Forthcoming in *Computers and Operations Research*

An Investigation of Mating and Population Maintenance Strategies in Hybrid Genetic Heuristics for Product Line Designs

by

P.V. (Sundar) Balakrishnan

Rakesh Gupta*

Varghese S. Jacob

October 7, 2004

*Corresponding author.

P.V. Balakrishnan is an Associate Professor of Marketing, Business Administration Program, University of Washington, Bothell, WA 98021, email:sundar@u.washington.edu

R. Gupta is an Assistant Professor of MIS at the University of Texas at Dallas, Richardson, TX 75083. email:rgupta@utdallas.edu

V.S. Jacob is a Professor of MIS at the University of Texas at Dallas, Richardson, TX 75083. email:vjacob@utdallas.edu

Keywords: Decision Support, Genetic Algorithms, Marketing, Attribute Importance, Malthusian, Dynamic Programming, Hybrid heuristics, Conjoint Analysis,

Abstract

This research builds on prior work on developing near optimal solutions to the product line design problems within the conjoint analysis framework. In this research, we investigate and compare different genetic algorithm operators; in particular, we examine systematically the impact of employing alternative population maintenance strategies and mutation techniques within our problem context. Two alternative population maintenance methods, that we term "Emigration" and "Malthusian" strategies, are deployed to govern how individual product lines in one generation are carried over to the next generation. We also allow for two different types of reproduction methods termed "Equal Opportunity" in which the parents to be paired for mating are selected with equal opportunity and a second based on always choosing the best string in the current generation as one of the parents which is referred to as the "Queen bee", while the other parent is randomly selected from the set of parent strings. We also look at the impact of integrating the AI approach with a traditional optimization approach by seeding the GA with solutions obtained from a Dynamic Programming heuristic proposed by others. A detailed statistical analysis is also carried out to determine the impact of various problem and technique aspects on multiple measures of performance through means of a Monte Carlo simulation study. Our results indicate that such proposed procedures are able to provide multiple "good" solutions. This provides more flexibility for the decision makers as they now have the opportunity to select from a number of very good product lines. The results obtained using our approaches are encouraging, with statistically significant improvements averaging 5 % or more, when compared to the traditional benchmark of the heuristic dynamic programming technique.

1. Introduction

Traditionally the development of task specific Decision Support Systems (DSS) have focused on the use of a specific modeling approach (see for example [15, 7, 22]). The best solution provided by the modeling technique was then meant to be used by the decision-maker for his/her needs. There are at least two major problems with this approach. First, a significant number of decision problems for marketing decisions like product designs, media scheduling and retail site location, are analytically complex and belong to the NP-complete or NP-hard class of problems. This means that there are no known solution techniques that can provide optimal solutions to these decision problems in reasonable (polynomial) amounts of time. Consequently for these complex problems, researchers have turned to various heuristic and machine learning-based procedures to obtain "good" solutions [11, 17, 18, 19, 25]. Unfortunately, these heuristic solutions often work extremely well for certain variations of the problem and do not work quite as well for others. For greater usage of such models, it is preferable that the models find good solutions by using different solution techniques and at the same time provide a larger list of good candidate solutions which the decision makers are free to choose or employ as the basis for further analysis [1, 3].

This research builds on prior work [2, 3, 4], focusing on developing near optimal solutions to the product line design problems within the conjoint analysis framework [13]. In this research, we focus our investigation on comparing different genetic algorithm operators for product line designs to gain better experience. In particular, we examine systematically the impact of employing alternative population maintenance strategies and mutation techniques within our problem context.

In this research, we consider explicitly two alternative strategies that we term "Emigration" and "Malthusian" for population maintenance. These govern the relative harshness of the condition in terms of how individual product lines in one generation survive to the next generation as they impact how the population of candidate solutions is maintained over the course of the simulation. The relative harshness of the environment as specified by our operationalization of the Malthusian strategy implies that there is a tradeoff. Namely, the possibility that a specific high quality string will dominate the population after this

procedure has been carried out a large number of times, leading eventually to the fact that the diversity of the population may be reduced.

We then also examine in detail two different types of reproduction methods. The first is termed "Equal Opportunity" in which the parents to be paired for mating are selected with equal opportunity. The second reproduction method is based on always choosing the best string in the current generation as one of the parents which is typically referred to as the "Queen bee", while the other parent is randomly selected from the set of parent strings. The deployment of the Queen Bee strategy leads to the possibility that in the context of our problem, there is the possibility that the population will become homogeneous over the generations. We also look at the impact of integrating the AI approach with a traditional optimization approach by seeding the GA with solutions obtained from a Dynamic Programming heuristic [21]. A detailed statistical analysis is also carried out to determine the impact of various problem characteristics and heuristic technique aspects on multiple measures of performance through means of a Monte Carlo simulation study. Process tracing the evolution of the performance of these alternative implementations is also provided to give the reader some additional insight as to their relative impact.

Our results indicate that such proposed procedures are able to provide multiple "good" solutions. This provides more flexibility for the decision makers as they now have the opportunity to select from a number of very good product lines. The results obtained using our approaches are encouraging. We obtain statistically significant improvements averaging 5 % or more, when compared to the traditional benchmark of the heuristic dynamic programming technique.

The rest of the paper is as follows: in the second section we provide a brief description of the proposed approaches to tackle the well-known product line problem, and describe the different GA implementations we plan to investigate along with some limited computational comparisons. In section three, we describe the Monte Carlo simulation study and provide the detailed comparative computational results. Finally, in the last section, we provide our conclusions, discussions and directions for future work.

2. Product Line Problem and Solution Approaches

The product line design problem is known to be NP-hard [21], and consequently in practice it is difficult to obtain optimal solutions given the constraints of time. To address this issue, researchers have proposed alternative heuristic techniques to solve this general problem (see [14, 15, 27, 20, 8, 24, 28]). We now provide below the mathematical program formulation of the problem along with brief descriptions of the techniques directly relevant to our interest.

2.1 Mathematical Programming Formulation

The integer programming formulation for the product line design problem by [21] is as provided below:

Problem P :

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

s.t:

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

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

$$x_i - \sum_{m\in\Psi} x_{im} \geq 1 - M \qquad i \in \theta' \tag{4}$$

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

Where we have, the following parameters:

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

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

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

$\theta = \{1, 2, ... I\}$ 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 buyer $i \in \theta$

$\beta_{ijk} = W_{ijk} - W_{ij^*_{ki}k}$ 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)

Decision Variables:

$$x_{jkm} = \begin{cases} 1 \text{ if level j} \in \varphi_k \text{ appears in item } m \in \Psi \\ 0 \text{ otherwise} \end{cases}$$

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

$$x_i = \begin{cases} 1 \text{ if customer } i \text{ does not choose to switch from status quo product} \\ 0 \text{ otherwise} \end{cases}$$

In the above Problem P, the objective function (1) is specified to maximize the fraction of consumers who choose to adopt a product other than their status quo. Note that this is equivalent to minimizing the number of customers who choose to stay with their status quo product. We also specify with constraint (2) that there be at most one level of each attribute in each product (or item) in the product line. The restriction in constraint set (3) requires 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 1 only 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), ensure the integrality and binary nature of the decision variables.

2.2 Solution Approaches

The above integer program formulation for a particular instance of the product line design problem can utilize commercial integer program solvers such as CPLEX, GAMS etc. to attempt to obtain the solution. The difficulties of these solvers utilizing variations of the Branch and Bound approach for NP-Complete or NP-Hard problems are well known [9]. We show later our computational experience in solving a sample of product line design problems employing a commercially available solver (CPLEX, version 8.1) and thus

demonstrate the intractability of continuing down this path. Consequently, we next describe briefly some of the different algorithmic approaches available to tackle this problem.

Dynamic Programming Approach

Kohli et al [20] have suggested a dynamic programming approach to solving the product line design problem. If we consider M products and K attributes with J_k possible levels for attribute $k \in K$. Then this approach proceeds by constructing at step k-1, $M \bullet J_{k-1}$ partial product profiles for each level of attribute k (where $k \leq K$). Each partial product profile with level j of attribute k is constructed by augmenting level j of attribute k the $M \bullet J_{k-1}$ partial profiles constructed in step k-2. Of these partial product profiles, the best M are selected based on the incremental number of consumers for whom at least one selected partial product profile has a positive relative-part-worth utility. An advantage of this procedure is that it is relatively quick and easy to use. Conceptually, this technique essentially involves building a tree of attribute levels and confining the search to specific branches of this tree using heuristics about what would constitute a good solution if one were to stop at that level of the tree. This method however, often leads to several good solutions and possibly the optimal solution being discarded early in the solution process. Thus the final solution achieved using this technique may be relatively far away from the optimal. For a detailed discussion of this procedure, the reader is referred to [21].

