Development of Hybrid Genetic Algorithms for Product Line Designs

P. V. Balakrishnan, Rakesh Gupta, *Member, IEEE*, and Varghese S. Jacob, *Member, IEEE*

***Abstract*—In this paper, we investigate the efficacy of artificial intelligence (AI) based meta-heuristic techniques namely genetic algorithms (GAs), for the product line design problem. This work extends previously developed methods for the single product design problem. We conduct a large scale simulation study to determine the effectiveness of such an AI based technique for providing good solutions and bench mark the performance of this against the current dominant approach of beam search (BS). We investigate the potential advantages of pursuing the avenue of developing hybrid models and then implement and study such hybrid models using two very distinct approaches: namely, seeding the initial GA population with the BS solution, and employing the BS solution as part of the GA operator's process. We go on to examine the impact of two alternate string representation formats on the quality of the solutions obtained by the above proposed techniques. We also explicitly investigate a critical managerial factor of attribute importance in terms of its impact on the solutions obtained by the alternate modeling procedures. The alternate techniques are then evaluated, using statistical analysis of variance, on a fairy large number of data sets, as to the quality of the solutions obtained with respect to the state-of-the-art benchmark and in terms of their ability to provide multiple, unique product line options.**

***Index Terms*—AI, beam search, GA, hybrid genetic algorithms, meta-heuristic techniques.**

I. Introduction

AMONG THE very important problems that managers of multiproduct firms have to deal with is the issue of designing and selecting a set of products to fill a product line. Such problems are important in an increasing number of industries wherein a menu of choices is provided from which consumers can select a product whether it be in the arena of health insurance, cell phone plans, pizzas, or automobiles. In such hyper-competitive industries, modifications to the design and appropriate selection of items in the product lines have an enormous impact on the revenues and market shares of the firms. Given the increasing nature of global competition, it is not surprising to see the firms battling for even marginal increases in market share as the consequence of even small improvements translate into major revenue streams.

A market research approach that has been employed with some success in attempting to address this issue derives from the conjoint analysis framework. There have also been attempts to design user friendly systems to assist managers using market research data to help deal with the product line design problems [1]–[3]. The development of such systems to address particular tasks have focused on the use of a specific modeling approach and in providing a specific recommendation (see for example [4]–[6]). The "single best" approach, therefore, in terms of both an analytical technique as well as in terms of the "solution," denied the managers a preferred list of solutions to discuss and choose from, but rather tends to have one imposed on them. To address the above two concerns of having greater confidence in and usage of such models by managers, it is preferable that the decision aids provide a "list" of good solutions and provide bench marks for comparative purposes from established modeling techniques [1], [7].

Artificial intelligence (AI) techniques have been proposed as alternative solution approaches to mathematical programming procedures [8]–[16] to overcome some of the difficulties faced by methods employing traditional mathematical programming techniques. Expert system methodologies or machine learning techniques such as genetic algorithms (GA) or neural networks are often used to provide "good" solutions to these problems. Researchers have also begun to consider combinations of these techniques [11], [17], [18] with varying degrees of success.

This research extends prior work due to [1], [19] that focused on developing near optimal solutions to the single product design problems within the conjoint analysis framework [20]. In this paper, we go well beyond the previous work, by proposing, developing and comparing different techniques for product line designs, as well as by investigating the potential benefits of developing hybrid techniques to intermarry the best aspects of each of these different procedures.

More specifically, in this paper, we investigate the efficacy of an AI based meta-heuristic technique namely GA for providing good solutions. Toward this end, we bench mark the performance of this against the current dominant and maximally different approach of beam search (BS) proposed by [21] which in turn was based on the earlier dynamic programming (DP) heuristics of Kohli and Sukumar [22]. We further investigate, whether it is fruitful to pursue the avenue of developing hybrid models that might encompass the best features of the two approaches but could ostensibly suffer from greater development effort. We then propose, implement and study two types of hybrid models using very distinct approaches: namely, seeding the initial GA population with the BS solution, and employing the BS solution as part of the GA operator's process.

Further, given the potential susceptibility for inefficiency in search due to many possible permutations of product line strings, we go on to examine the impact of two alternate string

Manuscript received August 24, 2001; revised November 6, 2002. This paper was recommended by Associate Editor H. Takagi.

P. V. Balakrishnan is with the University of Washington, Bothell, WA 98011 USA (email: sundar@u.washington.edu).

R. Gupta and V. S. Jacob are with the University of Texas at Dallas, Richardson, TX 75083 USA (email: rgupta@utdallas.edu; vjacob@utdallas.edu).

Digital Object Identifier 10.1109/TSMCB.2003.817051

representation formats[1] on the quality of the solutions obtained by the above proposed techniques. Given managers' natural interest on the issue of attribute importance [23] we explicitly investigated this factor in terms of its impact on the solutions obtained by the alternate modeling procedures. Finally, the alternate techniques are then evaluated, using statistical analysis of variance and process tracing charts, on a fairly large number of data sets, not only to assess the quality of the solutions obtained with respect to the state-of-the-art benchmark but also to evaluate their ability to provide multiple, unique options.

The paper is structured as follows: in the next section we briefly describe the nature of the product line problem and its applications. In Section III we describe in detail the various approaches to solving this problem and some limited computational comparisons with standard mathematical programming procedures using CPLEX for a restricted set of product line instances. Section IV provides computational results of all of the proposed heuristic solution approaches on a large number of "realistically sized" problems and compares their performance and develops statistical inferences from the results. The conclusions, discussions and directions for future work wrap up the final section.

II. Product Line Problem

Within the context of product line design we focus on the share-of-choices problem. Similar to [21] and [22] we assume that the utility for a particular level of an attribute for a set of buyers/consumers has been determined using conjoint or hybrid conjoint analysis through market research. In standard conjoint analysis, relevant attributes are selected and each is assigned a discrete number of levels. Next, scaled preference data is obtained (from buyers/consumers) for complete product profiles which are defined within a fractional factorial design. From this data it is possible to compute individual part-worth values for each attribute level, thereby giving us the utility function of each consumer. For a good review of these techniques the reader is directed to [24]. We assume also, that each consumer has a current favorite (i.e., "status-quo") brand in the market place. Using this information, we can construct a "relative part-worths" utility matrix for each consumer. Each element of this matrix would correspond to the difference between the consumer's utility for a particular level of an attribute and the utility to the consumer by the level of that attribute in the consumer's status-quo brand. The goal then is to design a line (i.e. a set) of products that will maximize the product share for the firm, i.e., maximize the number of customers who switch to one of our new products away from their status-quo product.

The product line problem is clearly a significant one for firms having multiple items in the same product category. For instance, wireless phone companies offer a number of different plan options, Knorrs makes several flavors of soups, Levers markets a line of bar soaps, GM makes different types of cars, etc. The goals here range from the desire to maximize the market share to maximizing consumer welfare for nonprofit organizations. Clearly, costs associated with design and marketing of the products can be astronomical and hence failure of the product in the marketplace can be devastating to a firm. Thus it is critical to provide the decision-maker with tools to analyze the problem.

Since the product line design problem is NP-hard [22], for most problem instances of interest it is not practical to obtain optimal solutions in reasonable amounts of time. For instance, in the case of a product category with 8 attributes with 5 levels each, the total number of possible design configurations total 390 625. This makes the managerial task of selecting even, say, three products from these possible designs for our Product Line combinatorially more complex as there are now over 10^{16} candidate solutions. Consequently, a number of heuristic solution techniques have, therefore, been proposed and discussed to help solve this and variations of the more general problem ([2], [21], [25]–[28]).

The current state of the art search technique is one based on beam search and has been developed by [21]. Prior to the development of beam search based techniques for product line design, the best available technique was a dynamic programming approach which was suggested by Kohli and Sukumar [22]. These search techniques (such as beam search) suffer from limitations due to the fact that they are deterministic in nature. For example, Nair *et al.* [21] present at least one pathological set (additional ones can be created) of relative part-worth's for which the beam search heuristic will always result in an arbitrarily bad solution. The dynamic programming heuristic due to [22] suffers from similar issues. These issues as will be explained later, are not a major problem for genetic search based techniques such as those we develop. In addition, previous researchers have explored the use of Genetic Algorithms for a variant of the product line design problem which take into account pricing information for each product and budget constraints for each consumer [29]. The solution techniques presented by [29] included beam search (which we describe in the next section) and genetic algorithms. Brief descriptions of the techniques directly relevant to our interest are provided in the next section.

III. Solution Approaches to the Product Line Design Problem

This section briefly describes and examines different solution approaches to the Product Line Design problem. We first present the problem as a 0-1 integer program and discuss the implications of using a traditional integer program solver to computationally solve it. We also describe two heuristic solution approaches, the first is a beam search based heuristic due to [21] and the second is the genetic algorithm based technique that we have developed. Further, we discuss the applicability of developing hybridized techniques that utilize both of the previously mentioned heuristics to solve the same problem. We conclude this section with a computational comparison of all four of these approaches on a restricted set of problems to gain some experience with the relative ability of the proposed approaches to tackle bigger problem instances.

A. Traditional Mathematical Programming Formulation

We utilize the 0-1 integer programming formulation for the product line design problem provided by [14] shown as follows.

[1]We thank the reviewers for the suggestions that led us to this line of investigation.

a) *Parameters:*

$\Omega = \{1, 2, \ldots K\}$ set of K attributes.

$\varphi_k = \{1, 2, \ldots J_k\}$ set of J_k levels for attributes k.

$\Psi = \{1, 2, \ldots M\}$ set of M items to be selected (each item is a product).

$\theta = \{1, 2, \ldots I\}$ set of I consumers or buyers.

W_{ijk} = part worth utility of level $j \in \varphi_k$ of attribute $k \in \Omega$ for consumer $i \in \theta$.

${j^*}_{ki}$ = level of attribute $k \in \Omega$ that appears in status quo product for buyers $i \in \theta$.

$\beta_{ijk} = W_{ijk} - W_{i{j^*}_{ki}k}$ part worth utility of level j of attribute k relative to part worth of level j^*_{ki}.

