Proceedings of the 2004 IEEE
Conference on Cybernetics and Intelligent Systems
Singapore, 1-3 December, 2004

A Heuristic Genetic Algorithm for Product Portfolio Planning

Jianxin Jiao
School of Mechanical and Production Engineering
Nanyang Technological University
Singapore
mjiao@ntu.edu.sg

Yiyang Zhang
School of Mechanical and Production Engineering
Nanyang Technological University
Singapore
Zhang_yi_yang@hotmail.com

Abstract—**For any manufacturing company, product portfolio planning constitutes one of the most important decisions regarding how to offer the "right" products to the target market. Essentially, such decisions exhibit a typical combinatorial optimization problem, which deems to be very complex and hard to solve using conventional optimization techniques. Enumeration is inhibitive if the problem size is extremely large. Genetic Algorithms (GAs) have been proven to excel in solving combinatorial optimization problems. This paper develops a heuristic GA to tackle the product portfolio planning problem.**

I. INTRODUCTION

Genetic algorithms (GAs) were first developed in John H. Holland's work in the early 70's. It is a highly parallel mathematical algorithm that transforms a set of individual mathematical objects, each with an associated fitness value, into a new population using operations patterned after the Darwinian principle of reproduction and survival of the fittest after naturally occurring genetic operations [1]. The genetic algorithm (GA) is one of the evolutionary algorithms, which is sparkled by the evolution in nature. Evolutionary algorithms are composed of four streams: Genetic algorithm, evolutionary planning, evolutionary strategy; and Genetic programming. The perspective of Evolutionary algorithms is that though the intelligence can't be planned directly, the evolution can be used to get intelligence indirectly.

For manufacturing companies, the most important thing is to provide the "right" product to the target market [2]. Under the production capability, manufacturing companies can provide different product alternatives which are composed of a bound of attributes with different levels. Since high product variety cause high cost, it is unwise to provide all the product alternatives within the capacity. The best strategy should be planned carefully to meet customers' requirements with cost as less as possible.

Such combinatorial optimization problems are very complex and hard to solve using conventional optimization techniques. Enumeration is inhibitive if the problem size is extremely large. Comparing with traditional calculus-based or approximation optimization techniques, the GAs have been proven to excel in solving combinatorial optimization problems [3]. The GA approach adopts a probabilistic search technique based on the principle of natural selection by survival of the fittest and merely uses objective function information, and thus is easily adjustable to different objectives with little algorithmic modification [4]. An important feature of the GA is that it allows product profiles to be constructed directly from attribute level part-worth data [5]. This is particularly preferable to reference set enumeration if the number of attributes and their levels is large and most multi-attribute products represented by different attribute level combinations are both economically and technologically feasible [6].

The GA approach to problem solving has some advantages over other optimization and search procedures. A GA is based on a population of points, instead of a single point (as with branch-and-bound and other techniques), which increases exploratory capability. Objective function information is used directly for evaluation, rather than derivatives used by gradient search techniques. Genetic algorithms evaluate specified candidate solutions completely versus building profiles one attribute at a time. Except for this, GAs work with a direct coding of parameters, rather than use the parameters themselves.

Towards this end, this paper develops a heuristic GA to tackle the product portfolio problem. A comprehensive methodology for product portfolio planning is developed, aiming at leveraging both customer and engineering concerns. The remainder of the paper proceeds as follows. In the next section, product portfolio planning problem is formulated. Section 3 presents the development of the mathematical model for the product portfolio planning problem. An optimization framework and the structural properties of the model are discussed in Section 4. A case study of notebook computer portfolio planning is reported in Section 5. The paper is concluded with a discussion of managerial implications in Section 6.

II. PROBLEM FORMULATION

A product, can be regarded as a bound of attributes, that is, $A \equiv \{a_k \mid k=1,\ldots,K\}$. Each attribute, $\forall a_k \in A$, possesses a few levels, i.e., $A_k^* \equiv \{a_{kl}^* \mid l=1,\cdots,L_k\}$.

