ASME 2016 International Design Engineering Technical Conferences and
Computers and Information in Engineering Conference
IDETC 2016
August 21-24, 2016, Charlotte, North Carolina

IDETC2016-59534

INTELLIGENT GENETIC ALGORITHM CROSSOVER OPERATORS FOR MARKET-DRIVEN DESIGN

Kevin M. Young
Graduate Research Assistant
North Carolina State University
Raleigh, NC 27695, USA
kmyoung4@ncsu.edu

Scott M. Ferguson[1]
Associate Professor
North Carolina State University
Raleigh, NC 27695, USA
scott_ferguson@ncsu.edu

ABSTRACT

Heuristic algorithms have been adopted as a means of developing solutions for complex problems within the design community. Previous research has looked into the implications of genetic algorithm tuning when applied to solving product line optimization problems. This study investigates the effects of developing informed heuristic operators for product line optimization problems, specifically in regards to optimizing the market share of preference of an automobile product line. Informed crossover operators constitute operators that use problem-related information to inform their actions within the algorithm. For this study, a crossover operator that alters its actions based on the relative market share of preference for each product within product lines was found to be most effective. The presented results indicate a significant improvement in computational efficiency and increases in market share of preference when compared to a standard scattered crossover approach. Future work in this subject will investigate the development of additional informed selection and mutation operators, as well as problem informed schema.

1 INTRODUCTION

Advancements in market research have allowed for more richer representations of customer preferences in heterogeneous markets [1–7]. Such quantitative estimates of market heterogeneity provide designers with a better understanding of the tradeoffs a customer is willing to make between product configuration and product price. Exploring these tradeoffs has revealed untouched market segments and exposed the need for improved product line offerings that would allow companies to capture larger portions of market share while also increasing profits. A product line is defined as “the set of related products that are offered by a single company [8].” A commonly made assumption is that the product can be decomposed into product features, and that those features can be described in levels. The design variables for a product line optimization then become the feature levels used in each product being designed. In addition to feature levels, previous research has established the foundation of using price as a design variable [9–11], increasing the size of the design string and often making the problem mixed-integer.

Large product line design problems were initially considered to have more than 20 binary variables [12], but Lou [13] expanded on this definition to include product lines of 6 - 8 products with 20 - 24 variables per variant, resulting in problems with at least 1.8E+33 possible configurations. To highlight the computational challenges posed by these problems, Belloni et al. [14] introduced a problem with approximately 5E+15 feasible product line solutions. Solving this problem using complete enumeration would take over 5,000 years, and over 1 week using a branch-and-bound algorithm (at a rate of 30,000 evaluations per second). As product development times shorten [15], Belloni et al. [14] further explain that most managers would consider one week of computation to be an upper limit of acceptable time. If limited to a single day of computation, only 8.6E-7 percent of the total design space could be explored.

The complexity associated with product line configuration problems has led the engineering design community to focus research efforts on the development of heuristic optimization techniques to determine solutions and/or solution spaces for these problems [16–21]. These heuristic methods have proven to produce more complete and thorough solutions spaces when compared to greedy and rule-based approaches.

In previous work, a genetic algorithm (GA) was selected for product line optimization problems because it provided ample tuning opportunities, robust performance when handling NP-hard problems, and was easily adapted to mixed-integer design string formulations [22]. This work led to the development of strategies that improved population initialization by using customer preference information to drive starting point selection.

[1]Corresponding Author

Referred to as targeted initialization, products were identified that maximized respondent utility using preference estimates from discrete choice surveys. These optimal products were then combined to create the initial population for a genetic algorithm.

This research demonstrated that targeted populations consistently generated product lines that yielded a higher market share of preference when compared to randomly initialized populations [22]. In the context of this work, market share of preference is defined as the sum of the share of preference for each product within a product line. These targeted populations also converged to an optimum at a faster rate, as shown in Figure 1. Results from this research also yielded significant improvements in design frontiers when tested on multi-objective problems [23].

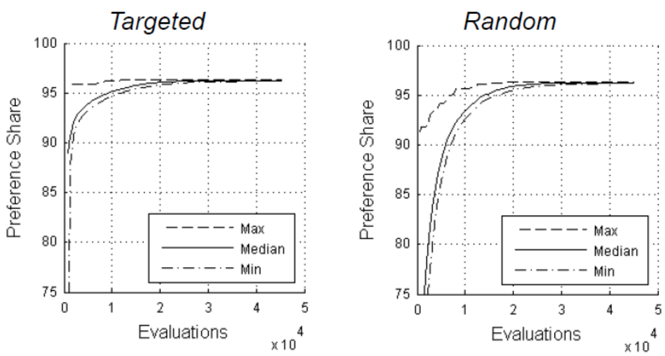

Figure 1. Targeted vs. Random Share Convergence [22]

Improving the initial population gives the algorithm a better starting point, but design string variations are achieved by the crossover and mutation operators. The goal of this paper is to expand the use of customer preference information available from discrete choice models to improve algorithm performance by developing an informed crossover operator. Four different crossover operators are explored, and the algorithmic performance improvements offered by each are identified. In summary, this work explores the development of informed crossover operators, or crossover operators that use problem-specific data to develop heuristic rules that guide their behavior, and how these developed operators perform when used with product line optimization problems.

2 BACKGROUND

This work focuses on using data from discrete choice models to improve genetic algorithm performance. The following section encompasses a brief history of discrete choice analysis and research efforts within the product design field, an overview of how this data was used in previous work focused on population initialization, and a description of how other research efforts have used problem-specific data to improve algorithm performance.

2.1 Discrete Choice Analysis in Market-Based Design

Early engineering design methods used conjoint analysis [24–26] or the S-Model [27] to gather data for the estimation of customer preference. Discrete choice analysis was then considered because of the added realism, despite the increased model complexity, which requires a respondent to make a single selection from a set of product alternatives [28–30]. Adopting choice-based surveys in engineering design research led to the first applications of a logit model [31,32], where customer-perceived value of a product is commonly modeled as the sum of the attribute part-worth values associated with product configuration and price. Under the assumption that the error term follows an extreme value distribution [29,33] the probability of choice for a respondent is defined by Equation 1. This equation represents the probability of consumer $\boldsymbol{i}$ choosing product $\boldsymbol{l}$ among the alternatives $\boldsymbol{k = 1{:}K}$, where $\boldsymbol{X_{ijl}}$ are the product configuration attributes and $\boldsymbol{\beta_j}$ are the part worth estimates

$$p_i^l = \frac{\exp(\sum_j X_{ijl}\beta_j)}{\sum_{k=1}^{K}\exp(\sum_j X_{ijk}\beta_j)} \tag{1}$$

The multinomial logit model describes the basic form from which many other discrete choice models are derived. Estimating customer heterogeneity was made possible by random utility models and leveraged in engineering design with nested logit [6,31], latent-class multinomial logit [3,34,35], and hierarchical Bayes mixed logit [2,3,7] formulations. Estimating product utilities leads to two potential avenues for analysis: using a probabilistic share of preference decision rule or using a first choice decision rule. As shown in Equation 1, a probabilistic choice rule assumes that each consumer develops a probability for choosing a product (p_i), and these probabilities are factored into their choice (but consumers still have the potential to select a product with a lower purchase probability). However, such a decision rule is not always representative of true consumer behavior – that is, a consumer has to make a single choice when making a purchase. A first choice decision rule assumes that consumers will select the product with the highest utility, regardless of other product utilities.