$\theta' \subset \theta$ set of buyers whose status quo product is not offered by seller (i.e., currently buying a competing product).

b) *Decision Variables:*

$$x_{jkm} = \begin{cases} 1, & \text{if level } j \in \varphi_k \text{ is chosen for attributes k} \\ & \text{ in item } m \in \Psi \\ 0, & \text{otherwise} \end{cases}$$

$$x_{im} = \begin{cases} 1, & \text{if item } m \text{ provides } i \text{ with utility} \leq \\ & \text{ status quo product} \\ 0, & \text{otherwise} \end{cases}$$

$$x_i = \begin{cases} 1, & \text{if customer } i \text{ does choose to switch from} \\ & \text{ status quo product} \\ 0, & \text{otherwise.} \end{cases}$$

Problem P:

$$Maximize \frac{1}{|\theta'|} \sum_{i \in \theta'} x_i \tag{1}$$

$$s.t:$$

$$\sum_{j \in \varphi_k} x_{jkm} = 1 \; k \in \Omega, m \in \Psi \tag{2}$$

$$\sum_{k \in \Omega} \sum_{j \in \varphi_k} \beta_{ijk} x_{jkm} + x_{im} > 0 \quad i \in \theta', m \in \Psi \tag{3}$$

$$x_i \leq M - \sum_{m \in \Psi} x_{im} \qquad \forall \quad i \in \theta' \tag{4}$$

$$x_{jkm}, x_{im}, x_i = 0, 1 \text{ and integer } i \in \theta', j \in \varphi_k, \; k \in \Omega, m \in \Psi. \tag{5}$$

The objective function (1) in Problem P maximizes the fraction of consumers that choose to adopt a product other than their status quo (this is equivalent to minimizing the number of customers who choose to stay with their status quo product). Constraint (2) requires that there can be at most one level of each attribute in each product (or item) in the product line. Constraint set (3) restricts x_{im} to be 1 if item m provides customer i with as much or less utility than his/her status quo product. Constraints (4) require x_i to be 0 if none of the selected items (or products) provide higher utility than the status quo utility of consumer i. The final set of constraints (5), enforce the integrality and binary nature of the decision variables.

Having formulated the integer program P for a particular instance of the product line design problem, we can utilize any commercial integer program solver such as CPLEX, LINDO, GAMS etc. to solve it. The majority of these solvers utilize variations of the Branch and Bound approach to solving the resulting integer program. The difficulties faced with using such approaches for NP-Complete or NP-Hard problems are well known [30]. At the end of this section we provide details of our computational experience in solving a few product line design problems while using CPLEX as a solver.

B. *Beam Search Based Approach*

Nair *et al.* [21] have suggested a beam search (BS) based approach to solving the product line design problem. Specifically, if we consider M products and K attributes with J_k possible levels for attribute $k \in K$, then the beam search approach utilizes K relative part-worths matrices (i.e. the β matrix from the previous formulation), and initializes work matrices $\mathrm{A}_l(\bullet)$ for each stage (termed as a "layer") l of the search. Each row of these work matrices correspond to different consumers, while each column corresponds to a partial product profile. For the first stage $(l = 1)\mathrm{A}_1(k) = \beta_k$ for each attribute $k \in K$. This approach proceeds by iteratively combining the work matrices $\mathrm{A}_l(\bullet)$ taking the matrices two at a time and forming matrices $\mathrm{E}_l(\bullet)$ of combined levels. Next from each such matrix E_l, the b most promising combinations of levels are chosen to form new columns in matrices $\mathrm{A}_{l+1}(\bullet)$ in the next layer. This procedure is repeated, as many times needed until only one work matrix remains. Each column of this remaining work matrix corresponds to one complete product. Together, this matrix represents b complete products and each is considered to be first of b different product lines.

For each one of the b products created so far, we now need to design M-1 other products to form a complete product line. Firstly, the original data set is pruned to remove all consumers which choose the first product over their status quo. The above process is then repeated to find one second product and then iterated until M products are created and the product line is complete. At the end of this process we will have b different product lines, the best among them (i.e., the product line consisting of products which are chosen by the most consumers over their status quo) is selected as the final product line designed. Intuitively, therefore, the beam search technique follows a "build-up" or incremental sort of approach by considering combinations of different levels of attributes at each step in its search.

C. *Genetic Algorithms*

Holland [12] first proposed the concept of genetic algorithms (GA). The basis for the algorithm was the observation that a combination of sexual reproduction and natural selection allows nature to develop living species consisting of individuals that are highly adapted to their environment. In applying this technique to the product line design problem, each individual (also referred to as a string) would represent a possible line of products (see next section for encoding). The technique exploits the fact that each product profile (string) contains some features (sub-strings) that are desirable and hence contribute to its being evaluated highly. When one exchanges genetic material between two strings (through a process known as crossover) the expectation is that this exchange will produce offspring that combines some good features of the parents. If this exchange is carried out between two "good" strings then the chances of producing high quality offspring are higher. Therefore, while moving from one generation to the next, the product profiles making up the subsequent population are likely to be of a higher quality than those of the preceding generation. In addition the GA approach also includes the use of a mutation operator that can cause changes (biologically akin to introduction of new genes) in the population of strings and hence avoid the local minima trap.

The GA approach has several advantages in the manner in which it performs its search process, which distinguishes it from typical optimization and search procedures. The interested reader is referred to [11] for further details. In the context of the managerial problem that we address, while it is critical for a decision-maker to be provided with a high quality product profile, it is just as important to present him/her with a wide variety of feasible and high quality product profiles. Given such a set of choices, the decision-maker can subsequently utilize any number of subjective criteria to evaluate these different profiles. Since a genetic algorithm proceeds by maintaining a diverse population of product profiles at each generation, it is readily evident that the GA approach holds a significant advantage over other traditional optimization methods in this regard. Also as mentioned earlier, traditional search techniques (such as beam search) suffer from other limitations due to the fact that they are deterministic in nature. For example, Nair *et al.* [21] present at least one pathological set (additional ones can be created) of relative part-worth's for which the beam search heuristic will always result in an arbitrarily bad solution. We tested the GA based procedures we have developed on this instance and another similarly created instance and found that the GA finds the best (optimal) solution as compared to the BS heuristic. This of course is an inherent advantage of all GA based techniques, since the GA procedure is not restricted to remain in any particular part of the search space and therefore always has a positive probability of finding the best solution available even if it is in a very obscure portion of the search space.

The fundamental steps followed in a genetic algorithm are as follows [11], [12]: We start with an initial population of individuals (in this case product line profiles) which are either randomly generated or produced by some heuristic. This population is subjected to genetic operators (reproduction, crossover and mutation) and their fitness (or quality) evaluated. Some of the members of the resulting population are discarded while the remaining are carried over to the next generation and the entire cycle is repeated until a pre-specified stopping condition is met. The stopping condition could be based on the number of generations (i.e. such cycles) that have been evaluated so far, or some other measure such as the improvement in quality in the most recent generation, over the previous "n" generations (say).

Within the context of our problem, the candidate solution set of strings (i.e., product profiles) are generated initially from the first chromosome pool (i.e., initial generation). The size of the chromosome pool (i.e., the number of strings) N, is generally maintained constant in successive generations. In the next few subsections we describe the genetic representation and operators used to generate candidate products i.e. reproduction, crossover and mutation.

1) Encoding an Individual Product Line: We extend the encoding utilized by [1] to the product line design problem. Specifically, if there are M products or items (length of the line) in a product line, K_m attributes in product $m (m \in M)$, L_{mk} levels for attribute k belonging to product m then each product line has P positions where P is calculated by

$$P = \sum_{m=1}^{M} K_m.$$

Each position would correspond to a sub-string that indicates the level of attribute k in item (or product) m. Hence, consider the following example:

Suppose one is interested in marketing two brands of shampoo, i.e., the product line is made up of two products. Further, the shampoo category has three attributes with levels L1 = 2, L2 = 3, L3 = 3 as shown in Table I where attribute 1 is conditioner, attribute 2 is scent and attribute 3 is an anti-dandruff ingredient.

A product line "L" of length two, therefore, would be represented as

$$L = 1, 2, 2|2, 1, 3.$$

Where the first shampoo in the line does not have a conditioner, has a cinnamon scent and has a mild anti-dandruff ingredient. The second shampoo has a conditioner, jasmine scent and a strong anti-dandruff ingredient.

Justification of encoding: A number of researchers in the past have stressed the critical part an encoding/representation plays in the effectiveness of a genetic algorithm [11], [31], [32]. Among the design principles for constructing useful representations due to [32] is the principle of "minimal redundancy" which specifies that ideally, each member of the space being searched should be represented only by one chromosome (or string). Notice that the previously described representation, does not adhere to this principle since the same line of products can be represented by $M!$ different representations (where M is the number of products in the line). For example, the product line L shown above has the following alternative representation:

$$L' = 2, 1, 3|1, 2, 2.$$

As [32] and others have pointed out, lack of adherence to the principle of "minimal redundancy" may cause inefficiencies in the search since the crossover between two identical product lines (for example L and L' above) could result in two entirely different sets of offspring. Conversely, it might be argued that one possible advantage of such a representation is that crossover allows the exploration of more members of the search space (ie the search is more broad). To alleviate this concern, we devise an alternative representation where each product lines is stored such that the products in the line are arranged in lexicographic order. That is, for a product line with a length of M products, the string would be such that: product $1 \leq \mathrm{product}2 \leq \ldots \leq \mathrm{product}M$ (e.g., 1,1,1 < 1, 1, 2 < 1, 2, 1 for a product line with three attributes and three products). For this representation/encoding, therefore, the string L$'$ (above) would not exist as it is not a valid representation and L would be the only (unique)

TABLE I
EXAMPLE PRODUCT LINE CONSISTING OF DIFFERENT BRANDS OF SHAMPOO

Attribute	Level 1	Level 2	Level 3
Conditioner	Not Present	Present	-
Scent	Jasmine	Cinnamon	Anti-septic
Anti-dandruff ingredient	None	Mild	Strong

encoding of the product line. Hence the principle of minimal redundancy is adhered to. We call, for brevity, this encoding a "sorted" representation while the previous encoding is termed as the "unsorted" representation. We employ both of these representations in our computational investigation.

Generating an initial population: Our GA based procedures work by generating an initial population of product lines based on the above encoding. These product lines can be generated such that each initial population has a diverse representation (which is randomly generated) or it can be "seeded" with certain user specified product lines (possibly those generated by another method).

2) Evaluating Fitness of a Member of the Population: Given that the relative-part-worth utility for each consumer for a particular level of each attribute is assumed to be available (obtained by conjoint analysis for example), for a given product line, we calculate the relative-part-worth utility that a particular consumer would achieve for each product in that line. If the relative utility for that product is greater than 0.0 for a particular consumer, he/she would choose that product over his/her status-quo choice. By calculating the ratio of the total number of consumers *not* choosing to switch to even one new product in the product line over their status-quo choices, and the total number of consumers, we get a measure of the ratio of consumers that are unsatisfied (or unaffected) by this product line. If we subtract this ratio from 1.00, we get the fraction of consumers that would choose at least one of our products over their status-quo choices if this product line were adopted. This is the market share metric that we seek to maximize. In other words this is the fitness of this particular product line.

Once the fitness of each string has been evaluated, operators such as crossover and mutation can be applied. This results in a new generation of strings that can then be evaluated again.

