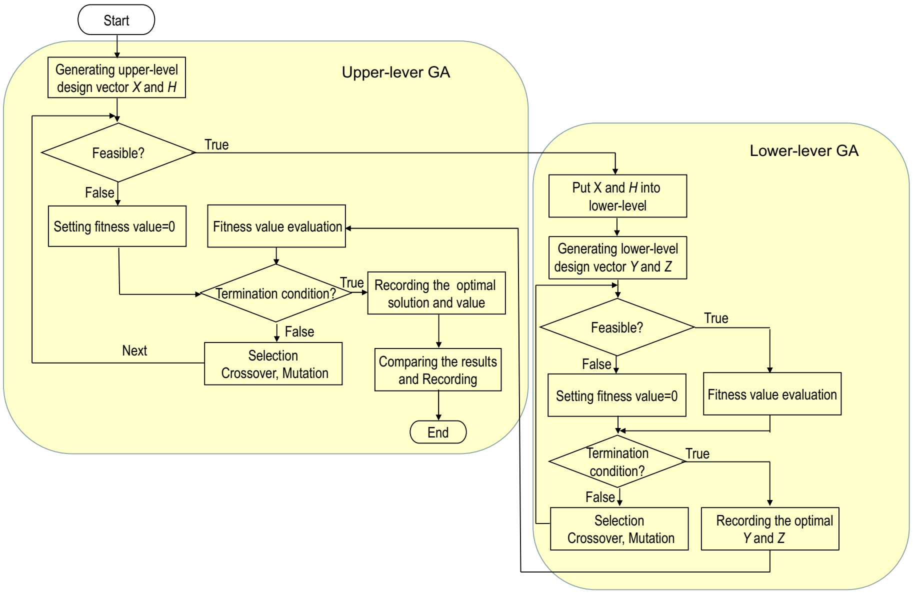
Start
Upper-lever GA
Generating upper-level design vector *X* and *H*
Feasible?
True
False
Setting fitness value=0
Fitness value evaluation
Termination condition?
True
Recording the optimal solution and value
False
Next
Selection Crossover, Mutation
Comparing the results and Recording
End
Lower-lever GA
Put X and *H* into lower-level
Generating lower-level design vector *Y* and *Z*
Feasible?
True
False
Setting fitness value=0
Fitness value evaluation
Termination condition?
True
False
Selection Crossover, Mutation
Recording the optimal *Y* and *Z*

Fig. 4 The solution flow of the bi-level nested GA

Case study

A case study of PPM optimization problem for the notebook computer products is reported to demonstrate the potential of the hierarchical joint optimization model. The product structure of notebook computer mainly consists of display, hard disk, graphics card, processer, battery and memory, etc.

Upper-level

Based on the service handbook and domain expertise, all the attributes and feasible attribute levels and their corresponding costs of notebook computer for leader are listed in Table 1.

To specify utility function U_{ij}^{F} in upper-level, w_{jk}^{F}, u_{ikl}^{F} and π_{ij}^{F} should be confirmed first. w_{jk}^{F} can be determined up to market survey; u_{ikl}^{F} can be obtained by conjoint analysis; and π_{ij}^{F} is formed from u_{ikl}^{F}. Conjoint analysis is a market survey method for product design. According to conjoint analysis method, the first step to calculate u_{ikl}^{F} is to given a set of product which contains the total number of J possible combinations. Then 20 orthogonal setup of product profiles can be generated using the Taguchi Orthogonal Array Selector in SPSS software, as shown in Table 2. While inviting 60 consumers to do a fractional factorial experiment with the 20 products. The result of conjoint analysis is shown in Table 3. For illustrative simplicity without losing the context, I consider one leading market for this leader–follower PPM problem ($I = 1$), also assume that $Q_i = 10{,}000$, and $\mathrm{J}^{+} = 4$.