2.2 Product Line Design using Targeted Initialization

As described previously, past research by the authors used preference estimates from discrete choice surveys to improve population initialization for a genetic algorithm. This approach was developed around the theory that if a population was initialized with products that target the objectives of the problem, and not just randomly, improvements to solution quality and algorithm performance could be realized. The targeted initialization process can be seen in Figure 2.

The approach described in Figure 2 can be extended to both single and multiobjective problem formulations. When considering only a single objective, the targeted objective relates to the main objective (such as maximizing market share of preference). In addition to product configuration variables, price markup variables were included in the problem formulation. These variables indicated the price charged for a feature beyond base cost, and were added to the design string as a floating-point number.

To create the initial population, respondents from the discrete choice survey were selected randomly. For each respondent selected, an objective was targeted (in the case of a two objective problem formulation, this selection was simulated via a coin flip). Using the preference estimates for that respondent, and pre-defined values for the price markup variables, an "ideal" product configuration was created. These individually optimized products were then combined to seed a product line and product lines were combined to create the initial population.

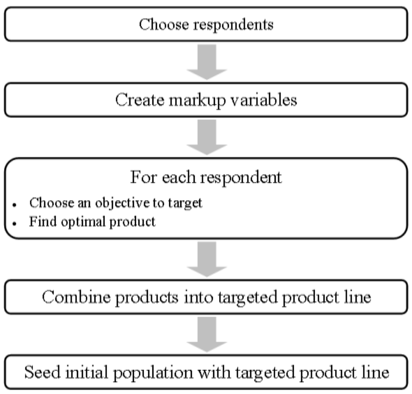

Figure 2. Enhanced Targeted Initialization Approach [22]

This informed initialization approach resulted in improved algorithm efficiency and solution quality. When multiple objectives were considered, significant improvements in final hypervolume were achieved when compared to solutions run with an initial populations generated by a Latin hypercube.

2.3 Current Crossover Modification Research

Previous research efforts into various other complex optimization problems have ventured towards altering crossover operators to enhance algorithm performance on single objective problems. Many of these efforts have focused on the traveling salesman problem. Zhou et al. [36] created offspring designs based on comparative parental performance between nodes, while Vahdati et al. [37] compared the distances between two bounding locations of a selected city for both parent designs. Experimental design procedures were used by Ho and Lee [38] to create a level-based technique that employed effects-based data from the parent strings to generate more robust offspring. A real-encoded crossover was proposed by Garcia-Martinez et al. [39] who created offspring within the fitness neighborhood of one parent, while the neighborhood size was defined by the other parent. Others have tailored crossover operators to suit specialized problems, such as a capacitated vehicle routing problem [40]. Overall, these modifications improved algorithm effectiveness while preventing premature convergence.

Building on the motivation of these efforts, this research aims to combine the informed operator method established in the targeted population work with a market-based crossover operator that uses information from the market domain to improve algorithm performance and solution quality.

3 THEORY

As discussed in the previous section, formulating the product line optimization problem involves establishing the design string setup. Using this formulation, informed crossover operators were developed for testing. This section details the design string formulation and the development of these informed crossover operators. For each technique, pseudo-code is also presented.

3.1 Design String Formulation

The product line optimization problem formulation includes both pricing and feature configuration in the design string. Product configuration variables represent the different product features considered, and can take on a discrete value indicating the level of that feature as included in the discrete choice survey. Price variables are encoded as real integers varying between 0 and 1, where a price value of 0 implies that the feature is sold at cost, and a 1 indicates 100% markup. This model assumes that feature prices are constant across consumers. This requires that the same pricing markup levels are applied to all features repeated within a product line. Figure 3 provides a graphic indicating a sample design string with n pricing variables and m products with k features each.

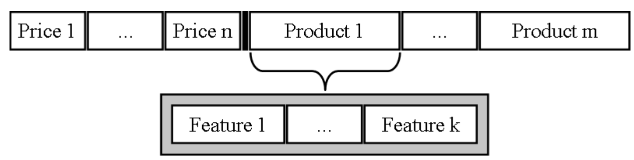

Figure 3. Sample Design String Formulation

Consider a product defined by 3 attributes. Attribute A has 3 levels, Attribute B has 2 levels, and Attribute C has 3 levels. The first part of a design string describing a product line made up of these products would have 8 pricing variables (for the 8 total levels) encoded as positive continuous variables. For a product line with 3 products, this would be followed by 9 discrete variables describing the levels used to define each product.

3.2 Pricing Calculation

The cost of a product is determined by summing the cost of each feature included in a product and the base price of the product. When calculating the cost of each product from the design string, the product feature string is converted using a binary representation based on the number of levels of each feature offered. For example, if level 3 was selected from a feature attribute with 8 levels, that corresponding section of the design string would be interpreted at 00100000. This is then multiplied component-wise by the price variable string to yield the overall markup added to the base price. Figure 4 details an example of how corresponding price markups are determine from the design string.

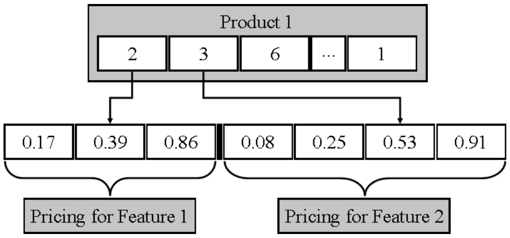

Figure 4. Pricing Determination Using Design String

3.3 Informed Crossover Operator Descriptions

Having established a design string setup, the construction of informed operators can be covered. For the purposes of this work, an informed operator is defined as a heuristic operator that uses problem data to alter its function. For this type of market-based product line optimization problem, the key information being utilized is customer preference estimates, product pricing, and the market share of preference of each product configuration. Four different crossover operators were created that use this problem information (Lowest Share Crossover, Lowest k-Share Crossover, Mixed Share Crossover, and Price Sorting Crossover).

The concepts presented in the four informed crossover operators alter the product/feature component of the design string, as defined in Section 3.2. All of the developed informed crossover operators utilize an altered form of scattered crossover, a standard crossover operator that performs bit-wise exchanges based on a pre-determined uniform probability [39]. These alterations are all influenced by the consumer preferences developed in Equation (1) by using share of preference data to sort the product lines by various means.

