I, Sriram Kannan

hereby submit this work as part of the requirements for the degree of:

Doctorate of Philosophy (Ph.D.)

in:

Business Administration

It is entitled:

An Exact Algorithm for the Share-of-Choice Problem

This work and its defense approved by:

Chair: Dr. Jeffrey D. Camm

Dr. David J. Curry

Dr. James J. Cochran

Dr. Amitabh S. Raturi
An Exact Algorithm for the
Share-of-Choice Problem

A dissertation submitted to the
Division of Research and Advanced Studies
of the University of Cincinnati

in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

in the Department of Quantitative Analysis and Operations Management
of the College of Business Administration

Spring Qtr, 2006

by

Sriram Kannan

B.Sc., University of Madras, 1988
M.M.S., University of Bombay, 1994
M.S., University of Cincinnati, 2001

Committee

Jeffrey D. Camm, Ph.D. (Chair)
David J. Curry, Ph.D.
James J. Cochran, Ph.D.
Amitabh S. Raturi, Ph.D.
Abstract

Products and services can be thought of as bundles of attributes that are each made up of many levels. Consumers consider tradeoffs between various attributes and levels before making a purchase decision for a product. A popular marketing research technique called conjoint analysis is used to study these tradeoffs. The data on consumer preferences is generally decomposed using a regression procedure into utility values that the consumer associates with each level of every attribute. In the design of a new product, it is critical to determine a judicious combination of attribute levels that is likely to perform well in a market containing competitor products. Using these utility measures, a firm can design products that would best meet consumer expectations and the seller’s objectives.

A popular way to measure the success of a new product is to calculate the amount of market share it is expected to capture in a competitive market. Researchers have proposed the share-of-choice problem whose objective is to maximize the number of respondents from a conjoint study who prefer a new product to their status-quo products. The share-of-choice problem is NP-hard. The linear-programming based branch-and-bound algorithm is inefficient in solving real world instances of the share-of-choice problem. Several heuristic procedures to solve the share-of-choice problem have appeared in the literature. In this research, an exact algorithm is proposed to solve real world instances of the share-of-choice problem to optimality. An experiment is designed to test the run-time performance of the exact algorithm using a set of simulated problems that mimic real world problem characteristics. We also suggest potential extensions of the exact algorithm to solve modified versions of the share-of-choice problem, and also to solve the related product line design problem.
I am very fortunate to have found a research advisor in Professor Camm. This dissertation would be impossible without his guidance, support, encouragement and tolerance for slackness. I am very grateful to my dissertation committee members Drs. Camm, Curry, Cochran and Raturi, for giving me the opportunity to work on this interesting and challenging research topic. I had excellent guidance from my advisors in all aspects of research. I am indebted to Dr. Curry for his help with the simulation design, Dr. Cochran for his help with the statistical analysis, Dr. Camm for help with all optimization issues. My thanks are due to Dr. Raturi for agreeing to serve on my committee. I have greatly benefited from the faculty members in the Quantitative Analysis & Operations Management and Marketing departments of the College of Business, University of Cincinnati. I am indebted to all of them.

I am much obliged to my parents for their faith in me, for their constant support and encouragement in all my endeavors. I am obliged to my brother and all my friends who have been very supportive. Finally, I am grateful to the QAOM Department for providing me with a UGS and a graduate assistantship throughout my graduate study so that I could focus on doing research and teaching.
Copyright © Sriram Kannan 2006.
All rights reserved.
Table of Contents

Abstract.............................................................................................................................................. i
Acknowledgments ............................................................................................................................ ii
Table of Contents ........................................................................................................................... iv
List of Figures ....................................................................................................................................... vii
List of Tables ......................................................................................................................................... viii
Chapter 1.0: A Literature Review of Conjoint Analysis......................................................... 1
  1.1 Introduction to CA.................................................................................................................. 3
  1.2 The CA procedure ............................................................................................................... 8
  1.3 Family of CA techniques .................................................................................................... 11
    1.3.1 Which Conjoint method to use? .................................................................................. 17
  1.4 CA desirable characteristics .............................................................................................. 19
    1.4.1 CA assumptions ......................................................................................................... 20
    1.4.2 Issues to resolve in conjoint studies ........................................................................ 21
  1.5 Pooled data models ............................................................................................................. 23
  1.6 Uses and limitations of conjoint analysis ......................................................................... 26
    1.6.1 Strengths and weaknesses of conjoint methods ....................................................... 30
  1.7 Suggestions for improvement and new developments .................................................... 34
  1.8 Conjoint market simulators ............................................................................................... 41
    1.8.1 Properties of a good simulator .................................................................................. 45
    1.8.2 Conjoint analysis software ....................................................................................... 47
  1.9 Optimizing the product design ......................................................................................... 51
Chapter 2.0: Conjoint Analysis Applications ............................................................................ 53
  2.1 Statistics of conjoint analysis applications in the industry ............................................. 53
  2.2 Selected conjoint analysis applications ........................................................................... 55
Chapter 3.0: Algorithmic approaches for product design...................................................... 63
  3.1 Choice rules for market simulators .................................................................................. 65
    3.1.1 Share of Utility/ BTL Choice Rule ........................................................................... 65
    3.1.2 Logit Choice/Share of Preference Rule ................................................................. 67
    3.1.3 Maximum Utility/Share of First Choices Rule ..................................................... 68
3.1.4 Randomized First Choice Rule ................................................. 69
3.2 A product-positioning model ...................................................... 71
3.3 The Share-of-Choice Problem ..................................................... 75
  3.3.1 Divide-and-Conquer Heuristic .............................................. 75
  3.3.2 Dynamic Programming Heuristic ......................................... 77
  3.3.3 Shortest Path Heuristic ...................................................... 83
  3.3.4 Genetic Algorithm Heuristic ............................................... 86
  3.3.5 Nested Partitions Heuristic .................................................. 89
3.4 The Advanced Simulator Module for Product Optimization .......... 94

Chapter 4.0: An exact algorithm for the share-of-choice problem ........ 97
  4.1 Formulation of the share-of-choice problem ............................ 98
  4.2 LP-based or logic-based branch-and-bound method? ................. 100
  4.3 Implicit Enumeration ............................................................ 102
  4.4 Phase I of the Exact Algorithm ................................................. 106
    4.4.1 Pre-processing Steps ......................................................... 106
    4.4.2 Greedy Algorithm ............................................................. 111
    4.4.3 Lagrangian relaxation of the share-of-choice problem .......... 112
    4.4.4 Determining the branching order ...................................... 117
    4.4.5 Checking for column dominance ...................................... 119
  4.5 The logic-based fathoming of nodes in the search tree ............... 119
    4.5.1 Logic Test I ................................................................. 120
    4.5.2 Logic Test II ................................................................. 120
    4.5.3 The heuristic algorithm used to aggregate constraints ........... 124
  4.6 Phase II of the Exact Algorithm ............................................. 128

Chapter 5.0: Algorithmic approaches for product line design ............. 136
  5.1 Buyer’s Welfare, Seller’s Welfare and Share-of-choice Models ...... 138
    5.1.1 Greedy Heuristic ............................................................ 139
    5.1.2 The X-System Solver ...................................................... 143
    5.1.3 Dynamic Programming Heuristic ..................................... 148
    5.1.4 Beam Search Heuristic .................................................. 153
    5.1.5 Ideal Point Heuristic ...................................................... 157
# List of Figures

<table>
<thead>
<tr>
<th>Figure 1.1</th>
<th>Alternative approaches to measuring preference structures</th>
<th>Page Nos.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 3.1</td>
<td>First choice and logit rules - Simulation methods compared</td>
<td>69,237</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Traversing the search tree</td>
<td>103,238</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>A flow diagram of Phase I of the exact algorithm</td>
<td>106,239</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>Flow diagram of Phase II of the exact algorithm</td>
<td>128,240</td>
</tr>
<tr>
<td>Figure 8.1</td>
<td>Runtime difference between CPLEX 8.0 Solver and the exact algorithm</td>
<td>205,257</td>
</tr>
<tr>
<td>Figure 8.2</td>
<td>Ratio of CPLEX solution over optimal solution</td>
<td>206,258</td>
</tr>
<tr>
<td>Figure 8.3</td>
<td>Ratio of average runtimes at high level over average runtimes at low level for each factor</td>
<td>209,259</td>
</tr>
<tr>
<td>Figure 8.4</td>
<td>Ratio of greedy solution over optimal solution</td>
<td>214,260</td>
</tr>
<tr>
<td>Figure 8.5</td>
<td>Ratio of optimal solution over sample size</td>
<td>215,261</td>
</tr>
<tr>
<td>Figure 8.6</td>
<td>Percentage of greedy solutions that are found to be optimal</td>
<td>215,262</td>
</tr>
<tr>
<td>Figure 8.7</td>
<td>Percentage of subgradient solutions that are found to be optimal</td>
<td>216,263</td>
</tr>
<tr>
<td>Figure 8.8</td>
<td>Percentage of optimal solutions found during branch-and-bound stage</td>
<td>217,264</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page Nos.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.1</td>
<td>A set of attributes and levels for designing a PC</td>
<td>4,230</td>
</tr>
<tr>
<td>Table 1.2</td>
<td>Sports car in the mid-price range (attributes and levels of conjoint stimuli)</td>
<td>4,230</td>
</tr>
<tr>
<td>Table 1.3</td>
<td>Traditional Conjoint: Card-Sort Method (5 attributes)</td>
<td>12,231</td>
</tr>
<tr>
<td>Table 1.4</td>
<td>Traditional Conjoint: Card-Sort Method (15 attributes)</td>
<td>12,232</td>
</tr>
<tr>
<td>Table 1.5</td>
<td>ACA Survey - Self-Explicated “Priors” Section</td>
<td>15,232</td>
</tr>
<tr>
<td>Table 1.6</td>
<td>ACA Survey - Conjoint “Pairs” trade-offs (Preferences)</td>
<td>15,233</td>
</tr>
<tr>
<td>Table 1.7</td>
<td>ACA Survey - Conjoint “Pairs” trade-offs (Importances)</td>
<td>15,233</td>
</tr>
<tr>
<td>Table 1.8</td>
<td>ACA Survey – Calibration Concept to obtain purchase likelihood scores</td>
<td>15,234</td>
</tr>
<tr>
<td>Table 1.9</td>
<td>Choice Based Conjoint Question</td>
<td>15,234</td>
</tr>
<tr>
<td>Table 2.1</td>
<td>A partial list of conjoint applications</td>
<td>53,235</td>
</tr>
<tr>
<td>Table 2.2</td>
<td>A sample list of conjoint applications</td>
<td>53,236</td>
</tr>
<tr>
<td>Table 8.1</td>
<td>Results for two real world problems</td>
<td>203,241</td>
</tr>
<tr>
<td>Table 8.2</td>
<td>Computational results for the simulated test problems</td>
<td>204,242-6</td>
</tr>
<tr>
<td>Table 8.3</td>
<td>Model 1 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time</td>
<td>208,247</td>
</tr>
<tr>
<td>Table 8.4</td>
<td>Model 2(a) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time</td>
<td>208,248-9</td>
</tr>
<tr>
<td>Table 8.5</td>
<td>Model 2(b) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time</td>
<td>209,250-2</td>
</tr>
<tr>
<td>Table 8.6</td>
<td>Model 3 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the greedy solution over the optimal solution expressed as a percentage</td>
<td>212,253-4</td>
</tr>
<tr>
<td>Table 8.7</td>
<td>Model 4 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the optimal solution over the sample size expressed as a percentage</td>
<td>212,255-6</td>
</tr>
</tbody>
</table>
Chapter 1.0: A Literature Review of Conjoint Analysis

Products and services can be defined as bundles of attributes that are each made up of many levels. Consumers consider tradeoffs between various attributes and levels before making a purchase decision for a product. A popular marketing research technique called conjoint analysis is used to study these tradeoffs. The data on consumer preferences is generally decomposed using a regression procedure into utility values that the consumer associates with each level of every attribute. In the design of a new product, it is important to determine a judicious combination of attribute levels that is likely to perform well in a market containing competitor products. Using these utility measures, a firm can design products that would best meet consumer expectations and their objectives.

A popular way to measure the success of a new product is to calculate the amount of market share it is expected to capture in a competitive market. Researchers have proposed the share-of-choice problem whose objective is to maximize the number of respondents from a conjoint study who prefer a given (new) product to their status-quo products. The share-of-choice problem is NP-hard and the linear-programming based branch-and-bound algorithm is inefficient in solving real world instances of it. Although several heuristic procedures to solve the share-of-choice problem have appeared in the literature, we wish to propose an exact algorithm to solve real world instances of the said problem to optimality. An experiment is designed to test the run-time performance of the exact algorithm using simulated problems that mimic real world problem characteristics. We also suggest potential extensions of the exact algorithm to solve modified versions of the share-of-choice problem, and also to solve the related product line design problem. This chapter provides a review of the conjoint analysis (CA) methodology.
CA is a popular statistical tool used by marketing researchers for aiding in product design. A discussion on the CA technique finds a place in most market research textbooks, such as in Tull and Hawkins (2003). A comprehensive introduction to CA theory and its applications in product design and positioning is given in Green and Krieger (1993). Green and Srinivasan (1990) provide a survey of new developments in CA. Green et al. (2001) summarize the developments in CA over the last thirty years and list some of the major contributions to CA during the period. Orme (2003B) has written an introduction to CA methodology avoiding technical jargon. An excellent source for pedagogical material and research papers on CA is available on the Internet at sawtoothsoftware.com, maintained by Sawtooth Software Inc., a leading provider of software for CA applications and consulting expertise in CA.

CA methodology has evolved over the years through new ideas proposed by researchers. Today, a large variety of products are designed optimally using CA. Green and Srinivasan (1990) list two trends that have resulted in increased popularity of the CA technique among industry practitioners. The first trend is “the development and diffusion of packaged personal computer (PC) software designed to make CA easily and inexpensively accessible to industry practitioners”. The second trend is “the scaling up of conjoint studies from their focus on narrowly defined tactical problems to higher-level strategic issues”. Carroll and Green (1995) also state two important trends in CA applications. The first trend calls for “a demand for techniques that can handle a large number of attributes and levels”. The second trend shows “an interest in data collection methods that are choice-based, instead of ratings, paired comparisons or self-explications”. Orme (2005) quotes that, “Thousands of conjoint studies are conducted
each year, over the Internet, by fax, using person-to-person interviews or mailed paper
surveys. Leading organizations are saving a great deal of money on research and
development costs, successfully using the results to design new products or line
extensions, reposition existing products, and make more profitable pricing decisions”.

The chapter is organized into the following sections. CA is defined in Section 1.1
followed by a description of the CA procedure in section 1.1. The family of conjoint
techniques and the situations in which each are recommended for use are given in section
1.3. The desirable characteristics of CA, its underlying assumptions and problem issues
form the topic of section 1.4. A description of pooled data models is given in section 1.5.
The uses and limitations of the CA technique and the strengths and weaknesses of
individual conjoint methods are elaborated in section 1.6. Suggestions for improvement
and new developments are given in section 1.7. The importance of conjoint market
simulators, their properties and a description of software products that are available for
conducting market simulations and sensitivity analyses form the topic of section 1.8. The
last topic introduces the issue of finding an optimal product design. We end the chapter
with an outline of the topics of each of the chapters to follow.
1.1 Introduction to CA

We begin by defining the following terminology (italicized). An attribute is a key
feature of a product, such as price or brand. An attribute level is one of the values or
specifications of an attribute. For example, price can be set at levels $10, $12 or $15, and
the brand could be Toyota, Honda or Ford. A profile is a description of a product or
service by its attribute levels. For example, the profile of a car can be described by
choosing a single attribute level from each of the attributes such as price, make, model,
size, color, power etc. A choice set is a set of alternative profiles that respondents evaluate during a study. Two examples of products with several attributes and each attribute having at least two attribute-levels are described below. First, consider a product such as a Personal Computer (PC). To successfully design a PC, its manufacturers have to consider several attributes each with many levels. Some of the relevant attributes and its levels are shown in Table 1.1. Krieger et al. (2004) provide another example of a product, namely a sports car in the mid-price range. The relevant attributes and levels used in the study are shown in Table 1.2.

---

Tables 1.1 and 1.2 inserted here
---

Academicians and practitioners have defined CA in many ways and some of them are reproduced below. Wind (1982) states that CA “measures the joint effect of two or more independent variables (attributes) on the ordering of a dependent variable (overall liking, likelihood of trying, chances of switching, intention to buy, believability, worth or some other evaluative measure)”. Green and Srinivasan (1990), and Yoo and Ohta (1995) define CA as “any decompositional method that estimates the structure of a consumer’s preferences, given his or her overall evaluations of a set of alternatives that are pre-specified in terms of the levels of different attributes”. Green et al. (2001) define CA as “a technique for measuring trade-offs for analyzing survey responses concerning preferences and intentions to buy, and a method for simulating how consumers might react to changes in current products or to new products introduced into an existing competitive array”.
The CA technique is used to model purchase behavior, McCullough (2002). The respondents make trade-offs of competing profile descriptions. Then, the CA technique uses these trade-off responses to identify purchase preferences for each respondent. When asked directly about their purchase preferences the respondents may not disclose them, and in many cases may not even be aware of their purchase preferences. The CA technique is used by a decision maker to deal with options that simultaneously vary across two or more attributes, Krieger et al. (2004).

Krieger et al. (2004) have stated the following with regard to the origin of CA. “Luce and Tukey are credited with the initial work in what is today known as CA. The authors were interested in the conditions under which measurement scales for both dependent and independent variables exist, given only (1) order information on the joint effects of the independent variables, and (2) a hypothesized composition rule. The authors called their approach conjoint measurement. Since market researchers are more interested in parameter estimation and scaling, they adopted the name CA”. According to Orme (2004), the word “conjoint” does not refer to respondents evaluating attributes of products considered jointly (italics intended). Instead, he considers that the adjective “conjoint” derives from the verb “to conjoin” meaning “joined together”.

Orme (2005) provides the following reasoning for the CA terminology. Asking respondents about their preferences for the attributes that comprise a product does not reflect how respondents choose a product in real life. Respondents (consumers) generally evaluate competing product profiles in a study (purchase situation). Since each profile includes multiple conjoined product features, CA is the terminology used to refer such studies. Huber (1987) states the following reasoning for using the CA terminology.
“Consider the statement that one prefers a $28000 convertible to a $22000 sedan. This statement implies that the benefit of a convertible over a sedan is greater than $6000. Psychometricians were able to show that by putting together a number of such preference statements, it is possible to derive intervally-scaled additive partworth utilities that could underlie these preferences. A number of tests were specified to determine if such an interval scale is justified, given the preference orderings. Conjoint measurement provided a theory for creating a measurement scale from judgments on compound or conjoint objects. The marketing researchers adapted conjoint measurement to reflect reality and chose to use the term CA”.

Concept testing and CA (of a new product) are different techniques. Krieger et al. (2004) list the following differences between these two techniques. “Concept testing is used when the concept is simple and inexpensive. The costs of being wrong are relatively low, the concept is easy for the consumer to understand and assimilate. Only short-term decisions need to be made and the concept is most often an extension of an existing concept (examples include toothpaste, soft drink, hair conditioner, breakfast cereal etc.). On the other hand, CA is used when the concept is complex, the costs of being wrong are relatively high, and the consumer’s decision-making process is relatively complex and time consuming. Also, CA aids in designing a product. The BASES system from A. C. Nielsen is widely used for concept testing”.

The rationale for using the CA approach as described in Wind (1982), Herman and Klein (1995), and Orme (2002C, 2005) is given below. In a CA study, respondents express their preferences for product profiles composed of different combinations of the levels of various attributes. But, their preference values for individual attributes and
levels are not known from their overall response to these product profiles. Respondents evaluate product profiles composed of multiple conjoined attributes. Hence, it is possible to figure out the preference scores that might have been assigned to the attributes and levels of the product that would have resulted in those overall evaluations. If the importance of each attribute and its levels can be quantified, then it is possible to model consumers’ reaction to any product defined by those attributes, whether or not that product was used to elicit preferences. It is assumed that the value of a product is equal to the sum of the values of each of its attributes, for any given respondent.

Given a choice, buyers would seek products with all the desirable attributes at the lowest possible price. Similarly, if sellers were given a choice, they would seek to maximize profits by producing products that buyers would prefer over those offered by the competition, at the lowest possible cost and selling them at the highest possible price. The following example demonstrates the value of a CA study and is borrowed from Orme (2005). Consider a credit card product. Some of the important product attributes are the brand, interest rate, annual fee and credit limit. A straightforward approach has been to ask the following direct questions about each of the attributes to understand preferences of each respondent.

- What brand do you prefer?
- What interest rate would you like?
- What annual fee would you like?
- What credit limit would you like?

It is expected that most respondents would prefer a popular brand that carries a low interest rate, does not charge an annual fee and offers a high credit limit. But such a
product may not be viable for the seller. So, this direct questioning approach clearly fails. Another approach is to ask the respondents to assign importances to the various attributes. But, respondents would often state that most attributes are important because there is no need for them to discriminate between the attributes. Hence, this approach is also not helpful. It is clear that the respondents must be forced to make trade-offs while making a choice, as in a CA study. A detailed description of the CA procedure is given in the next section.

1.2 The CA procedure

The steps in the CA procedure have been described in Wind (1982), Dobson and Kalish (1993), Herman and Klein (1995), Lilien and Rangaswamy (1998), Easton and Pullman (2001), Krieger et al. (2004), and Orme (2005). The steps in a CA procedure are given below, based on the description in the aforementioned research papers.

1. Develop a set of attributes and the corresponding attribute levels. If a new product is to be introduced in an existing market, the attributes and levels must represent the set of existing competitor products. Use of focus groups, consumer interviews and internal corporate expertise can be made to identify the attributes and levels. The issues involved in generating attributes and levels are discussed in Krieger et al. (2004).

2. Develop attribute bundles (product profiles) for evaluation by potential consumers. Fractional factorial designs help reduce the number of stimulus descriptions to a small fraction of the total number of possible combinations, Green et al. (1997). Fractional factorial designs are used to develop the attribute bundles whenever there are a large number of attributes considered in the study.
that cause the evaluation task to become difficult and time consuming. To reduce the number of attribute bundles that a respondent needs to evaluate, researchers have come up with approaches like the hybrid conjoint model, adaptive conjoint model, bridging designs etc. While developing attribute bundles, care should be taken to ensure that they are realistic.

3. Determine the evaluation criteria, such as intention to buy, intention to switch from current brand, appropriateness for consumption in various occasions, overall liking, and likelihood of trying.

4. Decide on how to present the stimulus for evaluation by respondents, such as verbal description, pictorial presentation, advertisement format, video presentation, and static or dynamic (2 or 3-dimensional) images over the Internet. The data collection methods can be computer based, questionnaire based, personal interview based, through mail, through telephone or using the World Wide Web, Cohen (1997).

5. Alternate product profiles are presented to respondents in the study for evaluation by a suitable criterion. The sample of respondents may belong to a chosen market segment. Respondents may give their preference evaluations by using pair-wise comparison of attribute bundles, by rank ordering attribute bundles, by using a rating scale or by making a choice from a choice set. Examples of the types of data collected are given below.