A set of potential product profiles, $Z \equiv \{z_j \mid j=1,\cdots,J\}$, are generated by choosing one of the levels for certain attributes. Each product, is defined as a vector of specific attribute levels, i.e.,

$$\bar{z}_j = \left[a^{*}_{kl_j} \right]_K ,$$

where any $a^{*}_{kl_j} = \Phi$ indicates that product $\bar{z}_j$ does not contain attribute a_k. The best product combination is a set consisting of a few selected product profiles, i.e., $\Lambda = \{\bar{z}_j \mid j = 1,\cdots,J^{\dagger}\} \subseteq Z$, $\exists J^{\dagger} \in \{1,\cdots,J\}$, and thus denote the number of products contained in a product portfolio.

III. Model Development

The best product combination can be described as a mixed integer program, as below:

$$\textit{Maximize} \quad E[V] = \sum_{i=1}^{I}\sum_{j=1}^{J}\left(\frac{U_{ij}}{\beta e^{\frac{3\sigma_i^2}{\mu_i^2 - LSL^2}}}\right)\left(\frac{e^{\mu U_{ij}}}{\sum_{n=1}^{N} e^{\mu U_{in}}}\right)Q_i y_j , \tag{1}$$

$$U_{ij} = \sum_{k=1}^{K+1}\sum_{l=1}^{L_k}\left(w_{jk}u_{ikl}x_{jkl} + \pi_j\right) + \varepsilon_{ij} , \tag{2}$$

$$\sum_{l=1}^{L_k} x_{jkl} = 1 , \tag{3}$$

$$\sum_{k=1}^{K+1}\sum_{l=1}^{L_k}\left|x_{jkl} - x_{j'kl}\right| > 0 , \tag{4}$$

$$\sum_{j=1}^{J} y_j \le J^{\dagger} , \tag{5}$$

$$x_{jkl}, y_j \in \{0,1\}. \tag{6}$$

Equation (1) is to maximize the expected shared surplus by offering a product portfolio consisting of products, $\{\bar{z}_j\}_J$, to customer segments, $\{s_i\}_I$, each with size Q_i. Equation (2) refers to conjoint analysis – ensures that the composite utility of segment s_i for product $\bar{z}_j$ can be constructed from part-worth utilities of individual attribute levels, $\{A^{*}_k\}_{K+1}$. Equation (3) suggests an exclusiveness condition – enforces that exactly one and only one level of each attribute can be chosen for each product. Equation (4) denotes a divergence condition – requires that several products to be offered must pairwise differ in at least one attribute level. Equation (5) indicates a capacity condition – limits the maximal number of products that can be chosen by each segment. $J^{\dagger}$ is the upper bound on the number of products that the manufacturers want to introduce to a product portfolio. Equation (6) represents the binary restriction with regard to the decision variables of the optimization problem.

IV. Heuristic GA Procedure

Traditionally, the main procedure of GAs is divided into six steps: initialization, evaluation, selection, crossover, mutation, and replacement [7]. In this paper, developing a heuristic GA involves various steps. The proposed algorithm

2. A product portfolio is represented by a chromosome consisting of a string. Each fragment of the chromosome (i.e., substring) represents a product contained in the portfolio. Each element of the string, called gene, indicates an attribute of the product. The value assumed by a gene, called allele, represents an index of the attribute level instantiated by an attribute. A portfolio (chromosome) consists of one to many products (fragments of chromosome), exhibiting a type of composition (AND) relationships. Likewise, each product (fragment of chromosome) comprises one to many attributes (genes). Nevertheless, each attribute (gene) can assume one and only one out of many possible attribute levels (alleles), suggesting an exclusive all (XOR) instantiation.

The format of an allele may be either a binary or integer number [4]. The binary format is the most general form widely used for modeling the binary-selection type of problems [7]. In our case, each attribute (genes) may assume multiple levels (alleles), resulting in a multi-selection problem. Therefore, the integer format is adopted for representing multiple choices among attribute levels.

Given $J^{\dagger} \le J$ products to be selected for a product portfolio, and $K+1$ attributes in each product, $\bar{z}_j$, a generic string of the chromosome is defined to be composed of J substrings, with $J - J^{\dagger}$ empty substrings corresponding to those unselected products, and contains a total number of genes, with each substring consisting of $K+1$ genes.