3) Population Maintenance Strategy: We employ the commonly used emigration strategy to govern how individual product lines in one generation are carried over to the next generation. This strategy impacts how the population of individuals is maintained during the simulation. Assuming that we have N strings in the new population, in the emigration strategy the strings selected for reproduction and the offspring created from them, form the members of the total of N strings in the new population.

4) GA Operators—Reproduction: With a total of N strings in the population, we allow reproduction to be carried out in the following way. We deterministically select s (where $s < N$) strings for reproduction based on their fitness. This selection is carried out such that higher fitness strings are the first to be selected. That is, the s highest quality strings in any generation are selected for reproduction. Next we allow an equal opportunity based technique where of the strings selected for reproduction, we allow the parents to be selected with equal opportunity (i.e., randomly paired up from the set of parent strings).

5) GA Operators—Crossover: In the previous section we had described how strings are selected for reproduction. These selected strings are now considered to be candidates for crossover. We implement crossover such that sub-strings can be exchanged between two candidate parent strings. This can be illustrated by the following examples. Consider the following two strings that have been selected for reproduction:

$$\begin{array}{ll} \text{Product line 1} - & 1,3,3,\,|\,3,2,4 \\ & \quad\;\;\updownarrow \quad\;\; \updownarrow \\ \text{Product line 2} - & 1,4,2,\,|\,3,3,5. \end{array}$$

Depending upon a user-specified number of attributes r (where $r < P$) to crossover, r attributes are randomly selected and exchanged between the two strings to get two new strings. There is no restriction on the products that these attributes are chosen from. That is, all r of these attributes could potentially be chosen from the same product in the product line (but this is not necessarily so). For instance, if $r = 2$ and we randomly select positions 3 and 5 (from the strings above) to crossover. Then the resulting strings would be

$$\begin{array}{ll} \text{Product line child 1} - & 1,3,2,\,|\,3,3,4 \\ \text{Product line child 2} - & 1,4,3,\,|\,3,2,5. \end{array}$$

Note that by using this method, all resulting children are feasible with respect to the given number of levels and attributes. If the sorted representation is used, then each product line child created in this fashion is sorted in lexicographic order after the crossover operation.

Radcliffe [32] also refines the definitions of schemata and o-schemata (due to [12] and [11], respectively) to specify what he terms as forma. A forma is the set of all its instances such that if a chromosome η is an instance of a forma ξ then η and ξ both contain the same values at certain positions (called defining positions) at which particular values are specified in the forma ξ. For example, the product lines 1 and 2 (above) both belong to

$$\xi_{12} = 1,^*,^*\,|\,3,^*,^*.$$

Where the *'s in ξ imply "don't care" values. Another of the design principles posited by [32] is that of "respect," i.e., crossing over two instances belonging to the same forma should result in the creation of another instance of the same forma. It is clear that with the unsorted representation, new strings created using crossover from parents belonging to the same forma will also belong to the same forma, i.e., crossover is "respectful." This however does not hold true for strings belonging to the "sorted" representation. For example the following two product lines:

$$\begin{array}{ll} \text{Product line 3} - & 1,3,2,\,|\,1,4,3 \\ \text{Product line 4} - & 1,4,3,\,|\,5,1,3 \end{array}$$

both belong to the same forma

$$\xi_{34} = 1,^*,^*\,|^*,^*,3$$

TABLE II
GA TECHNIQUES

TYPE	Representation Used	Integration with BS	
		Hybrid Mutation	Seed with BS
GASM	Unsorted	No	No
GASSM	Sorted	No	No
GAHM	Unsorted	Yes	No
GASHM	Sorted	Yes	No
GASMBS	Unsorted	No	Yes
GASSMBS	Sorted	No	Yes
GAHMBS	Unsorted	Yes	Yes
GASHMBS	Sorted	Yes	Yes

TABLE III
PARAMETERS USED FOR GENETIC ALGORITHM BASED METHODS

Parameters	Value of Parameter
Mutation	0.04
Population size	400
Number of Attributes to Crossover (M=4, K=7)	10
Number of Attributes to Crossover (M=4, K=9)	17
Number of Attributes to Crossover (M=7, K=7)	12
Number of Attributes to Crossover (M=7, K=9)	21
Number of generations	500

M Number of Products
K Number of Attributes

but if they were crossed over at positions 1 and 4 (say), then the resulting children (after sorting in lexicographic order) will be

$$\text{Product line child 3} - \quad 1,3,2,|\,5,4,3$$
$$\text{Product line child 4} - \quad 1,1,3,|\,1,4,4.$$

Note that the first child still belongs to forma ξ_{34}, but the second child does not. This (sorted) representation therefore does not adhere to the "respectful" crossover principle posited by [32]. This difference may be one of the reasons why GAs using the unsorted representation outperform those using the sorted representation in our simulation study.

6) GA Operators—Mutation: The new population created using the above procedure is now subject to mutation. All strings in the new population are subject to the possibility of mutation. That is, new children as well as parents stand the chance of being mutated. We have created two different mutation operators. The first is a standard mutation operator (SM) which works in the following way. Given a probability of mutation defined by the user, strings are randomly chosen (without replacement) with this mutation probability from the population. Then a single attribute is randomly picked in this chosen string and the level of that attribute is changed to a randomly chosen level within its feasible set. The second mutation operator is a hybridized mutation (HM) operator which is defined in some detail in the next section.

D. Hybridized Approaches

With the availability of a quick solution method such as the BS heuristic described earlier and a more time consuming iterative method like a GA, we next investigate the potential merits of integrating the two procedures to get a better overall procedure. It is well known that the performance of GA's depends to a large extent on the quality of the initial population. There are two schools of thought on this issue. One group of researchers is of the view that the initial solutions (i.e., the initial population of strings) should be random so that adequate diversity is available for the latter generations [33]. Another group of researchers have found promising results when better (higher quality) starting solutions are provided in the initial population (see for example [34]). One hybridization strategy would be to "seed" the initial population with the best product line created by using the BS heuristic and generate the remaining $N-1$ members in a random fashion. This technique proceeds in the following manner: We run the BS heuristic and the best product line obtained from this is used to seed the initial population of the GA simulation that is run subsequently. We consider this seed product line as a member of the initial population in our GA and generate the remaining $(N-1)$ members in a random fashion to make up a starting population consisting of N members. We now proceed with the GA as mentioned before, stopping when the stopping condition is fulfilled.

A second hybridization technique is based on modifying the mutation operator[2] to utilize the best product line created by using the BS heuristic. This hybrid mutation (HM) operator works as follows: At the beginning of the GA simulation, the best product line created using BS is stored as a "mutator" string. Now during each generation whenever a string is selected for mutation, an attribute within this string is selected at random and its value modified either using the SM operator (described above) or changed to the value present for that attribute in the mutator string. This (equally likely) choice between applying the SM operator and the HM operator is important since it avoids premature convergence of the population to the alleles of the mutator string.

The combined hybrid approaches can be potentially advantageous due to a number of reasons. First, one might expect that similar to the findings of [34], we can find a better product line as compared to that found by just the GA or the BS procedures in isolation. Second, it would be useful if the hybridized techniques can result in producing a final population of product lines that are better (on average) than those found by just the GA procedure. Thirdly, it is our expectation that the hybridized procedures will be able to provide high quality product lines more quickly than a pure GA technique. Computational comparisons follow in the subsequent sections.

E. Comparison of Solution Approaches

The Product Line design problem has been recognized as belonging to the class of NP-hard problems. This implies that optimal solutions to reasonable sized instances of the problem are difficult to obtain in reasonable amount of time. To validate this

[2]We thank an anonymous referee for suggestions that led us to add such an operator.

TABLE IV
COMPARISON OF CPLEX WITH GA AND BS SOLUTIONS

M	*K*	BS	GA Methods								CPLEX
			GASM	GASSM	GAHM	GASHM	GASMBS	GASSMBS	GAHMBS	GASHMBS	
7	7*	0.9214	0.9929	**1.0000**	**1.0000**	0.9857	0.9857	0.9786	0.9571	0.9429	0.6143
7	7	0.9444	0.9722	0.9792	0.9931	0.9583	0.9722	0.9583	0.9514	0.9514	**1.0000**
7	7	0.7853	0.8079	0.8079	0.8079	0.8079	0.8079	0.8136	0.8192	0.8023	**1.0000**
7	7	0.9493	0.9783	0.9783	0.9855	0.9783	**1.0000**	0.9855	0.9928	0.9855	0.6957
7	9**	0.9404	0.9934	0.9868	0.9801	**1.0000**	**1.0000**	0.9801	0.9868	0.9801	0.6556
7	9	0.8659	0.9329	0.9329	0.9207	0.9573	0.9085	0.9268	0.9207	0.9146	**1.0000**
7	9	0.8222	0.8778	0.8722	0.8722	0.8778	0.8722	0.8500	0.8444	0.8556	**1.0000**
7	9	0.9338	**1.0000**	0.9801	0.9735	0.9669	0.9735	0.9735	0.9536	0.9536	0.6490

Legend:

M	Number of Products
K	Number of Attributes
BS	Share of Choices due to Beam Search heuristic
GA	Share of Choices due to Genetic Algorithm based methods
CPLEX	Share of Choices due to CPLEX (version 7.5) software

Note: Share of choices values are represented as a percentage of the largest value obtained from among all (GA, CPLEX, BS) methods. Values obtained from CPLEX are those after 36,000 seconds of CPU time. CPLEX was not able to find the optimal solution for all instances within this time limit.
Items in bold indicate highest value in row
* 7 Attributes with 6 3 7 4 5 3 3 levels respectively
**9 Attributes with 7 3 5 5 6 3 3 7 5 levels respectively

point on the one hand and to demonstrate on the other the potential applicability of heuristic procedures such as our proposed GA approach, we solved eight instances of the product line design problems. These problem sets were tackled using a state of the art integer programming package (CPLEX version 7.5), the heuristic methods of beam search [21], our proposed GA and the hybridized GA methods were coded in C and run on a Pentium IV PC (1.0 GHz) with 512 Mb of RAM (running MS Windows 2000).