Additionally, scattered crossover was performed on the pricing variables in all four of the methods below (in addition to the informed crossover operators performed on the product/feature variables), and a product line size of five was chosen for initial testing based on past results [22]. It should also be noted that these four crossover operators were created through the ideas generated in an initial brainstorming session, and that a theoretically infinite number of crossover operators informed by share of preference data could be developed.

3.3.1 Lowest Share Crossover

This crossover operator hinges on the concept that the market share of preference of a product line can be increased by altering the poorest performing product. The operator in question sorts two product lines by relative market share of preference (the relative percentage of market share that each product in the line captures with respect to the other products in the line) in ascending order. This metric is determined by the consumer preference data relevant to the products included in the product line, and thus serves as an incorporation of this consumer preference data. The two poorest performing products from each line are selected, and scattered crossover occurs between these two products only, as depicted in Figure 5.

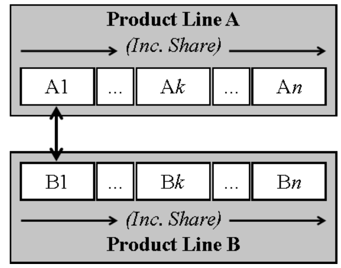

Figure 5. Representation of Lowest Share Crossover

A pseudo-code for this crossover operator is given by:

```
FOR i = 1 : (1/2)*size of population
    Choose rows i and i+1 in selected population
    Calculate each market share of preference
    Calculate relative market share of preference
    for each product in each line
    FOR j = 1 : number of products in product line
        Sort product lines in ascending order by
        relative market share
        Re-index products in both product lines
    END
    Perform scattered crossover on pricing variables
    and between first product only
END
```

3.3.2 Lowest k-Share Crossover

The Lowest k-Share Crossover operator is developed around the concept that overall performance of a product line is driven by its top performing products. Therefore, focus is placed on altering the poorer performing products. Similar to the Lowest Share Crossover operator, the Lowest k-Share Crossover operator sorts two product lines chosen in ascending order by their relative market share of preference. The lowest k products (where k is an integer varying from 2 to n) are then selected and crossed with their respective products in the other product line. Figure 6 depicts an example of this for n total products and k chosen products.

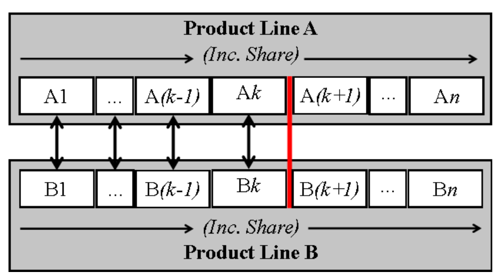

Figure 6. Representation of Lowest k-Share Crossover

A pseudo-code for this crossover operator is given by:

```
FOR i = 1 : (1/2)*size of population
    Choose rows i and i+1 in selected population
    Calculate each market share of preference
    Calculate relative market share of preference
    for each product in each line
    FOR j = 1 : number of products in product line
        Sort product lines in ascending order by
        relative market share
        Re-index products in both product lines
    END
    Perform scattered crossover on pricing variables
    FOR k = 1 : selected number of products
        Perform scattered crossover between products
        1 through k from each line
    END
END
```

3.3.3 Mixed Share Crossover

This crossover operator is motivated by the thought that product lines are often dominated by poorly and/or strongly performing products, and homogenizing the product line shares would be beneficial. The Mixed Share crossover operator takes two design strings and sorts one of the product lines in ascending order by relative share of preference and the other in descending order by relative share of preference. Scattered crossover then occurs between corresponding products, as shown in Figure 7.

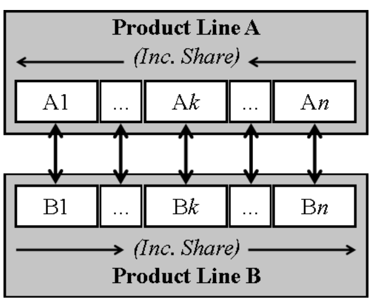

Figure 7. Representation of Mixed Share Crossover

A pseudo-code for this crossover operator is given by:

```
FOR i = 1 : (1/2)*size of population
    Choose rows i and i+1 in selected population
    Calculate relative market share of preference
    for each product in each line
    FOR j = 1 : number of products in product line
        Sort product line i in ascending order and
        product line i+1 in descending order
        Re-index products in both product lines
    END
    Perform scattered crossover on pricing variables
    and between products
END
```

3.3.4 Price Sorting Crossover

The Price Sorting Crossover operator uses price data from the product line optimization to cluster similarly priced items, with the thought that items in similar price categories often have shareable attributes that could be swapped to maximize line performance. This operator determines the price of each product in each line and sorts the product line by product cost. Scattered crossover then occurs between products in corresponding price brackets. Figure 8 depicts this for n total products. It is noted that while this is not an explicit use of consumer preference data, customer preference estimates for product price are determined, and therefore the developed operator is still considered an informed operator by the provided definition.

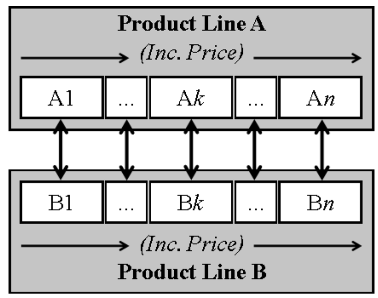

Figure 8. Representation of Price Sorting Crossover

A pseudo-code for this crossover operator is given by

```
FOR i = 1 : (1/2)*size of population
    Choose rows i and i+1 in selected population
    Calculate price of each product in each product
    line
    FOR j = 1 : number of products in product line
        Sort product lines in ascending order by
        price
        Re-index products in both product lines
    END
    Perform scattered crossover on pricing variables
    and between products
END
```

4 INITIAL TESTING OF CROSSOVER OPERATORS

The testing conducted on the various crossover operators was restricted to a more simplistic product line optimization problem so as to establish the most effective crossover operator that implemented problem data. The results of this section were then applied to a large-scale product line optimization problem for further testing and evaluation. It should also be noted that for the purposes of this preliminary study, only a single objective problem aimed at maximizing market share of preference is explored. Scaling this approach multi-objective optimization problems and testing it experimentally is left as future work.