Ranking – A Likert type scale using 1-7 to indicate low to high desirability,
Constant sum – The task is to allocate a total of 100 points among shown profiles,
Likelihood of purchase – Use scale from definitely would not to definitely would.
Choice modeling - Select one out of choice set of profiles, including no choice.

By independently varying the attributes that are shown to the respondents and observing the responses to the product profiles, the conjoint analyst can statistically identify the product attributes that strongly influence choice. Respondents usually complete between 12 to 30 conjoint questions. The questions are designed using experimental design principles of independence and balance of the attributes.

6. A categorical variable is assigned to each level of each attribute. With the product profile evaluations as dependent variables, the influence of each independent variable (attribute level) is estimated through a regression procedure such as ordinary least squares (OLS), Logit, non-metric or the recent hierarchical Bayes (HB) method. The regression coefficients are the partworths. The scaling unit is common across all attributes and this allows us to add up partworths across each attribute level to obtain the overall utility of any profile composed from the basic attribute levels. McCullough (2002) suggests the regression method to use for different types of data collected. Rating or ranking based partworths is estimated using OLS regression. Choice based partworths used to be estimated using logit regression at the aggregate level. But, now HB modeling is used to estimate utilities at the individual level.

Green and Krieger (1993) describe additive models as follows. In vector notation, a product profile is represented as \( X = (x_1, x_2, \ldots, x_M) \), where \( x_1, x_2, \ldots, x_M \) represent a chosen level from each of the \( M \) attributes that are required to completely describe product \( X \). The compositional model is defined as \( U_i(X) = \)
\[ \sum_{j=1}^{M} w_j \beta_j(x_j), \] where the utility of product X is given as the sum of the products of attribute importance weights \( w_j \) and the partworths \( \beta_j \). The partworths can be obtained from a regression-like decompositional procedure. The decompositional model is represented as 

\[ U_i(X) = \beta_0 + \sum_{j=1}^{M} \sum_{k=1}^{L_j} \beta_{jk} d_{jk}, \]

where \( \beta_0 \) is the intercept term, profile X is defined in terms of dummy variables \( d_{jk} \) (\( d_{jk} = 1 \) if attribute j is at level k, 0 otherwise) and \( L_j \) refers to the number of levels for attribute j. The functions \( \beta_k \) are referred to as partworth functions. Orme (2002B) discusses an example of a conjoint study and explains how partworths can be obtained using the statistical analysis tools within Microsoft Excel spreadsheet.

7. The partworths are used in many ways. Consumers can be segmented using cluster analysis. Using suitable choice rules that incorporate managerial objectives and constraints, optimal or near-optimal product designs can be identified, any new product concept can be evaluated, and forecasts of market share can be made. Market simulators also help to study how each respondent would react to a market scenario involving many competitor products as well as the effects of changes in price, brand or product line characteristics.

1.3 Family of CA techniques

There are different ways to elicit consumer preferences. Product profiles can be shown in many different ways to obtain respondent evaluations. According to McCullough (2002), CA as a family of techniques involves three branches, namely ratings based, choice based
and hybrid techniques. The alternative approaches to measuring preference structures are shown in Figure 1.1.

Figure 1.1 inserted here

The different CA techniques are summarized from Green and Srinivasan (1990), Green and Krieger (1993, 1996) and Green et al. (1997), Green et al. (2001), McCullough (2002), Orme (2003A, 2005), Krieger et al. (2004), and the ACA 5.0 technical paper.

1. Tradeoff-tables method: The respondent is shown a sequence of tables involving two attributes each and asked to rank the cell descriptions of each two-way table in terms of preference. The other attribute levels are assumed to be constant. Newer conjoint methods have displaced the tradeoff-tables method.

2. Constant sum method: Respondents assign 100 points across the full set of profiles, so as to reflect their relative desirability of the profiles.

3. Full profile conjoint method: Respondents receive a set of full profiles based on an orthogonal design, and respond by ranking and/or rating each profile on a likelihood of purchase scale. Product profiles are shown one-at-a-time. The profiles are sorted into ordered categories and then each is rated on a 0-100 likelihood of purchase scale. This method is also referred as the traditional conjoint method. Tables 1.3 and 1.4 each show a sample profile (prop card) that is evaluated by respondents.

Table 1.3 and Table 1.4 inserted here
The questionnaire can have questions that seek answers to either pairwise comparisons or single concept designs or both. “Showing one product at a time encourages respondents to evaluate products individually, rather than in direct comparison with a competitive set of products, Orme (2003A)”. Table 1.3 shows a prop card with six attributes. Each respondent typically needs to evaluate around 24-36 such prop cards. Respondents find the evaluating task to be challenging because they have to read a lot of information to complete the evaluation tasks. Table 1.4 shows a prop card with fifteen attributes. In this case, respondents need to evaluate around 60-90 cards. Most respondents are unable to complete these evaluation tasks because there is just too much information for them to process. In most cases, the respondents are forced to simplify by focusing only on the top few important attributes. To keep the evaluation tasks simple and meaningful, it is recommended that at most six to eight attributes be shown on a prop card.

Partial profile conjoint studies involve a sub-set of the set of attributes whereas full-profile studies involve one level from every attribute. Conjoint studies can be conducted using full-profile or partial-profile designs. Partial profile designs are typically used when there are a large number of attributes.

4. Self explicated preference method: The value of an option is calculated as the sum of the desirability of each attribute level times its attribute importance. This method is compositional.

5. Hybrid conjoint method: The respondents are administered a self-explicated evaluation task and a small set of full profiles for evaluation. The individual
partworths are composed of data obtained from both tasks. More specifically, respondents evaluate the desirability of each attribute level on a degree of desirability scale, say 0-10. Then, the importance of each attribute is rated on a constant sum scale, say 0-100. Finally, the purchase-likelihood of a full profile is obtained for each respondent on a purchase scale, say 0-100. For more information, refer to Green (1984), Green and Krieger (1996) who survey recent developments in hybrid modeling for CA and also describe, classify and compare several hybrid models.

6. Adaptive conjoint analysis (ACA) method: Initially, the respondents are each administered a self-explication task through which their more important attributes are identified. Next, a set of partial profile descriptions are shown, two at a time. The respondents evaluate each pair of partial profiles on a graded paired-comparisons scale to refine the partworths for their more important attributes. ACA uses the partial profile rather than the full-profile method. Both these tasks are administered through a computer. The respondents’ interaction with the computer is said to increase their interest and involvement with the task.

The term “adaptive” relates to the fact that the computer-based data collection is customized for each respondent. At each step, answers given to previous questions by a respondent are used to decide the next paired comparison question to ask. This way, the analyst can have better information about the respondent’s preferences. In the ACA method, the computer is used to conduct the interview and obtain the respondents answers. ACA is an example of a hybrid technique. Rich Johnson of Sawtooth Software Inc. is credited with the development of ACA.
in the 1980’s. ACA method can handle studies with more attributes than was possible with the full-profile conjoint method. Tables 1.5 to 1.8 illustrate how the ACA method is used to obtain respondent evaluations.

Table 1.5 – Table 1.8 inserted here

Table 1.5 is used to obtain desirability ratings for each attribute level corresponding to a given (color) attribute. Table 1.6 shows a paired comparison question (two partial profiles of an automobile) used to obtain consumer preferences. Table 1.7 is used to obtain attribute (color) importance. Finally, Table 1.8 is used to obtain purchase-likelihood ratings for a given partial profile (automobile).

7. Choice-based conjoint (CBC) method: Respondents are asked to either choose their favorite profile from a set of profiles shown in the evaluation task or indicate that they do not wish to choose any of the profiles. The CBC method is used to study the purchase process for products in competitive markets, in which respondents choose instead of rating or ranking product profiles. An example of the CBC method displaying a set of product profiles for evaluation can be seen in Table 1.9.

Table 1.9 inserted here

In CBC models, the purchase situation should be as realistic as possible. The idea is to study consumer behavior in a purchase situation rather than ask him about his preferences directly. The respondent is shown, one at a time, several
hypothetical purchase situations each of which specifies the set of brands that the respondent can choose from. The choice situations may differ in terms of the brands from which the respondent can choose, the attributes and levels of one or more of the brands, and situational attributes.

The CBC method became popular starting in early 1990’s and is the most widely used conjoint technique in the world today, Orme (2005). In CBC, the partworths (parameters) were traditionally estimated at the total sample (aggregate) level. Of late, the availability of latent class and HB estimation methods (discussed later in the chapter) can aid in analyzing data at the segment or individual levels. When there are a large number of attributes and levels used to describe a product, it is not advisable to show a set of full profiles for evaluation, for reasons such as information overload and respondent fatigue. The partial profile CBC method is recommended since it increases the number of attributes that can effectively be measured using the CBC method. Partial profile CBC applications are quite popular.

Discrete choice models are types of statistical analysis like a logit or a probit that can be applied to data obtained from a CBC study. Also, there are situations like the transportation-mode choice studies which can by analyzed by discrete choice models. In these models, the characteristics of each choice alternative may exist for one alternative (maintenance costs for a car, none for public transport) and may exist for all alternatives (time taken to get to work). Traditionally, survey and scanner data are analyzed using discrete choice models.
1.3.1 Which Conjoint method to use?

The conjoint method used should reflect how prospective buyers make decisions in a purchase situation. Several researchers have made valuable suggestions on which conjoint method is most appropriate for a particular research situation. In this section, we summarize the recommendations of Huber (1997), Green (2001), McCullough (2002), Orme (2003A) and Krieger et al. (2004).

The full-profile CA method is recommended in the following situations. 1) A potential product is introduced in a new or stable market in which competition is either absent or does not retaliate against new product entries. 2) The consumers make choices of products by considering only a limited number of attributes. Also, a desirable attribute-level is less important than an undesirable attribute-level in influencing choice. 3) The consumer makes a decision without requiring comparisons between pairs of product profiles. 4) The product entry is into an oligopoly, monopoly or emerging markets. 5) The product is new without close substitutes in the market. 6) The objective of the study is to obtain market shares of all brands in the market.

Self-explicated models are recommended in the following contexts. 1) Many attributes need to be considered by a consumer before making a decision. 2) Expectations about attribute-levels and associations among attributes are stable in the given market. 3) Consumers make a choice by considering each product by itself, without the competing products in mind.

Hybrid conjoint methods, which combine self-explicated scaling with full-profile conjoint or CBC methods, are recommended when the product comprises a large number of attributes. Many hybrid conjoint methods have been developed to simplify the
respondent’s evaluation task. Using experimental design techniques, the number of product descriptions to be evaluated is kept at a minimum as also the number of attributes presented in a single product profile description.

The ACA methods involving graded paired-comparisons of product profiles are appropriate for the following situations. 1) The new product is introduced in a market in which the competing products are explicitly compared with one another before a purchase decision is made. 2) The product fits in the category of high-involvement purchases, where respondents seek information on a range of attributes before making a final decision. 3) The within-attribute value steps between various levels are smooth and approximately linear.

CBC methods are most appropriate in the following situations. 1) To seek consumer response to competitive offerings, especially for brand and price studies. 2) Purchase decisions are made based on relatively few, popular attributes with a strong dislike for the undesirable levels of each attribute. 3) Consumers make purchase decisions on the basis of competitive differences among the given attributes. 4) The product is introduced in a highly competitive market, where the competitors are expected to retaliate. 5) The new product is an extension of an existing product. 6) The objective of the study is to obtain market shares of all brands in the market. 7) The product fits in the category of low-involvement purchases and the profiles are described on only a few attributes. 8) Attribute interaction effects need to be measured. 9) Respondent data consists of only a few questions answered by each respondent. The CBC method pools respondent data and can handle this situation as long as more number of respondents are included in the study.
1.4 CA desirable characteristics

For CA studies to be successful, certain problem characteristics are desirable. Huber (1987, 1997), Krieger et al. (2004) and Orme (2005) describe some of the desirable characteristics for CA that is summarized below.

1. The product is decomposable into a set of attributes and attribute levels.
2. The product choice is not impulsive, but a high-involvement decision.
3. The factorial combinations of basic attribute levels presented to respondents for evaluation are believable.
4. Any new product can be created from the basic attributes.
5. The product profiles can be realistically described verbally, pictorially or by any other means.
6. The different combinations of profiles are perceived to be reasonably homogeneous by the respondents.
7. The attribute levels should have a clear and unambiguous meaning.
8. The conjoint method must use tradeoffs that are similar to those in the market.
9. When respondents are asked to evaluate tasks involving a large number of attributes, they simplify their tasks by selecting a small subset of attributes that are important to them and make their decisions based on those attributes. Such simplification strategies of respondents should take place in purchase situations.
10. Conjoint profiles are designed to be orthogonal. An orthogonal design is one in which the levels of different attributes across profiles are uncorrelated. Such designs ensure that an estimate of one attribute is not affected by the estimate of the other attributes.
1.4.1 CA assumptions

CA studies are based on several assumptions. Some of the important assumptions are listed below based on the discussions in Huber (1987) and Orme (2004, 2005).

1. The value of a product is equal to the sum of the values of its constituent attribute levels, called the simple additivity rule. This rule is most frequently used in CA studies. Other rules such as an attribute-weighted additivity rule or a non-linear additivity rule also seem possible.

2. The respondents who are interviewed belong to the target segment and have answered the questions in a truthful manner.

3. The preference measurements are directly related to purchase choice. That is, individual preferences can be expressed in numerical terms that lead to purchase behavior.

4. The partworth estimation techniques are reasonably accurate and reliable.

5. All the attributes and levels that actually affect buyer choice have been included.


7. Equal consumer awareness of all competitive products in the product category.

8. Equal time on market and equal effectiveness of sales force.

9. The attributes are statistically independent of one another. That is, attributes are uncorrelated with one another. For example, brand and price attributes are found to be correlated and special care must be taken in CA studies involving both these attributes.

10. Each attribute-level is mutually exclusive of (distinct from) the other levels in that attribute.
11. Complex decision-making can be explained using a limited number of attributes and levels.

12. The conjoint data obtained from respondents is preferred as ordinal scaled rather than interval scaled.

1.4.2 Issues to resolve in conjoint studies

There are a few issues to resolve in conjoint studies failing which the results from a conjoint study may not be used to predict the purchase behavior of consumers. Some issues are discussed in this sub-section.

At first, consider the number of levels (NOL) issue. It is reported in Green and Srinivasan (1990) that the relative importance of an attribute increases with an increase in the number of levels (NOL) on which it is defined, in spite of holding the minimum and maximum values for the attribute fixed. Apparently, the addition of levels to an attribute forces the respondent to pay more attention to that attribute. Hence, more the number of levels in an attribute, greater the importance of that attribute in determining product preferences. Orme (1998B) makes the suggestion that “optimally weighting the self-explicated ratings with the paired comparison conjoint evaluations can help reduce the NOL effect”. According to Wittink et al. (1992A), “in problems with attributes that have varying number of levels, ACA that permits paired comparisons with non-dominated pairs are preferred since it helps reduce the NOL effect”. Clearly, more research is needed to alleviate this problem.

The second issue is that of the Attribute Range (AR). AR signifies the magnitude of difference between the maximum and minimum levels of an attribute. AR affects attribute importance and must be chosen carefully. Consider the following example
given in the ACA 5.0 technical paper. “If all airline tickets from city A to city B were to cost between $100 and $101, then price could not be important in deciding which airline to select. However, if cost varied from $10 to $1000, then price would probably seem very important”.

The third issue is that of Attribute additivity (AA). AA is a problem that occurs when a large number of attributes are included in a study. AA results in a large number of less important attributes overwhelming a few important attributes. Hence, the attributes must be determined carefully.

The fourth issue relates to a property called the independence from irrelevant attributes (IIA). Some of the consumer choice rules (discussed in chapter 3) suffer from the IIA property. The IIA property stated below is based on the discussion in Lilien and Rangaswamy (1998), and Chen and Hausman (2000). The IIA property states that the choice probabilities from any sub-set of alternatives depend only on the alternatives included in the set, and are independent of any alternatives not included. The IIA property causes the problem of share inflation of similar products. Consider the following example from Lilien and Rangaswamy (1998). Suppose an individual chooses between a Toyota Camry and a Honda Accord while purchasing a new car, and the probability of selecting either option is given to be 0.5. If the choice set were changed to a Blue Toyota Camry, a Green Toyota Camry, and a Honda Accord, it is expected that the probability of selecting the three alternatives would be 0.25, 0.25, and 0.5 respectively, assuming that the consumer pays no attention to color. But, some of the choice rules such as the share of utility and logit choice rules would predict the above probabilities to be each equal to 0.33, which is not what is expected.
Orme (1998A) shows that aggregate logit models suffer from the IIA problem much more than models that consider respondent heterogeneity, such as the latent class and individual-choice models. The use of hierarchical Bayes (HB) estimation technique in CBC studies to obtain individual-level partworths reduces the IIA problem, and improves the predictive validity of the model, Johnson (2000).

1.5 Pooled data models

Pooled data models include fully aggregated and partially disaggregated models. In this section, we describe the pooled data models that are discussed in Krieger et al. (2004) and Orme (2003A, 2005). In aggregate choice models, the partworths of the entire set of respondents in the study are estimated at the aggregate level after assuming respondent homogeneity. That is, a single set of partworths represents the average preferences of a population. In partially disaggregate models, the partworths are estimated at partially disaggregated levels such as clusters or latent classes, which are assumed to be homogeneous with respect to respondent preferences.

In latent class analysis, homogeneous respondent segments are first identified and then partworths are estimated for each segment. That is, partworths are assumed to be constant within segments. According to Orme (2005), “Latent class methods permit each respondent to have some positive probability of belonging to each class, and solve simultaneously for the utilities for each class and for each respondent’s probability of belonging to each class”. Latent class models and clustering models leverage the benefits of aggregate estimation while recognizing market heterogeneity and thereby represent a middle option between fully aggregated and individual-level conjoint models.
Disaggregate models are preferred to aggregate models in conjoint studies because aggregate models do not capture consumer heterogeneity, Green and Srinivasan (1990) and McCullough (2002). But, pooled data models are shown to be successful in the following situation. Krieger et al. (1998) suggest that pooled data models such as clusterwise, latent class or choice-based conjoint, will be successful in predicting market shares when researchers employ monotonic attributes. A monotonic attribute is one where more of a good attribute and less of a bad attribute is always preferred by respondents. In other words, aggregate models are not as successful in predicting market shares using holdout samples when many attributes are non-monotonic. A holdout sample represents conjoint data that is not used to estimate partworths, but withheld so that market shares can later be predicted using it.

The CBC method does not provide stable individual-level estimation because unlike rankings or ratings, choices contain less information, leading analysts to use aggregate models. In CBC evaluations, respondents indicate whether or not they would choose a product from a set of product options. From these evaluations, it is not known how much more desirable the chosen product is over those not chosen. Also, it is not possible to know the relative values of the non-chosen product profiles. Therefore, analysts pooled conjoint data using methods such as logit to model the “average” respondent. This leads to two problems that are mentioned in Orme (2005). 1) Since individual-level partworths were not available, it became difficult to use choice data to develop market segments. 2) Logit models assume that the only source of variability among respondents is random and hence do not consider variability among the
respondents. Therefore, the logit model is inappropriate when respondents belong to distinct consumer segments.

The development of the HB estimation technique has helped CBC studies considerably by producing better estimates of individual partworth values in spite of using shorter questionnaires. The HB method is described below based on the discussions in Johnson (1999, 2000), Allenby and Rossi (2003) and Orme (2000, 2005). Orme (2000) defines the HB method as follows. “The HB model is called “hierarchical” because it has two levels. At the higher level, it is assumed that individuals’ parameters (partworths) are described by a multivariate normal distribution. At the lower level it is assumed that, given an individual’s partworths, his or her probabilities of achieving some outcome, say product choice, is governed by a specific model such as multinomial logit or linear regression”.

The basic idea of HB modeling is to recognize that each individual is a member of a group of similar individuals. The knowledge of the probability distribution of individuals’ partworths can improve estimation for each individual. The HB model borrows information from other respondents to stabilize the partworth estimates when respondents provide multiple observations. It is assumed that the individuals are distributed multi-normally and HB estimates the mean vector and covariance matrix for that distribution. It uses that distribution as a “prior” in the Bayesian sense, to improve the estimation of each individual’s partworths. Therefore, the HB model handles respondent heterogeneity better than an aggregate model that works with an “average” respondent. Since respondents are expensive to interview, it is desirable to ask them to
evaluate more than one choice question. The answers to these additional choice questions enable the estimation of individual level partworths under the HB method.

Orme (2004) gives two reasons why the CBC method has overtaken ACA and other ratings-based conjoint methods. The first reason was the release of commercial software for discrete choice (CBC) by Sawtooth Software in 1993. The second reason was the application of HB methods to estimate individual-level models from discrete choice.

1.6 Uses and limitations of conjoint analysis

Conjoint analysis is a valuable technique that marketers can use to study consumer preferences. Suppose that respondent partworths were homogeneous across market segments. Naturally, most firms would end up with product profiles that are quite similar to one another. This would result in more or less equal market shares for each of the products entering the market. Green and Krieger (1993) give many reasons to discount the possibility of respondents having homogeneous partworths. On the other hand, if the respondent partworths were heterogeneous, then firms would first identify various market segments based on the partworths, decide on the segments to target, and then introduce product profiles that appeal to respondents in each of the target segments. The firms would also be interested in differentiating their product profiles from those offered by the competitors so that their products are positioned uniquely.

Krieger et al. (2004) describe the close relationship between market segmentation and product positioning which we quote as follows. “Buyers and sellers wish to find an ideal product or service offering that best satisfies both buyer preference and seller profit objectives. Preference heterogeneity for products can be related to either individual
variables (demographic, psychographic characteristics, current brand loyalty etc.) or situational variables (purchase occasion, type of purchase etc.) and their interactions. Segmentation approaches consider both buyer background characteristics and product attribute partworths. In the first case, marketers target the type of buyer they are looking for and then design the best product for that type of buyer. In the second case, marketers use the partworths themselves as a basis for clustering buyers’ attribute level preferences and then design the best product for each resulting buyer segment. Marketers are interested in identifying the association between consumer segments (obtained through preference scores or partworths) and demographics. This is to enable appropriate advertising and promotion activity aimed at these segments”.


**Uses of CA:**

1. CA is an efficient concept screening method. Concept testing is typically limited to a small number of product alternatives. In CA, consumers evaluate product profiles as they would in real purchase situations. The conjoint analyst decomposes the preference data to understand how the individual attributes are driving consumer evaluations of complete products.

2. The partworths obtained from a CA study can be used to identify the best product profile based on a chosen objective criterion.

3. The conjoint analyst can work with any dependent variable (evaluation criteria) of their choice.
4. The CA results can be obtained at the individual respondent level so that the segmentation of respondents based on the similarity of their response to tested concept profiles can be done.

5. Using a market simulator, market share for the tested product concept can be estimated. Using multi-dimensional scaling technique, the tested product concept can be positioned against various competitor products in various market segments. Also, sensitivity analysis of product profiles can be performed.