This dynamic programming heuristic can proceed in a number of different ways depending on the sequencing rules utilized for considering the order in which attributes are selected for branching. The basic rules we have investigated here are: Ascending (in which the attributes with the fewest number of levels are considered first), Descending (which is the converse of the ascending method), Random (in which attributes are ordered at random) and Original (where the original sequence of attributes is maintained as the sequencing rule i.e., attribute 1 is considered first, attribute 2 second and so on). The original sequencing rule is arbitrary in nature and can be considered to be equivalent to the "Random" sequencing rule. In our computational tests we compare the performance of all of the above methods to our GA based methods and to traditional mathematical programming.

Genetic Algorithms

By now this is a reasonably well known heuristic and we reference the reader to Holland [17] who first proposed this concept of Genetic Algorithms (GA) and refer the reader to [12] and [23] for further details. Within the context of our problem, an initial set of strings (i.e., product line profiles) is generated in a random fashion to form 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 initial generation, a set of product lines can be randomly generated so that it has a diverse representation. Alternatively, part of this initial population can be "seeded" with certain user specified product lines generated obtained by another solution method.

We employ the encoding utilized by [2] for the product line design problem, wherein with M products (or items) in a product line, K_m attributes in product m ($m \in M$), L_{mk} levels for attribute k belonging to product m each product line has $P = \sum_{m=1}^{M} K_m$ positions. In a solution string, each position corresponds to a sub-string indicating the level of the attribute k in a specific item (or product) m. The fitness of each string is then evaluated in terms of the market share formulation specified in the math programming problem in the prior sub-section. The new candidate solution strings (product lines) to populate the next generation are then selected based on the fitness criteria and the operators such as crossover and mutation are applied resulting in a new generation of strings which are then evaluated again. Clearly, choosing the best control parameters (such as probabilities of crossover and mutation) is not a trivial task. A good study on the optimization of these parameters is provided in [16].

Two alternative methods that we term "Emigration" and "Malthusian" strategies are deployed to govern how individual product lines in one generation are carried over to the next generation. These strategies impact how the population of individuals is maintained over the course of the simulation. In the case of the emigration strategy, the best strings are selected for reproduction and the offspring created from them form the members of new generation resulting in a constant total of N strings. On the other hand, in the Malthusian strategy, the offspring of reproduction are added back into the population of the previous generation and then the best N of this larger population is selected to survive to the next generation. Thus in

the Malthusian strategy, the most fit (or highest quality) N strings pass on to become members of the next generation. This implies that there is a possibility that the same (or similar) high quality strings will dominate the population after this procedure has been carried out a large number of times, but it is conceivable that the diversity of the population may be reduced. Note that we deterministically select s (where $s < N$) strings for reproduction based on their fitness resulting in higher fitness strings being the first to be selected.

We also allow for two different types of reproduction methods. The first is termed "Equal Opportunity" in which the parents to be paired for mating are selected with equal opportunity (i.e. randomly paired up from the candidate set of parent strings). The second reproduction method is based on always choosing the best string in the current generation as one of the parents which is typically referred to as the "Queen bee", while the other parent is randomly selected from the set of parent strings. This second strategy (i.e., "Queen Bee") of course implies that each resulting child will potentially carry traits that belong to the Queen in that generation. Hence there is the possibility that the population will become more and more homogeneous as time goes by. In the results section we shall comment in more detail on impact of this phenomenon on the quality of the product lines designed.

The offspring are created by the exchange of genetic material between the parents employing a crossover technique such that sub-strings can be exchanged between two candidate parent strings. We implement crossover such that, a user-specified number of attributes r are randomly selected and exchanged between the two parent strings to get two new resulting feasible offspring. The probability of mutation of the offspring is also defined by the user and is performed such that the feasibility of resulting strings continues to be maintained. The strings are randomly chosen (without replacement) with some probability from the population. Then a single attribute is randomly picked and the level of that attribute is changed to a randomly chosen level within its feasible set.

A Hybrid Approach (GA-DP)

We propose a possible hybrid approach that integrates the two above procedures with the possibility of obtaining a better method. Given that the performance of GA's depends to a large extent on the quality of the initial population, it can be argued that developing such a hybrid approach might have two possible

consequences. One school of thought suggests that the solution strings in the initial population should be as random as possible. This will help to ensure that there is adequate diversity for the latter generations [1]. An alternative school has found good results when the initial population comprises of better quality starting solutions (see for example [5]).

In our hybrid approach, we "seed" the initial population with four product lines (i.e., solutions) obtained from the DP heuristic. Specifically, the four solutions are obtained from running the four DP variants on the problem. Specifically, we employ DP with the following attribute sequencing rules of Descending (DPD), Ascending (DPA), Random (DPR) and Original (DPO). We then generate the remaining N-4 members of the initial population in a random fashion. We then proceed with the running of the GA algorithm variants as described above.

Beam Search Based Approach

Beam Search (BS) is a breadth-first (without backtracking) search method developed in the 1970's for use in Artificial Intelligence (AI) problems related to image and speech recognition. Unlike traditional breadth-first search, BS allows the user to specify a specific breadth or beam width (b) which controls the number of "promising" nodes to explore at any level in the search tree. At one extreme if b is equal to the maximum number of nodes possible at that level of the search tree, then BS becomes equivalent to complete enumeration. In general however, b is kept small to ensure that the search does not take an inordinately large amount of computational time and memory while still providing good results. Depending on the value of b therefore BS provides a number of different solutions much like the number of individuals present in the final generation of a Genetic Algorithm.

Nair et al. [24] provide a BS based approach for solving the product line design problem. Specifically, their method works as follows: if we consider M products and K attributes with J_k possible levels for attribute $k \in K$, then the BS approach utilizes K relative part-worths matrices (i.e. the β matrix from the previous formulation), and initializes work matrices $A_l(\bullet)$ 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 ($l = 1$) $A_1(k) = \beta_k$ for each attribute $k \in K$. This approach proceeds by

iteratively combining the work matrices $A_l(\bullet)$ taking the matrices two at a time and forming matrices $E_l(\bullet)$ 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 $A_{l+1}(\bullet)$ 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.

As mentioned above, the beam width parameter (b) used in BS allows the technique to provide multiple ("good") solutions for the problem in question. Similarly, the population of individuals in the final generation of a GA based procedures provides the same. These multiple solutions can then be utilized to evaluate tradeoffs based on other non-quantifiable managerial criteria as specified by a decision maker. At the same time, a GA based search procedure involves a randomization component (specifically the mutation and cross-over operators) absent from BS which can help avoid local optima (for a more detailed comparison of the two approaches the reader is referred to 2). This therefore is a major difference between the two solution procedures. For instance, through the use of these operators GA based search could create individuals in the population which did not exist in the previous generation and hence avoid getting stuck in a locally optimal solution.

Given the scope of this paper, we do not examine the Beam Search algorithm any further. For the purpose of this paper our focus will be limited to the DP and GA and their hybrid heuristics. In the next section, our computational experience with these alternative solution procedures will be described.

3. Computational Testing and Performance Evaluation

In this section, we describe in some detail our computational study, experience and assess the relative performance of the various heuristic configurations in a large scale Monte Carlo study. We however, first demonstrate the relative inability of successfully using a state of the art integer programming package (CPLEX version 8.1) on the one hand and the viability of the alternative heuristic methods of GA and the hybridized GA-DP as described before. These were all run on a SPARC-20 Computer System.

Study 1:

The goal here as stated before was to assess the greater relative potential of GA based and GA-DP hybrid heuristic procedures in arriving at good solutions for the product line problems in reasonable amounts of time relative to the enormous computational times involved with branch and bound methods. To address this, we solved 8 instances of the product line design problems.

First we defined the eight different types of GA and hybrid GA procedures based on alternative choices of population maintenance, mating strategy and integration with the dynamic programming heuristic. These eight procedures (GA-E, GA-M, GA-Q-E, GA-Q-M, GA-E-DP, GA-M-DP, GA-Q-E-DP and GA-Q-M-DP) and their features are listed in Table 1. The parameter values that utilized in our computational tests are shown in Table 2. In each of the hybrid of GA and DP techniques (i.e., GA-E-DP, GA-M-DP, GA-Q-E-DP and GA-Q-M-DP) the four product line solutions discovered by using the dynamic programming heuristics (DPD, DPA, DPR and DPO) are "seeded" as members of the initial population with the remaining members randomly generated. The problems attempted here had the following characteristics: $M = 7$ items (products) in the product line, $K = 7$ or 9 attributes in each product and a market sample of 200 consumers. Our experience indicates the complexity of these problems as they were difficult to solve to optimality.

Insert Table 1 and Table 2 about here.

In Table 3, we provide data on the comparative performance of theses four different classes of solution approaches described in the previous section, namely, CPLEX; DP; GA; and GA-DP. Note that as CPLEX uses a branch and bound algorithm for arriving at the best integer solution it is time consuming and requires a large amount of memory since a sizeable branch and bound tree needs to be stored during the solution process. In Table 3, the best objective function value obtained by the CPLEX procedure after 20,000 seconds of CPU time is reported. This time limit for stopping that was set though arbitrary, arose from a need as several early runs on CPLEX took more than a week (over 600,000 seconds) of computational time and had still not converged, requiring them to be terminated since the branch and bound tree was consuming excessive (computer memory) space. This at one level is not dramatically surprising since the general integer programming problem is known to be NP-complete [10].