Further introduce an allele equal to 0 as the default value for every gene. This indicates that the corresponding attribute is not contained in a product. Then with L_k possible levels for an attribute, a_k, the corresponding gene may assume an allele from the set, $\{0,1,\ldots,L_k\}$, meaning that a total number of $L_k + 1$ alleles are available for each gene. This corresponds to the fact that an attribute, a_k, may assume a de facto level, that is, $\exists a^{*}_{kl} \in \{\Phi, a^{*}_{k1}, \ldots a^{*}_{kl}, \ldots, a^{*}_{kL_k}\}$. If all genes throughout a substring assume $\{0\}_{K+1}$ alleles, then it means that the corresponding product is not selected in the portfolio. In this way, a chromosome enables a unified structure, through which various portfolios consisting of different numbers of products can be represented within a generic product portfolio, $\Lambda = \{\bar{z}_j\}_J$. Each individual portfolio can be instantiated from the same generic product portfolio by indirect identification of zero or non-zero alleles for all substrings.

For example, the chromosome shown in Fig. 2 suggests that product $\bar{z}_J$ is not selected for the portfolio (i.e., $y_J = 0$) as the corresponding substring is totally empty. As far as product $\bar{z}_1$ is concerned in Fig. 2 (i.e., $y_1 = 1$), the 1st allele assumes a value of 2 indicating that the 1st attribute of the product chooses the 2nd attribute level associated with this attribute (i.e., $x_{112} = 1$). The last ($K+1$) allele of the 1st substring suggests that the price attribute takes on the 3rd price level for product $\bar{z}_1$. On the other hand, the k-th allele assumes a value

ps that can be
on, constraint
termination.

folio planning
shown in Fig.

of 0, indicating that the k-th attribute is not contained in product $\bar{z}_1$.

B. Implementation Issues

1) Initialization: Initialization involves generating initial solutions to the problem. The initial solutions can be generated either randomly or using some heuristic methods. Considering

developed for this paper follows a total of step
divided into four basic points: initializatio
handling, reproduction, crossover, mutation, and

A. Generic Encoding

A generic strategy for encoding the portf
problem is illustrated in Fig. 1, with an example

the feasibility of product configurations, an initial population of product portfolios of size is determined. The population size, M , directly affects the computational efficiency of a GA. A larger population size gives the algorithm a higher chance of success by exploring a larger solution space; but leads to more calculations. While a standard value could be suggested by extensive experimentation, this research sets a default population size of 20 chromosomes.

2) Handling of Configuration Constraints: In order to obtain feasible solutions, each chromosome must satisfy certain configuration constraints on product generation from combinations of attribute levels. They constitute two types of constraints: compatibility constraints and selection constraints. Compatibility constraints refer to the restrictions on choices of attribute levels (e.g., size compatible) and are generally described as IF THEN rules. Selection constraints refer to those conjoint, exclusiveness, divergence and capacity conditions as postulated in Eqs. (1-6).

A number of methods of constraint handling have been reported in the literature, such as the repairing, variable restricting, penalizing, and modifying generic operator methods [7]. This research adopts a rejecting strategy. Whenever a new chromosome is generated, a constraint check is conducted with respect to all types of constraints, and only those valid ones are kept in the population.

[illegible]

parent selection and reproduction process. Parent selection is a process that allocates reproductive opportunities among chromosome population. A biased selection enables the convergence of the search; but sometimes may lead to premature convergence. From many selection rules, this research adopts the roulette wheel selection process. A reproduction probability is assigned to each chromosome based on its fitness value. Then the roulette wheel is filled using the respective cumulative probabilities of every chromosome. The areas of the sections on the wheel depend on the fitness values of the associated chromosomes, with fitter chromosomes occupying larger areas in this biased roulette wheel, thus increasing their chances of survival. The roulette wheel selection can be implemented by generating random numbers between 0 and 1 in accordance with the cumulative reproduction probabilities [8].