Limitations of CA:

1. The CA method depends on trade-off questions that limit the capacity of the method to solve problems with large numbers of attributes and levels. The respondent may be unwilling to submit to a long interview, which if forced, can lead to data of questionable quality. In such cases, the researcher may be forced to consider a narrower focus in the interview.

2. When faced with a difficult task, respondents may resort to simplifying tactics and the resulting partworth estimates distorts their true preferences. Also, inconsistencies in respondent evaluations cause errors in utility estimation.

3. Repetitive questioning in CA reveals the purpose of the study to the respondents who then do not respond naturally to questions put to them.

4. Through the use of CA, it is desired to determine the most important attributes and levels that influence buyer choice. But, CA requires the researcher to decide which attributes and levels are important enough to include in the study. So, the researcher’s bias enters in the study.
5. Each respondent is assumed to choose a product based on the same objective criteria. But, consumer choice is based on subjective and idiosyncratic perceptions of a product.

6. It may be necessary to calibrate the choice shares to align them more closely to actual market shares. Factors such as advertising, distribution and promotion may affect actual market shares but are not reflected in the product attribute evaluations.

7. The CA approach assumes that consumers are rational in comparing choice alternatives and hence the CA approach works best for considered purchases rather than impulsive purchases.

8. The trade-off task in CA studies is unrealistic since real alternatives do not present themselves for evaluation often enough.

9. It is difficult for evaluation tasks to simulate real world purchase situations.

10. When using a conjoint simulator, there are many choice rules that transform consumer preferences into product choice. The different choice rules lead to different estimates of market share.

11. When segmenting respondents, cluster analysis is often used to group them based on similarity of their utility values. Again, depending on the type of cluster analysis and the way data is sorted leads to different results.

12. In CA studies with a large number of attributes and levels, interaction effects are often ignored to keep the profile evaluations within a manageable number.

13. It may not be possible to consider a product involving certain attribute levels due to constraints in manufacturing feasibility, cost etc. The way out this restriction is
to eliminate unacceptable product profiles in the evaluation process. But, this
may cause non-orthogonality of the experimental design and lead to unreliable
results.

14. Market simulators often assume a choice rule wherein respondents may buy a
product even if it has a low utility value. But, respondents may decide not to
purchase a product with a low utility value.

1.6.1 Strengths and weaknesses of conjoint methods

There are many useful techniques for modeling consumer preferences and no one
technique dominates over all the others, Orme (2002). Green and Krieger (2002) refer
to a study they conducted in 1998 to compare the performance of six different conjoint
methods by the same input data based on several objective criteria. The self-explicated
data model did not perform well while the conjoint and hybrid conjoint models
performed best. This result was not acceptable to researchers using self-explicated
method. So, Gibson and Marder (2002) suggested that the self-explicated and conjoint
methods be compared using an experiment design that is acceptable to all researchers.

Self-explicated preference method:

The strength of the self-explicated preference method is that it can handle a large
number of attributes and levels. The weaknesses of the self-explicated preference
method are listed below based on the description in Green and Srinivasan (1990) and

1. Respondents have limited abilities to judge the relative importance of large
numbers of attributes with high reliability.

2. Redundancy of attributes may lead to double counting.
3. Self-explicated importances do not reveal differentiation between attributes. Respondents do not discriminate enough between most important and least important attributes while using self-explicated importance ratings method.

4. The emphasis in consumer choice studies should be on selections that respondents make to profiles in choice sets and not on attribute importances and attribute level desirabilities.

5. If some attributes are correlated among themselves, it is difficult for the respondent to provide ratings for the levels of an attribute holding all else equal.

6. Questions on the importance of socially sensitive factors may lead to biased answers. It is reported in a study that MBA students ranked salary as the sixth most important factor when asked directly. But, the importance weights derived from CA indicated that salary was number one in importance.

7. It is difficult to associate attribute importances with preference evaluations for a complete product.

ACA method:

The strengths and weaknesses of the ACA method described below are based on the description in Orme (2003A, 2005). The strengths of the ACA method are listed below.

1. The ACA method has the ability to measure many attributes without wearing out the respondents. Respondents find the interview more interesting and engaging and perceive it to take less time than a full-profile conjoint study.

2. The ACA interview is efficient. There is a high ratio of information gained for the effort taken by the respondents.
3. The ACA method can be used even with very small sample sizes.

The weaknesses of the ACA method are listed below.

1. When comparing two products to rate those, the consumer is assumed to choose one of them. There is no provision to not choose either of them. Also, partial-profile presentations are less realistic.

2. Respondents may not be able to assume that attributes not shown are held constant.

3. The ACA method must be computer-administered. The software either resides in a PC or can be accessed through the Internet.

4. The ACA method is a main-effects model and attribute interactions cannot be included.

5. The ACA method tends to understate importance of price, and within each respondent assumes all brands have equal price elasticities. Therefore, it is not good at pricing research.

CBC method:


1. The CBC method is more realistic for respondents. The choice questions mimic real world purchase situations by asking respondents to choose from a set of available products, while other CA methods ask respondents to rate, rank or compare products.
2. CBC can study the impact of price and attributes on a set of predefined groups. It can quantify brand equity for a chosen brand by measuring the price premium it commands over other brands.

3. CBC can investigate interactions and alternative-specific effects.

4. CBC can include “No Choice” alternative, or multiple “Similar” alternatives.

5. Paper or Computer/Web based interviews are possible.

6. In CBC, unlike in other CA methods, each product profile need not share the same set of attributes and levels.

7. CBC can study choice behavior using a nested choice structure.

8. In CBC, the estimation of preference utilities and simulation of choices take place simultaneously and the user does not have to suggest a simulation choice rule.

9. Segmentation of respondents can be done using latent class analysis and individual-level estimation of partworths can be done using the HB method.

The weaknesses of the CBC method are listed below.

1. The CBC study design is inefficient since respondents have to read a lot of information before making a choice. Strength of preference could not be measured and hence each respondent’s preferences could not be modeled separately. Instead, aggregate models of preferences were developed across groups of respondents, which are subject to the IIA problem.

2. CBC methods usually require larger sample sizes than with full-profile or ACA methods.

3. The choice tasks are more complex, so respondents can process fewer attributes. A limit of six attributes is recommended because choice tasks involving more
than about six attributes are likely to confuse respondents and may encourage response simplification strategies.

4. CBC data analysis is more complex than with full-profile or ACA methods, particularly if individual-level estimates of partworths are to be made using the HB method.

5. Aggregating respondents assumes respondent homogeneity, which is unrealistic.

6. Choices are inefficient. They tell us only what product is preferred, not how much. In CBC, the analyst obtains far less information than would be available had the task been to rate or rank each alternative in the set.

7. CBC tasks are difficult to design for the average user.

1.7 Suggestions for improvement and new developments

Three types of questions need to be asked in a conjoint study, McCullough (2002). The study begins with the warm-up task, where questions are asked to stabilize the responses. The next set of questions represents the conjoint task and is used to estimate the partworths. The final set of questions represents the holdout task and is used to validate the model after the partworths have been estimated. Also, it is important to pretest conjoint studies to make sure they can be implemented. Testing the assumptions, such as the additivity of utilities, must be done before estimating the partworths.

Toubia et al. (2003) propose and test (1) a new adaptive question design method that attempts to reduce respondent burden while simultaneously improving accuracy, and (2) a new estimation procedure based on centrality concepts (interior points of a polyhedra). For each respondent, the question design method adapts the design of the next question using that respondent’s answers to previous questions. The estimation
method relies on interior point techniques and is designed to provide robust estimates from relatively few questions. Adaptive question design methods are said to suffer from endogeneity bias. That is, the nth question depends on the respondent’s answers to the previous (n-1) questions. If the previous (n-1) questions have any response errors, then bias enters the design method. The new adaptive question design method helps us get better partworth estimates faster, but with the risk of endogeneity bias.

Green and Krieger (1993) and Orme and King (1998) have discussed the implementation of conjoint studies on the Internet. Internet usage of conjoint applications is growing rapidly and conjoint researchers have started using the Internet to conduct email and online surveys. Orme and King (1998) describe the pros and cons of using email and online surveys. They provide details of a trivial full-profile conjoint study that was implemented on the Internet successfully. They also describe the implementation of pairwise and single-concept presentations for full profile conjoint, and report both methods to be equally effective.

Dahan and Srinivasan (2000) describe an Internet-based product concept testing method that was developed to depict virtual prototypes of new product concepts instead of physical prototypes. The virtual prototypes of product concepts were static as well as dynamic, and they demonstrated how the product works. Results from the study indicate that the virtual prototypes produced market shares that closely approximate those obtained using physical prototypes. These virtual prototypes cost lesser to build and test, and hence a larger number of product concepts can be tested within the same budget. The Internet environment is ideally suited for obtaining data on consumer preferences with great speed and without geographic limitations. There are a few issues that need to
be addressed and are described next. Sampling is difficult to control over the Internet. Hence, careful panel recruitment and control are essential. Sensory experiences are difficult to communicate over the Internet. Technical issues like Internet bandwidth may be a concern. It is important to note that unlike attribute-based, full profile conjoint approach, the Internet-based virtual prototype conjoint surveys do not measure individual attribute partworths, but rather the utility for the product concept as a whole.

Most conjoint studies involve both brand and price attributes. But, brand and price are correlated and thus violate one of the assumptions in CA. Johnson et al. (1996A) give a review of the ways price can be handled in conjoint studies. In pricing studies, it is necessary to capture interactions of brand and price. The authors suggest that by using a main-effects-only model, partworths for each brand and each price can be obtained. But, this would assume that the same set of partworths for price applied to each brand. Alternatively, a composite attribute can be created to handle interactions. But, attributes with many levels are difficult for respondents to evaluate. Also, the NOL effect (see sub-section 1.4.2) causes respondents to believe that attributes with more numbers of levels are more important than they should be.

Ding et al. (2005) suggest the usage of monetary incentives while seeking preference information from subjects in a conjoint study. These monetary incentives are to be actually used during the conjoint study. Based on an experimental study it is claimed that incentive-aligned conjoint method predicts actual purchase behavior much better than hypothetical conjoint methods. Since adequate monetary incentive is provided, consumers do not discriminate on price and it seems that the incentive-aligned conjoint method has a serious problem. But, it is argued that in the absence of incentives,
respondents routinely discount their budget constraints while evaluating product concepts. The authors admit that implementing incentive-aligned conjoint method to expensive or complex products or new products would be a challenge. More applications of incentive-aligned conjoint studies are needed to verify the authors’ claim.

In many product markets, firms choose to modify certain attributes of products to reflect changing consumer preferences, improvement in technology, changes in economies of scale, and competition from other brands, Ofek and Srinivasan (2002). The authors propose a model to estimate the dollar value a market attaches to potential product modifications. Competition from other brands, potential for market expansion, and heterogeneity in consumer preferences are modeled within a multinomial logit framework. The key findings of their research work are given below. Consumers are to be differentially weighted based on their probability of purchasing the firm’s product. In particular, consumers with high or low purchase probabilities should receive less weight in the model. Also, consumers whose utilities have a larger random component should be weighted less. The dollar value for product modifications depends on the firm that is asking the question. One of the limitations of the study is that price reaction to improvement in attributes, one at a time, is considered. But, more than one attribute can be improved and the price reaction studied. We also note that Orme (2001) lists the methods used to convert conjoint utilities into monetary equivalents and makes valuable suggestions in avoiding pitfalls.

Managers usually determined market segments based on a combination of demographic characteristics of consumers and their judgment. Green et al. (1989) proposed a componential segmentation (CS) model to find optimal products for a set of
given market segments and optimal segments for a given set of products. In the CS model, the market segments and products are assumed to jointly influence consumer response. The authors explain the CS model using a pharmaceutical product as example.

A balanced or orthogonal design is one in which each level appears an equal number of times. In an orthogonal design, the columns of the design matrix are independent. Most commercial CA software, like Bretton-Clark’s *Conjoint Designer*, let analysts design experiments that use orthogonal main effects, Krieger et al. (2004). In CA, experimental designs are used to construct hypothetical products that are evaluated by respondents, Kuhfeld (1997). Kuhfeld (1997) says that with an orthogonal experimental design, the parameter estimates are uncorrelated and therefore have maximum precision. However, many practical problems in CA may have to do without orthogonal designs. To handle such problems, the author discusses nonorthogonal designs. Consider the example of a product which has many attributes, and each attribute has different numbers of levels. Also, suppose that some product combinations are unrealistic, such as the combination with the best product features at the lowest price. In the above situation, nonorthogonal designs have to be used.

Herman and Klein (1995) offer a few suggestions to improve the predictive power of CA. The main thrust of their paper is regarding the difficulties in modeling products with a large number of attributes and levels. 1) They say that the minimum number of profiles to evaluate is equal to the number of levels over all the attributes. Using too few profiles is said to cause prediction error. 2) If the impact of certain attributes on the overall reaction to a product concept is known, it is advised to keep those attributes out of the CA study. This technique is called constrained estimation. But, this technique cannot
be used on attributes for which consumers have idiosyncratic preferences. 3) Holdout evaluations are evaluations that are held out of the CA analysis. After the utility function for a respondent is estimated, it is used to predict how the respondent evaluated the holdout profiles. If the holdout profiles can be predicted well, then the respondent’s real world preferences can also be predicted well. It is important to ensure that the holdout profiles are as realistic as possible.

Allenby et al. (1995) report that many individual-level partworths obtained from typical conjoint studies are of the wrong algebraic sign (that is, a sign that is inconsistent with prior expectation). They investigate the predictive accuracy of partworth models that incorporate prior knowledge. A Bayesian approach is used to incorporate prior information about partworths into the analysis of conjoint studies. Incorporating ordinal restrictions into the estimation of partworths is reported to result in estimates that have greater predictive accuracy and validity (based on known factors such as a negative utility for a steep price increase of an attribute level).

Consider the task of estimating the heterogeneity in the consumer’s partworths in a CA study. The OLS estimation method requires each respondent to evaluate more profiles than the number of attributes, resulting in lengthy questionnaires for complex multi-attribute products, Lenk et al. (1995). Long questionnaires are observed to cause low response rates, and it is also believed that long questionnaires induce response bias. Lenk et al. (1995) “call for experimental designs and estimation methods that can recover partworth heterogeneity using shorter questionnaires”. Unlike popular estimation methods such as the Ordinary Least Squares (OLS) method, the HB models do not require the individual design matrices to be of full rank. The authors’ work analytically
investigates the tradeoff between the number of profiles per respondent (subject) and the number of respondents on the statistical accuracy of the estimators that describe the partworth heterogeneity. The result of the study indicates that shorter questionnaires can be used for studying complex products and services.

Zufryden (1988) proposes a model that uses CA to predict trial and repeat purchase patterns of new frequently purchased products. The beta distribution is shown to be a good choice to model heterogeneity of individual-choice probability over the consumer population. Holdout concepts were used to validate actual preference with predicted preference obtained from partworths data. Orme (1998C) provides the theory and tools to help conjoint analysts make sample size decisions.

Johnson et al. (1996B) discusses issues in CBC studies by seeking answers to questions such as the following. 1) How many choice tasks to ask each respondent? 2) Is it better to ask a respondent to evaluate more tasks or let each task be evaluated by more respondents? 3) Is there a systematic change in respondents’ answers as the interview progresses? 4) How long does it take respondents to answer choice questions? 5) Should we ask for just first choice for each set of concepts, or is it useful to ask for second choices as well?

The design issue in CBC is to determine the alternatives that should be included in the choice sets. Currently, most choice designs are not adaptive, and the particular choice sets individuals receive are independent of anything known about them. Johnson et al. (2003) ask if the information about an individual’s attribute evaluations can enable us to ask better choice questions. Chrzan and Orme (2000) discuss different approaches
to designing CBC experiments and several kinds of effects one might want to model and quantify in such experiments.

Green and Srinivasan (1990), Green and Krieger (1993) and Green et al. (2001) make the following suggestions for future research. 1) Developing product-positioning models that explicitly consider competitive action and reaction strategies as well as incorporation of other marketing control variables such as advertising, sales promotion and distribution. 2) Developing new simulator-optimizers that can maximize either financial return or market share. 3) Develop better models for product design and product-line design. 4) Develop profit optimization models after ascertaining the costs of the different components of a product, which are actionable and managerially useful.

1.8 Conjoint market simulators

A market simulator is used after conjoint partworths have been estimated, primarily for the following purpose. The conjoint partworths data is used to simulate market choices, which is what manager’s are most interested in. It is important to note that different models can lead to different simulator-based share estimates, depending on the composition of the competitive product profiles and the choice rule, Krieger et al. (2004). For an introductory article to conjoint market simulators refer Orme (2003C).

The early conjoint simulators took the partworths of respondents and information about competitors as input and produced predictions of market share, revenue and profit for each product as output. The respondents were expected to choose the product for which they professed the highest total utility. Conjoint simulators that are available today in the market perform many types of functions and provide various user-specified
sensitivity analyses. A list of conjoint simulator output options and potential sensitivity analysis options can be found in Green and Krieger (1993).

Krieger et al. (2004) explain the working of a typical market simulator, which is quoted as follows. “The utility functions of respondents are entered into the computer choice simulator along with a base profile that represented the management’s judgment about the kind of positioning that seemed most attractive. A number of alternative options are tested separately against the base case and a share-of-choice is obtained for each alternative. A selected few of the alternatives with highest share-of-choice are retained for further testing. A suitable choice rule is selected and the simulator computes each chosen alternative’s total utility for each respondent. The respondent selects the alternative that best meets the choice rule”.

Orme and Huber (2000) state the following. “A good simulator is like gathering a representative sample of a company’s consumers in one room to vote on product concepts and competitive scenarios. For each scenario, the consumers are offered a choice among offerings, each of which is described by differing levels on a common set of attributes”.

Conjoint market simulators make many important assumptions, which are listed below based on the description in Orme (2005).

1. All attributes that affect buyer choices in the real world have been accounted for.

2. Products are equally available (efficient distribution) at all marketplaces.

3. Respondents are aware of all the products in the market.

4. Long-range equilibrium exists. That is, time on market is equal for all products.

5. The sales force is equally effective for all products.
6. Out-of-stock conditions do not exist for any product.

Cattin and Wittink (1982) have pointed out that it is not advisable to use the CA estimates of market shares as accurate predictions. This is because there are other marketing factors that also influence a product’s market share but not included in the research, such as advertising and sales promotion. The market share estimates from conjoint simulators are often fine-tuned to match the actual market shares, which are obtained using holdout concepts. Care must be taken to prevent misuse of the simulator results by trying to accurately predict market shares this way. Orme (2005) suggests that the conjoint users should only look at the results diagnostically. In fact, Orme (1996) gives several reasons why conjoint models do not predict market share, which are listed below. Most of these reasons are closely tied to the aforementioned conjoint assumptions.

1. Conjoint assumes perfect information. In the conjoint interview, respondents are educated about available brands and features. In the real world, obscure brands have less chance of being purchased. Conjoint cannot fully account for differences in awareness or preference developed through advertising and promotion.

2. Conjoint assumes all products are equally available.

3. Conjoint respondents might not accurately reflect potential buyers. They may not have the ability to purchase at the current time.

4. Conjoint results reflect the potential market acceptance of products, given proper promotion, distribution and time.

5. In the real world, product life cycles drive market shares.
Huber (1997) suggests that conjoint results should be used for long-term prediction of market shares. In his opinion, the current market shares can best approximate short-term market shares. The author further states that firms need to know answers to the following questions to be able to predict long-term market shares.

1. What will consumers choose if they attend to the displayed attributes?
2. What will the firm do if consumers’ competitive expectations change?
3. What will consumers do if they think about how much they value an attribute?

The uses of conjoint market simulators are listed below based on the description in Orme and Huber (2000), Krieger et al. (2004) and Orme (2005).

1. Lets us specify competitive scenarios and estimates the share-of-choice (preference) using respondents’ estimated partworths.

2. Lets us conduct “what-if” scenarios to investigate issues such as the values of modifications to an existing product, a new product design, a product positioning, a pricing/marketing strategy, impact of a new product on market shares of existing products, and anticipating the impact of competitive moves on the firm’s products.

3. Lets us design new products that optimize a chosen objective criterion, such as maximizing profit or minimizing cost. That is, it lets us identify the best profile out of a limited set of simulated or user-specified profiles using a suitable choice rule.

4. Lets us investigate product line extension issues. For example, does the new product cannibalize our own share, or does it take most of its share from competitors?
5. Lets us estimate demand curves and cross-elasticity curves.

6. Provides an important input into demand forecasting models.

1.8.1 Properties of a good simulator

Orme and Huber (2000) and Huber (1987) state that if market simulators are to mimic market behavior, they have to satisfy three critical properties called differential impact, differential substitution and differential enhancement. These three properties are explained below.

First, simulators must display differential impact. By differential impact, it is meant that the impact of a marketing action depends on how close a given product alternative is to the purchase threshold, for a consumer to choose it over other products in the choice set. The threshold point occurs when the value of the given product alternative is close to the most valued alternatives in the choice set and hence the consumer has a good chance of choosing the given product. An incremental benefit to the product in question is most likely to win over consumers at the purchase threshold. The marketers can focus attention on consumers at the threshold of making a choice. It is advised to ignore those consumers who are either very unlikely (low probability) or very likely (high probability) to buy the product in question. In other words, it is better not to put managerial effort on alternatives that are already chosen or would never be. Gain in market share comes mainly from those consumers at the purchase threshold.

Second, simulators must display differential substitution. By differential substitution, it is meant that a new product offering should take share disproportionately from similar ones. That is, the new product takes a greater proportion of share from competing products that are similar to it and a lesser proportion of share from competing
products that are dissimilar to it. For a firm to be successful, its product line must consist of products that are designed to maximize share taken from competitors while minimizing loss of share of its own products. Take the example of a soft drink product. Pepsi is said to have taken most of the share from Coke, but had little impact on, say, the fruit-flavored soft drink category.

Third, simulators must display differential enhancement. By differential enhancement, it is meant that pairs of highly similar product alternatives should display extreme probabilities of choice, with the better product dominating the share-of-choice. A dominant product is one that is equal on most attributes but slightly better than its competing product on others enabling it to get a very high share. It is important to note that in the real world, there is some random buyer behavior so that the dominated product should not end up with a zero probability of purchase.

Orme (2005) suggests that to reduce the IIA problem in simulators, three properties called differential substitution, differential cross-elasticities and differential self-elasticities must be satisfied. The first property is already explained. Orme (2005) uses a “red-bus, blue-bus and train” example to explain the other two properties, which we borrow. Assume that the bus riders are price sensitive. An increase in price of blue-bus causes people to shift to red-bus. So, the model has captured differential cross-elasticities. Now, assume that the bus riders are price sensitive while the train riders are not. If train prices are raised, few train riders shift to buses. Whereas, if bus prices are raised, many bus riders may shift to trains. So, the model has captured differential self-elasticities.
Orme and Huber (2000) state, “The correspondence of simulators to market behavior can be improved by adding variability to the choice models that are used in the simulators”. Also, market simulation models that reflect heterogeneity are usually more accurate and flexible than models that do not. For example, latent class models capture heterogeneity of the market, and as a result display differential substitution effects.