Insert Table 3 here

It can be seen from Table 4 that even after 20,000 seconds of computation time CPLEX failed to produce any solutions that were even close to those found by the different "flavors" of GA (which averaged approximately 30 seconds for each simulation). Interestingly, none of the Dynamic Programming heuristic methods performed as well as the GA or the GA-DP methods for all eight of the problems considered. The heuristic DP methods, however, to their credit took considerably less time (2-3 seconds) to reach their solution. But the solution value obtained was substantially inferior to the GA or the GA-DP solution. It, therefore, seemed logical to think that a hybrid solution procedure which is designed to combine the features of both the GA and DP technique might be successful, and which might thus explain the relative success in this limited set of the integrated (GA-DP) methods.

Study 2

In the second study, a systematic and detailed comparison of the various heuristic approaches, and problem characteristics of number of products in line, number of attributes, presence (or absence) of attribute importance on specific dependent variables are considered. We briefly describe the Monte Carlo simulation study; discuss the performance of various GA, GA-DP and DP techniques, and the impact of the problem characteristics on the performance of the heuristics.

A 2x2x2 full factorial design to assess the importance of number of products in the line (4 or 7); the number of attributes (7 or 9) in the product category; and the presence (or absence) of attribute importance was employed. Note that for each of the problems, the number of levels for each attribute was randomly chosen to be a number from 3 to 7. The part worth utility values were generated randomly from a uniform distribution and then normalized within consumers, thereby allowing interpersonal comparisons between consumers, and consumer was randomly assigned an idiosyncratic status-quo product profile (see [2] for additional details on the data generation process). We employed 10 replicates of each problem instance resulting in a total of 80 different problem sets. Each problem set was solved for the "optimal" product line employing four variants of the DP heuristic and the eight GA based heuristic techniques. An analysis of variance (ANOVA) tests on a number of different performance measures was then conducted to assess the effect of the alternative factors.

Insert Table 4 about here.

The primary dependent variable of performance was to compute for each of the eight GA or GA-DP heuristic the ratio of the best solution obtained to the best across the four DP solutions. We define this as:

$$\text{PBST} = \frac{\text{BST}}{\max\{\text{DPD}, \text{DPA}, \text{DPR}, \text{DPO}\}}$$

where:
BST = Best GA or GADP product line
DPD = Best product line due to DPD
DPA = Best product line due to DPA
DPR = Best product line due to DPR
DPO = Best product line due to DPO

Additional performance metrics of interest for each of the eight GA types that we considered besides the best solution (BST) are the worst string in the population (WORST), the average solution value of the product lines in the final population (AVG), the standard deviation (SD) of the strings in the final population, and the generation number at which the best solution is found (GENNO). We also explicitly evaluate the

impact of the problem factors and solution types on the number of unique strings in the final GA population. To this end, we count the number of unique strings with the best fitness 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 final population. See Table 5 for the legend of list of notations of the various performance characteristics.

Insert Table 5 about here.

A statistical analysis of variance of the heuristic runs across the 80 data sets "solved" for optimal product lines employing each of the eight GA types are reported in Table 6. Across all 640 simulation runs on average the GA based heuristics significantly outperform by a factor of over 5.3% the best solutions found across the four DP heuristics.

Insert Table 6 about here.

A quick examination of the column PBST from Table 6 shows that we find that each of the eight GA based heuristics (GA-E, GA-M, GA-Q-E, GA-Q-M, GA-E-DP, GA-M-DP, GA-Q-E-DP and GA-Q-M-DP) consistently outperformed the best solution from among the four DP based procedures. In fact, a simple t-test for the difference in the objective function attained between DP and GA based procedures shows that the GA based techniques (all the eight variants) provide a share of choices significantly greater than the best of the DP based techniques ($p < 0.0001$). The worst performance on average which was obtained using GA-Q-E still outperformed the best across four variants of DP by more than 2.5%. The best performance averaged across all 80 data sets was achieved with GA-M-DP and it outperformed by over 6.7% the best across variants of DP. Clearly, the type of GA technique utilized (GA Type), makes a statistically significant difference in the best product line designed after 100 generations.

The GA techniques which utilize the Malthusian strategy for population maintenance, namely GA-M, GA-Q-M, GA-M-DP, and GA-Q-M-DP resulted in higher quality product lines being designed on average than their counterpart GA techniques (GA-E, GA-Q-E, GA-E-DP, and GA-Q-E-DP) which employed an Emigration strategy (see Table 6). Our intuition for this is that since the Malthusian strategy allows only the

highest quality strings to survive in the next generation and culls the weakest at every generation, this implies that the best few strings are selected to be parents. The Malthusian strategy, therefore, increases the chances that the higher quality strings of the current generation will survive till the next generation and any weak offspring produced will get discarded. Another key observation deals with the diversity of the resulting population of product lines given the different methods. As expected, while the Malthusian strategy forces the lower quality strings to "die out" sooner, this also results in the diversity in the population being reduced significantly as compared to techniques using the emigration strategy. From Tables 6 and similarly from Table 7 it can be observed that application of the Malthusian strategy results in a final population of product lines that is of high quality but lower diversity for the good (FIVE and OTOFV) but less than the best (ONE) solution obtained.

The use of the Queen bee based reproductive technique, on the other hand, seems to result in strings that are lower in quality on average (PBST) than those employing an Equal opportunity mating strategy. However, the use of this mating strategy mechanism tends to converge towards a high quality population earlier (GENNO) than the other techniques. We hypothesize, that this phenomenon may be due to premature convergence onto local minima. This phenomenon seems consistent with the schemata theory of Goldberg [12]. The standard deviation of the quality of strings in the final population (SD) is also statistically different across the different GA techniques ($p < 0.0001$). Further evidence for the argument that the Queen bee based procedures suffer from premature convergence is provided since the Queen bee based techniques result in the lowest standard deviation values of the share of choices of the product lines in the last generation.

Insert Figures 1, 2, 3 and 4 about here

The hybrid techniques integrating GA and DP heuristics (i.e., GA-E-DP, GA-M-DP, GA-Q-E-DP and GA-Q-M-DP) all tend to find better solutions (PBST, BST) and better average candidate product lines (AVG) than their counterpart pure genetic algorithm based methods (i.e., GA-E, GA-M, GA-Q-E and GA-Q-M). The process tracing results of the performance metric AVG across the 100 generations are shown in Figures 1 to 4. These depict the change in the average fitness of the population over a typical simulation run

of the GA and show graphically this tendency toward premature convergence (GENNO) on the part of the hybrid approaches. We speculate that this is due to the four relatively high quality product lines being included in the initial generation in the integrated methods which drives the population to converge sooner. Again among the integrated methods the Queen bee based methods result in the quickest convergence (i.e. the best candidate product line is found in the earliest generation).

Insert Figures 5, 6, 7, and 8 about here

The process tracing results (see Figures 5-8) show 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 pure genetic algorithm based methods the best product line found initially is of substantially lower quality than that when using the integrated methods. This is not surprising since the integrated methods have an initial population that includes product lines found using the DP methods. However, over a simulation run one can see the both pure GA as well as integrated 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 (i.e. without waiting for a long simulation run) then the integrated methods might be better suited for him/her as opposed to the pure GA techniques.

The impact of the problem characteristics such as the number of attributes, number of products in the line and presence or absence of attribute importance, and the GA type on the performance metrics of interest specified previously (see Table 5) were explicitly teased out using analysis of variance (see Tables 6 and 7).

Insert Table 7 about here

The managerially pre-specified number of products, either four or seven, in the product line (p <0.0001) significantly impacts, as should be expected, all of the measures of unique strings (ONE, FIVE and OTOFV) in the last generation and in a positive direction (Table 6). Increasing the number of products or number of attributes, increase the number of unique strings in the last generation. The length of the product line (PLINE) significantly affects all of the performance metrics. The number of attributes (NATR) however does significantly affects all of the performance metrics except PBST ($p>0.06$). That is, going from seven to

nine attributes does not have a huge impact on performance improvement. But, the interaction of NATR with Attribute Importance (ATIMPT) is significant though does not account for a lot of the variance explained in the performance variables PBST, SD, ONE, and FIVE. As the problems become more complex the DP based methods become less effective while the GA methods continue to perform well (PBST).

The three-way interaction[1] for problem size and complexity (PLINE*NATR*ATIMPT) was specifically examined. This three-way interaction in terms of complexity and size had no significant impact on any of the performance metrics in terms of the quality of the solution found. It did however significantly impact the number of unique solutions (ONE, FIVE and OTOFV) discovered by the GA heuristic approaches. The two-way interaction for problem size PLINE*NATR had a statistically significant impact on the improvement in the best solution found relative to the solution found using DP heuristics (PBST). It also significantly affected the number of good unique strings (ONE and FIVE) found in the last generation but had no impact on the other performance metrics. Interestingly, the two-way interaction of the type of GA heuristic employed and the number of items required in the product line (GATECH*PLINE) had a statistically significant impact on all of the performance metrics except the generation number (GENNO) in which the best solution was first identified. Clearly, more studies of this interaction terms effect are needed in the future to see if this holds up as it can play an important role the managerial and analysts decisions.