We have defined, in this regard, eight different types of GA and hybrid GA procedures based on choices of string representation, mutation technique and seeding with the BS heuristic. These eight procedures are GA with standard mutation (GASM), GA with sorted representation and standard mutation (GASSM), GA with hybrid mutation (GAHM), GA with sorted representation and hybrid mutation (GASHM), GA with standard mutation and seeding (GASMBS), GA with sorted representation, standard mutation and seeding (GASSMBS), GA with hybrid mutation and seeding (GAHMBS) and GA with sorted representation and hybrid mutation and seeding (GASHMBS). All of these eight procedures and their features are summarized in Table II. Further, the values of GA parameters that we have utilized in our computational tests are shown in Table III. It must be kept in mind that these parameters are problem specific and while derived from the study due to [19], they may need to be modified for other problem contexts. In general we found that lower values of mutation resulted in premature convergence to a suboptimal product line design, while values of mutation higher than 0.04 resulted in lack of significant improvement due to too much of random variation and best product line being obtained toward the end of the simulation run. The number of attributes to crossover also make a significant difference to the convergence of the GA techniques. It was found (empirically) that for the problems with smaller number of products a higher number of attributes to crossover was more beneficial as compared to problems with larger number of products. In each of the integrated (i.e., hybrid GA and BS) techniques (GAHM, GASHM, GASMBS, GASSMBS, GAHMBS and GASHMBS) the hybrid mutation operator and (or) the best BS created product line is introduced as a member of the initial population (the remaining members are randomly generated). The problem instances had the following characteristics: $M = 7$ items (products) in the product line, $K = 7$ or nine attributes in each product with (6 3 7 4 5 3 3) levels in the seven attribute case and with (7 3 5 5 6 3 3 7 5) levels in the case of nine attributes. The number of consumers was kept constant at 200. Note that, this limited test was separately performed from the latter set of computational runs, since the objective here was to compare the effectiveness (and or ineffectiveness) of our heuristic search techniques with an optimal solution technique such as CPLEX. As our results indicate, these problems were difficult to solve to optimality.

Table IV provides comparative data on the performance of the four different classes of solution approaches namely, CPLEX; BS; GA; and hybridized GA. CPLEX uses a branch and bound algorithm for arriving at the best integer solution. This method is time consuming and requires a large amount of memory since a sizeable branch and bound tree needs to be stored during the

TABLE V
COMPARISON OF GA AND HYBRID GA TECHNIQUES

GATECH	PBEST Ratio	BGA	WORST	AVG	SD	GENNO	Average Number of Unique Strings		
							ONE	FIVE	OTOFV
GASM	**1.0336**	**0.6305**	0.4549	**0.6113**	**0.0387**	**248.74**	2.72	**40.86**	58.62
GASSM	1.0303	0.6285	0.4684	0.6045	0.0340	195.82	**2.80**	33.11	**60.13**
GAHM	1.0290	0.6279	0.4914	0.6110	0.0270	147.43	2.66	37.72	53.31
GASHM	1.0295	0.6282	0.4916	0.6113	0.0270	149.75	2.59	37.64	53.68
GASMBS	1.0245	0.6249	0.4639	0.6009	0.0340	115.40	2.51	31.67	59.79
GASSMBS	1.0247	0.6250	0.4639	0.6011	0.0342	110.40	2.64	31.05	59.76
GAHMBS	1.0200	0.6222	**0.4960**	0.6089	0.0236	78.23	2.34	37.88	45.79
GASHMBS	1.0197	0.6220	0.4955	0.6087	0.0236	72.97	2.34	37.68	46.50
p-Value	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)

Items in bold indicate highest value in column

Legend:

GATECH	Genetic Algorithm technique used
BGA	Share of choices of best product line in final population of GA
BBS	Share of choices of best product line designed by Beam Search
$\text{PBEST} = \frac{\text{BGA}}{\text{BBS}}$	
WORST	Share of choices of worst product line in final population of GA
AVG	Average share of choices of product lines in final population of GA
SD	Standard Deviation of share of choices of product lines in final population of GA
GENNO	Generation number at which best product line was found
ONE	Number of unique product lines with best share of choices in final population of GA
FIVE	Number of unique product lines with share of choices within 5% of the best share of choices (but less than the best value)
OTOFV	Number of unique product lines with share of choices between 5 and 10% of the best share of choices in the final population of GA

solution process. In Table IV, we report the best objective function value obtained by the CPLEX procedure after 36 000 seconds (10 h) of CPU time. This time limit is arbitrary and was used after several runs on CPLEX took more than a week (over 600 000 s) of computational time and had to be terminated since the branch and bound tree was consuming excessive (computer memory) space. This is not surprising since the general integer programming problem is known to be NP-complete [35].

It can be seen from Table IV that even after 36 000 s of computation time CPLEX failed to produce any solutions that were confirmed to be optimal, although it did find better solutions that the heuristic techniques in half of the cases. Further, the beam search based method did not perform as well as the GA or the hybridized GA methods for all eight of the problems considered. The heuristic BS method, however, took a considerably shorter amount of time (2–3 s) to reach its solution as compared to the GA techniques which took approximately 30 s on average. But the solution value obtained via BS was substantially inferior to any of the GA based techniques. Therefore, it seemed logical to think that an integrated or hybrid solution procedure which is designed to combine the features of both the GA and BS techniques might be successful, and which might thus explain the relative success in this limited set of the integrated (GA and BS) methods.

IV. COMPUTATIONAL TESTING AND PERFORMANCE EVALUATION

In this section we compare various heuristic configurations and make some qualitative inferences as to their applicability and use in product line design problems. Specifically, we investigate the impact of various problem characteristics such as size (number of products in line, number of attributes), presence (absence) of attribute importance on specific dependent variables. This section consists of three parts. The first part describes the methodology employed by us to carry out the performance evaluation and identifies the dependent and independent variables of interest. The second part compares the performance of various GA techniques. The last section discusses the impact of various problem characteristics on the performance of the heuristics and dependent variables.

A. Methodology

Given that we are interested in characterizing the behavior of the genetic algorithm based methods developed in this paper (i.e., the eight different GA based variants we have developed), we employ a $2 \times 2 \times 2$ full factorial design to assess the importance of the following factors:

a) number of products in the line (4 and 7);

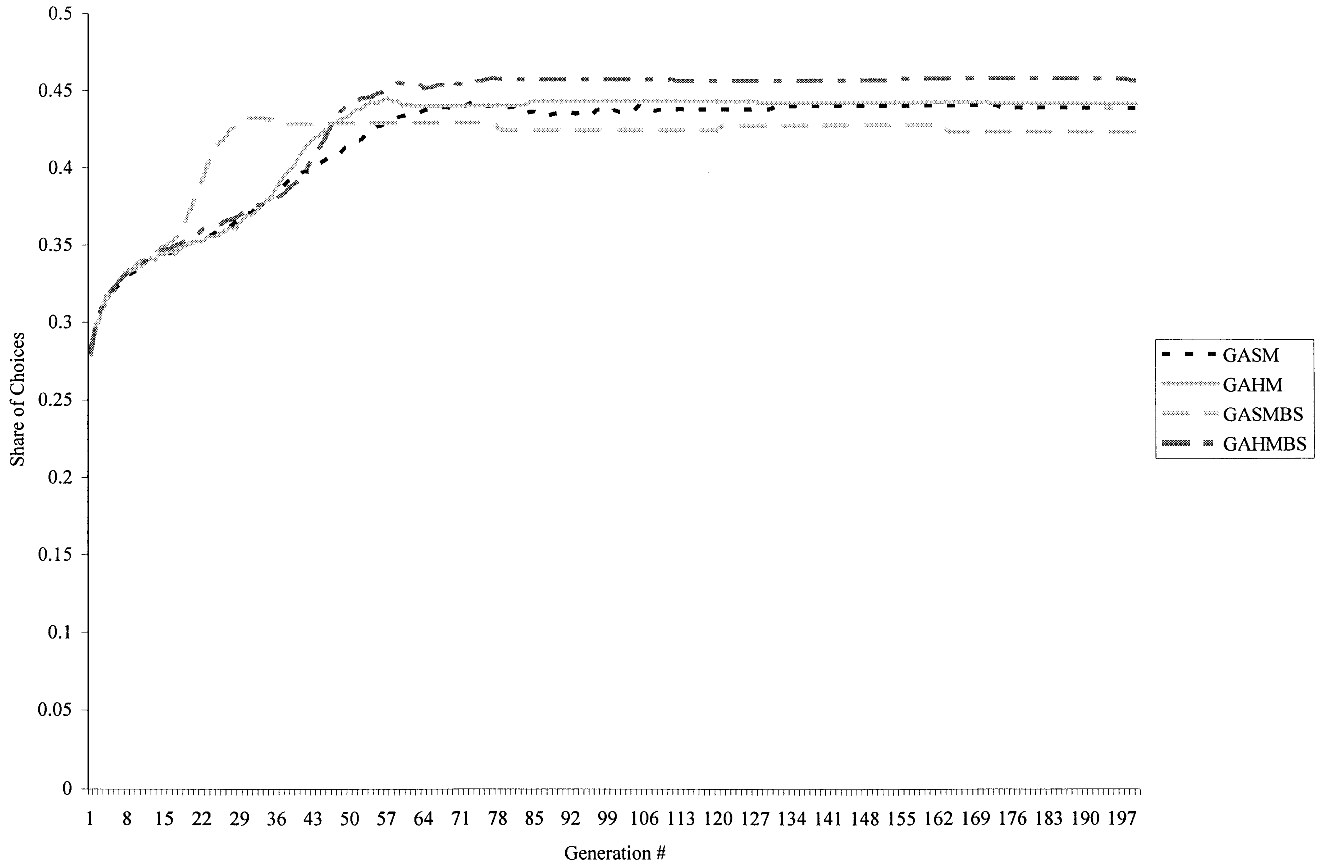

Fig. 1. Average quality of population over the course of a simulation (GASM, GAHM, GASMBS, GAHMBS).

b) the number of attributes (7[3] and 9[4]) in the product category;
c) the presence (absence) of attribute importance, on the performance of the heuristic methods.

Problem instances were generated with either seven or nine attributes, with the number of levels of each attribute ranging between three and seven and with 200 consumers in each data set. Utility values were assigned in the following manner, for each consumer a utility value for every level of each attribute was generated from a uniform distribution between zero and one. Next, these utility values across all attributes for that consumer, were scaled (normalized) to sum to a total of one. For problem instances with the presence of attribute importance, this previously generated (and normalized) utility data matrix was modified in the following manner: For each such instance the attribute with the largest number of levels was designated as the most "important" attribute. Next for each consumer, a randomly chosen level for this "important" attribute was assigned a utility value of 1 + its previous utility value. Following this, the utility values for the consumer were rescaled, so as to sum up to one as before. For each consumer, therefore, the chosen attribute is guaranteed to have a significantly higher utility value relative to the utility values for the other attributes. Next, each customer was assigned one level of each attribute as his/her status quo level (in a random manner) and the "part-worth" utility values of all levels calculated by subtracting the utility of this status quo level from that due to other levels of that attribute. For each combination of the factors above, we created 10 replicates of the data set resulting in a total of 80 different problems.

Next, we deployed each of our eight different techniques to design product lines for these 80 problem instances. In keeping with other researchers [36], [37] in the past, we ran our GA based techniques 10 different times for the same problem instance. This resulted in a total of 6400 different runs. We then carried out a series of analysis of variance (ANOVA) tests [37] on the results to see the effect of the factors described above on a number of dependent variables of interest which are discussed below.