Table 1 List of attributes and their levels of upper-level

No.	Attribute	Code	Attribute levels	Cost
A1	Display	A11	13.3″ HD	248
		A12	14.0″ HD	330
		A13	15.6″ HD	450
A2	Hard disk	A21	250 GB 5400 rpm hard drive	255
		…	…	…
		A25	128 GB solid state drive	580
A3	Graphics card	A31	Intel HD graphics	659
		…	…	…
		A35	NVIDIA GTX860 M 4 GB	2290
A4	Main board	A41	B85	890
		…	…	…
		A43	X99	2580
A5	Processer	A51	Intel core i5, 2 cores	1460
		…	…	…
		A55	Intel core i7, 6 cores	2855
A6	Battery	A61	6-Cell lithium-ion	440
		A62	9-Cell lithium-ion	635
A7	Memory	A71	4G DDR3	248
		A72	8G DDR3	380
A8	Case	A81	Metal	170
		A82	ABS	96

Table 2 Orthogonal setup of product profiles for conjoint analysis of upper-level

#	Display	Hard disk	Graphics card	Main board	Processer	Battery	Memory	Case
1	13.3″ HD	500 GB	Intel HD graphics	B85	Intel core i5, 2 cores	2–3 h	4G DDR3	ABS
2	14.0″ HD	500 GB	NVIDIA GTX540 M 2 GB	B85	Intel core i5, 2 cores	2–3 h	4G DDR3	ABS
…	…	…	…	…	…	…	…	…
20	15.6″ HD	128 G Solid	NVIDIA GTX860 M 4 GB	X99	Intel core i7, 6 cores	Above 5 h	8G DDR3	Metal

Table 3 Part-worth utilities of upper-level

Attribute	Code	Attribute levels	u_{ikl}^{F}
Display	A11	13.3″ HD	1.69
	A12	14.0″ HD	1.87
	A13	15.6″ HD	2.25
Hard disk	A21	250 GB 5400 rpm hard drive	1.45
	…	…	…
	A25	128 GB solid state drive	1.76
Graphics card	A31	Intel HD graphics	2.34
	…	…	…
	A35	NVIDIA GTX860 M 4 GB	3.19
Main board	A41	B85	1.92
	…	…	…
	A43	X99	3.40
Processer	A51	Intel core i5, 2 cores	2.58
	…	…	…
	A55	Intel core i7, 6 cores	4.23
Battery	A61	6-Cell lithium-ion	−0.82
	A62	9-Cell lithium-ion	1.58
Memory	A71	4G DDR3	2.23
	A72	8G DDR3	2.47
Case	A81	Metal	−0.57
	A82	ABS	1.61

Meanwhile, other compatibility problems, such as, the consistent matching of processer and memory, display and battery should be formulated as constraint equations, as the following:

$$h_{j55} \neq h_{j72} \tag{8}$$

$$h_{j13} = h_{j62} \tag{9}$$

Lower-level

The lower-level PPM optimization problem is as same as the upper-level, but the parameters of products are different for their diverse production capacity and market positioning. The attributes, attribute levels and the corresponding costs of products of the follower are shown in Table 4.

Table 4 List of attributes and their levels of lower-level

No.	Attribute	Code	Attribute levels	Cost
B1	Display	B11	13.3″ HD	276
		B12	14.0″ HD	360
		B13	15.6″ HD	510
B2	Hard disk	B21	250 GB 5400 rpm hard drive	294
		…	…	…
		B24	2T 7200 rpm hard drive	488
B3	Graphics card	B31	Intel HD graphics	718
		…	…	…
		B34	NVIDIA GTX860 M 4 GB	2440
B4	Main board	B41	B85	920
		…	…	…
		B43	X99	2710
B5	Processer	B51	Intel core i5, 2 cores	1730
		…	…	…
		B54	Intel core i7, 6 cores	3005
B6	Battery	B61	6-Cell lithium-ion	475
		B62	9-Cell lithium-ion	680
B7	Memory	B71	4G DDR3	266
		B72	8G DDR3	410
B8	Case	B81	Metal	191
		B82	ABS	104

And with the same conjoint analysis method in "Upper-level" section, I can obtain a set of 20 orthogonal setup of product profiles and the part-worth utilities, are separately shown as Tables 5 and 6.