Table 1. MP3 Player Attributes and Price Levels

Level	Photo/Video/Camera	Web/App/Ped	Input	Screen Size	Storage	Background Color	Background Overlay	Base Price
1	None	None	Dial	1.5 in diagonal	2 GB	Black	No Pattern/Graphic Overlay	$49
2	Photo Only	Web Only	Touchpad	2.5 in diagonal	16 GB	White	Custom Pattern Overlay	$99
3	Video Only	App Only	Touchscreen	3.5 in diagonal	32 GB	Silver	Custom Graphic Overlay	$199
4	Photo and Video Only	Ped Only	Buttons	4.5 in diagonal	64 GB	Red	Custom Pattern and Graphic Overlay	$299
5	Photo and Lo-Res Camera	Web and App Only		5.5 in diagonal	160 GB	Orange		$399
6	Photo and Hi-Res Camera	App and Ped Only		6.5 in diagonal	240 GB	Green		$499
7	Photo, Video, and Lo-Res Camera	Web and Ped Only			500 GB	Blue		$599
8	Photo, Video, and Hi-Res Camera	Web, App, and Ped			750 GB	Custom		$699

Table 2. MP3 Player Cost per Feature

Level	Photo/Video/Camera	Web/App/Ped	Input	Screen Size	Storage	Background Color	Background Overlay
1	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00
2	$2.50	$10.00	$2.50	$12.50	$22.50	$5.00	$2.50
3	$5.00	$10.00	$20.00	$22.50	$60.00	$5.00	$5.00
4	$7.50	$5.00	$10.00	$30.00	$100.00	$5.00	$7.50
5	$8.50	$20.00		$35.00	$125.00	$5.00	
6	$15.00	$15.00		$40.00	$150.00	$5.00	
7	$16.00	$15.00			$175.00	$5.00	
8	$21.00	$25.00			$200.00	$10.00	

Table 3. MP3 Player Competition Design

	Product 1	Product 2	Product 3	Product 4	Product 5	None
Photo/Video/Camera	Photo, Video, and Hi-Res Camera	Photo, Video, and Hi-Res Camera	Photo, Video, and Hi-Res Camera	Photo, Video, and Hi-Res Camera	Photo, Video, and Hi-Res Camera	N/A
Web/App/Ped	Web and App Only	Web and App Only	Web and App Only	Web, App, and Ped	Web, App, and Ped	
Input	Dial	Touchscreen	Touchscreen	Touchscreen	Touchscreen	
Screen Size	1.5 in Diagonal	4.5 in Diagonal	4.5 in Diagonal	4.5 in Diagonal	6.5 in Diagonal	
Storage	16 GB	16 GB	16 GB	64 GB	160 GB	
Background Color	Silver	Silver	Silver	Custom	Green	
Background Overlay	Custom Pattern and Graphic Overlay	Custom Graphic Overlay	Custom Pattern and Graphic Overlay	Custom Pattern and Graphic Overlay	Custom Graphic Overlay	
Price	*$132.59*	*$211.39*	*$216.39*	*$438.89*	*$504.14*	*$0.00*
Preference Share	*25%*	*27%*	*20%*	*15%*	*10%*	*3%*

4.1 MP3 Problem

The first case study analyzed in this paper concerns an MP3 product line optimization problem. The preference model for the MP3 problem was constructed from a choice-based conjoint survey fielded to 205 respondents. Choice task questions were based on the 12 attributes detailed in Table 1. The price for each feature level is detailed in Table 2. Sawtooth's CBC/HB [41] software was used to develop a mixed discrete choice model that estimated the part-worth coefficients for each respondent. The "none" option was also included in the choice tasks, and its part-worth was also estimated. Development of the price levels for each feature is detailed in [22]. Competition was also included in the MP3 market simulation to increase the difficulty of establishing an optimal product line. These competitive products were developed by creating an optimal product line using the outside good as competition, and their makeups are shown in Table 3.

4.2 Testing Procedure

The four crossover operators proposed in the previous section were encoded in MATLAB. These operators were incorporated into a GA with the following test standards, and then tests were then run on each crossover operator with the basic scattered crossover serving as a test control.

- Trials: 10 per experimental setup
- Convergence: 500 generations
- Population Size: 2 times design string length
- Selection: tournament with 4 designs per tourney
- Crossover Rate: 0.8
- Mutation: Uniform with rate set at 0.05

For each experimental setup, the optimal objective function value and the number of stall generations was recorded. In this work, stall generations is defined as the number of generations where the optimal design from the population does not change. The total generations taken to find optimal design were calculated as the difference between the 500 total generations and the number of stall generations.

4.3 Informed Crossover Performance

Figure 9 details the average market share of preference from the five crossover operators tested. The only proposed crossover operator that yielded improved algorithm performance was the Lowest k-Share Crossover, producing results that seemed closely matched the results of Scattered Crossover. Due to the relatively small number of attributes and feature levels associated with this testing problem, it was expected that improvements in objective function would be minimal at best. Prior work has demonstrated that this problem could be solved by a non-modified GA, but that convergence rates could be improved.

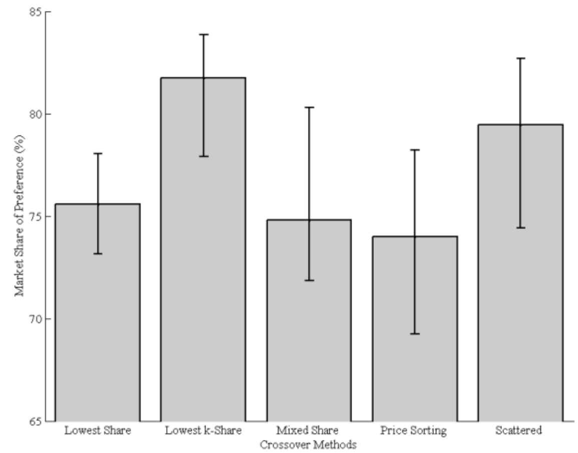

Figure 9. Market Share of Preference vs. Crossover Operator for MP3 Player Test Problem

Table 4 summarizes the data collected on the number of generations to reach the optimal design from each crossover operator. The Lowest k-Share Crossover again yields similar results to Scattered Crossover, warranting further study. It should also be noted that, while the Price Sorting Crossover yielded improvements in algorithm efficiency, it did not produce significant improvements when evaluated for improved market share of preference. It can be theorized that Lowest Share Crossover underperformed computationally due to the low impact of crossover. Conversely, the Mixed Share and Price Sorting Crossover operators likely underperformed due to presence of too much mixing via crossover, potentially breaking up high share products.

Table 4. Generational Results of Initial Crossover Testing for MP3 Player Problem

Crossover Method	Average Generations	Standard Deviation of Generations
Lowest Share	176.500	152.426
Lowest k-Share	116.200	57.823
Mixed Share	160.700	101.987
Price Sorting	92.400	20.430
Scattered	105.100	19.576

4.4 Lowest k-Share Crossover Tuning

Due to the fact that k was assumed to be the ceiling of half the number of products in the product line for the initial testing, further testing was pursued when developing this operator. A test was established using the MP3 Player problem with multiple product line sizes and different k values to establish an optimal crossover size. Product line sizes of 5, 6, and 7 products were tested with the Lowest k-Share Crossover, where k was varied from 1 to n (with n being the total number of products in the line). The core objective remained to increase market share of preference, and the results of these trials are presented in Figure 10.