According to Huber (1987), there are two main uses of a holdout task. First, it identifies respondents whose responses to a conjoint evaluation task are unlikely to correspond to their choice behavior. Such respondents are given less importance (weight) in the simulation. Second, it helps in relating the conjoint model output to actual choice. If a conjoint model does not correspond to actual choice, it can be changed or improved.

1.8.2 Conjoint analysis software

Early in this chapter, we quoted Green and Srinivasan (1990) regarding two trends that have made the CA technique popular among industry practitioners. The first trend was the availability of packaged PC software to help design and implement CA studies easily and inexpensively. The second trend was the use of packaged PC software to study higher-level strategic issues by performing various sensitivity analyses. In this subsection, we provide a partial list of conjoint software and briefly describe their functionalities.

Green et al. (1981) have described the POSSE (Product Optimization and Selected Segment Evaluation) software that is designed to conduct conjoint analyses and market simulation studies. The major components of the POSSE software include experimental design, utility function estimation, choice simulators, objective function
estimation, optimization, sensitivity analysis and time path forecasting. The optimization is carried out over different objective functions using response surface methods. The authors have explained the features and functionality of the POSSE software system through the example of a pickup truck designed for consumers in the USA.

Green and Krieger (1992, 1993) have described the SIMOPT (SIMulation and OPTimization) model to optimally position a product or product-line using conjoint data. The SIMOPT model can perform the following types of analysis.

1. A competitive analysis helps managers with competitive strategy by providing market shares and sales returns for each competing brand,

2. A sensitivity analysis reveals market share or sales revenue changes when a single level of an attribute in a product is modified,

3. An optimal attribute level analysis determines the best profile for a given firm by assuming specific attribute levels associated with each competitor product,

4. A cannibalization analysis determines the optimal profile that maximizes a given objective function such as market share or sales return, when all existing products of the firm have been considered. The amount of cannibalization of existing products due to the new product is output,

5. A Pareto frontier analysis identifies all the profiles that are undominated with respect to sales return and market share. The user can choose one or more of the undominated profiles for introduction into the market.

Green and Krieger (1993) have also described a case-study type of analysis using the SIMOPT software. The case study includes details of the study design, the research
questions, sensitivity analysis, optimization, Pareto frontier analysis, line extensions and competitor retaliation.

Krieger et al. (2004) mention three classes of models used by choice simulators, which are reproduced below.

1. Simulators that are designed to implement if-then tactics on the part of a single supplier with competitors,

2. Optimizers that solve for the highest market share or return conditioned on assumed or known competitor strategies, and

3. Dynamic optimizers that examine longer term returns that reflect several rounds of competitive interplay using the concept of Nash Equilibrium.

An extension of the SIMOPT model called SIMDYN (SIMulation via DYNamic modeling) is described in Krieger et al. (2004). Krieger et al. (2004) state that, “The SIMDYN model uses game theoretic modeling in which a group of competitors are allowed to sequentially choose product profiles, based on strategies adopted by earlier competitive moves”. The model uses the concept of Nash equilibrium. Nash equilibrium is defined as a market equilibrium condition in which firms cannot make further improvements for themselves by moving away from the Nash equilibrium. SIMDYN considers a sequence of competitive moves and countermoves dynamically to determine those sets of attributes and levels for a given product that are robust to competitor retaliations. They describe another model called the SIMACT model, which has the ability to handle interaction terms besides main effects.

Bretton-Clark has a commercial software package for CA applications. The names of the individual products are the Conjoint Designer, the Conjoint Analyzer,
Simgraf, Linmap and Bridger. The statistical analysis software products such as SAS and SPSS consist of modules that aid in conjoint analysis. Sawtooth Software, Inc. is a market leader in the development of software for collecting and processing conjoint data and in the design of choice simulations. The company provides tutorials and seminars to assist conjoint analysts in developing expertise in applying conjoint methodology. It maintains an active website sawtoothsoftware.com that has links to research papers on conjoint methodology. The Advanced Simulation Module (ASM) is a commercial product from Sawtooth Software Inc. The ACA 5.0 technical paper lists the following functions that are performed by ASM.

1. Simulate respondent preferences for new or modified products and output the share-of-choice for each product that is tested.
2. Explore “what if” scenarios, such as changes in pricing, product formulation or marketing activities.
3. Can be used to efficiently find optimal products, based on the objective criterion of utility, share, revenue or profit.

The “ASM for product optimization” technical paper provides the following examples of the capabilities of ASM.

1. New product introduction without competition,
2. New product introduction with competition,
3. Searching for multiple products simultaneously,
4. Searches using attribute-based prices,
5. TURF (total unduplicated reach and frequency) like problems
6. Maximize appeal subject to a maximum cost, and
1. Minimize cost subject to meeting a performance threshold.

1.9 Optimizing the product design

Market simulators can quickly evaluate a product profile and determine how good it is with respect to the firm’s objective criterion. Marketers often want to know the best product profile from the set of all possible product combinations. The ASM software has sophisticated algorithms that search for products that optimize market share or profitability, refer the “ASM for product optimization” technical paper for details. The ASM optimizer takes into consideration the characteristics of products currently available in the market and the choices of a heterogeneous population of buyers. The ASM optimizer is reported to use response surface methods to solve problems with objective criteria such as maximizing market share or profits.

The focus of this research is to device an exact algorithm that can solve real world instances of the share-of-choice problem to provable optimality. The share-of-choice problem falls under a class of problems that are classified as NP-hard, as proved in Kohli and Krishnamurti (1989). Hence, solving real world instances of the share-of-choice problem to provable optimality within a reasonable amount of time is found to be difficult.

We provide below an outline for the focus of each of the chapters that follow. Chapter 2 will present a survey of conjoint analysis applications in the industry. Chapter 3 will discuss various choice rules used by conjoint simulators, a product positioning model, and the various heuristics that are described in the literature to solve the share-of-choice problem. Chapter 4 will present an exact algorithm that is newly proposed to solve the share-of-choice problem to provable optimality. Chapter 5 will present many
product line design models that are described in the literature. Chapter 6 will discuss potential extensions of the exact algorithm to solve variants of the product design problem. It will also discuss uses of the exact algorithm in solving the product line design problem. Chapter 7 will discuss a simulation design to generate a set of test problems that mimic real world problem characteristics. An experimental design to identify factors influencing solution times will also be presented. Chapter 8 will present the computational results and statistical analysis of the results. Chapter 9 will present conclusions and suggest potential for future work.
Chapter 2.0: Conjoint Analysis Applications

Thousands of conjoint analysis (CA) studies are conducted in the market each year, Orme (2005). This chapter presents a summary of CA applications that are described in the literature. Statistics pertaining to CA applications in the industry are given in section 2.1 and section 2.2 provides the details of selected industry applications of CA. Green and Krieger (1993) and Green et al. (2001) give a list of conjoint applications in the industry that are shown in Tables 2.1 and 2.2.

Table 2.1 and Table 2.2 inserted here

2.1 Statistics of conjoint analysis applications in the industry

Wittink and Cattin (1989) report the following results of a survey conducted on the commercial use of CA.

1. Consumer goods account for 59% of conjoint studies, industrial goods 18%, while financial government, transportation and other goods account for the remaining 23%.

2. New product or concept evaluation, competitive analysis, price and market segmentation represent the main types of applications.

3. Personal interviewing is the most popular data gathering procedure, although computer interactive methods are gaining momentum.

4. Typical sample sizes are 300-400 respondents and typical product profiles are composed of approximately eight attributes, each attribute consisting of a mean number of 3 levels.
5. Full profile designs were popular for stimulus construction in 61% of conjoint studies. Rating and rank order response scales accounted for 85% of conjoint studies. OLS estimation procedure accounted for 54% of conjoint studies.

6. The conjoint studies were validated by a comparison of the predicted market shares to actual market shares. The predicted market shares depended on the choice rule, sampling method and the completeness of the concept description.

7. Computer interactive data collection methods were becoming popular largely because respondent interests in computer interactive tasks were high.

Wittink et al. (1992B) have surveyed the incidence of CA applications by European market research vendors and report the following.

1. The sampling frame consisted of 553 agencies and the survey instrument was sent to each of them. About 29% of the agencies are said to have responded.

2. A majority of commercial applications involved consumer goods, followed by industrial goods.

3. Pricing was the most popular reason given for project purpose.

4. Data is collected through a personal interview. The ACA method is most popular followed by the full-profile conjoint method.

5. Stimulus was presented verbally in most of the applications and rating scales represented the dominant measurement approach. Preference intensity was the dominant variable used to predict marketplace choice behavior and OLS was the dominant estimation procedure.

6. The sample size varied from 30 to 1000, with an average of 268 respondents.
7. With regard to data aggregation, 39% of respondents indicated a higher frequency for the use of pre-specified segments than for clusters based on respondent similarity in judgments or in parameter estimates. The respondent preference for the cluster-based methods accounted for 24%.

8. Predictive validity was done using holdout judgments in 9% of conjoint studies.

9. About 56% of respondents indicated that market share analysis was conducted.

2.2 Selected conjoint analysis applications

We describe a selected set of CA applications below, ordered by the date in which it appeared in a research paper. Zufryden (1983) describes a CA application to evaluate and design a new course offering at the University of Southern California. The new course offering was the “Market Demand and Sales Forecasting” course, targeted at the MBA students. A sample of 15 students participated in the conjoint study. A fractional factorial design was employed. Five attributes that contain a total of thirteen levels within them were identified. The attributes were the technical course content, student career orientation, class format, level of computer application and level of mathematical sophistication. The results of the CA study indicated that level of computer application and class format attributes were considered to be most valuable by the students. The predictive validity of the model was tested using holdout concepts and reported to be good.

Goldberg et al. (1984) discuss an application using categorical CA to study price premiums for hotel amenities. In the authors’ opinion, a key issue facing analysts is how to handle the price attribute while developing stimulus profiles for evaluation by respondents. The problem is that most of the times, a given price level is significantly
correlated with other product attribute levels. There is also the issue of pricing a bundle of attribute levels as in the case of hotel amenities. Product bundling refers to the idea of giving the customer an option to purchase a subset of attributes (product bundle) at a price that is lower than buying each of the attributes in the subset at their individually specified price. That is, the sum of the individual prices of attributes in the product bundle is higher than the price of the product bundle. The research interest was to find out if one could predict a respondent’s evaluation of a bundle of hotel amenity-price combinations as a simple linear function of the summed partworths of the entities making up the bundle. The authors address these issues in their model and apply it to data on lodging preferences collected by a large hotel chain. The problem consisted of 180 respondents and 43 attributes containing a total of 100 levels within them. In spite of the complexities involved in the model, the results of the study indicated that the model had good predictive validity.

Page and Rosenbaum (1987) describe how the Sunbeam Appliance Company redesigned its food processor product line using CA. The authors describe details of the study, which include the design attributes, alternate product designs, market segments, and competitive positions. A market simulation was conducted to predict market shares of alternative product line configurations before deciding on a product line to introduce in the market. Twelve attributes were identified, which contained a total of 31 attribute levels. A set of 500 respondents evaluated 27 profiles by using a main effects orthogonal array design. The total interview time was about 40 minutes per respondent. The appendix section of the paper provided details of the CA procedure. Wittink (1989) offered suggestions for improving the validity and usefulness of the product redesign

Wind et al. (1989) describe one of the largest commercial conjoint studies in terms of the number of attributes and levels. The study involved the design of a new hotel concept called “Courtyard by Marriott”. The management objectives were that the new hotel 1) should offer consumer’s good value for money, 2) should not cannibalize Marriott’s other hotel offerings, and 3) should establish a market positioning that offered a competitive advantage. The number of attributes considered was 50, which were classified under seven facets such as building size, landscaping, room size and function, food, lounge features, service features, leisure options and security. In total, 167 attribute-levels were considered and 601 respondents were interviewed. A hybrid conjoint design was used. Price was not treated as a separate attribute. Instead, a room-price per night was listed along with each profile description of a potential hotel design. Market simulations were conducted using the conjoint data and predicted the following. 1) The market share for a given hotel concept, 2) Cannibalization of existing Marriott hotels, 3) Sources of business (the other hotel chains), and 4) Characteristics of the market segment attracted to a given hotel concept. The results of the study suggested the positioning of a “special hotel at a very comfortable price”. The most effective validation of the study was the success of the “Marriott by Courtyard” chain of hotels.

Green and Krieger (1992) discuss the application of a product-positioning model using the CA technique to a pharmaceutical product. The application is described in case-study format and the software product SIMOPT was used for the study. The product was a new dietary food supplement for administration to hospital patients. The
product is taken orally and used to provide nutrition in cases where disorders of the mouth, esophagus, or stomach preclude the ingestion of solid foods. Nine attributes containing a total of thirty-two attribute-levels were developed using information gathered from focus group interviews with physicians, dietitians, marketing staff and the research staff. The management was interested in designing of an optimal product, maximizing financial return and in introducing additional products. A total of 356 respondents participated in the study and they were all subjected to a mail-telephone-mail interview. A hybrid conjoint approach was used to collect and analyze the data. Certain background characteristics of the respondents were collected for segmentation purposes. The SIMOPT software provided several analyses (that were listed in the previous chapter) to help management in coming up with a valuable competitive strategy.

Graf et al. (1993) describe a conjoint study to evaluate and design health care unit programs. The main objective for a health care unit was to satisfy patient and physician needs better than the competition. The study was conducted by management to measure the simultaneous impact of hospital unit attributes on patient and physician satisfaction with the hospital. The management also desired to identify a service configuration that maximizes return on investment. The conjoint study was undertaken to help identify attribute importances (including the price attribute), to segment the patient (respondent) population, and to estimate market shares that are used for capacity planning. A full profile conjoint study was used. Conjoint data was collected from about 100 respondents (patients) divided into a dozen focus groups. Conjoint data was collected in a similar fashion from a small set of physicians. Some of the attributes in the study include facilities for newborn care, room switching, hospital location, hospital size, visitor policy,
availability of obstetrical specialists and neonatology affiliation with other hospitals. The results of the study indicated that room switching and hospital size were the most important attributes. Based on the results, a service configuration for the health care unit was chosen that maximized return on investment.

Green and Krieger (1993) discuss an application of CA to a large-scale study of consumer evaluations of airline services. Twenty-five attributes such as in-flight services, décor of cabins, flight schedule, routing of aircraft and price were considered in the study. The partworths were developed for a specific route (origin and destination cities) and on a purpose-of-trip basis. The partworths were used to simulate market share for the study’s sponsor on a specific route assuming retaliatory actions of the various competitors. The results of the study were not discussed.

Zufryden and Dreze (1998) describe a CA method that uses the World Wide Web to design optimal new products. The idea is to collect electronic data from visitors to a web site, and then use the data to redesign the web site such that the firm’s objectives are met. This method was applied to a firm that promotes and markets music CD’s online. The firm wanted to redesign the web site so that during a visit by a potential customer, it maximized both the time spent and the number of pages accessed by the visitor. The assumption was that the amount of time spent at the web site and the number of pages accessed at the web site were directly related to the amount of exposure to promotional material, which in turn could influence the customer’s purchase behavior. A total of 788 visitor observations were collected and used in the CA. A non-linear integer-programming model was solved using response surface methodology to identify the optimal web site design. The results of the study were not discussed.
Vavra et al. (1999) describe a conjoint study on the EZPass system for electronic toll collection (ETC). Some background information on the EZPass system is given below, as quoted by the authors. “In the year 1992, a task force was formed among executives of seven regional transportation agencies in the New York/New Jersey area. The mission of the task force was to investigate the feasibility and desirability of adopting ETC for the interregional roadways of the area. The ETC consisted of providing commuters with small transceivers (tags) that emit a tuned radio signal. Receivers placed at toll booths would receive the radio signal and identify the commuter associated with the particular signal. Commuters establish ETC accounts that are debited for each use of a toll-based roadway or facility, thus eliminating the need for the commuter to pay by cash or token. Because the radio signal can be read from a car in motion, ETC can reduce traffic jams at toll plazas by allowing tag holders to pass through at moderate speeds”.

The chief objective of the CA study was to design EZPass based on users of the toll-ways. A secondary objective was to assess the commuter demand for EZPass. A videotape demonstration of the EZPass system was developed to educate respondents about its features. Seven attributes, which contained a total of 26 attribute levels, were identified. A fractional factorial design was used to obtain preferences data. Respondents were recruited for the task by contacting them over the telephone and screened for suitability. Then, a survey kit was mailed to them that consisted of a self-administered questionnaire. A total of 6500 commuters were selected through quota sampling, of which only 3369 respondents successfully completed the survey (a 52% response rate). The results of the study indicated that the attributes “price of toll with
EZPass” and “number of lanes” were regarded as the most important. Market simulations predicted a 49% adoption rate for the EZPass system and actual usage rate (after the EZPass was implemented) was found to be 44%, therebyvalidating the conjoint study.

Krieger et al. (2004) discuss a CA application used for the Baltimore Ravens football team. The team management wished to develop a new logo for the team in the 2002-03 season. It was also desired to arrive at a monetary value that consumers were willing to pay for each logo. A CA study was used to test the appeal of various logos that are used on T-shirts, caps, sweatshirts or blazers, when considered along with a bunch of attributes such as logo color, quality of T-shirt, style of T-shirt and price of the T-shirt. The results of the study were not discussed.

Krieger et al. (2004) and Green et al. (2003) discuss a CA application to a service product called TrafficPulse. TrafficPulse is a new service, which provides subscribers a continuous 24/7 update on traffic conditions, travel times, and alternate routes in case traffic congestions occur. Being a new service, prospective consumers had to be educated about the service before their evaluations of its potential worth could be obtained. The following technical information about the service is quoted. “TrafficPulse employs digital technology and its network is built with partnership from the U.S. Department of Transportation. It delivers wireless data to the U.S. National Transportation Data Center. The resulting traffic information is then distributed over the Internet, radio, TV broadcasts and wireless access. The data is delivered to automobiles via digitally based, on-board telemetric devices”. The study objectives of Mobility Technologies, Inc., the developers of TrafficPulse were to investigate the market
potential for the new service product, and learn which price and product features were most attractive to a subscriber. Respondents from the city of Pittsburgh were chosen for market testing their reaction to the new service concept. A total of 401 respondents were interviewed and the average interview completion time was 30 minutes. Respondents were given a $25 cash incentive for participating in the study. Using the partworths data, market potential and attribute importances were estimated. Also, the returns were forecasted for various pricing strategies.

Luo et al. (2005) discuss a CA application to the design of a new consumer durable product that is robust to variations in both product performance and consumer preferences. After the partworths are estimated, a multi-objective genetic algorithm is proposed to identify better product designs. The authors summarize the result of their work by stating that “Most new product development work is carried out in cross-functional teams, and it is necessary to have coordination processes that are efficient and effective to harness the power of such teams”.

This is a relatively short chapter and naturally the summary given below is also brief. We started by listing several industry applications of CA. The statistics pertaining to CA usage in the industry were then presented. Finally, a selected set of CA applications was described in some detail. The descriptions of the CA applications are not meant to be exhaustive. Interested readers are directed to the reference papers (sources) for fuller details. In the next chapter, we discuss algorithmic approaches to product design.
Chapter 3.0: Algorithmic approaches for product design

Two popular approaches for product design are available to marketers. The first approach is the multi-dimensional scaling (MDS) based models for product positioning. The second approach is the conjoint analysis (CA) based optimal product design models. Product positioning refers to the positioning of a product in a perceptual attribute space so that it closely matches the consumer perception of the various product attributes, Thakur et al (2000). The perceptual attributes are generally composed from two or more basic attributes. For example, the length of a car is a composite (perceptual) attribute that is composed of basic attributes such as seating comfort, driving safety, ease of parking and fuel economy, Gavish et al (1983). On the other hand, product design involves selecting the levels of all the basic attributes that go into making a product by using utilities obtained from a CA study. Both the approaches assume that the choice for a given product is related to buyers’ preferences for the various attribute levels that comprise it.

Since the CA method for product design is already discussed in the first chapter, we now discuss the MDS method. MDS methods are proposed in the literature to optimally position products, refer Green and Krieger (1989) for a set of reference articles on the subject. Green, Carroll and Goldberg (1981) discuss many problems in using the MDS based approach to optimally position products in a perceptual attribute space. The problems are that MDS models require excessive computation time to solve, the solution algorithms cannot guarantee a global optimal solution, and users have difficulty in obtaining input data for the perceived attributes. Our primary interest in this research is in CA based optimal product design. Therefore, we describe only one MDS based
approach in detail. An MDS based product positioning model proposed by Gavish et al (1983) is described in section 3.2.

Green and Krieger (1989) raise the following questions for managers to consider when dealing with optimal product design.

1. Is the firm introducing a new product in a mature market with competitor products?
2. Is the firm introducing a single product or a line of products in the market?
3. If a product-line is planned, will the products be introduced sequentially or simultaneously?
4. What objective criteria (market share, net profit, return on investment, buyer welfare etc.,) are to be optimized?
5. Should competitor reactions to new product introduction be modeled?
6. Should potential buyers be weighted differently in the objective function to reflect their buying frequency and/or buying volume?
7. How should market segments be assigned to different products and how should products be assigned to different market segments?

To answer to above questions, managers seek information such as the type of product, the target market segment, the set of competitor products, and the consumer demographic characteristics. The various models proposed in the literature and described in this chapter have attempted to address these questions.

The chapter is organized as follows. In section 3.1, we discuss consumer choice rules that are popularly used in market simulations. The sub-sections 3.1.1, 3.1.2, 3.1.3 and 3.1.4 describe the share of utility rule, the logit choice rule, the first choice rule and
the randomized first choice rule respectively. In section 3.2, we describe a product-positioning model that uses the MDS technique. In section 3.3, we define the share-of-choice problem and then provide a detailed account of five different heuristics that have been proposed in the literature to solve it. More specifically, sub-sections 3.3.1, 3.3.2, 3.3.3, 3.3.4 and 3.3.5 describe a divide and conquer heuristic, a dynamic programming heuristic, a shortest path heuristic, a genetic algorithm heuristic and a nested partitions heuristic respectively. In section 3.4, we present the product optimization capability of the Advanced Simulator Module from Sawtooth Software Inc. We conclude the chapter with a summary.

3.1 Choice rules for market simulators

Many choice rules are available to translate consumer utilities into product choices. McCullough (2002) and the “ASM for product optimization” technical paper list five types of choice rules used in market simulations, which are the (1) first choice, (2) share of preference, (3) share of preference with correction for similarity, (4) purchase probability, and (5) randomized first choice. The purchase probability rule, also known as the purchase likelihood rule is used to simulate product choice for a new product entry in a market with no competitors. Purchase likelihood estimates give an indication to the direction of buyer preference, but their absolute values should not be assumed to mean anything significant. We will be describing the rest of the four choice rules in this section.