And I also assume that the number of products $M^{+} = 4$. The very important constraint in the lower-level is from the upper-level. The constraint $\sum_{m=1}^{M} P_{im}^{f} Q_i y_m \leq Q_i - \sum_{j=1}^{J} P_{ij}^{F} Q_i x_j, \forall i = 1, 2, \ldots, I$ means that the market size for the follower is not the whole size of market segment but the rest part from the leader's market share.

Similarly, according to the practical situation of product technical design, other technical constraints, such as, attribute level B13 is not compatible with B61, can be given by Eqs. (10)–(12).

$$z_{m13} \neq z_{m61} \tag{10}$$

Table 5 Orthogonal setup of product profiles for conjoint analysis of lower-level

#	Display	Hard disk	Graphics card	Main board	Processer	Battery	Memory	Case
1	13.3″ HD	250 GB	Intel HD graphics	B85	Intel core i5, 2 cores	2–3 h	4G DDR3	ABS
2	14.0″ HD	500 GB	NVIDIA GTX540 M 2 GB	B85	Intel core i5, 4 cores	2–3 h	4G DDR3	ABS
…	…	…	…	…	…	…	…	…
20	15.6″ HD	2T	NVIDIA GTX860 M 4 GB	X99	Intel core i7, 4 cores	Above 5 h	8G DDR3	Metal

Table 6 Part-worth utilities of lower-level

Attribute	Code	Attribute levels	u_{ikl}^{F}
Display	B11	13.3″ HD	1.54
	B12	14.0″ HD	1.79
	B13	15.6″ HD	2.33
Hard disk	B21	250 GB 5400 rpm hard drive	1.62
	…	…	…
	B24	2T 7200 rpm hard drive	1.81
Graphics card	B31	Intel HD graphics	2.22
	…	…	…
	B34	NVIDIA GTX860 M 4 GB	2.99
Main board	B41	B85	1.47
	…	…	…
	B43	X99	2.69
Processer	B51	Intel core i5, 2 cores	2.33
	…	…	…
	B54	Intel core i7, 6 cores	3.83
Battery	B61	6-Cell lithium-ion	−0.82
	B62	9-Cell lithium-ion	1.58
Memory	B71	4G DDR3	2.85
	B72	8G DDR3	2.97
Case	B81	Metal	−0.89
	B82	ABS	1.22

$$z_{m24} \neq z_{m41} \tag{11}$$

$$z_{m33} \neq z_{m42} \tag{12}$$

Solution and results

I adopt a bi-level nested GA to solve this optimization model which is a nonlinear programming with 0–1 variables. For each feasible chromosome of upper-level population, the corresponding lower-level optimal value is sent back to the upper-level, then the upper-level fitness value is calculated, as shown in Fig. 4. According to Jiao (2007, 2013), for the more complex combination of attribute model, the fitness population size is 100 for GA.

For a medium-scale combinatorial configuration problem like this notebook computer product portfolio, I empirically set the population size at 100 for both the upper- and lower-level, crossover probability is 0.6, mutation probability is 0.005, and the generation is 200. Figure 5 shows the GA processes with the upper-level, after around 160 iterations the calculation verges to convergence. Similar outputs of the lower-level GA for PPM can be obtained as well.

Figure 6 shows how the upper- and lower-level GAs are nested during their convergence processes, indicating the leader's and follower's PPM problems are competing during the optimal design of notebook computers. Finally after around 160 generations, both the upper- and lower-level fitness values reach their optima, suggesting that the leader's and follower's PPM problems arrive at equilibrium solutions of the Stackelberg game. The optimal product portfolios for leader and follower are shown in Tables 7 and 8.