One of the primary variables of interest is the ratio of the best GA solution to the best BS solution which we define as

$$\mathrm{PBST} = \frac{\mathrm{BGA}}{\mathrm{BBS}}$$

where

BGA Best product line due to GA based technique.

BBS Best product line due to BS.

Other dependent variables of interest are the best GA solution (BGA); the number of unique strings (product lines) in the final population (STR); the worst string in the final population (WORST); the average value of the product lines in the final population (AVG); the standard deviation of the fitness of the

[3]7 Attributes with 6 3 7 4 5 3 3 levels respectively.

[4]9 Attributes with 7 3 5 5 6 3 3 7 5 levels, respectively.

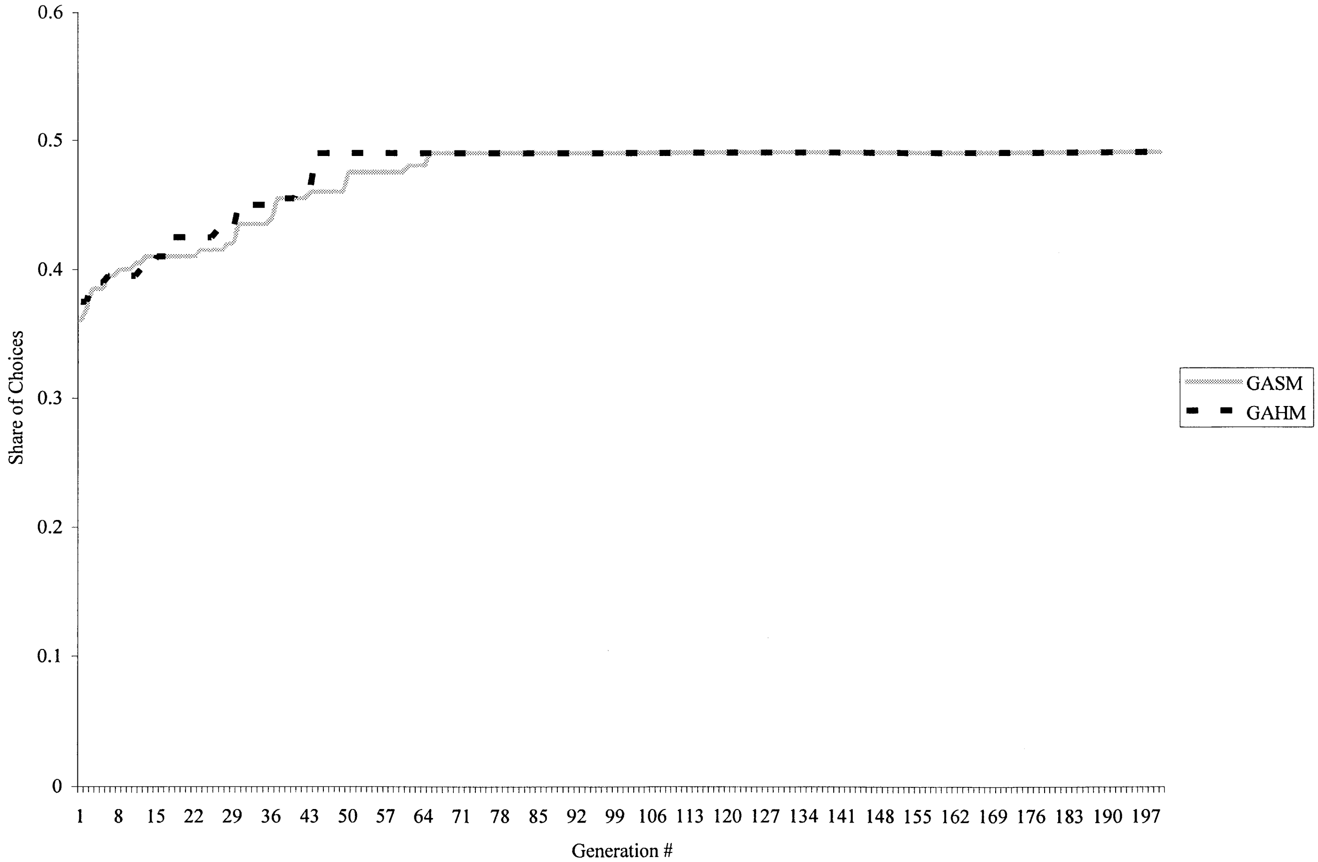

Fig. 2. Impact of hybrid mutation on best product line (GASM versus GAHM).

strings (i.e., market share of product lines) in the final population (SD); and the generation number at which the best solution is found (GEN).

Furthermore, we also evaluate the impact of the factors on the number of unique strings in the final GA population. For this we need to compare strings in the population to see if they are different. This comparison works in the following manner: two product lines are considered to be different if there exists at least one product in the first product line which is not present in the second product line, two products are considered different (non-identical) if they differ in the level of at least one attribute. Using this comparison function, we measure the number of unique strings with the best evaluation (ONE), the number of unique strings with an evaluation within 5% of the best value but less than the best value (FIVE) and the number of unique strings with an objective function value less than 5% and no worse than 10% of the best evaluation (OTOFV), in the population of the final generation.

B. Comparative Performance of GA and Integrated GA Techniques

Upon analyzing the data from our simulation runs we find that all of the GA based techniques consistently outperform the BS based procedure (see Table V). Of the total 6400 simulation runs the GA based procedures did as well (or better) than BS in 6140 (95.93%) cases and were strictly better (than the BS solution) in 5300 (82.81%) instances. On average (among all eight GA techniques tested over the 6400 runs) the share of choices is 2.6% better (than the BS method) for product line profiles created using GA based methods. While in the best case a GA based procedure produced a product line with a share of choices that is 12.75% higher than the BS based heuristic and in the few (260 out of 6400) cases where the GA methods (GASM, GASSM, GAHM and GASHM) were outperformed by BS, the worst GA solution resulted in a share of choices which was 6.1% less than the BS solution. The hybridized methods with seeding (GAHMBS, GASHMBS, GASMBS, GASSMBS) always produce product line designs at least as good as the BS solution and in 80.2% of the cases produce designs strictly better than the BS solution. Note that, both of these techniques (GA and BS) suffer from the same limitation in that neither provides information on how far the solution is from the optimal market share. However, due to the NP-complete nature of the product line problem, this is at this point a limitation which is difficult to overcome.

An ANOVA procedure considering the impact of the type of GA based technique utilized (of the eight listed in Table II) on the dependent variables mentioned above, provides us with interesting results (Table V). The kind of GA technique utilized, makes a significant difference in the best product line designed after 500 generations, interestingly enough not in the manner we had expected.

We find that techniques which utilize the unsorted representation with the standard mutation operator and no seeding (i.e. with no integration with BS) perform the best, on average. In other words, integration with the BS heuristic does not

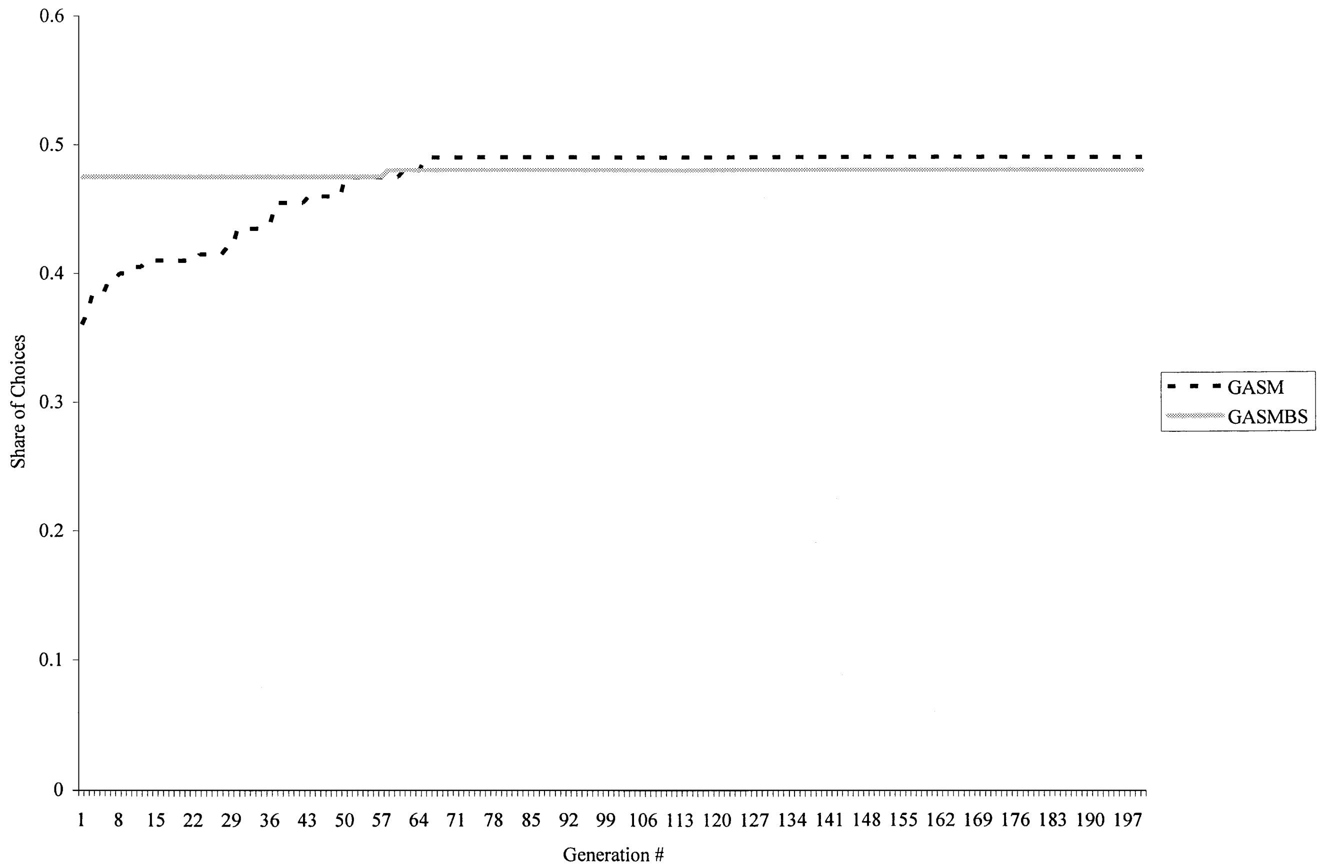

Fig. 3. Impact of seeding on best product line (GASM vs GASMBS).

positively impact the quality of the best product line designed. While this is in variance with our intuitive expectations, this is in keeping with some of the past literature [16], [39]. Another key observation deals with the diversity of the resulting population of product lines given the different methods. As expected, the integrated techniques cause some loss in diversity since the BS string used to seed the population or use in hybrid mutation causes convergence of the population to the same (or similar) strings. This phenomenon is consistent with the schemata theory that was posited by Goldberg [11]. According to [11], subsequent generations in a GA algorithm will see the proliferation of short high value substrings (schemata) of high fitness strings belonging to earlier generations thereby resulting in an improvement in the overall fitness of the population of strings. Given that generally the BS string will be of relatively high quality (as compared to other randomly generated members of the initial population), the population of following generations would be expected to be heavily influenced by this "good" BS generated product line. This would thereby tend to reduce the search to the local area around this particular product line.