3.1.1 Share of Utility/ BTL Choice Rule

Bradley, Terry and Luce proposed the BTL choice rule, also popularly known as the share of utility rule. The BTL rule is described in Easton and Pullman (2001) as follows.
To estimate choice probabilities and market for a proposed new product, individual utility estimates are aggregated. According to the BTL model, the probability that a consumer selects a product profile X from a set of N competing product profiles is equal to X’s share of the total utility provided by all the N product profiles. The market share for product X is computed by adding the choice probabilities for product X over all the respondents in the study and dividing by the total number of respondents in the study. The assumption is that the higher the utility of a product to a consumer, the greater the probability that the consumer will choose it over competitor’s products, Lilien and Rangaswamy (1998). The BTL choice rule is mathematically represented using the alpha choice model, as given in Easton and Pullman (2001). The probability that consumer i chooses product ‘k’ from N competing products is given as:

$$P_i(X_k) = \frac{U_i^\alpha(X_k)}{\sum_{b=1}^{N} U_i^\alpha(X_b)} \quad \text{where} \quad 1 \leq \alpha \leq \infty, \quad U_i^\alpha(X_k) \geq 0, \quad \text{and} \quad \sum_{b=1}^{N} U_i^\alpha(X_b) > 0, \quad k \in N$$

Equation (3.1) represents the BTL choice rule when $\alpha = 1$. The parameter $U_i^\alpha(X_k)$ represents the utility of consumer i for product k.

A consumer is expected to choose the product for which he/she associates the highest probability, provided it is available. Suppose that the consumer cannot find the product that he/she is most interested in. In such a case, the consumer is expected to seek the product with the next highest probability, and so on. Hence, the BTL choice rule is said to be suitable for low-involvement, frequently purchased products, Lilien and Rangaswamy (1998).
3.1.2 Logit Choice/Share of Preference Rule

The assumption in this choice rule is that the utility values are mean realizations of a random process. So, the brand with maximum utility varies randomly from one purchase situation to another, Lilien and Rangaswamy (1998). The consumer is assumed to choose the product with which he/she associates the highest probability. The market share for a product X is computed by adding the choice probabilities over all the respondents in the study and dividing by the total number of respondents in the study. The logit choice rule is stated in the following mathematical form. Consider the following notation.

I - index set of consumers,
J - index set of products,

\( u_{ij} \) - The utility value of consumer i for product j (can take positive/negative value),

\( e^{u_{ij}} \) - The antilog of \( u_{ij} \) to the ‘base e’,

\( p_{ij} \) - The probability that consumer i will choose product j, and

\( m_j \) - The market share for product j.

The probability is given as

\[ p_{ij} = \frac{e^{u_{ij}}}{\sum_{j \in J} e^{u_{ij}}} \]

The market share is given as

\[ m_j = \frac{\sum_{i \in I} p_{ij}}{|I|} \]

We find that in the logit choice rule, the product utilities are exponentiated as shown above so that they result in positive numbers. The market share results obtained from the logit rule can be fine-tuned so that the maximum and minimum values of shares can be adjusted as desired. If all the partworths are multiplied by a large constant, the logit rule approaches the first choice rule. If all the partworths are multiplied by a small constant, the predicted shares will all become nearly equal. As mentioned before, the
share of utility and logit choice rules, both suffer from the property called independence from irrelevant alternatives (IIA). The share of preference rule with correction for similarity corrects for product similarity by decreasing estimated shares of products in proportion to their similarity to other products. This helps in reducing the IIA effect.

3.1.3 Maximum Utility/Share of First Choices Rule

The share of first choices model is defined below based on the description in Green, Carroll and Goldberg (1981) and Easton and Pullman (2001). The share of first choices rule or the maximum utility rule states that a consumer selects product profile X from a set of N competing product profiles if product profile X provides him/her the maximum utility over all the N competing product profiles. The market share for product profile X is the percentage of respondents for whom product profile X provides the maximum utility. The probability that consumer i will choose product profile ‘k’ from N competing products can be obtained from equation (3.1) when $\alpha$ is large, say $\alpha = 1000$. Under this choice rule, the consumer is expected to choose a single product that maximizes his total utility, and not consider any other product that may be available. Hence, this choice rule is used to evaluate high involvement purchases such as cars, Lilien and Rangaswamy (1998). Also, the optimal profile under this choice rule is very sensitive to changes made to the profile, so that minor changes can result in drastic changes to predicted market share.

The first choice model is not a realistic model of consumer behavior for the reasons given below.

2. Buyers are not known to purchase a product with a 100% probability. Rather, they are known to seek variety. Also, there are out-of-stock situations for a product.

3. The shares of popular products are exaggerated and the shares of unpopular products are underestimated.

4. First choice rule can produce volatile share changes and demand functions.

5. It lacks the random factors that characterize actual purchase decisions and hence the share estimates from first choice simulations generate extreme probabilities, too close to zero or one. For a comparison between the logit and first choice rules, figure 3.1 displays ‘choice probability’ versus ‘product utility’ graphs for each of the two rules.

In spite of the above reasons, Orme (2005) feels that the first choice rule can be tuned to be more realistic after multiplying the product utilities by a scale factor.

3.1.4 Randomized First Choice Rule

In the randomized first choice (RFC) rule, the utility of a product alternative is a function of its utility plus a random error term added to the partworths plus a random error term added to the product alternative, Orme (2005). The random error term added to the partworths is unique for each attribute and respondent but is held constant across alternatives. The random error term is unique for each alternative and respondent. The appropriate error variance for partworths and product alternatives is found by fine-tuning the simulated shares to holdout choice shares or known market shares. If the error
variance is large, the simulated share is expected to be less extreme. As in the first choice rule, a respondent is assumed to choose the alternative with the highest utility. For a history of the randomized first choice rule, refer to Orme and Huber (2000).

The RFC rule can be expressed using the following relation, Orme (2005):

\[ U_i = X_i (\beta_i + g_a) + g_p. \]

That is, the utility of alternative \( i \) (\( U_i \)) is a function of the utility of that alternative plus random error of the part-worths (\( X_i (\beta_i + g_a) \)), plus random error of the product alternative (\( g_p \)). The terms \( X_i \) refer to the dummy-coded design matrix, and \( \beta_i \) to the partworth vector.

The “ASM for product optimization” technical paper gives the following explanation to describe how the random error terms are computed and choice shares are estimated. “The choices for each respondent are simulated many times using sampling iterations. At each iteration, the partworths are perturbed randomly. The perturbations are reported to be of two types. First, the partworths are perturbed by adding an error term for each attribute. The modified partworths are used to sum utilities for each product. Then the utility sums are further perturbed by adding an error term for the product. For each iteration, the respondent is allocated to the product with the highest perturbed utility. The results for all the iterations are averaged to produce the final estimate of choice share for each respondent”.

Some of the advantages of using the RFC method are given below.

1. The RFC method handles the issue of respondents whose choices are variable and not constant.

2. The IIA problem is reduced.
3. RFC simulations can be tuned so that the estimates of shares for popular and unpopular products are prevented from taking extreme values. That is, the magnitudes of perturbations of partworths and utilities can be adjusted. This way, the simulated shares will be closer to actual market shares. For example, as the magnitude of perturbations is increased, predicted shares of products become more similar to one another.

4. The RFC model adds variability to both the partworths and to the overall profile values in such a way that optimally matches simulators against experimental choice shares. It is reported by Orme and Huber (2000) that adding variability to the choice model used in the simulator improves holdout share predictions.

5. RFC lets you tune the amount of correction for product similarity between an IIA model and one that splits share in half for duplicate offerings.

A limitation of RFC rule is that the best use of RFC results from properly tuned “attribute” and “product” type errors, which tasks are time consuming. The “ASM for product optimization” technical paper and Orme (2005) list other limitations of the RFC rule. In spite of their limitations, the RFC and logit choice rules are regarded to be realistic models for consumer behavior.

3.2 A product-positioning model

Gavish, Horsky and Srikanth (1983) propose a model to optimally position a new product in an existing product class. The assumptions made in the model are given below.

1. The consumer and the firm are both expected to follow a two-stage decision process when choosing a product and offering a product respectively.
2. The consumer first decides his/her budget for buying from a product class. Then, the consumer identifies the set of products from the product class that meet his/her budget constraint, evaluates them with the help of a weighted multi-attribute utility model and chooses the product with the highest utility.

3. The consumer has an ideal level for each attribute under consideration and will choose a product from the set of competing products that is closest to his/her ideal point. Consumers may have different attribute weights and ideal levels.

4. The firm identifies potential product positions that are attractive to consumers in the attribute space consisting of ideal points and competing products. Then, the firm computes the profits resulting from each position and offers the best product in the market.

A mixed integer nonlinear programming (MINP) formulation is proposed to solve the firm’s problem of identifying an optimal new product position. The objective is to identify a point in the multi-dimensional attribute space that is closer than the existing products in the product class to the ideal point of as many consumers (in the study) as possible. The firm obtains the following input data necessary to solve the MINP problem. A sample of the population of consumers who intend to buy or have bought a product in the product class is chosen for the study. The authors recommend a sample size of the order of 300-500, to adequately represent market heterogeneity. The consumer ideal points and attribute weights are estimated using a multi-attribute utility model, and the location of existing products in the product class is determined based on their attribute features. We now define the notation used in the MINP problem formulation.
Sets:

M – The set of consumers in the study,

A – The set of attributes,

N – The set of competing products in the product class,

Parameters:

$y_{mk}$ - The ideal point on attribute k for consumer $m$, $m \in M$ and $k \in A$,

$w_{mk}$ - The weight given to attribute k by consumer $m$, $m \in M$ and $k \in A$,

$x_{nk}$ - The coordinate of product n on attribute k, $n \in N$ and $k \in A$,

H – A large positive number,

S – The acceptable region for positioning the new product $Z$,

$D_m(Z)$ - The weighted Euclidean distance of the ideal point of consumer $m$ from a product located at position $Z = (z_1, z_2, ..., z_{|A|})$; $D_m(Z) = \sqrt{\sum_{k \in A} w_{mk} (z_k - y_{mk})^2}$, $m \in M$, 

$R_m^2$ - The product closest to the ideal point of consumer $m$, computed using the formula

$$R_m^2 = \min_{n \in N} \left\{ \sum_{k \in A} w_{mk} (x_{nk} - y_{mk})^2 \right\}, m \in M,$$

Decision Variables:

$\delta_m$ - A binary variable that takes a value of 0 if the chosen product is not closer to the ideal point of consumer $m$ than all of the competing products, and a value of 1 if the chosen product is closer to the ideal point of consumer $m$ than each of the competing products,

$Z$ – The vector of coordinates of the product to be determined; $Z = (z_1, z_2, ..., z_{|A|})$.

The MINP model formulation is given below.
Max $\sum_{m \in M} \delta_m$ \hspace{1cm} (3.2)

Subject to:

\[
\left\{ \sum_{k \in A} w_{mk} (z_k - y_{mk})^2 \right\} - R_m^2 \leq (1 - \delta_m) M, \quad m \in M
\] \hspace{1cm} (3.3)

$\delta_m = \{0, 1\}, \ m \in M$ \hspace{1cm} (3.4)

$Z \in S$ \hspace{1cm} (3.5)

The above model maximizes the number of consumers for whom product Z is closer to their ideal point than any of the competing products in the product class. The objective function can be weighted to reflect consumer m’s purchase frequency and/or purchase volume. The authors report that solving the MINP problem to optimality is not feasible for realistic problem instances. In order to reduce the problem size, the authors show that the feasible region containing the optimal solution to the MINP problem has to satisfy several properties, leading to a reduced search space. A point search heuristic and a line search heuristic that each searches for the optimal solution in the reduced search space is illustrated with an example. The computational performance of the two heuristic procedures is compared on a set of common test problems. A simulated positioning example of inexpensive compact cars is also given with a detailed explanation.

The authors conclude their work with the following comments.

1. The positioning of the new product should be such that it is highly differentiated from the positioning of the competing products.

2. A major drawback of the model is that price is not explicitly built in.

3. The quality of the solutions identified by the point search and line search heuristics are not known since the optimal solutions are not known.
The Share-of-Choice Problem

According to the share-of-choice rule, an optimal product is defined to be the one that maximizes the number (share) of consumers (respondents) who prefer the said product to their status-quo products, Kohli and Krishnamurti (1989). The share-of-choice problem definition follows from the share-of-choice rule. That is, the share-of-choice problem is to find the product that maximizes the number of respondents who prefer the said product to their status-quo products. Consumers are expected to choose a given product if it has a higher utility than the utility of their status-quo product. The market share for the new product is computed as the ratio of the number of respondents who prefer it to their status-quo products, over the total number of respondents in the study. The share-of-choice problem is proven to be NP-hard, Kohli and Krishnamurti (1989). Five heuristic approaches that appear in the product design literature to solve the share-of-choice problem are explained in the following sub-sections.

3.3.1 Divide-and-Conquer Heuristic

Green and Krieger (1993) propose a divide-and-conquer (DC) heuristic to solve the share-of-choice problem as part of a simulation and optimization (SIMOPT) model for product design. Consider a new product with k attributes 1, 2, 3, ..., k. Each attribute comprises of two or more levels. The objective function value is assumed to be a linear combination of one level from each attribute and is written as $f(l_1^*, l_2^*, l_3^*, ..., l_k^*)$. It is possible to find the levels $l_1^*, l_2^*, l_3^*, ..., l_k^*$ that maximize share-of-choice using brute force enumeration of all possible solutions. In that case, the number of solutions to enumerate would be $\prod_{p=1}^{k} N_p$, where $N_p$ refers to the number of levels within attribute p. When
there are a large number of attributes and levels, there will be too many solutions to enumerate and hence the brute force enumeration method cannot solve the share-of-choice problem within a reasonable amount of time.

The DC heuristic works by dividing the set of attributes into many sub-sets. At a given iteration, the share-of-choice problem is solved sequentially over each sub-set by using a brute-force enumeration method. If the solution to the overall share-of-choice problem has improved in the current iteration over that found in the previous iteration, we update the best solution and repeat the process of optimizing over each sub-set. The DC heuristic stops when no further improvement in the solution can be found between two successive iterations. The pseudo-code for the DC heuristic is given after definition of notation.

\( \kappa \) - The current iteration number,

\( l^0, l^0_2, l^0_3, \ldots, l^0_k \) - A starting solution to the share-of-choice problem,

\( f(l_1, l_2, l_3, \ldots, l_k) \) - The objective function of the share-of-choice problem,

\( (1, \ldots, N_i), (N_1 + 1, \ldots, N_2), \ldots (N_{\eta-1} + 1, \ldots, k) \) - The \( \eta \) sub-sets of the attribute set,

Initialization Step: Let \( \kappa = 1 \). The incumbent objective function value is \( f(l^0_1, l^0_2, l^0_3, \ldots, l^0_k) \).

Step 1 – Solve a series of \( \eta \) share-of-choice problems using the brute force enumeration method. First, optimize over sub-set 1 and find \( (l^1_1, \ldots, l^1_N) \) that maximizes \( f(l_1, \ldots, l_N, l^{k-1}_{N_i+1}, \ldots, l^{k-1}_k) \). Next, optimize over sub-set 2 and find \( (l^1_{N_1+1}, \ldots, l^1_{N_2}) \) that maximizes \( f(l^1_1, \ldots, l^1_{N_i}, l^{k-1}_{N_i+1}, \ldots, l^{k-1}_k) \). Continue this solution process until all \( \eta \) sub-sets are optimized. Then, go to step 2.
Step 2 – If $f(l_1^e, ..., l_k^e) > f(l_1^{e-1}, ..., l_k^{e-1})$ then update the best incumbent solution value, increment $\kappa = \kappa + 1$ and go to step 1. Else, terminate the DC heuristic.

The authors make the following observations about the DC heuristic.

- The DC heuristic provides a local optimum solution and the chances of the local optima being the global optima decrease as the number of sub-sets is increased.

- The DC heuristic performs very well if the conjoint data within each attribute is independent and identically distributed. But, if there are many interactions among attributes it is likely that a sub-optimal solution is obtained. It is suggested that the sub-sets be formed in such a way that correlation of partworths across sub-sets of attributes is minimized. The SIMOPT software provides the user an option to create sub-sets using many different methods that help to alleviate problems due to interactions among attributes.

No study on the computational performance of the DC heuristic is given.

3.3.2 Dynamic Programming Heuristic

The dynamic programming (DP) heuristic discussed in this sub-section and the shortest path (SP) heuristic discussed in sub-section 3.5.3 are based on an integer programming formulation of the share-of-choice problem. The description of DP heuristic is based on Kohli and Krishnamurti (1987), and the description of SP heuristic and the integer programming formulation is based on Kohli and Krishnamurti (1989). The integer programming formulation of the share-of-choice problem is given after defining the following notation.

*Sets:*

$E$ – The set of edges in the graph,
N – The set of nodes (a dummy starting node, a dummy ending node and a node for each attribute-level) in the graph = \{1,2,...,|N|\},

R - The set of respondents in the study = \{1,2,...,r\},

A - The set of attributes = \{1,2,...,k\},

L_{k} - The set of all the attribute-levels in attribute k = \{1,2,...,S_{k}\}, where k \in A,

**Parameters:**

δ - A small positive number,

\(a_{ve}\) - The node-arc incidence matrix, where v \in N and e \in E; \(a_{ve} = 1\) if arc \(f_{e}\) departs from node v, \(a_{ve} = -1\) if arc \(f_{e}\) terminates into node v, and \(a_{ve} = 0\) if arc \(f_{e}\) does not meet node v,

\(\omega_{ijk}\) - The partworth of consumer i for level j from attribute k, where i \in R, j \in L_{k}, and k \in A; and \(\omega_{ij^{'}}k\) - The partworth of consumer i for level j’ from attribute k of his/her status-quo product, where i \in R, j’ \in L_{k}, and k \in A,

\(C_{ijk}\) - The partworth of level j from attribute k relative to the partworth of level j* from attribute k for individual i, given as \(C_{ijk} = \omega_{ijk} - \omega_{ij^{'}}k\),

\(C_{ie}\) - The relative partworth corresponding to the node into which arc e terminates; the \(C_{ie}\) values are normalized so that \(-1 < \sum_{e \in E} C_{ie}f_{e} < 1\),

**Decision Variables:**

\(f_{e}\) - A binary variable that equals 1 if edge e is chosen, 0 otherwise, where e \in E,

\(x_{i}\) - A binary variable that equals 0 if \(\sum_{e \in E} C_{ie}f_{e} \geq \delta\), 1 otherwise, where i \in R,
Min \( \sum_{i \in R} x_i \) \hspace{1cm} (3.6)

Subject to:

\[
\left( \sum_{e \in E} c_{ie} f_e \right) + x_i \geq \delta, \ i \in R
\]

(3.7)

\[
\sum_{e \in E} a_{ve} f_e = 1, \ v = 1
\]

(3.8)

\[
\sum_{e \in E} a_{ve} f_e = -1, \ v = |N|
\]

(3.9)

\[
\sum_{e \in E} a_{ve} f_e = 0, \ v \in N : v \neq 1 \text{ and } v \neq |N|
\]

(3.10)

\[
x_i = \{0,1\}, \ i \in R
\]

(3.11)

\[
f_e = \{0,1\}, \ e \in E
\]

(3.12)

The above model formulation can be represented as a graph. From a dummy source node, forward arcs connect to \( S_1 \) nodes each associated with a level of attribute 1.

From each of these \( S_1 \) nodes, forward arcs connect to \( S_2 \) nodes each associated with a level of attribute 2. This process is continued until forward arcs from each of \( S_{k-1} \) nodes connect to \( S_k \) nodes each associated with a level of the attribute \( k \) (last element in the set \( A \)). Finally, forward arcs from each of \( S_k \) nodes connect to a dummy destination node.

The objective function (3.6) minimizes the number of respondents from the set \( R \) for whom \( \sum_{e \in E} c_{ie} f_e < \delta \). Constraint set (3.7) forces \( x_i \) to take a value of 1 if \( \sum_{e \in E} c_{ie} f_e < \delta \), for a given respondent \( i \). Constraint (3.8) ensures that only one arc is chosen from those originating from the dummy source node and constraint (3.9) ensures that only one arc is chosen from those terminating into the dummy destination node. The constraint set (3.10) ensures that flow-in equals flow-out for each node that corresponds to an attribute-level (not the dummy source or dummy destination node). That is, if an arc flows into a node, then another arc has to flow out of the same node. Also, if no arc flows into a
node, then no arc can flow out of the node. Constraint sets (3.11) and (3.12) restrict the variables $x_i$ and $f_e$ to take only binary integer values 0 or 1. Solving the integer program (3.6 - 3.12) to optimality using the traditional branch and bound method is difficult even for moderately sized problems since the problem is known to be NP-hard. Hence, a DP heuristic is proposed to solve the share-of-choice problem.

The DP heuristic treats attributes as stages and attribute-levels as states. Forward arcs connect each state in a given attribute (stage) to all states in the succeeding attribute (stage). Each state corresponds to a distinct column from the $C_{ijk}$ matrix and each stage consists of a distinct set of columns from the $C_{ijk}$ matrix. The heuristic is implemented using forward recursion and works in the following manner. At first, each state (column) in stage 2 is combined with each state (column) in stage 1 by vector addition. For each state in stage 2 the best combination with a state from stage 1 is determined by identifying the combined column with the highest number of positive elements. We now have a set of aggregated columns equal in number to the number of levels in stage 2. Each of these aggregated columns represent a partial product profile. Next, we identify for each state in attribute 3, the best partial profile (identified in the previous stage) with which to combine. This process is continued until all attributes have been considered. In the end we choose the best among the complete product profiles that have been identified. The DP heuristic is described below in the form of a pseudo-code after the following definition of notation.

$\Phi$ - The null set,

$C(k)$ – The set of columns belonging to attribute $k$, where $k \in A$,

$C_{jk}(k)$ - The column corresponding to attribute-level $j$ of attribute $k$, where $j \in L_k$, 

80
$S^*(k)$ - A set of distinct aggregated-columns (an aggregated-column is generated by adding one column from each of the sets $C(1) \ldots C(k)$) retained at stage $k$.

$S_j(k)$ - A set of aggregated-columns generated by adding column $C_j(k)$ to each of the aggregated-columns in the set $S^*(k-1)$, where $k > 1$.

**Step 1** – Let $S_j(k) = \emptyset, j \in L_k, k \in A$ and $S^*(k) = \emptyset, k \in A$. Let $k = 1$. Initialize $S^*(k) = C(k)$. That is, the set of columns in $C(l)$ are added to the set $S^*(l)$. Increment $k = k + 1$. Go to step 2.

**Step 2** – For all $j \in L_k$, and $m(k-1) \in S^*(k-1)$, add the aggregated column $m(k-1) + C_j(k)$ to the set $S_j(k)$. Go to step 3.

**Step 3** – For all $j \in L_k$, identify the aggregated-column from the set $S_j(k)$ that has the largest number of positive elements and add it to the set $S^*(k)$. Go to step 4.

**Step 4** – Increment $k = k + 1$. If $k > |A|$ then go to step 5. Else, go to step 2.

**Step 5** – Decrement $k = k - 1$. Identify the aggregated-column, say $p^*$, in the set $S^*(k)$ with the largest number of positive elements. The product profile represented by $p^*$ is the best solution.

The DP heuristic can be adapted to reflect the following changes in the share-of-choice problem.