From the results presented in Figure 10, it can be seen that improved algorithm performance was often associated with k values equal to $n-1$. The improved performance of this crossover operator may be due to the fact that the strongest product in each line is left untouched, and the remaining products are subsequently altered (conceptually similar to the theoretical basis used to develop the Lowest Share Crossover). The findings of this test led to the selection of the Lowest $n-1$ Share Crossover as the adopted informed crossover operator to be used for large-scale testing.

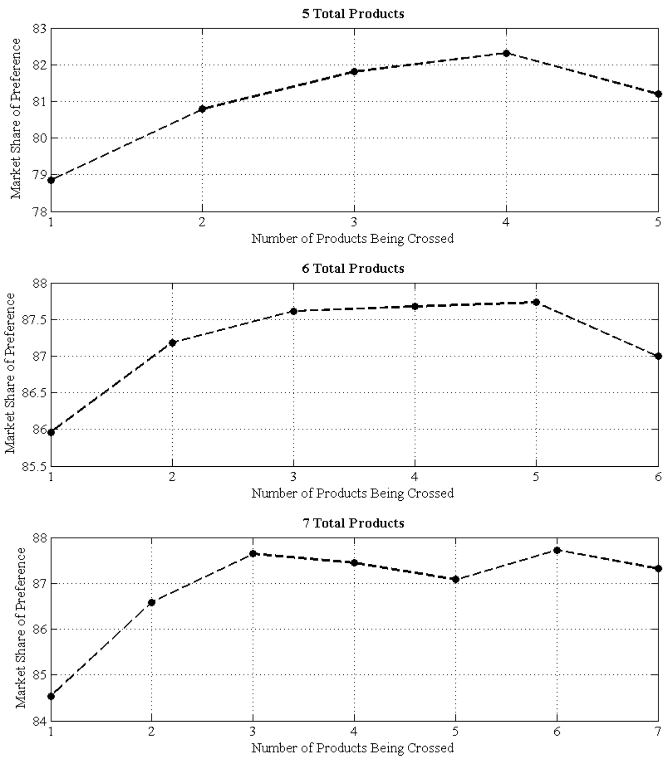

Figure 10. Results of Testing Varying k Values Across Different Product Line Sizes

LARGE-SCALE PROBLEM IMPLEMENTATION

The previous section established that the Lowest $\boldsymbol{n-1}$ Share crossover operator offered algorithmic gains for a smaller-scale market-based design problem. To test the effectiveness of this approach on larger problems, an automobile feature packaging case study will be investigated. The customer preference data for this study was captured from a choice-based conjoint survey fielded to 2,275 respondents, and the observed part-worth estimates were established using an HB mixed logit model that was fit using the Sawtooth CBC/HB software. Each respondent in the HB model has 73 total part-worths: 60 for the features offered, 12 for the price levels, and 1 for the "none" option (or outside good). Due to the proprietary nature of the problem, the feature and cost breakdowns cannot be shown. They are similar in nature to the breakdowns shown in Tables 1 and 2. Table 5 presents the total number of levels present for each of the 19 attributes. Accounting for all possible feature combinations, there are 1,074,954,240 possible feature combinations. Finally, the outside good serves as the competition source for the automobile feature packaging market simulator.

5.1 Experimental Setup

The experimental setup for testing the informed crossover operator involves altering the initialization method (between random and targeted), the crossover operator (between scattered and informed), the number of products in the product line (ranging from three to eight), and the model being used to determine product selection from respondent data (chosen between the discrete choice and first choice models). Due to the previous success demonstrated when using targeted population initialization (an informed initialization operator), this variable was included in the testing to determine if additional efficiencies and/or algorithm effectiveness would occur. The number of products offered in the product line, and the choice selection scheme were both modified during the experiment as a measure of robustness. The control for this experiment was a GA based on NSGA-II [42] with random population initialization and scattered crossover. The size of the population at each generation is ten times the number of design variables. The selection operator is tournament with four candidates. The mutation operator is adaptive with each bit having a 5% chance of mutating. Finally, the convergence criterion is set at 600 total generations, established from previous work.

Table 5. Automobile Feature Levels per Attribute

Attribute Number	Number of Levels
1	3
2	2
3	5
4	6
5	2
6	3
7	3
8	2
9	4
10	2
11	3
12	2
13	4
14	3
15	3
16	4
17	4
18	3
19	2

A series of 48 separate experimental setups were developed to account for every combination: two initialization methods, two crossover operators, two customer selection rules, and six different numbers of products offered in the line. Twenty different trials were run for each combination mentioned above, leading to 960 unique genetic algorithm runs yielding data on market share of preference and total generations to reach an optimal design. The summary of each unique test setup is shown in Table 6.

Table 6. Experimental Setup for Vehicle Problem

Number of Products	Initialization	Crossover	Customer Selection Rule	Runs Per Trial
3	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
4	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
5	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
6	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
7	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
8	Random	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20
	Targeted	Scattered	Probabilistic	20
			First Choice	20
		Informed	Probabilistic	20
			First Choice	20

5.2 Probabilistic Choice Rule Experimental Results

Following completion of the 480 runs using the probabilistic choice rule, the results were tabulated and analyzed. Figure 11 depicts the average market share of preference for varying product line sizes found using the four operator combinations (random initialization and scattered crossover, random initialization and informed crossover, targeted initialization and scattered crossover, and targeted initialization and informed crossover). From this figure, three performance groups can be isolated: the control combination of random initialization and scattered crossover as the poorest performer, the combination of targeted initialization and informed crossover as the strongest performer, and the combinations of a single informed operator with a control operator as middle-of-the-line performers when concerning objective function value. These trends confirm that the informed crossover operator enhances the market share of preference found when optimizing with a genetic algorithm.