Specifically, the GA technique with no seeding, standard mutation and the unsorted representation (GASM) produces product line designs which are as good as or better than the GA technique with seeding and standard mutation in 69.25% (and strictly better in 57.12%) of the problem instances. GASM produces product lines designs as good or better than the GA technique with no seeding, standard mutation and the sorted representation (GASSM) in 60.75% (and strictly better in 47%) of the cases respectively. In addition the GA techniques without any hybridization (GASM and GASSM) provide product line designs as good or better than the hybridized techniques (GAHM, GASHM, GASMBS, GASSMBS, GAHMBS, and GASHMBS) in 52.37% (and strictly better in 35.12%) of the problem instances respectively.

The standard deviation of the quality (i.e., the market share of product lines) of strings in the final population (SD) is different across the different GA techniques ($p < 0.0001$). Further evidence for the argument that the hybridization based procedures suffer from premature convergence is provided since the GASHMBS and GAHMBS techniques result in the lowest standard deviation values of the share of choices of the product lines in the last generation.

We find that on average the integrated techniques (GAHM, GASHM, GAHMBS, GASHMBS, GASMBS, GASSMBS) all tend to find their best candidate product line earlier than the pure genetic algorithm based methods (GASM and GASSM). Fig. 1 depicts the process tracing results showing the change in the average fitness of the population over a typical simulation run of the GA and show graphically this (premature) convergence. We speculate that this is due to the high quality product line (due to BS) being included in the initial generation (or used within mutation) in the integrated methods which drives the population to converge sooner. Again among the integrated methods the GASHMBS based methods result in the quickest convergence (i.e., on average the best candidate product line is found in the earlier generations).

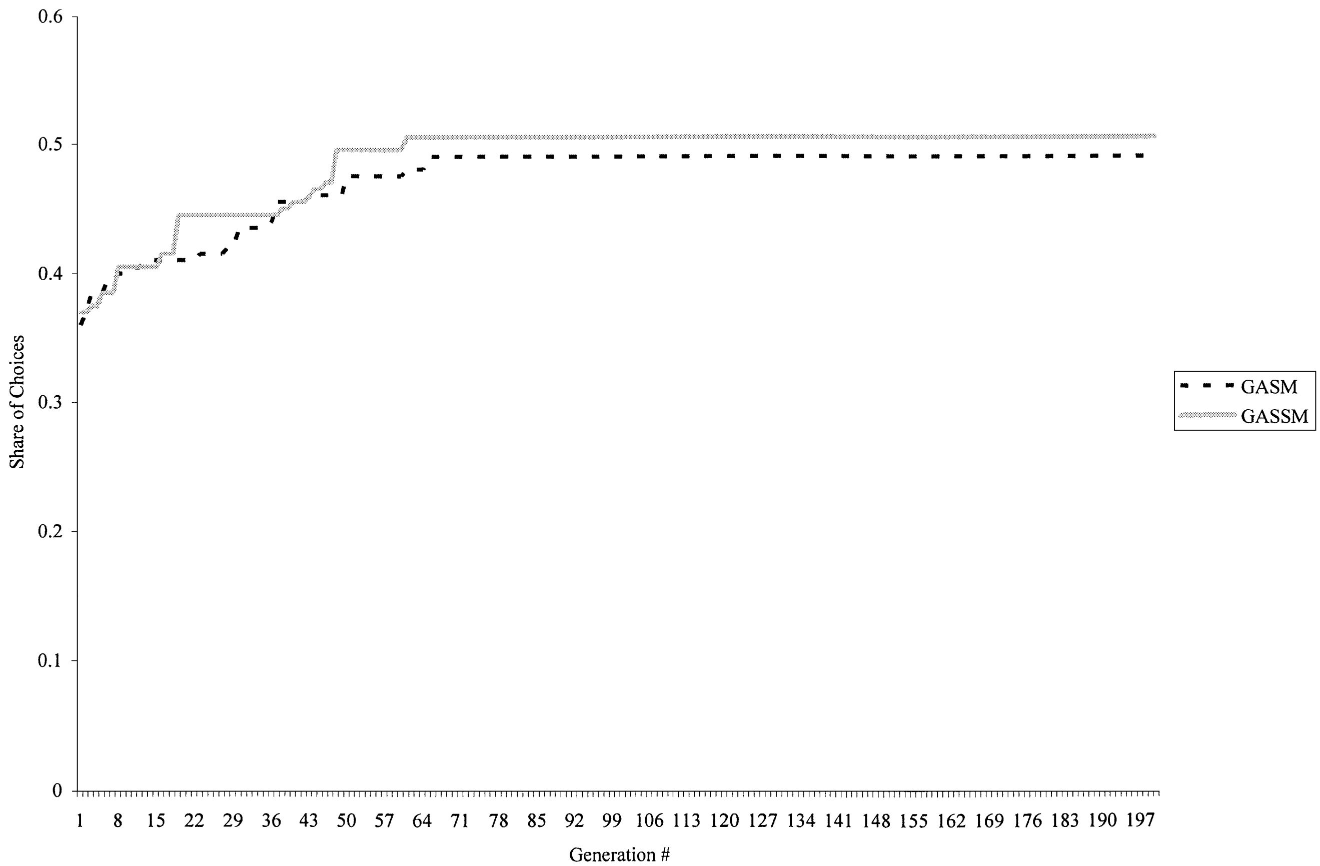

Fig. 4. Impact of GA representation on best product line (GASM versus GASSM).

Figs. 2 –5 show the process tracing results. Specifically, the depict the change in the value of the best product line found so far over typical simulation runs. It can be seen that in the case of each of the non seeded genetic algorithm based methods, the best product line found initially is of lower quality than that found using the integrated methods. This is not surprising since the integrated methods have an initial population that includes a product line found using the BS method. However, over a simulation run one can see the both pure GA as well as hybrid methods converge to find product lines that are nearly the same in quality. While this shows the robustness of the GA techniques, it also implies that if the decision maker is interested in getting to a high quality string quickly (e.g., when evaluating a number of "what if" scenarios iteratively) then the integrated methods would be better suited for him/her as opposed to the pure GA techniques.

C. Impact of GA Representation

We notice from the simulation results (Table V) that on average, the unsorted representation results in a higher quality of product line designs when compared to GA methods using the sorted representation (see Fig. 4). This again can be related to the diversity of the population which is higher for the unsorted representation as compared to the sorted representation. Recall, that in the unsorted representation there are many possible distinct ways to encode the same line of products—$M!$ different encodings to be exact, where M is the number of products in the product line. In addition for the unsorted representation, crossover between identical product lines (with different representations) will result in offspring which are nonidentical to each other. While this goes against the principle of "minimal redundancy" posited by [32], it does conform to the principle of "respectful" recombination which implies that crossover between strings belonging to the same forma result in strings which also belong to the same forma. The sorted representation, while being minimally redundant, does not allow respectful recombination and this apparently, causes its lack of performance. In other words, the representation allowing respectful recombination dominates over the one allowing minimal redundancy.

D. Impact of Problem Characteristics

This sub-section discusses the impact of problem characteristics such as problem size (number of attributes, number of products) and attribute importance on the dependent variables of interest identified previously. Detailed information of the statistical analysis is provided in Tables VI and VII.

1) Impact of Problem Size: The number of products in the product line ($p < 0.0001$) significantly impacts all of the measures of unique strings (ONE, FIVE and OTOFV) in the last generation and in a positive direction (Table VI). That is, with increases in number of products or number of attributes, the number of unique strings in the last generation increases. This is, of course, as expected since the number of different combinations of products and attributes increases with increases in either of the above and hence there are potentially a larger number

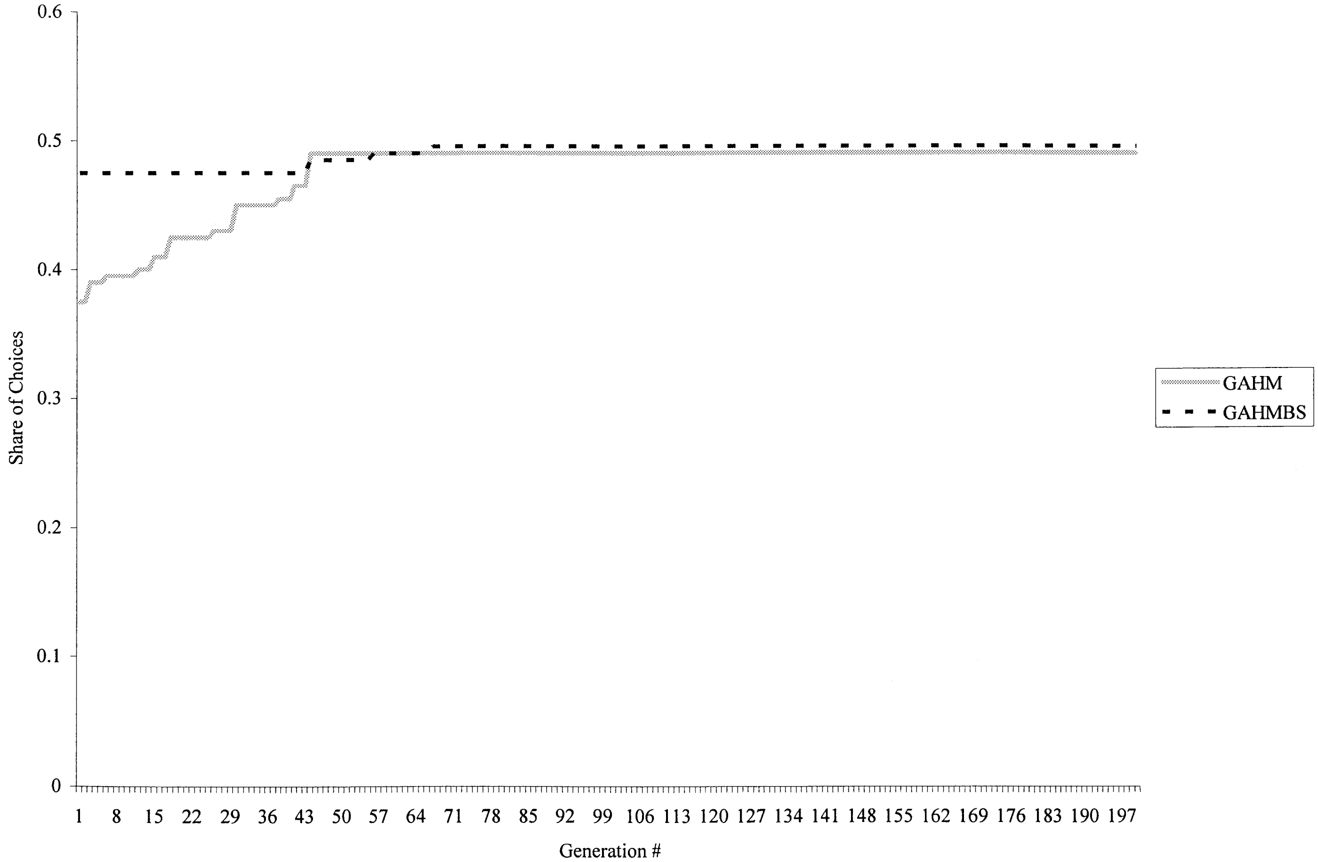

Fig. 5. Impact of seeding with hybrid mutation on best product line (GAHM versus GAHMBS).