(1) Respondents can be assigned weights based on their purchase frequency and/or purchase volumes. The best column chosen during step 3 of the DP heuristic is modified as the aggregated column with the largest sum of weights corresponding to elements in the aggregated column with positive values.
(2) Based on many factors such as consumer preference, technological feasibility, manufacturing cost etc., all the infeasible product profiles could be identified. Suppose there are N infeasible product profiles that are known in advance. In order to ensure that at least one feasible product profile appears in the final solution, the following modification is made to the DP heuristic.

Let $N_{jk}$ be the number of infeasible product profiles each of which consists of attribute-level $j$ from attribute $k$. Let $N_k = \max_{j \in L_k} (N_{jk})$, $k \in A$ and let $N_{\min} = \min_{k \in A} (N_k)$. Let $k^*$ be an attribute such that $N_{k^*} = N_{\min}$, so that each level of attribute $k^*$ appears in a maximum of $N_{\min}$ infeasible product profiles. To ensure at least one feasible solution, the DP heuristic is modified in two ways.

(i) The attribute $k^*$ is the last stage in the DP heuristic, and

(ii) At any stage $k$, $(N_{\min} + 1)$ aggregated-columns are retained in the set $S^*(k)$ if $\sum_{j \in L_k} |S_j(k)| > N_{\min}$.

Or else, all the aggregated-columns of $S_j(k)$, $j \in L_k$ are included in the set $S^*(k)$.

To test the DP heuristic, forty-eight test problems were generated using a 3x4x4 experimental design that employed (4, 6, 8) attributes, (2, 3, 4, 5) levels within attributes, and (100, 200, 300, 400) respondents as the design factors. These forty-eight problems were replicated four times and hence a total of one hundred and ninety two problems formed part of this study. The authors quote that “all partworths were randomly generated from a uniform distribution between [0,1] and then normalized within
individuals”. A comparison is made between the performance of the DP heuristic and a Lagrangian-relaxation heuristic. The DP heuristic is shown to be superior to the Lagrangian-relaxation heuristic in terms of solution times and the solution quality. A description of the Lagrangian-relaxation heuristic was not available.

Finally, the authors discuss two shortcomings of the DP heuristic with the help of a trivial example.

(1) The quality of the solution obtained from the DP heuristic is very sensitive to how the attributes are arranged. To obtain a better solution, it is suggested that the DP heuristic be run with several different orderings of the attributes.

(2) The quality of the solution obtained from the DP heuristic is very sensitive to the number of segments to which respondents in the conjoint study belong. It is suggested that the DP heuristic be run at the segment level and not at the market level.

A third shortcoming is to do with the computational performance of the DP heuristic.

(3) Large problem instances are likely to consist of many attributes and many levels within attributes. In such problems, suppose there are $N$ infeasible product profiles, where $N$ is large. In such a case, $N_{\text{min}}$ is also likely to be a large number. This will require the heuristic algorithm to hold a large amount of data in memory. Hence, the (modified) DP heuristic implemented as a computer program may fail due to inadequate memory resource.

3.3.3 Shortest Path Heuristic

In the SP heuristic, the $C_{ijk}$ values are aggregated across all the respondents in the study to simplify the share-of-choice problem and transform it to a shortest-path type of
problem. The network structure of the shortest-path type of problem is similar to that of
the DP heuristic. The attributes are stages and the attribute-levels are states. Forward
arcs connect each state in the first stage from a dummy source node ‘s’. Forward arcs
flow from each state in the last stage to a dummy destination node ‘d’. Forward arcs
connect each state in a given stage to all states in the succeeding stage. The objective of
the SP heuristic is to select a path from the dummy source node to the dummy destination
node along which the sum of arc-weights along the path is the largest. The simplified
problem is easy to solve as a dynamic program since the “principle of optimality” holds.
That is, the optimal solution to a series of sub-problems (finding an optimal path between
one stage and the next) form an optimal solution to the overall problem of finding an
optimal path from source node ‘s’ to destination node ‘d’. We define the following
notation.

\( \Phi \) - The null set,

\( C_{jk} \) - The sum across respondents of the relative partworths of level j of attribute k
(referred to as arc-weight); \( C_{jk} = \sum_{i \in \mathbb{R}} C_{ijk} \),

\( C(k) \) – The set of arcs belonging to attribute k, where \( k \in \mathbb{A} \),

\( C_j(k) \) - The arc corresponding to attribute-level j of attribute k, where \( j \in \mathbb{L}_k \),

\( S^*(k) \) - A set of partial-paths (a path from source node ‘s’ ending in a node from attribute
k) retained at stage k,

\( S_j(k) \) - A set of partial-paths generated after the arc \( C_j(k) \) is appended to each partial-
path in the set \( S^*(k-1) \), where \( k > 1 \).

The SP heuristic is described below in the form of a pseudo-code.
Step 1 – Let $S_j(k) = \Phi, j \in L_k, k \in A$ and $S^*(k) = \Phi, k \in A$. Let $k = 1$. Initialize $S^*(k) = C(k)$. That is, the set of arcs in $C(1)$ are added to $S^*(1)$. Increment $k = k + 1$.

Go to step 2.

Step 2 – For all $j \in L_k$ and $m(k-1) \in S^*(k-1)$, add the partial path $m(k-1) + C_j(k)$ to the set $S_j(k)$. Go to step 3.

Step 3 – For all $j \in L_k$, identify the partial-path from the set $S_j(k)$ with the largest value of the sum of its arc-weights, and add that partial-path to the set $S^*(k)$. Go to step 4.

Step 4 – Increment $k = k + 1$. If $k > |A|$ then go to step 5. Else, go to step 2.

Step 5 – Decrement $k = k - 1$. Identify the path, say $p^*$, in the set $S^*(k)$ with the largest value of the sum of its arc-weights. The product profile represented by $p^*$ is the best solution.

Suppose there are $N$ infeasible product profiles that are known in advance. Compute $N_{min}$ and identify attribute $k^*$ as in the DP heuristic. To ensure at least one feasible solution, the SP heuristic is modified in two ways.

(i) The attribute $k^*$ is the last stage in the SP heuristic, and

(ii) At any stage $k$, $(N_{min} + 1)$ partial-paths are retained in the set $S^*(k)$ if

$$\sum_{j \in L_k} |S_j(k)| > N_{min}.$$  Or else, all the partial paths in $S_j(k), j \in L_k$ are included in the set $S^*(k)$.

Sixty-four test problems were generated using a 4x4x4 experimental design that employed $(100, 200, 300, 400)$ respondents, $(4, 6, 8, 10)$ attributes, and $(2, 3, 4, 5)$ levels within attributes as the design factors. The partworths were randomly generated exactly
as in the DP heuristic. A comparison is made between the performance of the SP heuristic and the DP heuristic on the simulated test problems. The DP heuristic is shown to perform better than the SP heuristic in terms of solution times and solution quality. The SP heuristic also suffers from shortcomings (2) and (3) of the DP heuristic. But, the quality of the solution obtained using the SP heuristic does not depend on how the attributes are arranged (ordered).

3.3.4 Genetic Algorithm Heuristic

Balakrishnan and Jacob (1996) propose a genetic algorithm (GA) heuristic to solve the share-of-choice problem. The GA approach is popular with operations researchers, and is known to mimic the approach of natural selection in the evolution of species. The GA approach is applied to the product design problem using the following logic. A chromosome (product) is composed of genes (attributes) each of which can take a number of values (attribute levels) called alleles. Starting from an initial population of product profiles, the GA heuristic uses the genetic operators of reproduction, crossover and mutation to create better populations until some pre-specified termination criteria is reached. It is assumed that the partworths of consumers are obtained using conjoint or hybrid CA. The GA heuristic is explained below in further detail.

Initialization Phase: An initial population of chromosomes (product profiles) is randomly generated. The size of the population M is set at 100 (and maintained at 100 in each generation of the GA heuristic). Three genetic operators are used to generate a new set of products that are expected to be fitter than the previous generation of products. The fitness of a product is measured by computing the number of consumers in the study who
prefer the said product to their status quo products. The three genetic operators are
defined below.

Reproduction Phase: This phase involves picking a subset of products $m \ (m < M)$ from
the initial pool of candidate products. Based on their “fitness”, copies of their profiles are
generated. Half the fittest product profiles from the population survive into the next
generation so that the better candidate profiles are preserved and have an opportunity to
produce offspring.

Crossover Phase: In this phase, pairs of reproduced product profiles are chosen (with
replacement) for mating. Along specific positions on the product profile genetic material
are exchanged between the chosen pairs resulting in pairs of new product profiles called
as the offspring. The exchange of attributes is implemented using a uniform crossover
rule with probability value of 0.5. That is, the attributes that are exchanged are uniformly
distributed throughout the product profile rather than having contiguous segments of
attributes exchanged.

Mutation Phase: In this phase, a product profile is randomly chosen from the population
without replacement and its value at a specific location (attribute level) in the product
profile is modified. The mutation rate is set at 0.30. The number of product profiles
keeps increasing due to reproduction and crossover. Once the number of product profiles
equals twice the initial number (2M), the best half of these product profiles are retained
based on their “fitness” and the rest are deleted.

Stopping Rule: The GA heuristic iterates through the above three phases until termination
criteria are reached. The stopping rule used is the moving average rule that provides an
indication of convergence to a solution. The heuristic is terminated if the moving
average over the five previous generations is less than 2% of the fitness of the three best product profiles in the current generation.

The GA heuristic is shown to have bad performance in the worst-case. The data used to analyze the GA heuristic are the 192 problems used by Kohli and Krishnamurti (1987) for testing the DP heuristic. The GA heuristic is compared with several versions of the DP heuristic using these simulated problems for the share-of-choice problem and the buyer’s welfare problem. The buyer’s welfare problem can be easily solved in linear time to produce an optimal solution, as explained in Shi, Olafsson and Chen (2001). So, we focus on the results for the share-of-choice problem. Based on solution quality, hit rate (in finding the optimal solution) and standard deviation of error, the performance of the GA heuristic is reported to be superior to that of the DP heuristic.

A sensitivity analysis of the GA performance to changes in parameter values was conducted. A 5x4x3x2 full factorial experiment was designed to study the impact of different mutation rates (0, 0.01, 0.1, 0.25, 0.3), fraction of attributes participating in crossover (0, ¼, ½, ¾), chromosome pool population sizes (50, 100, 200) and stopping rules’ degree of improvement (2%, 0.2%) on GA performance with regards to solution quality and solution times. Hence, 120 runs of the GA heuristic was conducted for one replicate of a data set with 400 consumers and 8 attributes each with 5 levels, for the share-of-choice problem and the buyer’s welfare problem. We only concern ourselves with results for the share-of-choice problem. The results from the sensitivity analysis were subjected to a series of ANOVA tests. The main effects model had a $R^2$ value of .504 that was statistically significant with $p < 0.05$. Therefore, the parameters significantly affected the quality of the GA solution. Except for the mutation parameter,
all the other factors (parameters) impacted the quality of the GA solution. Another series of ANOVA tests were run to study the impact of the parameters on the GA solution times. The main effects model had a $R^2$ value of 0.798 that was statistically significant with $p < 0.05$. Again, except for the mutation parameter, all the other factors (parameters) impacted the quality of the GA solution.

Separate tests were conducted to study the impact of varying attributes sizes (number of levels) and different sequencing of attributes on the performance of the GA heuristic. The performance of the GA heuristic was compared to the performance of the DP heuristic and the former is again shown to perform better. The authors claim that interaction between attributes, and technological infeasibility of product profiles can both be incorporated into the GA heuristic. They also suggest using a decision support system (DSS) that triangulates on a solution, see Balakrishnan and Jacob (1995). In other words, the DSS provides maximally different solution methods so that the user is confident that the best solution obtained is close to optimal. Balakrishnan and Jacob (1995) present GENESYS, a DSS that provides three solution strategies for the product design problem. They are complete enumeration, the DP heuristic and the GA heuristic.

3.3.5 Nested Partitions Heuristic

Shi and Olafsson (2000) proposed the Nested Partitions (NP) method to solve global optimization problems. Subsequently, Shi, Olafsson and Chen (2001) used the NP method to solve the share-of-choice problem. The stopping criterion of the NP method (adopted by the authors) is a fixed amount of CPU run-time, when used to solve a set of test problems. Since the optimality of the solution obtained could not be verified, the NP
method will be referred to as the NP heuristic from now on. The notation used to
describe the model formulation is given below.

**Sets:**

R - The set of respondents in the study = \{1,2,...,r\},

A - The set of attributes = \{1,2,...,k\},

L_k - The set of all the attribute-levels in attribute k, where \( k \in A \),

**Parameters:**

\( u_{ijk} \) - The utility associated with consumer i and level j of attribute k,

\( q_i \) - The total utility of the status quo product used by consumer i,

\( TU_i(\theta) \) - The total utility that consumer i obtains from product \( \theta \); where

\[
TU_i(\theta) = \sum_{k\in A} \sum_{j\in L_k} u_{ijk} \theta_{jk},
\]

\( f_i(\theta) \) - The performance function for consumer i; where

\[
f_i(\theta) = \frac{\max(0, TU_i(\theta) - q_i)}{TU_i(\theta) - q_i}
\]

when \( TU_i(\theta) \neq q_i \) and \( f_i(\theta) = 0 \) when \( TU_i(\theta) = q_i \),

**Decision Variables:**

\( \theta_{jk} \) - A binary variable that is equal to 1 if attribute k is set to level j and 0 otherwise,

\( \theta \) - A complete product profile described using the \( \theta_{jk} \) variables, \( k \in A \) and \( j \in L_k \),

The share-of-choice problem is formulated as follows.

**Max** \( \sum_{i \in R} f_i(\theta) \) \hspace{1cm} (3.13)

Subject to:

\[
\sum_{j \in L_k} \theta_{jk} = 1, k \in A \hspace{1cm} (3.14)
\]

\( \theta_{jk} = \{0,1\}, j \in L_k, k \in A \hspace{1cm} (3.15) \)
The model (3.13) – (3.15) representing the share-of-choice problem is known to be NP-hard. The NP heuristic is proposed to solve the above model. The NP methodology is implemented as a search tree and constitutes four main steps of partitioning, random sampling, identifying a promising region and backtracking. The main idea of the NP method is to focus the search for a good solution in those regions of the feasible region that are more likely to have them. At the start of the NP heuristic, the feasible region is partitioned into disjoint sub-regions. Each level of the first attribute in the branching order is chosen (branched on) to generate a sub-region. The number of sub-regions that can be formed is equal to the number of levels within the attribute. Next, a promising region is chosen from among the sub-regions. The promising region is determined (for each sub-region) through a random sampling of the rest of the attributes that are not branched on yet. For each such attribute, a level is randomly chosen based on a probability distribution. The resulting complete product profile (also referred to as a sample profile) is evaluated in terms of the objective function (3.13). In solving the test problems, the number of sample profiles generated in each sub-region was fixed at 20. The best objective function value from amongst the sample profiles is denoted as the promising index and is computed for each sub-region. The promising region is the one with the highest value of the promising index. If the promising index is higher in value than the best incumbent solution, the latter is updated to the higher value. In the next iteration, the NP heuristic focuses on the promising region and partitions the promising region into sub-regions by fixing the levels of the next attribute in the branching order. The remaining part of the feasible region that is outside of the promising region is referred to as the surrounding region. The promising region is determined as described
earlier. Suppose the promising region happens to be the surrounding region. In this case, the algorithm backtracks to the surrounding region using the following backtracking rule. The depth of the search tree at the new region is to be fixed at half the depth of the current node in the search tree. The NP heuristic continues in this manner from iteration to iteration until a pre-specified termination criterion is reached. While solving the test problems, the termination criterion was a fixed amount of CPU run-time on the computer.

We note that the NP heuristic takes a global solution approach in the partitioning, random sampling and backtracking steps and a local solution approach in determining the promising region. The quality of the solution obtained using the NP heuristic is dependent on how best the promising region is determined. A chief advantage of using the NP heuristic is that any number of heuristic procedures can be used to determine the promising region. In fact, the authors use a greedy approach, the DP heuristic approach, the GA heuristic approach and a combination of two or all of these approaches in the NP heuristic to speed its convergence to a better solution. Incorporating known heuristic procedures in the sampling step of the NP heuristic are explained in detail by the authors who also trace the various steps in the NP heuristic algorithm from start to finish through a trivial example.

The NP heuristic and its variants (which are embellished with various combinations of local search heuristics) are tested on a set of seven problems that are randomly generated. The parameters in these problems vary as follows. The numbers of attributes are 5, 6, 7, 8, 9, 10 and 20 for the seven problems. The number of levels within an attribute is set equal to the number of attributes in a problem instance. For example, a problem with 5 attributes has 5 levels within each of them. No explanation is given for
designing the number of levels in this manner. Each of the seven problems has 400 consumers. The partworths are generated from a uniform distribution as in Kohli and Krishnamurti (1989). All the problems were run for a fixed amount of CPU run-time. For some unknown reason, the authors do not disclose what the fixed amount of CPU run-time is! The test results show that the NP heuristic is better than the DP heuristic in terms of solution quality. But, the GA heuristic performs better than the NP heuristic in terms of solution quality. The NP heuristic performs better when strengthened with the other local search heuristics and best when combined with the GA and greedy search heuristics (NP/GA/GS). The seven problems were run for ten replications each. The standard deviation of the solution quality is shown to be lowest for the NP/GA/GS heuristic.

New sets of three problems with 5, 10 and 20 attributes respectively are generated. The number of levels within an attribute equals the number of attributes in the problem instance (as before), and the number of consumers stays fixed at 400. The partworths are generated from a beta distribution with parameters $\beta_1 = \beta_2 = 2$. The NP heuristic and its variants are run on these test problems. The test results again show that the NP heuristic strengthened with GA and greedy heuristics (NP/GA/GS) performs best in terms of solution quality. Based on this result, the authors conclude that the partworths preferences of consumers do not affect the performance of the NP/GA/GS heuristic in a significant way.

The NP method is shown to converge provably to the global optimal solution with probability one in finite time, Shi and Olafsson (2000). Yet, the NP method cannot prove optimality of a given solution, no matter how long the run time on a computer (except for
the special case when a solution has 100% coverage). Also, the efficiency of the NP method in quickly converging to an optimal solution depends on the partitioning strategy, the estimation procedure that identifies the most promising index and the backtracking strategy, among other possible factors.

The authors compare and contrast the NP method with the popular branch and bound (B&B) method. Both methods partition the feasible region by systematically branching on a variable (sub-region) and creating a search tree in the process. This is where the similarity ends. The B&B method is an exact method that establishes a bound on the best performance of a sub-region (by evaluating an objective function) and based on the bound determines whether to branch further or backtrack to another sub-region. Hence, upon natural termination of the B&B method, a provably optimal solution is obtained. On the other hand, the NP method does not pursue all branches in a search tree. It only branches on promising regions. The NP method does not establish bounds to exclude sub-regions from the search procedure. Hence as reported earlier, it is not possible to prove optimality of a solution obtained from using the NP method. The advantages of the NP method over the B&B method are dependent on problem type and problem size. Suppose that for a given problem instance, the bound for a sub-region cannot be established or the quality of the bound is poor. In such a case, the B&B method would be inefficient since most of the search tree has to be explored to find a provably optimal solution.

3.4 The Advanced Simulator Module for Product Optimization

We mentioned earlier that Sawtooth Software, Inc. is a market leader in the development of conjoint software and that the Advanced Simulation Module (ASM) is their
commercial product. The technical paper “ASM for product optimization” describes the optimization capability of ASM, which is reviewed below.

The ASM optimizer has in-built search algorithms to find products that optimize an objective criterion, such as market share or profitability. The ASM suite of algorithms consists of five optimization methods that are listed below.

1. Exhaustive search method – The algorithm searches exhaustively by examining every combination of permitted levels of all attributes. For large problem instances, this method is known to be highly inefficient.

2. Grid search method – In this method, a starting solution is chosen. Then, an operation is performed in which attributes are selected in random order. All permitted levels of each attribute are examined as per the order, holding other attributes constant, and the best level is retained for each attribute. The operation is conducted over several iterations. The iteration process continues until a particular iteration fails to find a better solution.

3. Gradient search method - This method works by finding a combination of attributes to change simultaneously, using a “steepest ascent” method to find the top of a peak in the response surface.

4. Stochastic search - In this method, an attribute is chosen in a random fashion and its level is also changed in a random fashion. If the change results in an improvement, the change is accepted, or otherwise it is rejected. This step is repeated from iteration to iteration. Iterations stop when there is no improvement between successive iterations.
5. Genetic search – The Genetic search is adapted from the GA heuristic described in section 3.3.4.

It is important to note that except for the exhaustive search method, the rest of the four methods do not search exhaustively and hence cannot prove optimality of the solutions obtained by them.

We summarize the main sections of this chapter. Two approaches to optimal product design, called the MDS method and the CA method respectively, are initially discussed. Then, we explained the share of utility, the logit choice, the first choice and the randomized first choice rules. Next, we discussed an MDS based optimal product positioning model. We then defined the share-of-choice problem for optimal product design and discussed five different heuristics that were proposed in the product design literature to solve it. Finally, the optimization capabilities of the ASM software are described. In the next chapter, an exact algorithm is proposed to solve the share-of-choice problem to provable optimality using a B&B procedure.
Chapter 4.0: An exact algorithm for the share-of-choice problem

In this chapter, an exact algorithm to solve the share-of-choice problem to provable optimality is proposed. The chapter is divided into the following sections. Section 4.1 presents a binary integer programming formulation of the share-of-choice problem. Section 4.2 discusses the difficulties involved in solving the share-of-choice problem using a traditional linear programming (LP) based branch-and-bound method and the relative advantages of using our exact algorithm. Section 4.3 provides a detailed account of the implicit enumeration scheme used to traverse the search tree.

Section 4.4 describes Phase I of the exact algorithm. This section is divided into the following five sub-sections. Sub-section 4.4.1 lists three pre-processing steps performed on the input data. Sub-section 4.4.2 presents a greedy algorithm used to solve the share-of-choice problem. The objective function value of the greedy solution serves as a starting (incumbent) solution. Sub-section 4.4.3 defines a Lagrangian relaxation of the share-of-choice problem and then describes a subgradient optimization procedure to solve it. The objective function value of the Lagrangian-relaxed share-of-choice problem serves as a valid upper bound on the optimal objective function value of the share-of-choice problem. The potential for identifying an improved incumbent solution to the share-of-choice problem during the subgradient optimization procedure is also discussed. Sub-section 4.4.4 describes the branching rule for the exact algorithm. Sub-section 4.4.5 discusses the potential for checking column dominance, which could reduce the problem size and hence the run-time of the algorithm.

Section 4.5 consists of three sub-sections. Sub-sections 4.5.1 and 4.5.2 explain the details of each of two logic-based rules that are used to fathom a node in the search
tree, through the help of a trivial example. Sub-section 4.5.3 presents a heuristic algorithm that is used to generate data for testing the second logic-based rule. Section 4.6 describes Phase II of the exact algorithm, wherein we state the branch-and-bound procedure that is used to solve the share-of-choice problem to optimality. We conclude the chapter with a summary.

4.1 Formulation of the share-of-choice problem

The share-of-choice problem is formulated as a binary integer program, as suggested by Cochran (1997). The formulation is given below with a definition of the notation used.