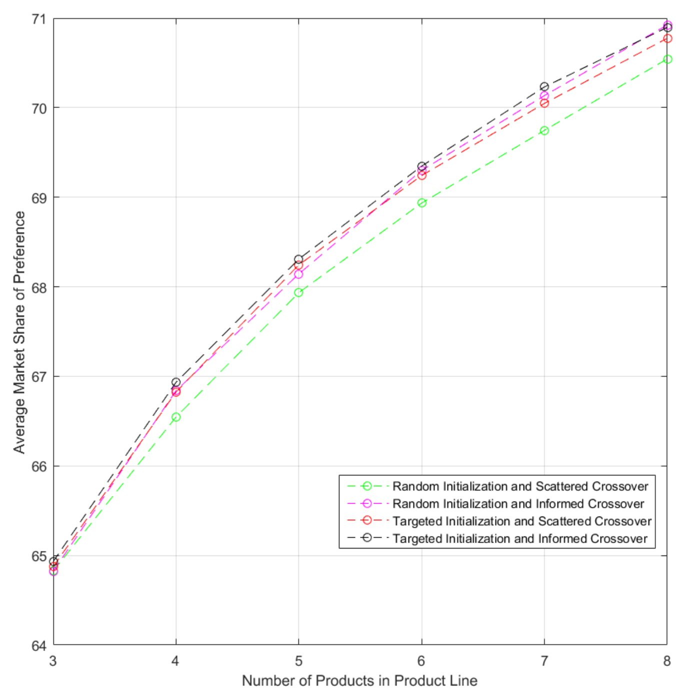

Figure 11. Average Market Share of Preference vs. Product Line Size Utilizing the Probabilistic Choice Rule

The above figure also serves to highlight the improvements in algorithm effectiveness when informed operators are applied to large-scale market-base design problems. Improvements in effectiveness are to be expected when problem data is implemented into the framework of the algorithm searching the design space, and it can be theorized that these improvements will become more significant as the design space becomes complex or as multiple objectives are considered.

When evaluating algorithm efficiency, the aforementioned stall generations evaluator was utilized. The number of stall generations was tracked for each trial and then subtracted from the 600 generation convergence limit to yield the number of generations required to reach the reported optimal product line configuration. Figure 12 highlights the range of generational limits reached with each of the operator combinations when using the probabilistic choice rule. As can be seen in Figure 12, implementation of the informed crossover dramatically decreased the number of generations necessary for the GA to reach an optimal solution. Computational efficiency (calculated as the percent change between the control group and the studied group) saw an average increase of 40% when utilizing the informed operators.

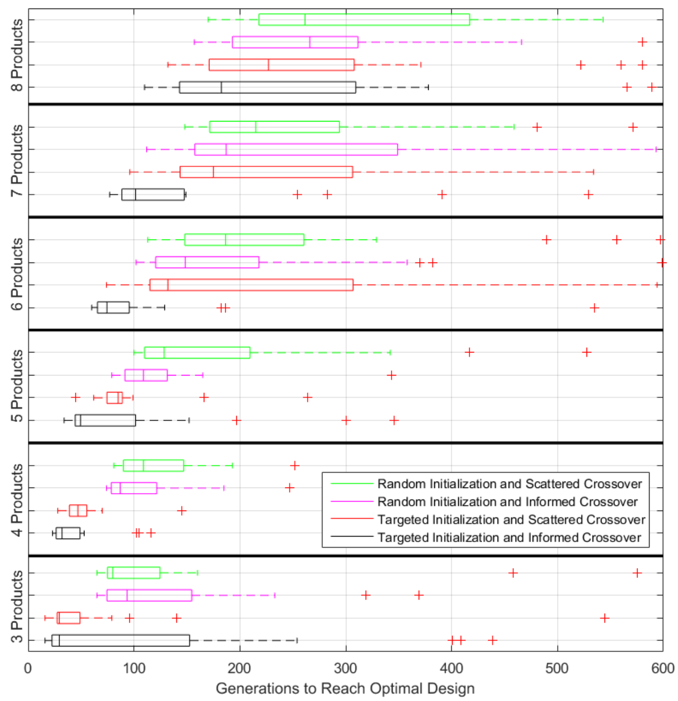

Figure 12. Comparison of Generations Required to Reach an Optimal Design Across Product Line Sizes Using the Probabilistic Choice Rule

5.3 First Choice Rule Experimental Results

Following confirmation that the developed crossover operator increases algorithm performance using the probabilistic choice rule, tests were completed using the first choice rule. A primary motivation for this was to ensure robustness across different choice rules of respondent behavior. A first choice rule places a strong emphasis on the highest performing product – it receives 100% of that respondent's share while all other products receive 0%. Figure 13 reveals that the objective values displayed by the algorithm when run with the first choice rule follows similarly noted trends when compared with the probabilistic choice rule. It is also interesting to note that objective performance is slightly lower when utilizing the first choice rule when compared with the probabilistic choice rule, due to the non-probabilistic nature of the first choice rule.

When comparing algorithm efficiency (via generations to reach an optimum), it can be seen from Figure 14 that the developed crossover operator performs just as effectively when using the first choice rule as compared to the probabilistic choice rule. Computational efficiency saw an average percent increase of 54% when utilizing the informed operators together with the first choice rule, which is 14 percentage points higher that when the same tests are run using the probabilistic choice rule. The trends noted in Figures 13 and 14 indicate that the developed informed crossover operator is robust enough to handle multiple market models and notably improves algorithm performance. While these improvements in algorithm effectiveness are not substantial, they are still provided, and in a significantly fewer number of generations.

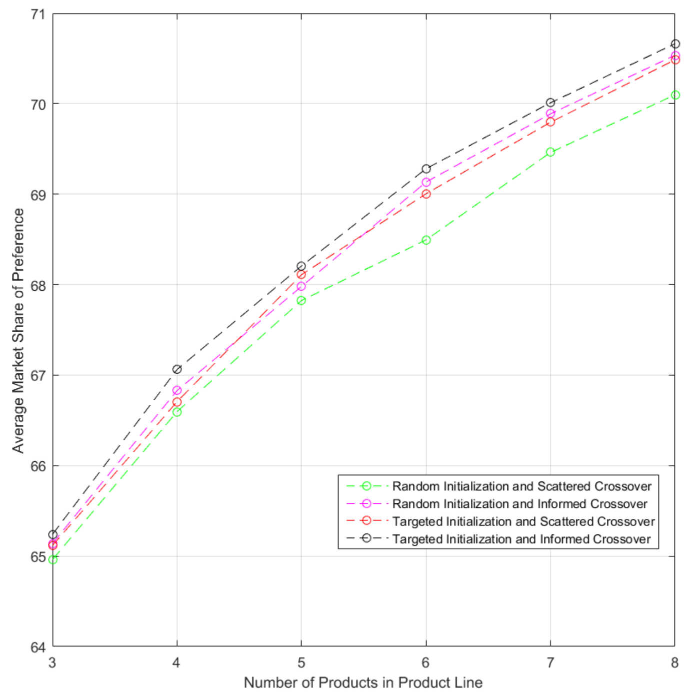

Figure 13. Average Market Share of Preference vs. Product Line Size Utilizing the First Choice Rule

Annex A contains two figures that re-organize the generational information presented in Figures 12 and 14, grouping them by operator/model combinations as opposed to number of products in the product line. These figures depict the trends exhibited by algorithms with changes in line sizes.