We found that the absence or presence of attribute importance significantly affects the best solution (BST) found relative to DP (PBST). It is interesting to observe that the presence of attribute importance seems to result in lower PBST values ($p=0.0001$). That is, this seems to make the problems significantly easier to solve for the DP methods. While Table 6 shows the performance of different GA based methods when tested on problem instances with and without attribute importance, Table 7 shows the performance of different GA based methods when tested on the problem instances in the absence of attribute importance. We note that the PBST value in Table 7 ranges from 1.026 to 1.092 indicating an improvement ranging from 2.6% to 9.2%. The DP methods clearly seem to perform better in the presence of attribute importance due to the fact that given four different sequencing rules (DPA, DPD, DPR and DPO) it is likely that one of the

[1] We thank a reviewer for suggesting this line of inquiry.

four DP based methods chooses the correct level of the most important attribute for the product line. This will result in a product line which will in all likelihood be very close to the best product line since the most "important" attribute will make the largest contribution to the objective function as opposed to the other lesser important attributes.

4. Conclusions

In this research, we have provided an empirical demonstration of the viability of Genetic Heuristic techniques for the product-line design problem – a problem that is known to belong to the NP-hard class of problems. We have also employed several additional alternative solution approaches using artificial intelligence. Specifically, we have deployed successfully a relatively novel hybrid approach that is based on the integration of two different techniques, namely Genetic Algorithms and Heuristic Dynamic Programming. This approach based on these results seems to have some merit given the practical difficulty of applying traditional mathematical programming techniques to this important problem to obtain good solutions in reasonable time. In particular, we have carried out a Monte Carlo simulation study to investigate impact of alternative population maintenance strategies and mating techniques on the multiple performance measures. We then conducted an elaborate statistical analysis to examine and obtain insights into the impact of seeding genetic heuristics with DP solutions as well as various problem specific characteristics such as the length of the product line, attribute importance and number of attributes in the product category.

The GA based heuristics provided superior solutions than the Dynamic Programming techniques, and by integrating these methods in the majority of cases we are able to achieve solutions superior to any of these techniques when employed in isolation. In general, the integrated techniques also resulted in quicker results and therefore need to be clubbed with a population maintenance technique such as the Malthusian strategy we have employed so as to increase the diversity in the population while at the same time maintaining its quality. For the analysts, it is clear that choices made in implementation and deployment of such genetic heuristics can affect the quality of the solutions obtained. Additionally, the recognition that there is a multi-dimensional nature to evaluating the quality of the obtained solutions implies that there will always be

tradeoffs to be made. This also implies that utilizing cybernetic principles of deploying a variety of solution approaches has much to recommend.

The results of this program of study therefore seem to suggest that GA based solution techniques hold significant value not only by themselves, but also in conjunction with other solution techniques such as dynamic programming or 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. While a few percent improvements in market share for a firm may not seem like a very large amount, one must keep in mind the size of the over all market and the jockeying for position. Automotive manufacturers, where the magnitude of the market size are huge, such as Chrysler for instance have befitted from employing conjoint analysis in the design of their successful launch of the Neon car [25]. Clearly, firms could conceivably benefit from a small improvement in market share due to the use of heuristic's such as the ones presented in this research for product line design, as it would have a significant, positive impact to their organization.

Acknowledgement

The authors acknowledge the contribution of Yi-ching Kao for her extensive help with the coding of the various heuristic techniques. Prof. Balakrishnan acknowledges the generous hospitality of the Graduate School of Business of The University of Chicago for hosting him during his sabbatical when some of the work on this paper was done.

References

[1] R. K. Ahuja, J. B. Orlin, "Developing Filter Genetic Algorithms", INFORMS Journal on Computing, Vol. 9, No. 3, 251-253, 1996.

[2] P.V. Balakrishnan, R. Gupta and V. S. Jacob. "An Investigation of Hybrid Genetic Algorithm Techniques for Product Line Design", IEEE Transactions on Man Systems Cybernetics-Part B vol. 34, No. 1, pp. 468-483, 2004

[3] P. V. Balakrishnan and V. S. Jacob, "Triangulation in Decision Support Systems: Algorithms for Product Design", Decision Support Systems, 14, 313-327, 1995

[4] P. V. Balakrishnan and V. S. Jacob, "Genetic Algorithms for Product Design", Management Science, Vol. 42, No. 8, 1105-1117, 1996

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

[6] D. W. Coit, A. Smith, "Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach", Computers & Operations Research, v23n6, 515-526, 1996

[7] J. Couillard, "A Decision Support System for Vehicle Fleet Planning", Decision Support Systems, Vol. 9, No. 2, 149-159, 1993

[8] G. Dobson and S. Kalish, "Heuristics for Pricing and Positioning a Product-line Using Conjoint and Cost Data", Management Science, Volume 39, pp. 160-175, 1993.

[9] M. L. Fisher, "Worst Case Analysis of Heuristic Algorithms", Management Science, 26, 1-17, 1980

[10] M. R. Garey, D. S. Johnson, "Computers and Intractability A Guide to the Theory of NP-Completeness", Freeman, 1979.

[11] F. Glover and M. Laguna. "Tabu Search". Kluwer Academic Publishers, Boston, Massachusetts, 1997.

[12] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, 1989.

[13] P. E. Green, J.D. Carrol, and S.M. Goldberg, "A General Approach to Product Design Optimization via Conjoint Analysis," Journal of Marketing, Vol. 45, 17-37, 1981

[14] P. E. Green, and A.M. Krieger, "Models and Heuristics for Product Line Selection", Marketing Science, Vol. 4, No. 1, 1-19, 1985

[15] P. E. Green and A. M. Kreiger , "An Application of a Product Positioning Model to Pharmaceutical Products", Marketing Science, Vol. 11, 117-132, 1985

[16] J. J. Grefenstette, "Optimization of Control Parameters for Genetic Algorithms", IEEE Transactions on Systems, Man and Cybernetics, SMC-16, 122-128, 1986

[17] J. H. Holland, "Adaptation in Natural and Artificial Systems", The University of Michigan Press, MI, 1975.

[18] J. J. Hopeld and Tank, D. W. "Neural computation of decisions in optimization problems." Biological Cybernetics, 52, 141-152, 1985.

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

[20] R. Kohli and R. Krishnamurti, "A Heuristic Approach to Product Design", Management Science, Vol. 33, No. 12, 1523-1533, 1987

[21] R. Kohli and R. Sukumar, "Heuristics for Product Line Design using Conjoint Analysis", Management Science, Vol. 36, 311-322, 1990

[22] D. Levine, "Application of a hybrid genetic algorithm to airline crew scheduling", Computers & Operations Research, v23n6, 547-558, 1996

[23] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", Springer-Verlag, Berlin, 1992.

[24] S. K. Nair, L. S Thakur and K. W. Wen,, "Near optimal solutions for product line design and selection: Beam search heuristics", Management Science; 41 , 767-785, 1995

[25] S. Player, "Hitting the Profitability Bulls Eye", Controller Magazine, p. 81, 1997

[26] H.-P. Schwefel. "Evolution and Optimum Seeking". John Wiley and Sons, Inc., New York, New York, 1995

[27] D. Sudharshan, J.H. May, and A.D. Shocker, "A Simulation Comparison of Methods for New Product Location", Marketing Science, 6 (Spring 1987), 182-201.

[28] L. S. Thakur, S. K. Nair, K-W Wen and P. Tarasewich, "A New Model and Solution Method for Product Line Design with Pricing", Journal of the Operational Research Society, 51, 2000, 90-101

Table 1

GA Heuristic Approaches Investigated

Symbol of GA Approach	Population Maintenance Strategy	Mating Technique	Seeded with DP Solutions
GA-E*	**E**migration	Equal Opportunity	No
GA-M	Malthusian	Equal Opportunity	No
GA-Q-E	Emigration	Queen Bee	No
GA-Q-M	Malthusian	Queen Bee	No
GA-E-DP	Emigration	Equal Opportunity	Yes
GA-M-DP	Malthusian	Equal Opportunity	Yes
GA-Q-E-DP	Emigration	Queen Bee	Yes
GA-Q-M-DP	Malthusian	Queen Bee	Yes

*For reasons of space, in some of the Tables and text, the hyphens in the GA type are eliminated.