Max \[ \sum_{i \in R} y_i \] (4.1)

Subject to:

\[ \sum_{j \in L} U_j x_j - H_i y_i \geq \delta, \ i \in R \] (4.2)

\[ \sum_{j \in L_k} x_j = 1, \ k \in A \] (4.3)

\[ x_j = (0,1), \ j \in L \] (4.4)

\[ y_i = (0,1), \ i \in R \] (4.5)

Sets:

R - The set of respondents = \{1,2,3,...,p\}

A - The set of attributes = \{1,2,3,...,q\}

L - The set of all the attribute-levels = \{1,2,3,...,r\}

Φ - The null set

L_k - The set of levels belonging to attribute k, k \in A

L_k \subset L, k \in A

L_k \cap L_j = \Phi, k \in A, j \in A, j \neq k
\[ L = \bigcup_{k \in A} L_k \]

**Parameters:**

- \( U_{ij} \): The partworth of respondent \( i \) for level \( j \), where \( i \in R \) and \( j \in L \)
- \( H_i \): The hurdle value for respondent \( i \), where \( i \in R \)
- \( \delta \): A small positive number fixed at a value of 0.0001

**Decision Variables:**

- \( x_j = 1 \) if level \( j \) is chosen, 0 if not, where \( j \in L \)
- \( y_i = 1 \) if the hurdle level of respondent \( i \) is exceeded, 0 if not, where \( i \in R \)

The objective function (4.1) maximizes the number of respondents for whom the total utility of a new product design is greater than their associated hurdle value. Constraint set (4.2) ensures that a given respondent is counted in the objective function only if the total utility of a new product design is higher than the hurdle value associated with that respondent. Constraint (4.3) ensures that only one level is chosen from each attribute set, so that the new product design is feasible. The \( x \) and \( y \) variables are constrained to take only binary values by the constraint sets (4.4) and (4.5).

We eliminate the constant \( \delta \) appearing in constraint set (4.2) by making the following changes. Let \( H_i^1 = H_i + \delta, i \in R \). Constraint set (4.2) can now be re-written as \( \sum_{j \in L} U_{ij} x_j - H_i^1 y_i \geq 0, i \in R \) (4.2a). Each constraint in the revised constraint set (4.2a) also ensures that the total utility of a new product design for a given respondent \( i \) exceeds his or her hurdle value. In the next section, we discuss the demerits of an LP-based branch-and-bound approach to solve the share-of-choice problem to optimality.
4.2 LP-based or logic-based branch-and-bound method?

We now consider nature of the input data to the share-of-choice problem. The constraint (utility) matrix is made up of elements (partworths) that have real values. The density of the constraint matrix is close to 100%. Density of a matrix is the ratio of the total number of non-zero elements in the matrix over the total number of elements in the matrix, expressed as a percentage. After the pre-processing steps (see section 4.4) are applied to the input data, the density of the constraint matrix is reduced. But even in the best case, when all attribute sets have exactly two levels (refer to the second pre-processing step in section 4.4), the density of the constraint matrix will be about 50%. Also, the right-hand side (hurdle) vector is made up of elements that have real values. In general, each respondent’s hurdle value is computed as the sum total of partworths of attribute levels comprising his or her status-quo product. Hence, the hurdle values are much larger than the values of individual elements (partworths) in the constraint matrix.

The share-of-choice problem does not have a special network structure that would yield a naturally integer solution when solving its LP relaxation. The share-of-choice problem is NP-Hard, Kohli and Krishnamurti (1989). Therefore, the worst-case behavior of an exact algorithm for solving the share-of-choice problem is expected to be exponential in the size of the problem input. Based on preliminary runs of simple test cases of the share-of-choice problem, the LP-relaxation solution is found to be highly fractional in the binary variables $x_j$ and $y_i$, where $j \in L$ and $i \in R$.

From the above discussion, it is expected that the quality of upper bound obtained by solving an LP-relaxation of the share-of-choice problem would be weak and therefore not very effective in pruning the search tree. Generally, the LP-based branch-and-bound
procedure works well if the integrality gap is relatively small. Integrality gap is defined to be equal to $1 - (LB/UB)$, where LB is the best incumbent solution (lower bound for the share-of-choice problem), and UB is the best bound (upper bound to the share-of-choice problem). If the integrality gap is large, the LP-based branch-and-bound procedure is expected to be inefficient.

In real-world share-of-choice problems, the number of $y$ variables is generally much greater than the number of $x$ variables. The traditional branch-and-bound method used to solve the share-of-choice problem, branches on all the binary variables $x$ and $y$. As noted earlier, both the $x$ and $y$ variables have a tendency to take on fractional values between 0 and 1 when a LP relaxation is solved at a node in the branch-and-bound tree. This causes the LP bound to be weak, and hence the solution times (to find a provably optimal integer solution) to be generally unacceptably large.

The exact algorithm proposed in this chapter prunes the search tree using logic-based bounds. It carries out optimization only over the $x$ variables, as previously shown in the implementation of a dual-based algorithm to solve the maximal-covering problem (MCP) by Downs and Camm (1996). The values of the $y$ variables are logically obtained as follows. If the constraint $\sum_{j \in L} U_{ij}x_j - H_i \geq 0$ is satisfied, then fix $y_i = 1$, else fix $y_i = 0$, $i \in R$. By keeping the $y$ variables out of the branching process, the size of the search tree is reduced considerably. At every node, the logic-based bounds are computed and used to decide whether to branch further or fathom.

We believe that a given instance of the share-of-choice problem will solve to optimality faster when using the proposed exact algorithm than when using a traditional branch-and-bound approach (available with commercial solvers like CPLEX). The next
section focuses on the computer implementation of the search tree used by the exact algorithm.

4.3 Implicit Enumeration

The share-of-choice problem is represented as a state space tree (search tree). The number of levels in the search tree (the depth of the search tree) is equal to the number of attributes in the problem. The number of nodes (variables) in each level (attribute) equals the number of possible settings of that attribute. The principle of backtracking is employed to traverse the search tree. During the traversal, branching (forward step) occurs at every level (attribute) of the search tree. In the branching process, the value of one of the free variables belonging to a given level (attribute) is set equal to 1. A backtrack step at a given level is taken by setting the value of the variable that is fixed at value 1 to a value of 0.

To reduce the size of the search space, the mechanism of pruning is used. Pruning of the search tree is achieved with the help of two logic-based rules. The logic-based rules identify the maximum number of additional constraints (respondents) that could be satisfied by further branching, when at a given level in the search tree. Based on the value of the (upper) bound obtained, a decision is made as to whether or not it is beneficial to continue branching on the next level below in the search tree. A review of the implicit enumeration scheme used to traverse the search tree is given below.

The following implicit enumeration scheme is based on the description found in Syslo, et al (1983). The exact algorithm explicitly or implicitly enumerates all the possible solutions to the share-of-choice problem. Most of the solutions are enumerated implicitly, with the help of the two logic-based bounds. The enumerative
procedure is a search tree composed of nodes and branches. As mentioned earlier, the exact algorithm branches only on the \( x \) variables. Each \( x_j : j \in L \) variable can be in one of three states: fixed at 1, fixed at 0, or free (fixed at value –1). A node corresponds to a solution to an attribute set. A new node is defined by fixing a \( x_j : j \in L_k \) variable (belonging to attribute ‘k’) to value 1 (forward step). If none of the \( x_j : j \in L_k \) variables is fixed at value 1, attribute ‘k’ is referred to as a free attribute. While branching on a free attribute ‘k’, exactly one \( x_j : j \in L_k \) variable that is free (at value –1) can be fixed at value 1. If none of the \( x_j : j \in L_k \) variables is free, the algorithm backtracks to a previous level (if one exists) or terminates (if the previous level is the root node). A node is re-visited by setting the \( x_j : j \in L_k \) variable fixed at value 1 to value 0 (backtrack step).

Consider a trivial example of the share-of-choice problem with the sets \( A = \{1,2\} \), \( L = \{1,2,3,4\} \), \( L_1 = \{1,2\} \), \( L_2 = \{3,4\} \), and \( R = \{1,2,3,4,5,6,7,8,9,10\} \). Let the \( x \) vector be initialized to \( x = [-1,-1,-1,-1] \), with entries interpreted as follows:

a) If \( x_k = 1 : k \in L \), it means that \( x_k \) has been fixed at value 1. Its complement \( x_k = 0 \) has not yet been considered.

b) If \( x_k = 0 : k \in L \), it means that \( x_k \) has been fixed at value 0, and the complement \( x_k = 1 \) has already been considered.

c) If \( x_k = -1 : k \in L \), it means that \( x_k \) is free to assume values of 0 or 1.

The traversal of the search tree is explained using the illustration in Figure 4.1.

Figure 4.1 inserted here
The node $L_0$ represents the root node, from where all branching begins. Assume an incumbent solution $Z_{\text{inc}}$ equal to 6. Also, consider a set $\overline{A}$ whose elements are derived from the set $A$, but are ordered according to a branching strategy (explained in sub-section 4.4.4). Let the set $\overline{A}$ instruct us to branch on attribute 1 first followed by attribute 2. Identify the first free attribute from the set $\overline{A}$, i.e., none of the variables belonging to the attribute is fixed at value 1. Attribute 1 (set $L_1$) is the first free attribute to be branched on, and hence variable $x_1$ which is free (at value $-1$) is fixed at value 1. Move to node $L_{11}$ by branching. Let the fathoming test (upper bound obtained using the logic-based rules) at node $L_{11}$ indicate that it is possible to improve $Z_{\text{inc}}$ by further branching. The next free attribute to branch on is attribute 2. The free variable $x_3$ is fixed at value 1. Move to node $L_{21}$ by branching. A feasible solution is now available since all the attributes are branched on. Suppose that the current feasible solution does not improve $Z_{\text{inc}}$. Since there are no free attributes to branch on, backtrack to node $L_{21}'$ by fixing $x_3$ at value 0. Now, attribute 2 is free. Let the fathoming test indicate that it is beneficial to branch on attribute 2. Branch on attribute 2 that is free by fixing variable $x_4$ at value 1. Move to node $L_{22}$. Suppose that the current feasible solution results in an objective function value equal to 8. Update $Z_{\text{inc}}$ to value 8. Backtrack to the node $L_{22}'$ by setting $x_4$ to value 0. Now, there are no free variables in the attribute set $L_2$. So, backtrack to the previous level in the search tree after setting $x_j = -1$, $j \in L_2$. Backtrack to the node $L_{11}'$ by fixing $x_1$ at value 0. Attribute 1 is free to be branched on and
variable $x_2$ is free. Let the fathoming test indicate that it is beneficial to branch on attribute 1. So, fix $x_2$ at value 1 and move to the node $L_{12}$. Suppose the fathoming test at node $L_{12}$ indicates that further branching is not going to improve $Z_{\text{inc}}$. Hence, the partial solution at node $L_{12}$ is fathomed. In other words, all completions of the partial solution have been implicitly enumerated. Backtrack to the node $L_{12}'$. Although attribute 1 is free to be branched on, there are no free variables in the attribute set $L_1$. So, backtrack to the previous level. But, the previous level happens to be the root node. The exact algorithm terminates at this point since it has returned to its starting point after traversing the search tree. The basic enumeration approach can now be summarized in a generic fashion as shown below.

Step 1 - Identify the first free attribute ‘k’, $k \in \bar{A}$. That is, none of the variables belonging to the set $L_k$ is fixed at value 1.

- If a free attribute ‘k’ such that $k \in \bar{A}$ is available, then go to Step 2.

- If a free attribute $k \in \bar{A}$ does not exist, it means that all attributes are branched on. A feasible solution is available at the current node, and if it yields an objective function value greater than the best incumbent ($Z_{\text{inc}}$), update $Z_{\text{inc}}$. Backtrack by setting the x-variable with value 1 in the attribute set $L_{[\bar{A}]}$ (last attribute element in the set $\bar{A}$) to a value of 0. Go to Step 1.

Step 2 - Identify a free variable $x_s$ such that $s \in L_k$.

- If a free variable $x_s$ is found, fix $x_s$ at value 1 and go to Step 3.
- If no free variable is found in $L_k$, backtrack to the previous level (attribute set $L_{k-1}$) of the search tree after setting $x_j = -1$, $j \in L_k$. Set the $x$-variable with value 1 in the attribute set $L_{k-1}$ to a value of 0. Go to Step 1.

Suppose that $k=1$ and no free variable is found in $L_1$. The exact algorithm can be terminated since backtracking to the previous level leads the algorithm back to the root node. That is, the search tree has been explicitly or implicitly enumerated.

Step 3 – Use logic-based rules to obtain the maximum possible coverage (bound) in the remaining free variables, belonging to attributes that have not yet been branched on.

- If the decision is to fathom at the current node, backtrack by setting $x_s = 0$ and continue the branching process by going to Step 1.
- If the decision is not to fathom at the current node, continue the branching process by going to Step 1.

4.4 Phase I of the Exact Algorithm

The different stages in Phase I of the exact algorithm are described in the following sub-sections. There are five sub-sections that are described below sequentially. A flow diagram of Phase I of the exact algorithm is shown in Figure 4.2.

4.4.1 Pre-processing Steps

**Step 1** - The logic-based fathoming rules can be efficiently implemented in the exact algorithm if $U_{ij} \geq 0$ for $i \in R$ and $j \in L$. Hence, the $U$ matrix and the $H^1$ vector are transformed by the following operations. A vector $M$ as defined in (i) below is used to implement step 1.
\[ M_i = \min \{0, \min_{j \in L} (U_{ij})\}, \ i \in R \]  
\[ U_{ij}^i = (U_{ij} - M_i), \ i \in R \text{ and } j \in L \]  
\[ H_i^2 = (H_i^1 - (M_i * |A|)), \ i \in R \]

The constraint set (4.2a) is now rewritten as \( \sum_{j \in L} U_{ij}^i x_j - H_i^2 y_i \geq 0, \ i \in R \) (4.2b).

**Proposition 4a:** An optimal solution to the share-of-choice problem with constraint set (4.2a) is also an optimal solution to the share-of-choice problem with constraint set (4.2b).

**Proof:** An optimal solution to the share-of-choice problem belongs to a finite set of solutions that are feasible to the constraint set (4.3). As proof, we show that for a given feasible solution, if a given constraint ‘i’ from (4.2a) is feasible, then constraint ‘i’ from (4.2b) will also be feasible and vice versa. In other words, we will reduce constraint ‘i’ from (4.2a) to constraint ‘i’ from (4.2b). So, the objective function value will be the same for each feasible solution with either constraint set (4.2a) or constraint set (4.2b). Consider a feasible solution \( \hat{x} \). The constraint (4.2a) for a given respondent \( i \), where \( i \in R \) is \( \sum_{j \in L} U_{ij} \hat{x}_j - H_i y_i \geq 0 \).

Since \( y_i \) is a binary variable that takes a value 1 if \( \sum_{j \in L} U_{ij} \hat{x}_j \geq H_i \) and 0 otherwise, it is not needed in the following analysis. From (ii) and (iii), constraint ‘i’ is rewritten as \( \sum_{j \in L} (U_{ij}^i + M_i) \hat{x}_j \geq (H_i^2 + M_i * |A|) \). In the feasible solution \( \hat{x} \), \( |A| \) elements of the \( \hat{x} \) vector are fixed at value 1 and the rest are fixed at value 0. Therefore, constraint ‘i’ can be rewritten as \( \sum_{j \in L} U_{ij}^i \hat{x}_j + M_i * |A| \geq H_i^2 + M_i * |A| \).
That is, \( \sum_{j \in L} U_{ij}^1 \hat{x}_j - H_i^2 \geq 0 \) which is the same as constraint ‘i’ from (4.2b).

Constraint ‘i’ from (4.2a) is reduced to constraint ‘i’ in (4.2b).

**Step 2** - The logic-based fathoming rules can be efficiently implemented in the exact algorithm if \( \min_{j \in L_k} (U_{ij}^1) = 0 \), \( i \in R \) and \( k \in A \). Hence, the \( U^1 \) matrix and the \( H^2 \) vector are transformed by the following operations. A matrix \( \alpha \) as defined in (iv) below is used to implement step 2.

\[
\alpha_k = \min_{j \in L_k} \left( U_{ij}^1 \right), \quad i \in R \text{ and } k \in A \quad \text{(iv)}
\]

\[
U_{ij}^2 = U_{ij}^1 - \alpha_k, \quad i \in R, \ j \in L_k \text{ and } k \in A \quad \text{(v)}
\]

\[
H_i^3 = H_i^2 - \sum_{k \in A} \alpha_k, \quad i \in R \quad \text{(vi)}
\]

The constraint set (4.2b) is now rewritten as \( \sum_{j \in L} U_{ij}^1 \hat{x}_j - H_i^3 y_i \geq 0 \), \( i \in R \) (4.2c).

**Proposition 4b:** An optimal solution to the share-of-choice problem with constraint set (4.2b) is also an optimal solution to the share-of-choice problem with constraint set (4.2c).

**Proof:** An optimal solution to the share-of-choice problem belongs to a finite set of solutions that are feasible to the constraint set (4.3). As proof, we show that for a given feasible solution, if a given constraint ‘i’ from (4.2b) is feasible, then constraint ‘i’ from (4.2c) will also be feasible and vice versa. In other words, we will reduce constraint ‘i’ from (4.2b) to constraint ‘i’ from (4.2c). So, the objective function value will be the same for each feasible solution with either constraint set (4.2b) or constraint set (4.2c). Consider a feasible solution \( \hat{x} \). The
constraint (4.2b) for a given respondent i, where \( i \in R \) is \( \sum_{j \in L} U_{ij} \hat{x}_j - H_i^3 y_i \geq 0 \).

Since \( y_i \) is a binary variable that takes a value 1 if \( \sum_{j \in L} U_{ij} \hat{x}_j \geq H_i^3 \) and 0 otherwise, it is not needed in the following analysis. From (v) and (vi), constraint ‘i’ is rewritten as

\[
\sum_{k \in A} \sum_{j \in L_k} (U_{ij}^2 + \alpha_{ik}) \hat{x}_j \geq H_i^3 + \sum_{k \in A} \alpha_{ik},
\]

i.e.,

\[
\sum_{k \in A} \sum_{j \in L_k} \alpha_{ik} \hat{x}_j + \sum_{j \in L} U_{ij}^2 \hat{x}_j \geq H_i^3 + \sum_{k \in A} \alpha_{ik}.
\]

In any feasible solution, one element of the \( \hat{x} \) vector belonging to each attribute k is fixed at value 1 and the rest are fixed at value 0. Hence, constraint ‘i’ is rewritten as

\[
\sum_{k \in A} \alpha_{ik} \hat{x}_j + \sum_{j \in L} U_{ij}^2 \hat{x}_j \geq H_i^3 + \sum_{k \in A} \alpha_{ik}.
\]

That is, \( \sum_{j \in L} U_{ij}^2 \hat{x}_j - H_i^3 \geq 0 \) which is the same as constraint ‘i’ from (4.2c).

Constraint ‘i’ from (4.2b) is reduced to constraint ‘i’ in (4.2c).

**Step 3** - The respondents in a share-of-choice problem instance may belong to different market segments and display idiosyncratic preferences for the various attribute-levels. Therefore, the hurdle and utility values associated with the respondents are likely to vary over a range of values. The run-time of the exact algorithm is strongly influenced by the branching order (to be discussed in sub-section 4.4.4) that in turn depends on the numerical values of elements in each column of the utility matrix. It is to be noted that the utility values in rows (respondents) having high hurdle values also tend to be high. In order that the branching order is not unduly influenced by large utility values (corresponding to high hurdle values), normalization of the hurdle values by reducing them all to a common value of 1 is recommended. The utility values in the \( U^3 \) matrix
may also need to be adjusted after the \( H^3 \) vector is normalized. Only the positive elements of the \( H^3 \) vector are normalized, as shown in the following pseudo-code.

\[
\text{if } H_i^3 > 0 \text{ then } h_i = 1.0, \text{ else } h_i = H_i^3, i \in R \quad \text{(vii)}
\]

\[
\text{if } h_i = 1.0 \text{ then } (\mu_{ij} = U_{ij}^2 / H_i^3, j \in L), \text{ else } \{\mu_{ij} = U_{ij}^2, j \in L\}, i \in R \quad \text{(viii)}
\]

The constraint set (4.2c) is now rewritten as \( \sum_{j \in L} \mu_{ij} x_j - h_i y_i \geq 0, i \in R \) (4.2d).

**Proposition 4c:** An optimal solution to the share-of-choice problem with constraint set (4.2c) is also an optimal solution to the share-of-choice problem with constraint set (4.2d).

**Proof:** An optimal solution to the share-of-choice problem belongs to a finite set of solutions that are feasible to the constraint set (4.3). As proof, we show that for a given feasible solution, if a given constraint ‘i’ from (4.2c) is feasible, then constraint ‘i’ from (4.2d) will also be feasible and vice versa. In other words, we will reduce constraint ‘i’ from (4.2c) to constraint ‘i’ from (4.2d). So, the objective function value will be the same for each feasible solution with either constraint set (4.2c) or constraint set (4.2d). Consider a feasible solution \( \hat{x} \). The constraint (4.2c) for a given respondent \( i \), where \( i \in R \) and for whom \( H_i^3 > 0 \), is

\[
\sum_{j \in L} U_{ij}^2 \hat{x}_j - H_i^3 y_i \geq 0. \quad \text{Since } y_i \text{ is a binary variable that takes a value 1 if}
\]

\[
\sum_{j \in L} U_{ij}^2 \hat{x}_j \geq H_i^3 \text{ and 0 otherwise, it is not needed in the following analysis. From (vii) and (viii), constraint ‘i’ is rewritten as } \sum_{j \in L} (\mu_{ij} * H_i^3) \hat{x}_j \geq h_i * H_i^3. \quad \text{The summation in the left-hand side of the constraint is over all } j \in L, \text{ and so } H_i^3 (\text{being a constant}) \text{ can be moved out of the summation. Therefore, constraint}
\]
‘i’ is rewritten as $H_i^3 \sum_{j \in L} \mu_{ij} \hat{x}_j \geq h_i * H_i^3$. Since $H_i^3 > 0$, $\sum_{j \in L} \mu_{ij} \hat{x}_j - h_i \geq 0$ which is the same as constraint ‘i’ from (4.2d). Constraint ‘i’ from (4.2c) is reduced to constraint ‘i’ in (4.2d).

4.4.2 Greedy Algorithm

We state the greedy algorithm that is used to obtain an incumbent solution (lower bound) to the share-of-choice problem. The following notation is used.

Φ - A null set,
Ψ - This set records the indices of attributes that are branched on at any given iteration,
Ω - This set records the indices of all the respondents (rows), whose hurdle values are exceeded at any given iteration,
κ - The iteration number,
b_i – An element of the vector that stores the hurdle value remaining to be covered at any given iteration, where $i \in R$,
$Z_{\text{greedy}}$ - The objective function value of the greedy solution, and
$Z_{\text{inc}}$ - Records the best incumbent solution value.

Step 1: Initialize $\Psi = \Phi$ and $\Omega = \Phi$. Initialize $b_i = h_i$, $i \in R$. Set $\kappa = 1$.