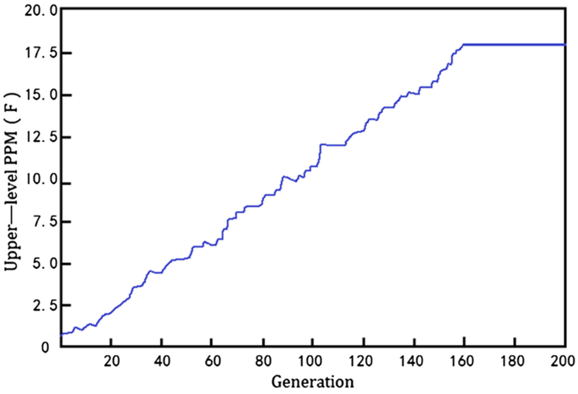

Fig. 5 Bi-level nested GA optimization process

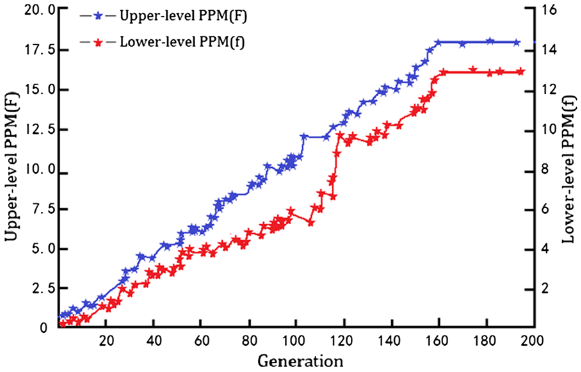

Fig. 6 Nested convergence of the upper- and lower-level

Table 7 The leader's optimal product portfolio

Attribute	Product variant			
	Product 1	Product 2	Product 3	Product 4
Display	13.3″ HD	14.0″ HD	15.6″ HD	15.6″ HD
Hard disk	1T 5400 rpm hard drive	1T 5400 rpm hard drive	2T 7200 rpm hard drive	128 GB solid state drive
Graphics card	Intel HD graphics	Intel HD graphics	NVIDIA GTX540 M 2 GB	NVIDIA GTX860 M 4 GB
Main board	B85	Z97	X99	X99
Processer	Intel core i5, 2 cores	Intel core i5, 2 cores	Intel core i7, 4 cores	Intel core i7, 6 cores
Battery	6-Cell lithium-ion	9-Cell lithium-ion	9-Cell lithium-ion	9-Cell lithium-ion
Memory	4G DDR3	4G DDR3	8G DDR3	8G DDR3
Case	ABS	Metal	Metal	Metal

Performance analysis

The optimal design

Examining the results of leader–follower PPM joint optimization in Tables 7 and 8 indicate that the optimal design of notebook computer product portfolios are logical and integrate as market-oriented products. The joint optimization results in Table 9

Table 8 The follower's optimal product portfolio

Attribute	Product variant			
	Product 1	Product 2	Product 3	Product 4
Display	13.3″ HD	13.3″ HD	14.0″ HD	14.0″ HD
Hard disk	500 GB 5400 rpm hard drive	1T 5400 rpm hard drive	2T 7200 rpm hard drive	2T 7200 rpm hard drive
Graphics card	Intel HD graphics	Intel HD graphics	NVIDIA GTX540 M 2 GB	NVIDIA GTX860 M 4 GB
Main board	B85	B85	Z97	X99
Processer	Intel core i5, 2 cores	Intel core i7, 4 cores	Intel Core i7, 4 cores	Intel core i7, 6 cores
Battery	6-Cell lithium-ion	6-Cell lithium-ion	9-Cell lithium-ion	9-Cell lithium-ion
Memory	4G DDR3	4G DDR3	8G DDR3	8G DDR3
Case	ABS	ABS	Metal	Metal

Table 9 Optimal values of the upper- and lower-level

Comparison	Evaluation criteria		
	Utility	Cost (RMB)	Objective function
Upper-level	68.84	20,781	17.73
Lower-level	60.59	21,524	12.89

indicate a reasonable leverage of leader and follower. I can also figure out that the nonlinear bi-level programming method can benefit both leader and follower well due to its decision-making mechanism by considering the market compititive factors and engineering parameters in PPM. Meanwhile this method can also gain a superior globle design scheme and it is better than non-joint one because this bi-level joint optimization correctly discribes the gaming relationship betweeen leader and follower in a market place.

The algorithms

To demonstrate the advantages of the nested GA, computational experiments are set up to compare its performance with other relevant algorithms. Related research has shown that genetic algorithm, simulated annealing algorithm and ATC methods perform well in product line design and product portfolio management problems (Balakrishnan et al. 2004; Belloni et al. 2008; Michalek et al. 2011). For the ATC method can only handles continuous variables, I just compare the former two methods for the scenario of $J = 4$, $M = 4$. And all the computations were conducted in Matlab on a Core i5 personal notebook computer.