based on Pareto analysis of fitness values among all chromosomes in the population. Then among the short-listed chromosomes, a "balanced product portfolio" rule is applied according to [9]. A balanced product portfolio means that all products contribute evenly or nearly evenly to the shared surplus; otherwise an unbalanced product portfolio. In general, a balanced product portfolio is more preferable, as it tends to perform more stable when there exist unexpected changes in the market. An unbalanced product portfolio, to the contrary, may suffer significantly when market changes diminish the performance of one or two dominating products in the portfolio. To quantify the extent of a balanced distribution of products' individual contributions to the entire product portfolio, an unbalance index is defined as the following:

$$\psi = \sqrt{\sum_{j=1}^{M}\left(\frac{E[V_j]}{E[V]} - \frac{1}{M}\right)^2} \tag{7}$$

where M is the total number of products in a portfolio, E[V] is the expected shared surplus of product $\bar{z}_j$, and E[V] is the expected shared surplus of all products. In an absolutely balanced portfolio, the shared surplus of portfolio is evenly distributed among all products. Therefore, the lower the value of the unbalance index is, the more balanced is the distribution of shared surplus (fitness) among the products. In practice, the

[illegible]

from the first parent from the beginning till the crossover point; and then the rest is added by copying from the second parent from the crossover point to the end. The order of combination is reversed for the other offspring. In regard to the generic chromosome, there are $J\cdot(K+1)-1$ cutting points.

The probability of crossover is characterized by a crossover rate, indicating the percentage of chromosomes in each generation that experience crossover. Crossover aims at producing new chromosomes that possess good elements of old chromosomes. Nonetheless it is also desirable to allow some chromosomes, in particular those good ones, to survive without change in the next generation (namely elitism). Therefore, this research adopts a crossover rate of 0.60. In practice, this value could be selected based on sensitivity analysis of trial examples using crossover rates that range, for example, 0.05-0.95.

To account for the occurrence of elitism, this research develops a heuristic. First, top 20% chromosomes are selected

5)Mutation: Mutation is applied to each offspring individually after crossover. It randomly picks a gene within each string with a small probability, referred to as mutation rate, and alters the corresponding attribute level at random. This process enables a small amount of random search, and thus ensures that the GA search does not quickly converge at a local optimum. But it should not occur very often, otherwise the GA becomes a pure random search method. Empirical findings have suggested a mutation rate of 0.001 as a rule of thumb to obtain good solutions. While reproduction reduces the diversity of chromosomes in a population, mutation maintains a certain degree of heterogeneity of solutions which is necessary to avoid premature convergence of the GA process.

6)Termination: The processes of crossover and reproduction are repeated until the population converges or reaches a pre-specified number of generations. A maximal generation number can be set ex ante at a large number. However, the algorithm may have found a solution before this number is ever reached. Then extra computations may have to be performed even after the solution has been found. Balakrishnan and Jacob have shown a moving average rule that can provide a good indication of convergence to a solution [10]. More specifically, the GA process terminates if the average fitness of the best three strings of the current generation has increased by less than a threshold, namely convergence rate, as compared with the average fitness of the best three strings over three immediate previous generations.

To leverage possible problems of termination by either convergence or maximal number of generations alone, this research adopts a two-step stopping rule to incorporate both. A moving average rule is used for the first stopping check. The convergence rate is set at 0.1%. In practice, this value could be determined based on sensitivity analysis of trial examples according to the particular problem context. Then a maximal number of generations is specified as the criterion for the second stopping check. In our case, a number of 1000 is used. Similarly, this value could be determined based on trial runs in