of distinct product lines with similar share of choices values. Again Table VII indicates the relative difference between the numbers of unique strings found in the last generation across the various GA techniques. The large number of relatively high quality strings found, are a significantly important feature of the GA based techniques. The reason being that given such a set of different yet high quality product lines, a manager can pick and choose the most useful product line out of the set while optimizing some secondary objectives [40] which are not included in the share of choices calculation.

Both the size of product line and the number of attributes significantly affect the standard deviation of the product lines in the last generation. We notice also that the number of products in the product line significantly affects the ratio PBEST as well as the variable BGA. In general an increase in the number of attributes (from seven to nine attributes) seems to marginally increase the value of PBEST and BGA. Our intuition for this is that as the problem becomes more complex (due to an increase in the number of attributes in each product in the line) the BS based heuristic become somewhat less effective while the GA methods still continue to perform well. This may be due to the fact that the number of possible product attribute level choices will increase with an increase in the number of attributes, hence the BS based procedures result in suboptimal choices. Similarly, the number of products in the product line ($p < 0.0001$) and the number of attributes ($p < 0.0001$), the GA technique ($p < 0.0001$) and attribute importance ($p < 0.0001$) all have a statistically significant impact the generation in which the best string is found.

2) Presence of Attribute Importance: The presence of attribute importance has a statistically significant effect on the ratio PBEST as well as the variable BGA. It is notable that GA based techniques are able to hone in on the presence of attribute importance. Specifically, the GA performs slightly better relative to the BS heuristic when attribute importance is present. While Tables V and VI show the performance of different GA based methods when tested on problem instances with and without attribute importance, Table VII shows the performance of different GA based methods when tested on the problem instances in the absence of attribute importance. It can be seen that the PBEST value in Table VII ranges from 1.019 to 1.032 (i.e., an improvement ranging from 2–3.2%) which is smaller than that observed in Table V. Our intuition for this is the following: The BS method performs worse in the presence of attribute importance due to the fact that given one particular level of an attribute contributes significantly to the utility of consumers as compared to all other attributes, it is possible that the beam search heuristic due to its "build up" nature misses out on this and thus does not find this particular attribute/level combination. The GA based searches on the other hand, which are based on simultaneous consideration of the complete attribute set do seem to be able to find this "important" attribute and incorporate it into the product lines designed. This tends to result in a product line which in all likelihood is very close to the best product line since the most "important" attribute

TABLE VI
ANOVA RESULTS: IMPACT OF PROBLEM CHARACTERISTICS ON PERFORMANCE

Parameter	Level	PBEST Ratio	BGA	WORST	AVG	SD	GENNO	Average Number of Unique Strings		
								ONE	FIVE	OTOFV
PLINE	4	1.028	0.535	0.393	0.518	0.028	92.77	2.08	19.23	44.13
	7	1.023	0.716	0.563	0.693	0.032	186.92	3.07	52.67	65.26
	p-Value	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)
NATR	7	1.021	0.606	0.463	0.588	0.0285	115.15	2.46	31.70	51.03
	9	1.031	0.645	0.492	0.623	0.0320	164.54	2.69	40.20	58.36
	p-Value	(0.07)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0109)	(0.0001)	(0.0076)
ATIMPT	NO	1.025	0.605	0.460	0.585	0.029	139.79	2.54	34.82	54.44
	YES	1.027	0.646	0.495	0.625	0.030	139.89	2.60	37.08	54.95
	p-Value	(0.0001)	(0.0115)	(0.0001)	(0.0001)	(0.0047)	(0.966)	(0.286)	(0.0001)	(0.0001)

Legend:

PLINE	Number products in product line
NATR	Number of Attributes in each product
ATIMPT	Presence/Absence of attribute importance

TABLE VII
PERFORMANCE OF GA AND HYBRID GA METHODS ON PROBLEM INSTANCES WITHOUT ATTRIBUTE IMPORTANCE

GATECH	PBEST Ratio	BGA	WORST	AVG	SD	GENNO	Average Number of Unique Strings		
							ONE	FIVE	OTOFV
GASM	**1.0327**	**0.6098**	0.4395	0.5794	**0.0379**	**248.48**	**2.81**	**39.33**	57.88
GASSM	1.0297	0.6079	0.4508	0.5841	0.0335	206.12	2.64	32.31	59.22
GAHM	1.0279	0.6071	0.4724	0.5902	0.0268	152.72	2.66	36.50	53.41
GASHM	1.0279	0.6072	0.4725	**0.5903**	0.0267	153.59	2.66	36.60	54.06
GASMBS	1.0237	0.6044	0.4465	0.5806	0.0335	106.96	2.40	30.58	59.26
GASSMBS	1.0241	0.6047	0.4482	0.5811	0.0335	108.37	2.54	29.88	**59.33**
GAHMBS	1.0199	0.6023	**0.4794**	0.5894	0.0231	71.00	2.30	36.59	46.07
GASHMBS	1.0197	0.6022	0.4782	0.5889	0.0232	71.11	2.36	36.80	46.32
p-Value	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)	(0.0001)

Items in bold indicate highest value in column

will make the largest contribution to the objective function as opposed to the other lesser important attributes. This is a particularly relevant factor as it relates to the managerially critical needs in brand positioning as well as in providing face validity for analytical techniques to a skeptical audience.

V. CONCLUSION

In this paper, we have proposed several alternative solution approaches for the product-line design problem The solution approaches we develop are derived from AI based genetic algorithm techniques. In addition, we have developed hybridized approaches that are based on the integration of Genetic algorithms and an entirely different solution technique (Beam Search). We demonstrate the practical problems involved in applying traditional mathematical programming techniques to this managerial problem. Further, we have conducted a fairly exhaustive and detailed simulation study complete with statistical analysis and process tracing to understand the relative impact of various problem specific characteristics on the different solution procedures that we have proposed. Finally, we develop two different GA representations for each individual string (both of which are grounded in previous research) and find that the representation which allows "respectful" recombination dominates (or outperforms) that allowing minimal redundancy.

It is also critical to note that the improved market shares (over the state of the art benchmarks) that result across a wide variety of scenarios due to the use of our heuristic(s), are statistically significant with very high confidence levels. The percentage improvement, which at may at first glance seem small, are not at all trivial (i.e., a 2.6% on average improvement) especially when taken in context of the battles for customer retention and the size of the markets. It is important to recognize that these product line design issues impact a significantly large market especially with respect to revenues. For example, a loss of just 5% of the market share has been termed as a "disaster" for General Motors due to its significant monetary value (5% of a $350 billion market worldwide) and resulted in contributing to the ouster of its President for North American Operations [41]. In specific problem cases, we find that the improvement can be significantly even larger—up to an astronomical 12% improvement over the benchmark.

While all the GA based techniques have been shown to provide superior solutions over the beam search techniques, we have clearly demonstrated that by integrating both of these methods we can in the majority of cases achieve solutions more quickly than the pure GA technique. Similar to other authors in the past, we notice that due to the presence of very high quality BS based product lines (either in the initial population or via mutation) the population in later generations converges rapidly to a solution which may be farther from the optimal than if this hybridization/integration were not present. However, in general the integrated techniques result in quicker results and guarantee that the result will always be at least as good as the heuristic used for integration and, therefore, useful in contexts where quicker solutions are needed.

We would like to emphasize the fact that GA based techniques provide a viable means to designing multiple high quality product lines since their final result yields a population of high quality product lines which can then be evaluated by a human decision maker using subjective criteria. This can be very effective in a decision support context where a manager/decision maker is interested in the evaluation of a number of different scenarios and alternatives for a product line. One must keep in mind though, that the unique strings/product lines that we refer to in this research might actually include many product lines which while may be "nearly the same" (that is, the majority of their attributes may be equal). Hence this benefit, of many high quality product lines in the final solution may be mitigated to some extent since the decision maker may be faced with many "similar" options, which are very close to each other. In addition, even when a list of product lines is provided to a manager, it is not trivial to evaluate which out of this list is the most applicable, given the number of different possible attributes/features (such as cost of production, economies of scale, high-level strategic objectives of company and so on) which may make one product line superior to others.

Another managerially important issue that might have a major effect on the choice of the product line selection lies in recognizing the organizational realpolitik that underlies such choices. Minimizing organizational conflict might require the recognition that in all of the above techniques the market share is being maximized at the product line level. This might, however, result in products/brands within the line that have large variances in share between them. The resulting dissatisfaction among the brand managers particularly from those assigned a low share item might result in organizational conflict which strategically might be undesirable. A potential approach to mitigate such problems might be to include an explicit criteria that while maximizing market share at the line level seeks to minimize the inequity [42] in share between brands in the line. Researchers such as [40] have in the past, suggested a multi-objectives genetic algorithm based on the multi-criteria decision aid system due to [43]. Our approach currently merely provides a ranking of various solutions obtained based on numerical values of market share and does not utilize such a multi-criteria approach. However, as noted, given that the GA based approaches deal with the entire product lines and don't suffer from the shortcomings of the traditional heuristics that employ a "build up attribute by attribute" approach, multicriteria objectives as specified by the decision-makers are more easily handled and could be incorporated in future research projects.

The results of this study, therefore, seem to suggest that GA based solution techniques hold significant promise not only by themselves, but also in conjunction with other solution techniques such as beam search. This sort of integrated approach can, therefore, allow the decision maker to build on the specific strengths of each such technique and arrive at a combined/integrated technique that is more robust and therefore, has wider applicability in real life marketing contexts.