Table 10 provides the results of comparison. I can figure out that no matter in terms of CPU time or Optimization objective, GA is much more efficient than SA method. These results suggest that, GA is more suitable for solving the PPM bi-level programming in this case.

Table 10 Algorithms comparisons for bi-level programming

Evaluation criteria	Comparison			
	GA		SA	
CPU time (h)	1.8		77.4	
Optimization objective	Upper-level	Lower-level	Upper-level	Lower-level
	17.73	12.89	16.99	12.01

Conclusions

PPM is different from the traditional single product or product line design problem, because it must handle a group of products with simultaneously optimizing the attributes of each product variant. The proposed nonlinear bi-level programming model which is applied to maximize the shared surplus of both leader and follower emphasizes the competitive relationships and tradeoffs between the leader and follower in leveraging their PPM problems. This method provides a new perspective to optimize PPM problem with different decision makers from both theory and practice. The bi-level nested genetic algorithm coincides with the bi-level decision structure of joint optimization. Encoding of attribute and product selection through a binary decision variable of each gene contributes to efficient solution of a class of non-linear combinatorial optimization programs. Results and analysis of the notebook computer case study demonstrate that the nonlinear bi-level optimization model is more reasonable and excellent in supporting leader–follower optimization for PPM problem and the proposed bi-level nested GA method is efficient for solving this challenging programming. Moreover, the introduced bi-level optimization framework and method can also be applied to other areas, such as, engineering design problem with multiple conflictive objects, supply chain coordination and optimization problem, and other optimization problems with game-theory relations.

Future research could focus on extending the bi-level programming model to multiple perspectives in PPM problem. With the considerations of the actual background in a competitive market, it could be a one-leader multi-follower bi-level programming for PPM problem. Solving such multi-follower model could be a very challenging issue that calls for both theoretical observation and efficient solution algorithms.

Acknowledgements
This research is supported by the "985 Project" of Beijing Institute of Technology. The author is looking forward to taking the opportunity to express his sincere gratitude to the anonymous referees for their valuable comments and suggestions that will be suggested.

Competing interests
The author declares that she has no competing interests.

Informed consent
Written informed consent was obtained from the patient for publication of this case report and any accompanying images. A copy of the written consent is available for review by the Editor-in-Chief of this journal.

Received: 25 January 2016 Accepted: 25 May 2016
Published online: 16 June 2016

References

Aust G, Buscher U (2012) Cooperative advertising and pricing decisions in a manufacturer–retailer supply chain: a game-theoretic approach. Eur J Oper Res 223(2):473–482

Bai C, Sarkis J (2013) A grey-based DEMATEL model for evaluating business process management critical success factors. Int J Prod Econ 146(1):281–292

Balakrishnan PV, Gupta R, Varghese SJ (2004) Development of hybrid genetic algorithms for product line designs. IEEE Trans Syst Man Cybern B Cybern 34(1):468–483

Barczak G, Griffin A, Kahn B (2009) Perspective: trends and drivers of success in NPD, practices: results of the 2003 PDMA best practices study. J Prod Innov Manag 26(1):3–23

Bard JF (1998) Practical bilevel optimization: algorithms and applications. Kluwer, Dordrecht

Belloni Alexandre, Robert F, Matthew S, Duncan S (2008) Optimizing product line designs: efficient methods and comparisons. Manag Sci 54(9):1544–1552

Bracken J and McGill JT (1973) The equivalence of two mathematical programs with optimization problems in the constraints. Oper Res 21(5):37–44

Chao RO, Kavadias S (2007) A theoretical framework for managing the NPD portfolio: when and how to use strategic buckets. Manag Sci 54(5):907–921

Chen J, Zhang H, Sun Y (2012) Implementing coordination contracts in a manufacturer Stackelberg dual-channel supply chain. Omega 40(5):571–583

Christiansen S, Patriksson M, Wynter L (2001) Stochastic bilevel programming in structural optimization. Struct Multidiscip Optim 21(5):361–371