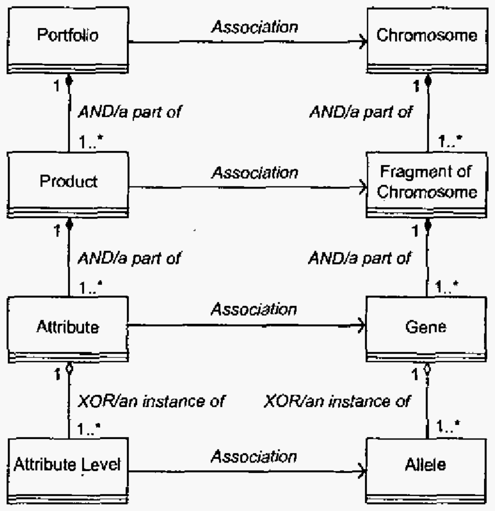

Figure 1. Generic enncoding of product portfolio

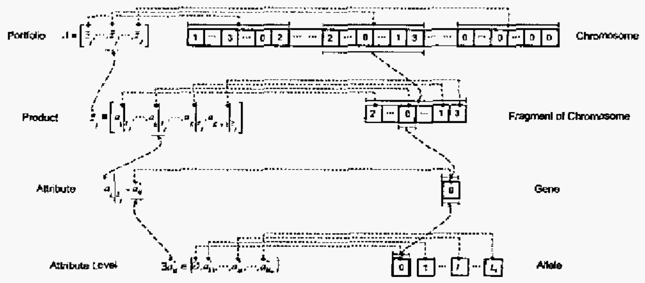

Figure 2. An illustration of generic encoding

V. Case Study

The proposed framework has been applied to the notebook computer portfolio planning problem for a world-leading computer manufacturing company. The company had conducted extensive market studies and competition analyses and projected the trends of technology development in the part-worth utility for each attribute level thus measure the product's market potential. The part-worth utility is listed in Tab 2 together with the part-worth standard time.

To determine an optimal notebook computer portfolio for the target three segments, the GA procedure is applied to search for a maximum of expected shared surplus among all attribute, product and portfolio alternatives. Assume that each portfolio may consist of a maximal number of $J^{\dagger} = 5$ products. Then a chromosome string comprises $9 \times 5 = 45$ genes. Each substring is as long as 9 genes and represents a product that constitutes the portfolio.

TABLE I. LIST OF ATTRIBUTES AND THE FEASIBLE LEVELS FOR NOTEBOOK COMPUTERS

a_i	Description	Code	Description
a_1	Processor	A1-1	Pentium 2.4 GHz
		A1-2	Pentium 2.6 GHz
		A1-3	Pentium 2.8 GHz
		A1-4	Centrino 1.4 GHz
		A1-5	Centrino 1.5 GHz
		A1-6	Centrino 1.6 GHz
		A1-7	Centrino 1.7 GHz
		A1-8	Centrino 1.8 GHz
		A1-9	Centrino 2.0 GHz
a_2	Display	A2-1	12.1" TFT XGA
		A2-2	14.1" TFT SXGA
		A2-3	15.4" TFT XGA/UXGA
a_3	Memory	A3-1	128 MB DDR SDRAM
		A3-2	256 MB DDR SDRAM
		A3-3	512 MB DDR SDRAM
		A3-4	1 GB DDR SDRAM
a_4	Hard Disk	A4-1	40 GB
		A4-2	60 GB
		A4-3	80 GB
		A4-4	120 GB
a_5	Disk Drive	A5-1	CD-ROM
		A5-2	CD-RW
		A5-3	DVD/CD-RW Combo
a_6	Weight	A6-1	Low (below 2.0 KG with battery)
		A6-2	Moderate (2.0 - 2.8 KG with battery)
		A6-3	High (2.8 KG above with battery)
a_7	Battery Life	A7-1	Regular (around 6 hours)
		A7-2	Long (7.5 hours above)
a_8	Software	A8-1	Multimedia package
		A8-2	Office package
a_9	Price	A9-1	Less than \$800
		A9-2	\$800 - \$1.3K
		A9-3	\$1.3K - \$1.8K
		A9-4	\$1.8K - \$2.5K
		A9-5	\$2.5K above

Through constraint check, only valid chromosomes are passed on for further evaluation. For every generation, a population size of M=20 is maintained, meaning that only top 20 fit product portfolios are kept for reproduction. The results of GA solution are presented in Fig. 3. As shown in Fig. 3, certain local optima are successfully overcome. The saturation period (350-500 generations) is quite short, indicating the GA search is efficient. This proves that the moving average rule is a reasonable convergence measure. It helps avoid such a possible problem that the GA procedure may run unnecessarily as long as 1000 generations.

Fig. 4 presents the results from two separate GA runs: one with elitism and the other without elitism. Their achievements of shared surplus are compared based on the top five portfolios in the respective final generations. In terms of the expected shared surplus, the solution with elitism overall yields a better performance than that without elitism (maximum 802K vs. 780K). In regard to the unbalance index measure, the solution with elitism produces an overall healthier structure.