Table 2

Parameters used for Genetic Algorithm based Heuristics

Parameters	Value of Parameter
Mutation	0.3
Population size	200
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
Stopping Criteria (# of generations)	100

Legend:

M = Number of Products in the Line

K = Number of Attributes in the Product Category

Table 3

Performance of GA, CPLEX and Dynamic Programming Solutions on Eight Problems

M	***K***	**DP**				**GA**								**CPLEX**
		DPA	**DPD**	**DPR**	**DPO**	**GAE**	**GAM**	**GAQM**	**GAQE**	**GAEDP**	**GAMDP**	**GAQEDP**	**GAQMDP**	
7	9	0.939	0.944	0.939	0.954	0.980	0.985	0.985	0.980	**1.000**	0.995	0.990	0.980	0.624
7	7	0.957	0.978	0.768	0.978	**1.000**	**1.000**	**1.000**	0.995	**1.000**	**1.000**	0.995	0.995	0.735
7	7	0.697	0.939	0.727	0.864	0.962	**1.000**	0.962	0.962	0.970	0.977	0.977	0.977	0.402
7	9	0.747	0.856	0.630	0.877	0.856	0.932	0.945	0.952	0.945	0.966	0.952	**1.000**	0.432
7	7	0.735	0.902	0.848	0.894	0.970	**1.000**	0.955	0.932	0.985	0.970	0.947	0.985	0.598
7	9	0.757	0.914	0.843	0.929	0.914	0.964	0.929	0.943	0.971	**1.000**	0.993	0.993	0.564
7	7	0.935	0.984	0.935	0.967	**1.000**	**1.000**	0.995	0.995	0.995	0.995	0.995	**1.000**	0.690
7	9	0.927	0.974	0.974	0.943	**1.000**	0.979	0.990	0.979	**1.000**	**1.000**	0.990	**1.000**	0.674

Legend:

M = Number of Products in the Line

K = Number of Attributes in the Product Category

Dpi Dynamic Programming Heuristic where I ε {**A**scending, **D**escending, **R**andom, **O**riginal} Sequence

Note: Objective function values are represented as a percentage of the largest value obtained from all methods. Values obtained from CPLEX (Version 6.0) are those after 20,000 seconds of CPU time.

Items in bold indicate highest value in column

Table 4:

Experimental Design for Study 2

Treatment	**Values**	
Number of Products (PLINE)	4	7
Number of Attributes (NATR)	7	9
Attribute Importance (ATIMPT)	Present	Absent
Number of Replications of each problem variant (PLINE*NATR*ATIMPT)	10	
Number of GA Approaches (GA Type)	8	
Number of DP heuristics employed (DPi)	4	

Table 5:

Notation and Legend of Performance Characteristics:

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

$$\text{PBST} = \frac{\text{Best of GA Type}}{\text{Best across all DP}}$$

BST	Best Share-of-Choices solution for the GA heuristic approach employed
WORST	Share of choices of worst product line in final population of GA Technique
AVG	Average of the share of choices of product lines in final population of GA Technique
SD	Standard Deviation of share of choices of product lines in final population of GA Technique
GENNO	Generation number at which best product line was first found
ONE	Number of unique product lines with best share of choices value 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 in the GA final population

Table 6: ANOVA of Simulation Runs: Investigation of Problem Characteristics and GA Approach on Performance

Parameter	**Level**	**No. of Obs.**	**PBST**	**BST**	**WORST**	**AVG**	**SD**	**GENNO**	**Average Number of Unique Strings**		
									ONE	FIVE	OTOFV
PLINE	4	320	1.060	0.636	0.559	0.626	0.014	49.52	2.68	17.61	11.55
	7	320	1.045	0.800	0.716	0.786	0.016	61.59	58.05	42.30	15.49
	p-Value		*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*
NATR	7	320	1.050	0.708	0.629	0.697	0.015	49.20	34.42	23.38	12.36
	9	320	1.055	0.728	0.645	0.725	0.0016	61.91	26.31	36.52	14.68
	p-Value		*(0.07)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0109)*	*(0.0001)*	*(0.0076)*
ATIMPT	NO	320	1.068	0.578	0.508	0.563	0.015	59.41	2.28	19.17	19.17
	YES	320	1.037	0.85	0.766	0.848	0.016	51.70	58.45	7.87	7.87
	p-Value		*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0047)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*
GA Type	GA-E	80	1.0487	0.715	0.545	0.685	**0.031**	**82.97**	12.8	**70.96**	**33.07**
	GA-M	80	1.0641	0.720	0.720	0.721	0.0010	67.68	30.67	22.16	0
	GA-Q-E	80	1.0254	0.702	0.546	0.684	0.030	40.75	31.83	31.81	23.45
	GA-Q-M	80	1.0535	0.717	0.717	0.717	0.0027	53.20	**42.98**	1.37	0
	GA-E-DP	80	1.0578	0.721	0.559	0.695	0.031	70.82	16.25	66.73	27.48
	GA-M-DP	80	**1.0670**	**0.726**	**0.724**	**0.725**	0.0005	53.76	38.43	7.97	0
	GA-Q-E-DP	80	1.0469	0.715	0.565	0.696	0.030	32.47	31.28	32.26	23.83
	GA-Q-M-DP	80	1.0605	0.723	0.722	0.722	0.0002	42.81	38.68	6.37	0
	p-Value		*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0047)*	*(0.0001)*	*(0.0001)*	*(0.0001)*	*(0.0001)*
Overall Mean		*640*	**1.053**	**0.719**	**0.638**	**0.706**	**0.018**	**55.56**	**30.4**	**29.9**	**13.5**

R-Square	0.33	0.98	0.97	0.98	0.95	0.41	0.51	0.51	0.65

Table 7: Performance of GA Heuristics Averaged Across 40 Problem Instances without Attribute Importance

GA Type	**PBEST Ratio**	**BST**	**WORST**	**AVG**	**SD**	**GENNO**	**Average Number of Unique Strings**		
							ONE	**FIVE**	**OTOFV**
GA-E	1.0558	0.569	0.425	0.530	**0.0305**	**86.2**	1.97	44.90	**50.52**
GA-M	1.0863	0.588	0.580	0.582	0.0017	72.70	2.30	20.87	0.72
GA-Q-E	1.0260	0.556	0.428	0.537	0.0286	48.50	2.17	18.57	32.72
GA-Q-M	1.0731	0.581	0.580	0.580	0.0004	59.72	2.12	2.42	0
GA-E-DP	1.0729	0.580	0.437	0.549	0.0302	68.92	**2.60**	**45.02**	37.67
GA-M-DP	**1.0924**	**0.592**	**0.588**	**0.589**	0.0009	62.75	2.55	7.22	0
GA-Q-E-DP	1.0614	0.576	0.444	0.555	0.0302	35.02	2.32	17.90	31.75
GA-Q-M-DP	1.0810	0.586	0.585	0.585	0.0004	41.50	2.2	1.07	0
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

Figure 1: Comparing GA-E vs. GA-E-DP: Evolution of Average Product Line Quality in Population