Colson Benoît, Marcotte Patrice, Savard Gilles (2007) An overview of bilevel optimization. Ann Oper Res 153:235–256

Cooper RG, Edgett SJ, Kleinschmidt EJ (1999) New product portfolio management: practices and performance. J Prod Innov Manag 16(4):333–351

Cooper RG, Edgett SJ, Kleinschmidt EJ (2001) Portfolio management for new product development: results of an industry practices study. R&D Manag 31(4):361–380

Cooper RG, Edgett SJ, Kleinschmidt EJ (2002) Portfolio management: fundamental to new product success. Wiley, New York

Dickinson MW, Thornton AC, Graves S (2001) Technology portfolio management: optimizing interdependent projects over multiple time periods. Eng Manag 48(4):518–527

Du G, Wang JZ (2009) Product family hierarchical associated design and its hierarchical optimization. J Syst Sci Inf 7(3):245–252

Esmaeili M, Aryanezhad MB, Zeephongsekul PA (2009) Game theory approach in seller–buyer supply chain. Eur J Oper Res 195(2):442–448

Ho TH, Tang CS (1998) Product variety management: research advances. Kluwer, Boston

Jeroslow R (1985) The polynomial hierarchy and a simple model for competitive analysis. Math Program 32:146–164

Jiao J, Tseng MM (2013) On equilibrium solutions to joint optimization problems in engineering design. CIRP Ann Manuf Technol 62(1):155–158

Jiao J, Zhang Y (2005) Product portfolio planning with custome–engineering interaction. IIE Trans 37(9):801–814

Jiao J, Simpson TW, Siddique Z (2007) Product family design and platform-based product development: a state-of-the-art review. J Intell Manuf 18(1):5–29

Kreng VB, Lee TP (2004) Modular product design with grouping genetic algorithm—a case study. Comput Ind Eng 46(3):443–460

Kwong CK, Luo XG, Tang JF (2011) A multi-objective optimization approach for product line design. IEEE Trans Eng Manag 58(1):97–108

McNally RC, Durmusoglu SS, Calantone RJ, Harmancioglu N (2009) Exploring new product portfolio management decisions: the role of managers' dispositional traits. Ind Mark Manag 38(1):127–143

Michalek Fred M, Feinberg Peter E, Adigüzel F, Panos YP (2011) Enhancing marketing with engineering: optimal product line design for heterogeneous markets. Int J Res Mark 28(1):1–12

Oliveto PS, He J, Yao X (2007) Time complexity of evolutionary algorithms for combinatorial optimization: a decade of results. Int J Autom Comput 4(3):281–293

Otten S, Spruit M, Helms R (2015) Towards decision analytics in product portfolio management. Decis Anal 2(4):244–269

Pachidi S, Spruit M, Weerd I (2014) Understanding users' behavior with software operation data mining. Comput Hum Behav 30:583–594

Sadeghi A, Zandieh M (2011) A game theory-based model for product portfolio management in a competitive market. Expert Syst Appl 38:7919–7923

Sakawa M, Katagiri H, Matsui T (2011a) Interactive fuzzy random two-level linear programming through fractile criterion optimization. Math Comput Model 54(11–12):3153–3163

Sakawa M, Katagiri H, Matsui T (2011b) Interactive fuzzy random two-level linear programming through fractile criterion optimization. Math Comput Model 54(11–12):3153–3163

Tolonen A, Shahmarichatghieh M, Harkonen J, Harkonen H (2015) Product portfolio management—targets and key performance indicators for product portfolio renewal over life cycle. Int J Prod Econ 146(1):281–292

van Hoesel S (2008) An overview of Stackelberg pricing in networks. Eur J Oper Res 189(3):1393–1402

von Stackelberg H (1952) The theory of the market economy. Oxford University Press, Oxford

Xiao T, Choi TM, Cheng TCE (2014) Product variety and channel structure strategy for a retailer-Stackelberg supply chain. Eur J Oper Res 233(1):114–124

Ye JJ, Zhu DL (2010) New necessary optimality conditions for bilevel programs by combining MPEC and the value function approach. SIAM J Optim 20(4):1885–1905