TABLE II. PART –WORTH UTILITY AND PART-WORTH STANDARD TIME

Level	Part-worth Utility (Customer Segment)			Part-worth Standard Time (Assembly & Testing Operations)	
	s_1	s_2	s_3	μ^t (second)	σ^t (second)
A1-1	0.75	0.65	0.62	497	9.5
A1-2	0.77	0.83	0.82	536	11
A1-3	0.81	0.78	1.18	563	12
A1-4	0.74	0.66	0.61	512	10.5
A1-5	0.77	0.86	0.89	556	11.8
A1-6	0.78	0.77	1.16	589	21
A1-7	0.81	0.79	1.18	598	21.1
A1-8	0.83	0.82	1.21	615	22.3
A1-9	0.84	0.85	1.22	637	24
A2-1	1.18	1.05	0.75	739	35
A2-2	1.21	1.47	1.18	819	37
A2-3	1.25	1.49	1.38	836	39
A3-1	1.02	0.5	0.4	659	24.5
A3-2	1.09	0.9	0.65	699	26.5
A3-3	1.12	1.15	0.93	725	32
A3-4	1.14	1.18	1.11	756	36
A4-1	1.33	0.97	0.63	641	26
A4-2	1.38	1.08	0.78	668	28
A4-3	1.52	1.13	1.08	707	29
A4-4	1.56	1.19	1.22	865	40
A5-1	0.86	0.93	0.78	293	4.4
A5-2	0.88	1.11	0.82	321	5.1
A5-3	0.92	1.35	0.83	368	5.5
A6-1	0.7	0.2	0.3	215	3.8
A6-2	0.9	0.7	0.8	256	4.0
A6-3	1.1	0.9	0.9	285	4.1
A7-1	0.7	0.6	0.3	125	1.6
A7-2	0.8	0.9	1.2	458	19.1
A8-1	1.2	1.1	1.2	115	1.55
A8-2	0.5	0.8	1.0	68	0.95
A9-1	0	0	0	0	0
A9-2	-1.75	-0.35	-0.2	-1.75	-0.35
A9-3	-2.25	-0.65	-0.47	-2.25	-0.65
A9-4	-2.75	-2.48	-0.6	-2.75	-2.48
A9-5	-3.5	-3.3	-0.95	-3.5	-3.3

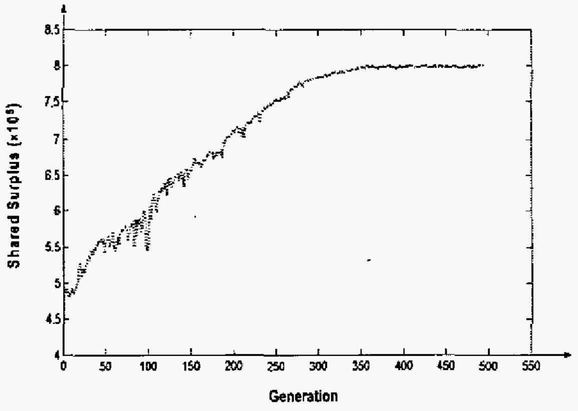

Figure 3. Results of GA solution

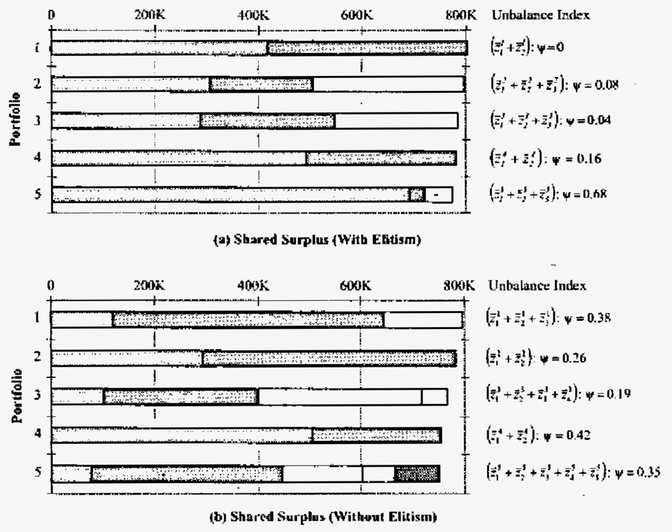

Figure 4. Comparison of GA solution with and without elitism

VI. Conclusions

Product portfolio planning differs from the conventional product line design problem in that it must not only optimize a mix of products but also in the meantime optimize the configurations of individual products in terms of specific attributes. This research develops shared surplus model accounting for both diverse customer preferences across market segments, and engineering costs that vary with the composition of a product portfolio. By integrating marketing inputs with detailed cost information attained through coordinated product and process platforms, the model captures the tradeoffs between the benefits derived from providing variety to the marketplace, and the cost savings that can be realized by selecting a mix of products that can be produced efficiently within a company's manufacturing capabilities.

penalties on those high cost-carrying attribute levels. This may be implemented as a hybrid constraint handling strategy during the GA procedure.