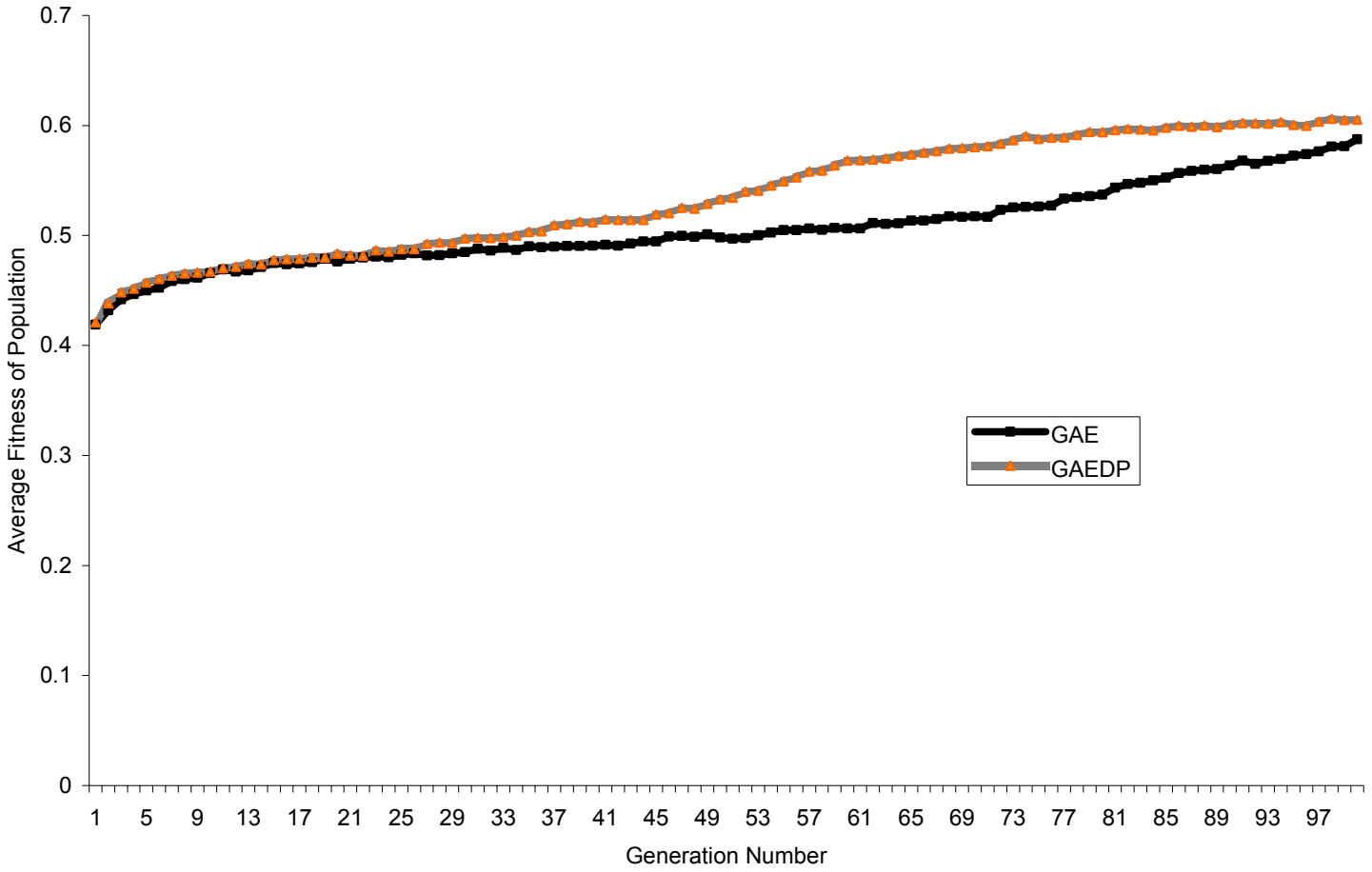


Figure 2: Comparing GA-M vs. GA-M-DP - Evolution of Average Quality of population

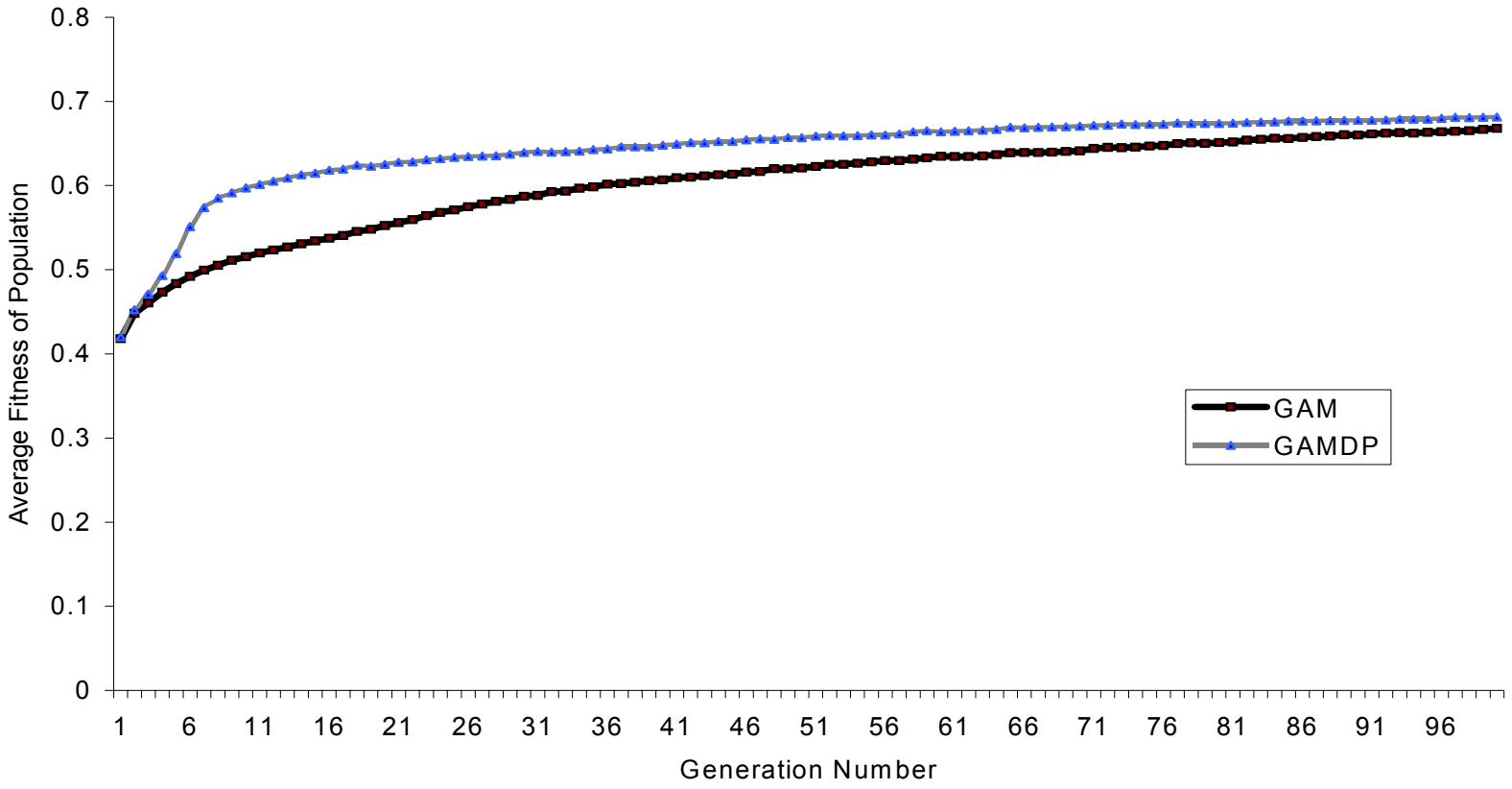


Figure 3: Comparing GA-Q-E vs. GA-Q-E-DP: Evolution of Average Quality of Product Lines

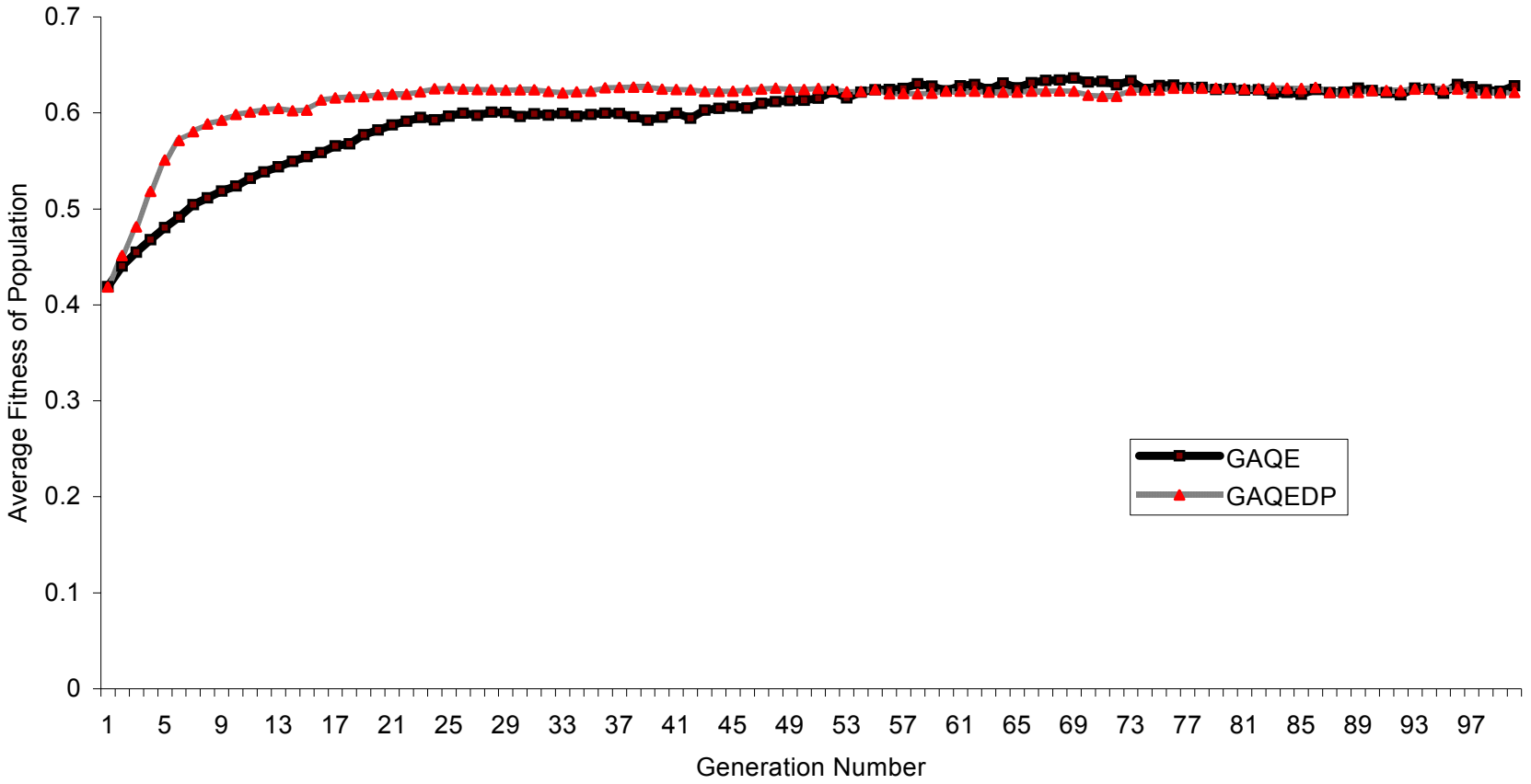


Figure 4: Comparing GA-Q-M vs. GA-Q-M-DP: Evolution of Average Quality of Product Lines