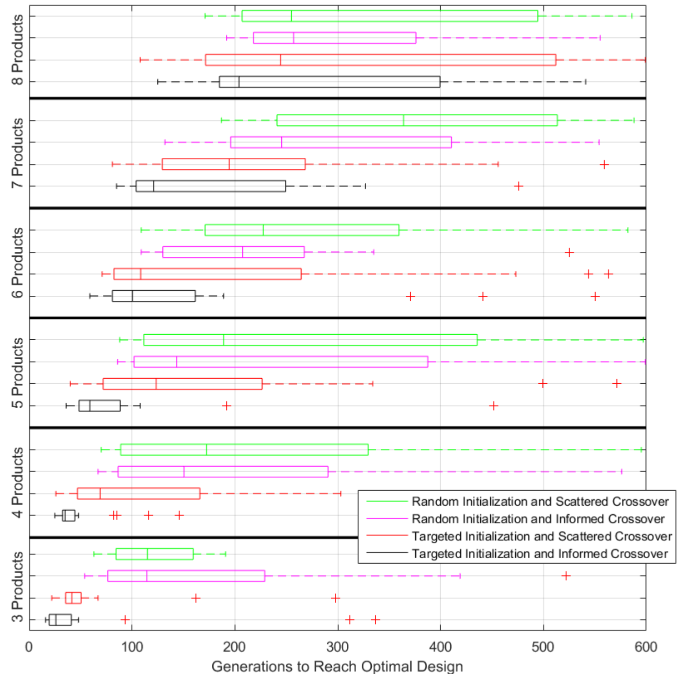

Figure 14. Comparison of Generations Required to Reach an Optimal Design Across Product Line Sizes Using the First Choice Rule

CONCLUSIONS

This paper presented an informed crossover operator aimed at improving computational efficiency and algorithm effectiveness by using problem information to influence operator actions. Section 2 provided background information in discrete choice analysis, previous informed operator research, and an overview of current state-of-the-art in problem specific crossover operators. Section 3 detailed a sample product line optimization problem and detailed four unique informed crossover operators developed for the purposes of this work. Following testing of these four operators, the strongest performing crossover operator was selected and re-tested for tuning. In Section 4, the informed crossover operator (referred to as Lowest $\boldsymbol{n-1}$ Share Crossover) was tested with and without a targeted initialization method against a standard scattered crossover and random initialization control group of data.

Testing of this informed crossover operator yielded minor increases in market share of preference, indicating minor improvement in algorithm effectiveness. It is speculated that with more complex product line design problems indicative of industry practice, or with a multi-modal design space, that the objective improvements would see markedly increased performance. This hypothesis will be explored in future iterations of this research.

The major contribution of this work is yielding significant improvements in computational efficiency by offering reduced generation counts required to reach an optimal solution, with improvements in generation counts ranging (on average) from 40% - 54%. These significant improvements in computational efficiency indicate that problem data should indeed be used when handling complex product line design problems, as this inclusion provides an effective and efficient means for handling difficult design spaces with potentially unknown optimality conditions. It was expected that these computational improvements would be seen, but the improvements were greater than expected and additionally paired with the benefits in algorithm effectiveness. Unlike the targeting of an initial population that involved numerous sub-optimizations, the computational cost associated with modifying crossover behavior using the proposed approaches is quite low, and thus the computational benefits offered by this inclusion are even more substantial. The problem data used to modify crossover is already calculated in evaluating the objective function of a newly generated design, and this information could be stored in a graveyard as the algorithm progresses, should the design string reappear in selection.

The author notes that the informed crossover operators developed for this research are dependent upon the combinatorial nature of the product line optimization problem (e.g. products can be re-arranged within a design string without altering design performance). Due to this unique design string setup, the informed crossover operators cannot be applied to other optimization problems. However, the concept surrounding using problem data to develop heuristic rules that inform crossover performance can be adopted and used in other complex optimization problems.

Future work on the subject of informed operators will first explore the effects of implementing the developed informed crossover operator on a multi-objective product line optimization problem. These multi-objective spaces could look to increase market share of preference, increase profit, decrease loss, etc. Another avenue of research in this area will explore the development and implementation of informed selection and mutation operators that also improve computational efficiency and algorithm effectiveness, ultimately leading to schema development that will be informed by problem information.

This work demonstrates the benefits of using discrete choice analysis in both initial populations and to inform crossover operators when analyzed for both algorithm efficiency and effectiveness. It is expected that these benefits would be seen when applied to other population based heuristic algorithms, and this remains a potential source of future work.

Finally, further avenues of research will look to expand the concept of informedly developed product line development into the product family and platform realms, using various other sources of information from consumer preferences and requirements to inform the optimization algorithm. These will then be tested with a newly developed market simulation model that will include complex product geometries to simulate product lines indicative of industry standards.

REFERENCES

[1] Michalek, J. J., 2006, "Balancing Marketing and Manufacturing Objectives in Product Line Design," J. Mech. Des., **128**(5), pp. 1196–1204.

[2] Michalek, J. J., Ebbes, P., Adigüzel, F., Feinberg, F. M., and Papalambros, P. Y., 2011, "Enhancing marketing

with engineering : Optimal product line design for heterogeneous markets," Int. J. Res. Mark., **28**(1), pp. 1–12.

[3] Sullivan, E., Ferguson, S., and Donndelinger, J., 2011, "Exploring Differences in Preference Heterogeneity Representation and Their Influence in Product Family Design," 37th Design Automation Conference, ASME, pp. 81–92.

[4] Turner, C., Ferguson, S., and Donndelinger, J., 2011, "Exploring Heterogeneity of Customer Preference to Balance Commonality and Market Coverage," ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, pp. 1–13.

[5] Huber, J., Arora, N., and Johnson, R., 1998, "Capturing Heterogeneity in Consumer Choices," Am. Mark. Assoc.

[6] Kumar, D., Hoyle, C., Chen, W., Wang, N., Levi, G. G., and Koppelman, F. S., 2009, "A hierarchical choice modelling approach for incorporating customer preferences in vehicle package design," Int. J. Prod. Dev., **8**(3), p. 228.

[7] Ferguson, S., Olewnik, A., and Cormier, P., 2004, "Exploring Marketing to Engineering Information Mapping in Mass Customization: A Presentation of Ideas, Challenges and Resulting Questions," ASME Design Engineering Technology Conference.

[8] Simpson, T. W., Maier, J. R. A., and Mistree, F., 2001, "Product platform design: method and application," Res. Eng. Des., **13**(1), pp. 2–22.

[9] Kumar, S., and Chatterjee, A. K., 2013, "A profit maximizing mathematical model for pricing and selecting optimal product line," Comput. Ind. Eng., **64**(2), pp. 545–551.

[10] Nichols, K. B., Venkataramanan, M. A., and Ernstberger, K. W., 2005, "Product Line Selection and Pricing Analysis: Impact of Genetic Relaxations," Math. Comput. Model., **42**(2005), pp. 1397–1410.