As witnessed in the case study, the strength of GA lies in the ability to carry out repeated runs without major changes of parameter values or defining different initial populations, thus improving the chance of finding an optimal or at least a near optimal solution. It is also possible to insert solutions obtained from other techniques into the initial population. Hence, rather than generating all the members of the initial population at random, the GA can use a prior knowledge about potential optima to arrange the initial population or improve on an existing solution that can performs as a kind of lower bound or benchmark for GA performance.

In the case study, we have tested the performance of the proposed heuristic GA on a small to medium sized problem. Although we do not know how well the heuristic GA would perform on large problems in an absolute sense, we believe, given the popularity of high performance computation, what important is not the computational efficiency of GA, but the quality of solutions returned by the GA procedure. From the experience of using the product portfolio balance rule, it is obvious that the heuristic GA also provides high flexibility with regard to the final decision making of a product portfolio. For example, the decision maker may be provided with quite a number of solutions using similar high fitness values that are of his expectations. In this way, the decision maker can instrument additional fitness criteria as appropriate for selecting the best product portfolio.

References

[1] X. Wang, and L. Cao, *Genetic Algorithm: Theory, application and software realization*, Xi'an Jiaotong University Press, Xi'an, 1997.

[2] B.D. Henderson, *The Product Portfolio*, Boston Consulting Group, Boston, MA, 1970.

[3] [illegible]

W.J. Steiner and H. Hruschka, "A probabilistic one-step approach to the optimal product line design problem using conjoint and cost data," unpublished.

.H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, 1992.

R. Kohli and R. Krishnamurti, "A heuristic approach to product design," *ement Science*, vol. 33, pp. 1523-1533, 1987.

air, L.S. Thakur, and K. Wen, "Near optimal solutions for product sign and selection: beam search heuristics," *Management Science*, , pp. 767-785, 1995.

en, and R. Cheng, *Genetic Algorithm and Engineering zation*, John Wiley and Sons, New York, 2000. R. Kohli and R. amurti, "A heuristic approach to product design," *Management* , vol. 33, pp. 1523-1533, 1987.

Obitko, 2003, *Introduction to Genetic Algorithms*, abe.felk.cvut.cz/~obitko/ga/, 2003.

and S. Azarm, 2002, "An approach for product line design n under uncertainty and competition," *Transactions of the ASME, l of Mechanical Design*, vol. 124, pp. 385-392, 2002., 2003, *ction to Genetic Algorithms*, http://labe.felk.cvut.cz/~obitko/ga/,

Balakrishnan, and V.S. Jacob, "Genetic algorithms for product " *Management Science*, vol. 42, pp. 1105-1117, 1996.

A heuristic Genetic algorithm is developed and applied to solve the combinatorial optimization problem involved in product portfolio planning. The study indicates that the GA works efficiently in searching for optimal product portfolio [illegible]
problem of introducing a new product portfolio with the objective of maximal shared surplus, the proposed framework could easily be adjusted to handle such complex problems as maximizing share-of-choices and extending an existing product portfolio by allowing for already existing items to be owned by the seller. This is supported by the flexibility of the GA procedure that merely uses objective function information, and therefore is capable of accommodating different fitness criteria without any substantial modification of the algorithm.

If interactions between attributes are to be considered, the additive main-effect utility model could be easily extended to include interaction terms without introducing additional decision variables. The interaction terms merely affect a string's fitness evaluation by taking into account the associated part-worth utilities for preference evaluation. A possible means to deal with technologically infeasible configurations of attribute levels is to incorporate related interaction terms into the cost functions and to impose certain

[3] V
c
u

[4] J
C

[5] F
Manag

[6] S.K. N
line des
vol. 41,

[7] M. G
Optimi
Krishna
Science

[8] M. C
http://la

[9] H. Li,
selectic
Journa
Introdu
2003.

[10] P.V.S.
design,