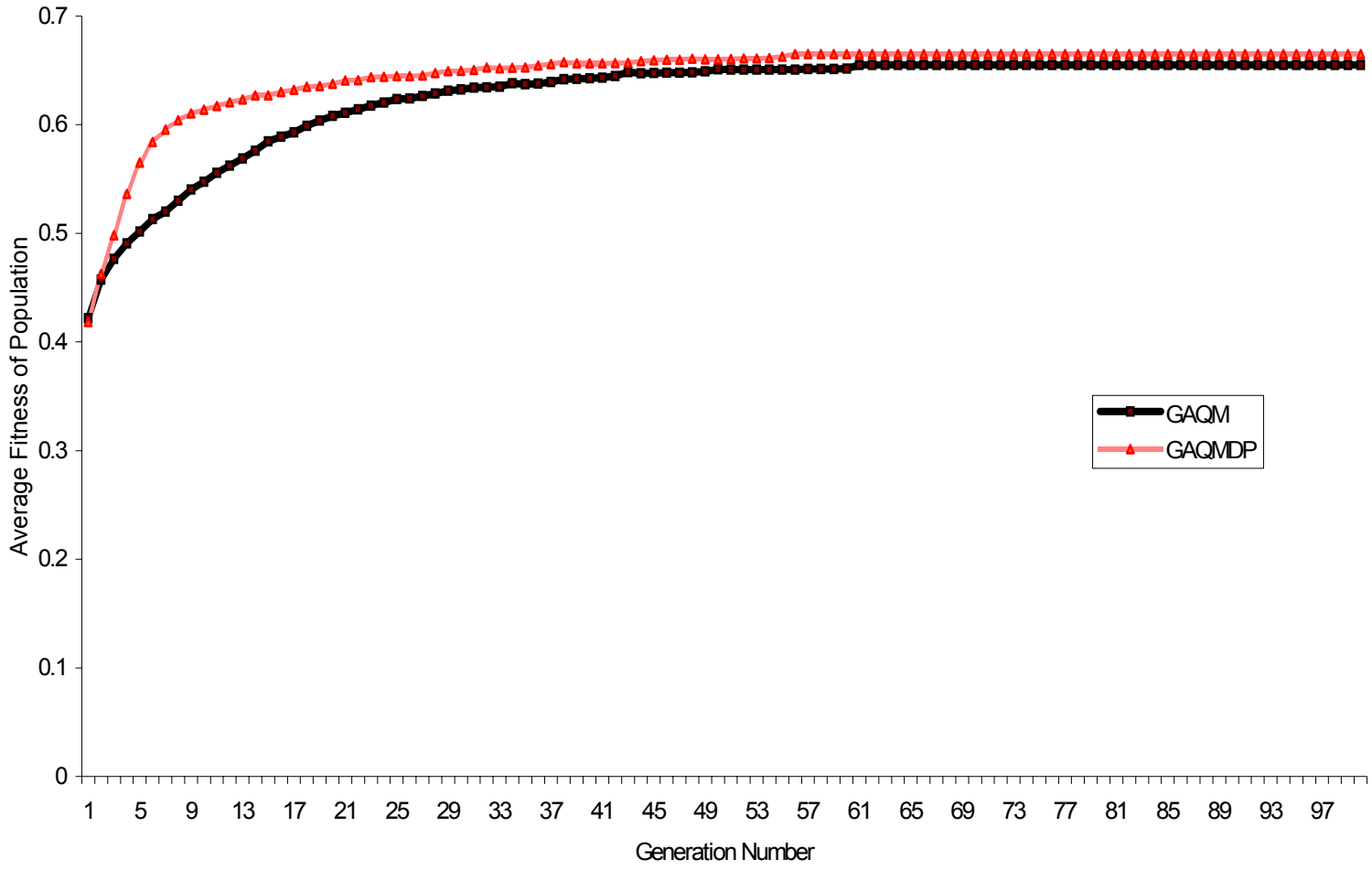


Figure 5: Best Product Line Value over the course of a simulation: GA-E vs. GA-E-DP

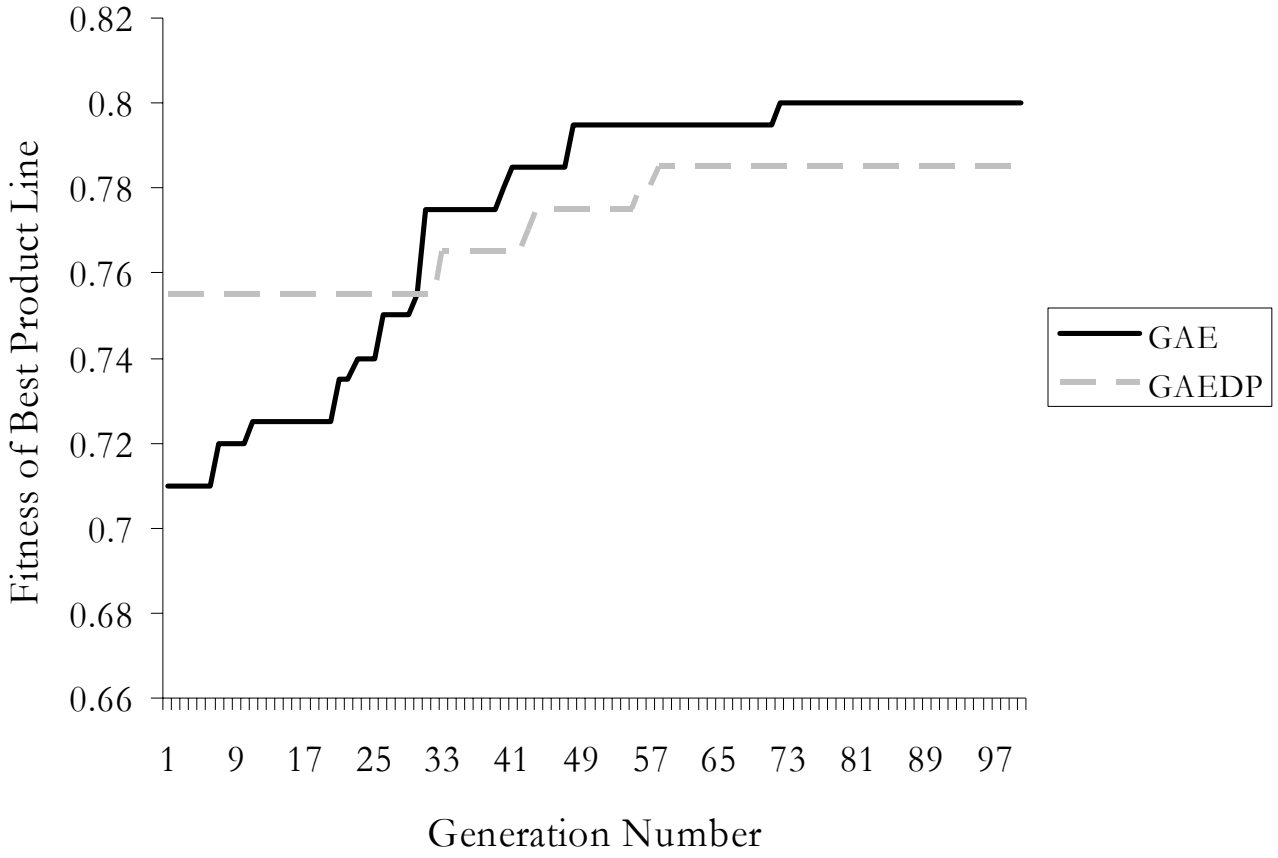


Figure 6: Best Product Line found over the course of a simulation: GA-M vs. GA-M-DP