[11] Qian, L., and Ben-Arieh, D., 2009, "Joint Pricing and Platform Configuration in Product Family Design With Genetic Algorithm," 14th Design for Manufacturing and the Life Cycle Conference, 6th Symposium on International Design and Design Education, 21st International Conference on Design Theory and Methodology, ASME, pp. 49–58.

[12] Siedlecki, W., and Sklansky, J., 1989, "A note on genetic algorithms for large-scale feature selection," Pattern Recognit. Lett., **10**(5), pp. 335–347.

[13] Luo, L., 2011, "Product Line Design for Consumer Durables: An Integrated Marketing and Engineering Approach," J. Mark. Res., **48**(1), pp. 128–139.

[14] Belloni, A., Freund, R., Selove, M., and Simester, D., 2008, "Optimizing Product Line Designs: Efficient Methods and Comparisons," Manage. Sci., **54**(9), pp. 1544–1552.

[15] Sanderson, S., and Usumeri, M., 1995, "Managing product families : The case of the Sony Walkman," Res. Policy, **24**(1995), pp. 761–782.

[16] Green, P. E., and Krieger, A. M., 1985, "Models and Heuristics for Product Line Selection," Mark. Sci., **4**(1), pp. 1–19.

[17] May, J. H., Shocker, A. D., and Sudharshan, D., 1983, "A Simulation Comparison of Methods for New Product Location," Mark. Sci., **6**(82-126), pp. 182–201.

[18] Kohli, R., and Krishnamurti, R., 1987, "A Heuristic Approach to Product Design," Manage. Sci., **33**(12), pp. 1523–1533.

[19] Dobson, G., Kalish, S., and Graduate, W. E. S., 1993, "Heuristics for Pricing and Positioning a Product-Line Using Conjoint and Cost Data," Manage. Sci., **39**(2), pp. 160–175.

[20] Nair, Suresh K.Thakur, L. S. K.-W. W., 1995, "Near Optimal Solutions for Product Line Design and Selection: Beam Search Heuristics.," Manage. Sci., **41**(5), pp. 767–785.

[21] Thakur, L. S., Nair, S. K., Wen, K.-W., and Tarasewich, P., 2000, "A new model and solution method for product line design with pricing," J. Oper. Res. Soc., **51**(1), pp. 90–101.

[22] Foster, G., Turner, C., Ferguson, S., Donndelinger, J., Foster, G., Turner, C., Ferguson, S., and Donndelinger, J., 2014, "Creating targeted initial populations for genetic product searches in heterogeneous markets," Eng. Optim., **46**(12), pp. 1729–1747.

[23] Foster, G., and Ferguson, S., 2013, "Enhanced Targeted Populations for Multiobjective Product Line Optii," ASME International Design Engineering Technical Conference and Computers and Informatoin in Engineering Conference, pp. 1–13.

[24] Li, H., and Azarm, S., 2000, "Product Design Selection Under Uncertainty and With Competitive Advantage," J. Mech. Des., **122**(4), pp. 411–418.

[25] Moore, W. L., Louviere, J. J., and Verma, R., 1999, "Using Conjoint Analysis to Help Design Product Platforms," J. Prod. Innov. Manag., **16**(1), pp. 27–39.

[26] Okudan, G. E., and Kim, M. C. T., 2013, "Perceived feature utility-based product family design : a mobile phone case study," J. Intell. Manuf., **24**(5), pp. 935–949.

[27] Cook, H. E., and DeVor, R. E., 1991, "On Competitive Manufacturing Enterprises I: The S-Model and the Theory of Quality," Manuf. Rev., **4**(2), pp. 96–105.

[28] Ben-Akiva, M., and Lerman, S. R., 1985, Discrete Choice Analysis: Theory and Application to Travel Demand, MIT Press, Cambridge, MA.

[29] Train, K. E., 2009, Discrete Choice Methods with Simulation, Cambridge University Press, New York.

[30] Louiviere, J. J., Hensher, D. A., and Swait, J. D., 2000, Stated Choice Methods: Analysis and Applications, Cambridge University Press.

[31] Michalek, J. J., Feinberg, F. M., and Papalambros, P. Y., 2005, "Linking Marketing and Engineering Product Design Decisions via Analytical Target Cascading," J. Prod. Innov. Manag., **22**(1), pp. 42–62.

[32] Chen, W., 2003, “An Approach to Decision-Based Design With Discrete Choice Analysis for Demand Modeling,” J. Mech. Des., **125**(3), pp. 490–497.
[33] Louviere, J. J., and Woodworth, G., 1983, “Design and Analysis of Simulated Choice or Allocation Experiments : An Approach Aggregate Data,” J. Mark. Res., **20**(4), pp. 350–367.
[34] Williams, N., and Kannan, P. K., 2011, “Product Design Retail Channel Structure Impact on Strategic Engineering Product Design,” **57**(5), pp. 897–914.
[35] Wassenaar, H., Chen, W., Chen, J., and Sudjianto, A., 2004, “An integrated latent variable choice modeling approach for enhancing product demand modeling,” ASME Design Engineering Technology Conference.
[36] Zhou, C., Zheng, J., and Li, W., 2008, “An improved heuristic crossover operator for TSP,” Fourth International Conference on Natural Computation, pp. 541–545.
[37] Vahdati, G., and Yaghoubi, M., 2009, “A New Approach to Solve Traveling Salesman Problem Using Genetic Algorithm Based on Heuristic Crossover and Mutation Operator,” International Conference of Soft Computing and Pattern Recognition, pp. 112–116.
[38] Ho, S.-Y., and Lee, K.-Z., 2003, “Genetic algorithm with intelligent crossover for colour quantization,” Imaging Sci. J., **51**(3), pp. 151–162.
[39] Lozano, M., Herrera, F., Molina, D., and Sa, A. M., 2008, “Global and local real-coded genetic algorithms based on parent-centric crossover operators,” Eur. J. Oper. Res., **185**(3), pp. 1088–1113.
[40] Nazif, H., and Soon, L., 2012, “Optimised crossover genetic algorithm for capacitated vehicle routing problem,” Appl. Math. Model., **36**(5), pp. 2110–2117.
[41] Sawtooth Software, I., 2009, “CBC Hierarchical Bayes Module.”
[42] Deb, K., Member, A., Pratap, A., Agarwal, S., and Meyarivan, T., 2002, “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II,” IEEE Trans. Evol. Comput., **6**(2), pp. 182–197.

ANNEX A

COMPARISON OF GENERATIONS REQUIRED TO REACH AND OPTIMAL DESIGN ACROSS PRODUCT LINE SIZES SORTED BY INITIALIZATION AND CROSSOVER OPERATORS