ACKNOWLEDGMENT

The authors wish to thank Y.-C. Kao, University of Texas at Dallas, for her help with the coding of the various heuristic techniques initially, and the anonymous reviewers, for the significantly detailed and insightful comments they provided which have greatly benefited this research.

REFERENCES

[1] P. V. Balakrishnan and V. S. Jacob, "Triangulation in decision support systems: algorithms for product design," *Decision Support Syst.*, vol. 14, pp. 313–327, 1995.

[2] P. E. Green and A. M. Kreiger, "A decision support model for selecting product/service positionings," *Eur. J. Operat. Res.*, vol. 147, pp. 187–202, 2002.

[3] Weirenga, Van Bruggen & Staelin, "The success of marketing management support systems," *Marketing Sci.*, vol. 18, no. 3, pp. 196–207, 1999.

[4] S. Swami, J. Eliashberg, and C. B. Weinberg, "SilverScreener: a modeling approach to movie screens management," *Marketing Sci.*, vol. 18, no. 3, pp. 352–372, 1999.

[5] P. E. Green and A. M. Krieger, "An application of a product positioning model to pharmaceutical products," *Marketing Sci.*, vol. 11, pp. 117–132, 1992.

[6] J. Couillard, "A decision support system for vehicle fleet planning," *Decision Support Syst.*, vol. 9, no. 2, pp. 149–159, 1993.

[7] L. Simpson, "Do decision makers know what they prefer?: MAVT and ELECTRE II," *J. Operat. Res. Soc.*, vol. 47, no. 7, 1996.

[8] S. Kirkpatrick, C. D. Gelatti, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[9] K. Smith, M. Palaniswami, and M. Krishnamoorthy, "A hybrid neural approach to combinatorial optimization," *Comput. Operat. Res.*, vol. 23, no. 6, pp. 597–610, 1996.

[10] F. Glover and M. Laguna, *Tabu Search*. Boston, MA: Kluwer Academic, 1997.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[12] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press, 1975.

[13] L. Burke, "Conscientious neural nets for tour construction in the traveling salesman problem: the vigilant net," *Comput. Operat. Res.*, vol. 23, no. 2, pp. 121–129, 1996.

[14] K. Tsuchiya, S. Bharitkar, and Y. Takefuji, "A neural network approach to facility layout problems," *Eur. J. Operat. Res.*, vol. 89, no. 3, pp. 556–563, 1996.

[15] A. Shtub, A. L. J. LeBlanc, and Z. Cai, "Scheduling programs with repetitive projects: a comparison of a simulated annealing, a genetic and a pair-wise swap algorithm," *Eur. J. Operat. Res.*, vol. 88, no. 1, pp. 124–138, 1996.

[16] C. C. Aggarwal, J. Orlin, and R. P. Tai, "Optimized crossover for the independent set problem," *Operat. Res.*, vol. 45, no. 2, pp. 226–234, 1997.

[17] D. W. Coit and A. Smith, "Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach," *Comput. Operat. Res.*, vol. 23, no. 6, pp. 515–526, 1996.

[18] D. Levine, "Application of a hybrid genetic algorithm to airline crew scheduling," *Comput. Operat. Res.*, vol. 23, no. 6, pp. 547–558, 1996.

[19] P. V. Balakrishnan and V. S. Jacob, "Genetic algorithms for product design," *Manag. Sci.*, vol. 42, no. 8, pp. 1105–1117, 1996.

[20] P. E. Green, J. D. Carrol, and S. M. Goldberg, "A general approach to product design optimization via conjoint analysis," *J. Marketing*, vol. 45, pp. 17–37, 1981.

[21] S. K. Nair, L. S. Thakur, and K. W. Wen, "Near optimal solutions for product line design and selection: beam search heuristics," *Manag. Sci.*, vol. 41, pp. 767–785, 1995.

[22] R. Kohli and R. Sukumar, "Heuristics for product line design using conjoint analysis," *Manag. Sci.*, vol. 36, pp. 311–322, 1990.

[23] P. E. Green and A. M. Kreiger, "Attribute importance weights modification in assessing a brands competitive potential," *Marketing Sci.*, vol. 14, no. 3, pp. 253–270, 1995.

[24] ——, "Recent contributions to optimal product positioning and buyer segmentation," *Eur. J. Operat. Res.*, vol. 41, pp. 127–141, July 1989.

[25] ——, "Models and heuristics for product line selection," *Marketing Sci.*, vol. 4, no. 1, pp. 1–19, 1985.

[26] D. Sudharshan, J. H. May, and A. D. Shocker, "A simulation comparison of methods for new product location," *Marketing Sci.*, vol. 6, pp. 182–201, 1987.

[27] R. Kohli and R. Krishnamurti, "A heuristic approach to product design," *Manag. Sci.*, vol. 33, no. 12, pp. 1523–1533, 1987.

[28] G. Dobson and S. Kalish, "Heuristics for pricing and positioning a product-line using conjoint and cost data," *Manag. Sci.*, vol. 39, pp. 160–175, 1993.

[29] L. Thakur, S. Nair, K. Wen, and P. Tarasewich, "A new model and solution method for product line design with pricing," *J. Operat. Res. Soc.*, vol. 51, no. 1, pp. 90–101, 2000.

[30] M. L. Fisher, "Worst case analysis of heuristic algorithms," *Manag. Sci.*, vol. 26, pp. 1–17, 1980.

[31] E. Falkenauer, "A hybrid grouping genetic algorithm for bin packing," *J. Heuristics*, vol. 2, no. 1, pp. 5–30, 1996.

[32] N. J. Radcliffe, "Forma analysis and random respectful recombination," in *Proc. 4th Int. Conf. Genetic Algorithms*, 1991.

[33] R. K. Ahuja and J. B. Orlin, "Developing filter genetic algorithms," *INFORMS J. Comput.*, vol. 9, no. 3, pp. 251–253, 1997.

[34] J. L. Bentley, "Experiments on Geometric Traveling Salesman Heuristics," AT&T Bell Laboratories, Holmdel, NJ, Computing Science Tech. Rep. 151, 1990.

[35] M. R. Garey and D. S. Johnson, *Computers and Intractability a Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[36] I. Lee, R. Sikora, and M. Shaw, "A genetic algorithm-based approach to flexible flow-line scheduling with variable lot sizes," *IEEE Trans. Syst., Man Cybern. B*, vol. 27, pp. 36–54, Feb. 1997.

[37] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *IEEE Trans. Syst., Man Cybern. B*, vol. 29, pp. 433–439, June 1999.

[38] R. A. Fisher, *Statistical Methods for Research Workers*. London, U.K.: Oliver and Boyd, 1925.

[39] S. Oman and P. Cunningham, "Using case retrieval to seed genetic algorithms," *Int. J. Computat. Intell. Applicat.*, vol. 1, no. 1, pp. 71–82, 2001.

[40] B. Rekiek, "Multiple-objectives genetic algorithm," in *Proc. Genetic Evolutionary Computat. Conf. (GECCO-99)*, Orlando, FL, July 1999, p. 401.

[41] D. Welch, The car guys take charge, in Business Week, Nov 26, 2001.

[42] P. V. Balakrishnan, A. Desai, and J. E. Storbeck, "Efficiency evaluation of retail outlet networks," *Environ. Planning B: Planning Design*, vol. 21, pp. 477–488, 1994.

[43] J.-P. Brans and B. Mareschal, "The PROMCALC and GAIA decision support system for multi-criteria decision aid," *Decision Support Syst.*, vol. 12, pp. 297–310, 1994.

P. V. (Sundar) Balakrishnan obtained his Ph.D. in marketing from The Wharton School, University of Pennsylvania in 1988. He also has a B.Tech. in textile engineering from I.I.T., Delhi in 1981, an M.S. in industrial engineering from the University of Texas at Arlington in 1983, and an A.M. in marketing from The Wharton School, Univ. of Pennsylvania in 1987.

He is currently an Associate Professor at the University of Washington, Bothell, of which he has served as Acting Director. He was also a Visiting Associate Professor at The University of Chicago and an Assistant Professor at The Ohio State University. His research papers have appeared in *Management Science*, *Psychometrika*, *Journal of Consumer Research*, *Decision Support Systems*, and the *European Journal of Operational Research* among others. His substantive areas of research are directed toward studying New Product Marketing & Strategy, Buyer-Seller Negotiations, Relationship Marketing, and Retail Locational Decisions. His research interests from a methodological standpoint are in working with innovative methodologies.

Rakesh Gupta (M'98) holds PhD and M.A. degrees in Management Information Systems from The Ohio State University in 1996 and an MBA from the University of California, Riversin in 1992. His undergraduate degree was in electrical and electronics engineering from the M.S.R. Institute of Technology, India, in 1990.

After receiving the PhD degree he joined the faculty of the Oklahoma State University where he was involved in the M.S. program in Telecommunications. In 1998 he joined the University of Texas at Dallas where he is currently an Assistant Professor of MSIS in the School of Management. His research areas include Wireless Networks, Machine Learning, Telecommunication Network Design, CATV Network Design, Location/Allocation of service facilities amongst others. Specifically, he has applied heuristic methods and algorithms to solve complex problems in Telecommunication Network Design and Routing, Product Design and Facility Location. Since 1998, he has been involved in developing and teaching telecommunications related courses in the Masters and Undergraduate MIS programs at UTD.

Dr. Gupta's research papers have appeared in the *European Journal of Operational Research*, *Decision Support Systems*, *Optical Networks* and others. He is a member of The Institute for Operations Research and the Management Sciences (INFORMS), The Decision Sciences Institute (DSI), IEEE Computer Society, and Association of Computing Machinery (ACM).

Varghese S. Jacob (M'84) obtained his Ph.D. degree in Management, majoring in management information systems, from Purdue University, West Lafayette, Indiana in 1986.

He is Senior Associate Dean and Professor of Management Science and Information Systems in the School of Management at the University of Texas at Dallas (UTD). Prior to joining UTD he was Associate Professor of Management Information Systems and Director of the Center for Information Technologies in Management in the Max M. Fisher College of Business at The Ohio State University. His research interests are in the areas of Artificial Intelligence, Data Quality, Decision Support Systems, and Electronic Commerce. His publications include articles in *Management Science*, *Information Systems Research*, *Decision Support Systems*, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, *European Journal of Operational Research*, *Psychometrika*, *Group Decision and Negotiation* and *International Journal of Man-Machine Studies*. He is the co-editor-in chief of the journal *Information Technology and Management* and serves as an Associate Editor for *Decision Support Systems*. He also serves on the editorial board of *Information Systems Frontiers: A Journal of Research and Innovation*.

Dr. Jacob is a member of The Institute for Operations Research and the Management Sciences (INFORMS), and Association of Computing Machinery (ACM). He has served as the Chair of the INFORMS Information Systems Society.