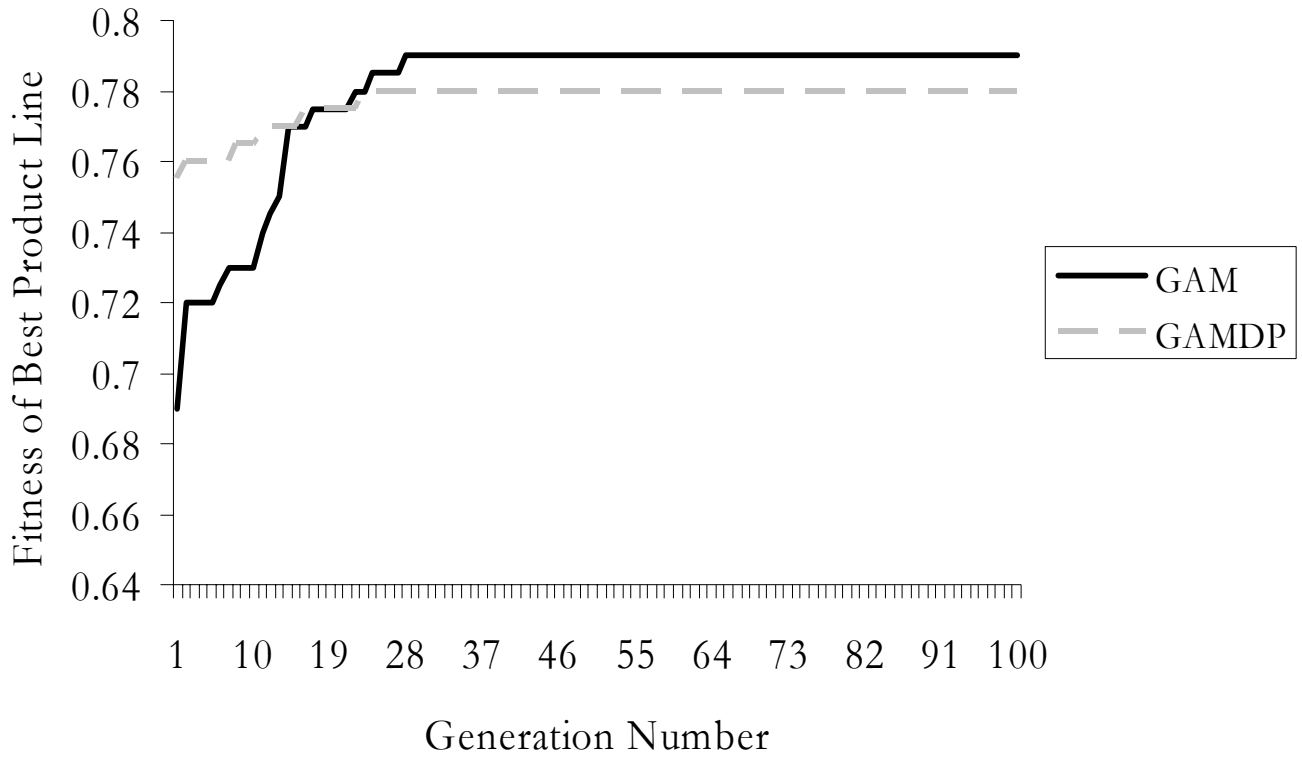


Figure 7: Best Product Line found over the course of a simulation: GA-Q-E vs. GA-Q-E-DP

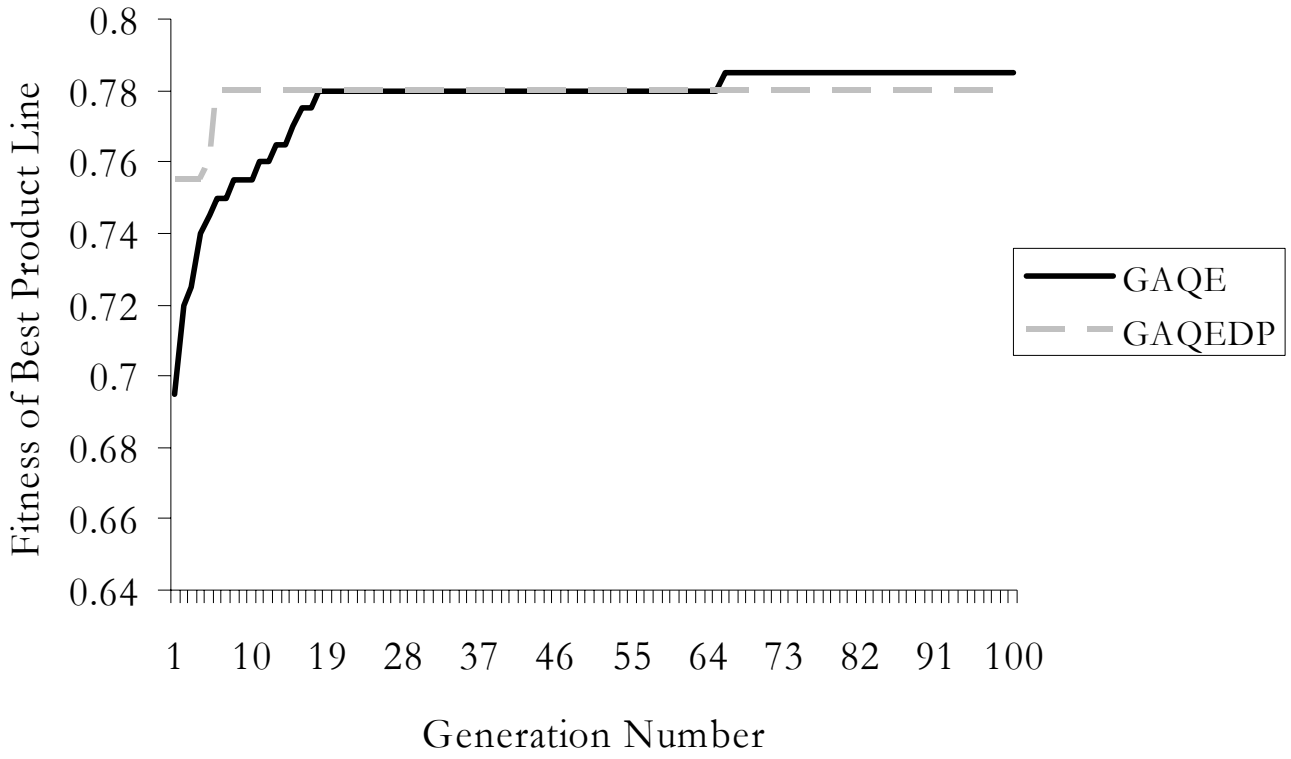
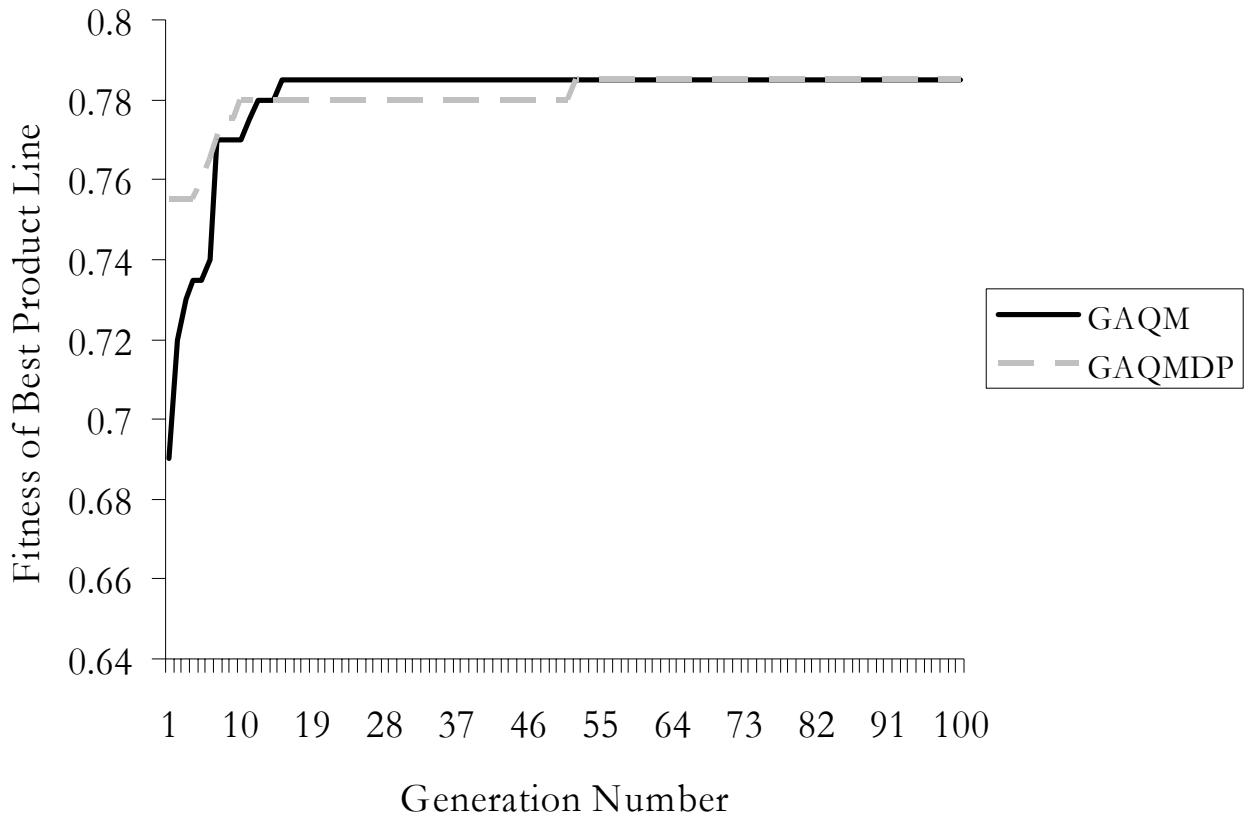


Figure 8: Best Product Line found over the course of a simulation: GA-Q-M vs. GA-Q-M-DP



CAPTIONS