Step 2: Let $j^*$ be such that $\sum_{i \in R, i \in \Omega} \min(\mu_{ij}, b_i) = \max_{j \in L, k \in A, k \notin \Psi} \left\{ \sum_{i \in R, i \in \Omega} \min(\mu_{ij}, b_i) \right\}$.

Step 3: Let $k^*$ be such that $j^* \in L_{k^*}$. Increment $\Psi = \Psi \cup k^*$.

Step 4: For all $i \in R$ and $i \notin \Omega$, set $b_i = b_i - \mu_{ij}$. If $b_i \leq 0$ increment $\Omega = \Omega \cup i$.

Step 5: Increment $\kappa = \kappa + 1$. If $\kappa > |A|$ go to step 6, else go to step 2.

Step 6: Set $Z_{\text{greedy}} = |\Omega|$. Set $Z_{\text{inc}} = Z_{\text{greedy}}$. Terminate the greedy algorithm.
A summary of the iterative steps in the greedy algorithm is given below. In step 1, the sets $\Omega$ and $\Psi$ are initialized to null sets, the value of the $b^j$ vector is initialized to the value of the $h^i$ vector, and the iteration count is set at 1. In step 2, the column $j^*$ (belonging to a free attribute) that makes the maximum contribution to covering the remaining hurdle values of uncovered respondents is identified. In step 3, the attribute set to which column $j^*$ belongs is identified and added to the set $\Psi$. In step 4, the $b^j$ vector is adjusted in value to account for the newly added column $j^*$ to the partial greedy solution. If any of the currently adjusted $b_i$ values is less than or equal to zero, element (respondent) $i$ is added to the set $\Omega$. In step 5 check if all the free attributes have been added to the set $\Psi$. If true, go to step 6, compute the greedy solution value, set the best incumbent solution value to the greedy solution value and terminate the algorithm. If false, go to step 2 and continue with the next iteration.

4.4.3 Lagrangian relaxation of the share-of-choice problem

The formulation of the share-of-choice problem is given below.

\[
\text{Max } \sum_{i \in R} y_i \tag{4.6}
\]

Subject to:

\[
\sum_{j \in L} \mu_j x_j - h_i y_i \geq 0, \ i \in R \tag{4.7}
\]

\[
\sum_{j \in L_i} x_j = 1, \ k \in A \tag{4.8}
\]

\[
x_j = (0,1), \ j \in L \tag{4.9}
\]

\[
y_i = (0,1), \ i \in R \tag{4.10}
\]
Note that changes to parameters have already been made in the pre-processing steps.

Now consider the following definition of a Lagrangian-relaxed share-of-choice problem (the set of constraints in (4.7) is relaxed), which we refer to as LSOC.

Min \( \text{LSOC} \left( \lambda_i, i \in R \right) \) \hspace{1cm} (4.11)

s.t. \( \lambda_i \geq 0, \ i \in R \) \hspace{1cm} (4.12)

The problem \( \text{LSOC} \left( \lambda_i, i \in R \right) \) is stated as follows.

Max \( \sum_{i \in R} y_i + \sum_{i \in R} \lambda_i \left( \left( \sum_{j \in L} \mu_j x_j \right) - h_i y_i \right) \) \hspace{1cm} (4.13)

s.t. \( \sum_{j \in L_k} x_j = 1, \ k \in A \) \hspace{1cm} (4.14)

\( x_j = (0,1), \ j \in L \) \hspace{1cm} (4.15)

\( y_i = (0,1), \ i \in R \) \hspace{1cm} (4.16)

Some additional notation is defined below.

\( \lambda_i \) - The Lagrangian multiplier associated with constraint ‘i’ from constraint set (4.7),

\( Z_{\text{lag}} \) - The objective function value of the LSOC problem, and

\( Z_{\text{lp}} \) - The objective function value of the LP relaxation of the share-of-choice problem.

The maximization objective function of the LSOC problem can be rewritten by re-arranging terms and restating \( \text{LSOC} \left( \lambda_i, i \in R \right) \) as follows.

Max \( \sum_{i \in R} (1 - \lambda_i h_i) y_i + \sum_{j \in L} \left( \sum_{i \in R} \mu_j \lambda_i \right) x_j \) \hspace{1cm} (4.17)

s.t. \( \sum_{j \in L_k} x_j = 1, \ k \in A \) \hspace{1cm} (4.18)

\( x_j = (0,1), \ j \in L \) \hspace{1cm} (4.19)

\( y_i = (0,1), \ i \in R \) \hspace{1cm} (4.20)

The LSOC problem is solved to optimality using a subgradient optimization (SGO) procedure that iterates between solving the inner maximization problem (4.17 –}
4.20) and the outer minimization problem (4.11 - 4.12), as described in the solution procedure for a dual-based algorithm by Downs and Camm (1996). The inner problem is to maximize the objective function value for a given value of the \( \lambda \) vector. The outer problem is to minimize the objective function value for a given value of the \( x \) and \( y \) vectors, by finding an optimal value for the \( \lambda \) vector.

The inner problem has the integrality property, which means the following. The objective function value of the inner problem with the binary integer restrictions (4.19 - 4.20), is equal to the objective function value of the inner problem with constraint sets (4.19 - 4.20) replaced by \( 0 \leq x_j \leq 1, j \in L \) and \( 0 \leq y_i \leq 1, i \in R \). Also, the solution to the inner problem can be easily obtained by inspection (without resorting to optimization), as shown below.

If \( (1 - \lambda, h_i) > 0 \), then set \( y_i = 1 \), else set \( y_i = 0 \), \( i \in R \).

Set \( x_j = 1 \), where \( j^* \) is such that \( \sum_{i \in R} \mu_{ij} \lambda_i = \max_{j \in L} \left( \sum_{i \in R} \mu_{ij} \lambda_i \right) \), \( k \in A \).

Note the following observation about the quality of the upper bound \( Z_{\text{lag}} \). Since the LSOC problem has the integrality property, \( Z_{\text{lag}} \) will be no better than \( Z_{\text{lp}} \). That is, \( Z_{\text{lag}} \geq Z_{\text{lp}} \). When \( Z_{\text{lag}} = Z_{\text{lp}} \), the SGO procedure is said to converge optimally. Also, the value of the \( x \) vector obtained while solving the inner problem (during an iteration of the SGO procedure) is feasible to the share-of-choice problem. Update \( Z_{\text{inc}} \) if any such feasible solution has a higher objective function value. Next, the SGO procedure that is used to solve the LSOC problem is discussed.

The subgradient optimization procedure uses the following notation.

\( \kappa \) - The iteration number,
$Z_{\text{int}}$ - The objective function value of the share-of-choice problem obtained from the solution to the inner problem (optimal value of the $x$ vector) found during the SGO procedure,

$Z_{\text{lsoc}} = $ The objective function value of the LSOC problem,

$t^\kappa$ - The step size at iteration $\kappa$ of the SGO procedure,

$\eta$ - A variable that keeps track of the number of iterations in the SGO procedure that have gone by without an improvement in $Z_{\text{lag}},$

$\theta^\kappa$ - A scalar used to calculate the step size $t^\kappa,$ at iteration $\kappa$ of the SGO procedure,

$\gamma^\kappa$ - The subgradient vector at iteration $\kappa$ of the SGO procedure,

The steps involved in the SGO procedure are given below.

Step 1 – Set $Z_{\text{lag}} = |R|$. Set $\lambda_i = 0, \quad i \in R$. Set $\theta^1 = 2$.

Set $\kappa = 1$. Set $\eta = 0$. Go to Step 2.

Step 2 – If $(1 - \lambda_i, h_i) > \lambda - \mu$ then set $y_i = 1$, else set $y_i = 0$, where $i \in R$.

Reset $x_j = 0, \quad j \in L$. Set $x_j = 1$, where $j^*$ is such that $\sum_{i \in R} \mu_{ij} \lambda_i = \max_{j \in L} \left( \sum_{i \in R} \mu_{ij} \lambda_i \right), \quad k \in A$.

Compute $Z_{\text{lsoc}} = \sum_{i \in R} (1 - \lambda_i, h_i) y_i + \sum_{j \in L} (\sum_{i \in R} \mu_{ij} \lambda_i) x_j$. Go to Step 3.

Step 3 – If $Z_{\text{lsoc}} < Z_{\text{lag}}$ then set $Z_{\text{lag}} = Z_{\text{lsoc}}$, and reset $\eta = 0$. Else, increment $\eta = \eta + 1$.

If $\eta = 3$ then reduce $\theta^\kappa = \theta^\kappa / 2$ and reset $\eta = 0$. Go to Step 4.

Step 4 – Set $Z_{\text{int}} = 0$. If $\left( \sum_{j \in L} \mu_{ij} x_j \right) - h_i \geq 0$ then, increment $Z_{\text{int}} = Z_{\text{int}} + 1$, for all $i \in R$.

If $Z_{\text{int}} > Z_{\text{inc}}$, then set $Z_{\text{inc}} = Z_{\text{int}}$. Go to Step 5.
Step 5 – Compute the subgradient \( \gamma_i^\kappa = \left( \sum_{j=1}^{L} \mu_j x_j \right) - h_i y_i, \ i \in R \). The step-size is computed using the formula \( t^\kappa = \frac{\theta^\kappa (Z_{\text{lag}} - Z_{\text{inc}})}{\sum_{i \in R} (\gamma_i^\kappa)^2} \). Compute the Lagrangian multipliers for iteration \( \kappa + 1 \) as \( \lambda_i^{\kappa+1} = \lambda_i^\kappa - t^\kappa \gamma_i^\kappa, \ i \in R \). If \( \lambda_i^{\kappa+1} < 0 \), then reset \( \lambda_i^{\kappa+1} = 0, \ i \in R \).

Go to Step 6.

Step 6 – Increment \( \kappa = \kappa + 1 \). If \( \kappa > 30 \) then stop the SGO procedure. Else, go to step 2.

A summary of the steps involved in the SGO procedure is given below. Step 1 is for initialization of parameters and variables. The Lagrangian multipliers \( \lambda_i, \ i \in R \) are all initialized to values of 0. The scalar \( \theta^\kappa \) used to calculate the step-size of the subgradient vector is initialized to value 2. The value of \( Z_{\text{lag}} \) is initialized to the total number of respondents in the study (maximum possible coverage). The iteration counter \( \kappa \) is set at value 1. The variable \( \eta \), which keeps track of the number of iterations that have gone by without an improvement in the value of \( Z_{\text{lag}} \), is initialized to value 0. In step 2, the optimal values of the \( x \) and \( y \) variables are obtained by solving the inner problem, and then \( Z_{\text{inc}} \) is computed. In step 3, set \( Z_{\text{lag}} = Z_{\text{loc}} \) if \( Z_{\text{loc}} < Z_{\text{lag}} \). If \( Z_{\text{loc}} \) does not improve in three consecutive iterations, reduce the value of the scalar \( \theta^\kappa \) by half. In step 4, update \( Z_{\text{loc}} \) if a better solution to the share-of-choice problem can be found using the value of the \( x \)-variables obtained from the solution to the inner problem identified in step 2. In step 5, the value of subgradient vector is calculated and used to compute the step-size. Then, the Lagrangian multipliers (\( \lambda \) vector) are adjusted using the following rule which penalizes violation of a constraint \( i, \ i \in R \). If
\[ \sum_{j \in L} \mu_j x_j - h_i y_i \geq 0 \] then, decrease \( \lambda_i \), else increase \( \lambda_i \), \( i \in R \). The Lagrangian multipliers are always kept at a positive value. In step 6, increment the iteration counter \( \kappa \) and if \( \kappa > 30 \) stop the SGO procedure. Else, go to step 2 and continue the next iteration.

### 4.4.4 Determining the branching order

The run-time of the exact algorithm depends upon the branching order over the attributes. Three different branching strategies for the exact algorithm were considered, all of which use a “depth first search” branching rule. The three branching rules are discussed below.

a) For the MCP, a popular branching strategy is to branch on the \( x_j \) variable that gives the maximum incremental contribution to the objective function value, see Downs and Camm (1996). This is a good branching strategy if the constraint matrix consists of elements with values 0 or 1, and the right-hand side vector has elements with value 1, as in the MCP problem. But, the input data for the share-of-choice problem, as discussed earlier in section 4.2, is unlike the input data for the MCP problem. Also, many attributes have to be branched on before the hurdle value can be exceeded and the objective function increases as a result. So, this branching strategy is not expected to result in faster run-times.

b) The second branching strategy is to identify and branch on the \( x_j \) variable that gives the maximum contribution to covering the uncovered rows of the hurdle vector. The hurdle vector is adjusted in value by accounting for the coverage provided by all the \( x_j \) variables fixed at value 1 at a given node. This branching strategy is expected to result in faster run-times.
c) The third branching strategy is to identify the attribute that has the largest utility values for the most respondents and branch on that attribute first. Next, the attribute with the second largest utility values for the most respondents is identified and branched on. This process is continued until all attributes are branched on. This branching strategy is also expected to result in faster run-times.

In choosing between branching strategies b) and c), the latter was preferred since it produced the best run-times on a set of test problems. The branching order is used during the branch-and-bound stage of the exact algorithm. A description of the algorithm corresponding to branching strategy c) is given below after some additional notation is defined.

κ - The iteration number,
S_k - This set stores the indices of respondents from the set R that have the maximum part-worth in attribute ‘k’ at a given iteration, k ∈ A,
Φ - A null set,
\(\overline{A}\) - This ordered set stores the branching order, and
β - A matrix defined as \(\beta_{ik} = \max_{j∈L_k}(\mu_{ij})\), where i ∈ R and k ∈ A.

Step 1: Initialize the sets \(S_k = \Phi\), k ∈ A, and the set \(\overline{A} = \Phi\). Set κ = 1.

Step 2: For all i ∈ R increment \(S_{k} = S_{k′} \cup i\), where \(k′\) is such that \(\beta_{ik′} = \max_{k∈A,k′∈A}(\beta_{ik})\).

Step 3: Let \(t^*\) be such that \(|S_{t^*}| = \max_{i∈A,t∈A}(|S_t|)\). Increment \(\overline{A} = \overline{A} \cup t^*\).

Step 4: Reset \(S_k = \Phi\), k ∈ A and increment κ = κ + 1.

Step 5: If κ > |A|, terminate the algorithm. Else, go to step 2.
4.4.5 Checking for column dominance

Consider two columns p and q from the constraint matrix that belong to the same attribute set, that is \( p \in L_k \) and \( q \in L_k \), where \( k \in \overline{A} \). Suppose that column p dominates column q. That is, \( \mu_{ip} \geq \mu_{iq} \), \( i \in R \). In that case, we can skip variable \( x_q \) during the branching process. This is because we can find an equivalent or better solution by replacing \( x_q \) with \( x_p \), whenever \( x_q \) appears in any feasible solution. But, column dominance is rarely observed in real world share-of-choice problems because the respondents in a conjoint study have unique preferences associated with each attribute-level. Therefore, we do not attempt to find dominated columns in the constraint matrix that could help reduce the computational burden on the exact algorithm.

4.5 The logic-based fathoming of nodes in the search tree

Two logic-based rules are used to help prune the search tree. The default fathoming rule is logic test I, described in sub-section 4.5.1. The other fathoming rule is logic test II, described in sub-section 4.5.2. Before the description of the two logic tests, note that in any computation involving the \( x_j \) variables, only the \( x_j \) variables fixed at values 1 or 0 are considered, and the free \( x_j \) variables that are fixed at value –1 are ignored. Assume a node in the search tree with the following particulars. Consider the sets

\[
\Psi = \left\{ k \mid \sum_{j \in L_k} x_j = 1 \right\}, k \in A, \text{ and } \Lambda = \left\{ k \mid \sum_{j \in L_k} x_j = 0 \right\}, k \in A. \]

That is, \( \Psi \) is the set of attributes that are branched on, and \( \Lambda \) is the set of attributes that are yet to be branched on. Note that \( \Psi \cup \Lambda = A \).
4.5.1 Logic Test I

Let \( b_i = h_i - \sum_{k \in \Psi} \sum_{j \in \Lambda_k} \mu_{ij} x_j, i \in R \). That is, vector \( b \) represents the remaining hurdle that must be satisfied at the current node after accounting for the set of attributes on which we have already branched. The values of the \( y \)-variables can be assigned as follows.

If \( \sum_{k \in \Psi} \sum_{j \in \Lambda_k} \mu_{ij} x_j + \sum_{k \in A} \min_j (\mu_{ij}) \geq h_i \), then \( y_i = 1, i \in R \). But, \( \min_j (\mu_{ij}) = 0 \), \( i \in R \) and \( k \in A \) (see step 2 in sub-section 4.4.1). Therefore if \( \sum_{k \in \Psi} \sum_{j \in \Lambda_k} \mu_{ij} x_j \geq h_i \) then \( y_i = 1, i \in R \).

That is, if \( b_i \leq 0 \) then \( y_i = 1, i \in R \) \hspace{1cm} (4.21)

Now consider the other case. If \( \sum_{k \in \Psi} \sum_{j \in \Lambda_k} \mu_{ij} x_j + \sum_{k \in A} \max_j (\mu_{ij}) < h_i \), then set \( y_i = 0, i \in R \).

That is, if \( \sum_{k \in A} \max_j (\mu_{ij}) < b_i \), then \( y_i = 0, i \in R \) \hspace{1cm} (4.22)

The values of the \( y \)-variables based on rules (4.21) and (4.22) hold true for any child node from the current node. At the node under consideration, let \( \tau = \{ i | b_i \leq 0 \} \), \( i \in R \), \( \omega = \{ i | \sum_{k \in A} \max_j (\mu_{ij}) < b_i \} \), \( i \in R \) and \( \rho = \{ i | i \not\in \tau \text{ and } i \not\in \omega \} \), \( i \in R \). An upper bound on the objective function value from this node is \( |\tau| + |\rho| \). Hence, if \( |\tau| + |\rho| \leq Z_{inc} \) then the node may be fathomed.

4.5.2 Logic Test II

Logic test II is based on the following proposition and the resulting corollary.
**Proposition 4d:** Consider a set $T$ such that $T \subseteq R$ and $|T| > 1$. Let $c = \sum_{i \in T} h_i$ and

$$\gamma_k = \max_{j \in L_k} \left\{ \sum_{i \in T} \mu_{ij} \right\}, \ k \in A.$$  
If $\sum_{k \in A} \gamma_k < c$, then there does not exist a product profile (set) $P$ such that $\sum_{j \in P} \mu_{ij} \geq h_i \ \forall \ i \in T$.

**Proof:** Suppose $\sum_{k \in A} \gamma_k < c$ and there exists a product $P$ such that $\sum_{j \in P} \mu_{ij} \geq h_i \ \forall \ i \in T$. Note that $\sum_{i \in T} \sum_{j \in P} \mu_{ij} \geq \sum_{i \in T} h_i = c$.

But, $\sum_{i \in T} \sum_{j \in P} \mu_{ij} \leq \sum_{k \in A} \max_{j \in L_k} \left\{ \sum_{i \in T} \mu_{ij} \right\} = \sum_{k \in A} \gamma_k < c$, and therefore product $P$ cannot satisfy all respondents in set $T$.

**Corollary:**

If $\sum_{k \in A} \gamma_k < c$, it follows that at least one respondent in the set $T$ is not satisfied with product $P$.

Logic test II uses the result from the above corollary to prune the search tree.

Before describing logic test II, the following notation is defined.

$W$ – A set whose elements represent aggregate constraints,

$\nu_{kj}$ - A matrix element which stores aggregate partworths, where $\nu_{kj} = \sum_{i \in W_k} \mu_{ij}, \ k \in W$ and $j \in L$,

$c_k$ – A vector element that stores the aggregate hurdle values, where $c_k = \sum_{i \in W_k} h_i, \ k \in W$. 

121
\(d_i\) – An element of the vector that stores the aggregate hurdle values remaining to be covered at a given node, that is, \(d_i = c_i - \sum_{k \in \Psi} \sum_{j \in L_k} v_{ij} x_{ij}, i \in W\).

Now consider the case that \(\sum_{k \in \Psi} \sum_{j \in L_k} v_{ij} x_{ij} + \sum_{k \in \Lambda} \max(v_{ij}) < c_i\), where \(i \in W\). That is,

\[
\sum_{k \in \Lambda} \max(v_{ij}) < d_i, \text{ where } i \in W
\] (4.23)

From (4.23) it can be concluded that one of the respondents belonging to the subset represented by the element \(i\) cannot be satisfied. This conclusion also holds true for any child node from the current node. At the node under consideration, let \(\omega = \left\{ i \mid \sum_{k \in \Lambda} \max(v_{ij}) < d_i \right\}, i \in W\). An upper bound on the objective function value from this node is \(|R| - |\omega|\). Hence, if \(|R| - |\omega| \leq Z_{inc}\) then the node may be fathomed.

To be able to fathom a node using logic test II, the number of aggregate constraints must be greater than or equal to the number of respondents in set \(R\) who are not satisfied by the best incumbent solution \(Z_{inc}\). That is, \(|W| \geq |R| - Z_{inc}\). Otherwise, \(|R| - |\omega|\) will always be greater than \(Z_{inc}\) and the fathoming test for logic test II will always fail. Also, it is empirically found that logic test II is most effective when the incumbent solution \(Z_{inc}\) is greater than 50% of \(|R|\). Hence, a node is not fathomed using logic test II when \(Z_{inc}\) is strictly less than 50% of \(|R|\).

The pruning from logic test II is efficient if the aggregate constraints are “tighter” than the individual constraints that are used to create them. By “tighter”, we mean the following. For aggregate constraints, the ratio of \(\sum_{k \in \Lambda} \max(v_{ij})\) over \(c_i\) must be as small
as possible. If the ratio is strictly less than 1.0, then even before branching begins, we
know that at least one of the respondents belonging to the sub-set i cannot be satisfied. If
the ratio is equal to or a little over 1.0, then the aggregate constraint is likely to hit
infeasibility faster than the case when all the individual constraints belonging to sub-set i
are separately tested using logic test I. A heuristic algorithm that creates “tight”
aggregate constraints is explained in sub-section 4.5.3. Before that, the two logic-based
tests are explained through a trivial instance of the share-of-choice problem.

Consider an attribute set A = \{1,2,3\} and a set of respondents R = \{1,2\}. Let
L_1 = \{1,2\}, L_2 = \{3,4,5\} and L_3 = \{6,7,8\}. The partworths and hurdle values pertaining
to the 2 respondents and the aggregate constraint (cluster \{1,2\}) are given in the table
below. Assume that the starting incumbent solution \( Z_{inc} = 1 \).

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Partworths from attribute 1 (L_1)</th>
<th>Partworths from attribute 2 (L_2)</th>
<th>Partworths from attribute 3 (L_3)</th>
<th>Hurdle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.4 0 0.3 0.4 0 0 0 0.2 0.3 1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0 0.6 0.5 0 0.3 0.2 0.5 0 1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster {1,2}</td>
<td>0.4 0.6 0.8 0.4 0.3 0.2 0.7 0.3 2.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Suppose attribute 1 (L_1) is branched on and variable x_1 is fixed at value 1. Logic
test I cannot help fathom the current node since it is possible to satisfy both respondents 1
and 2 by branching further and thereby improve \( Z_{inc} \). That is, it is possible to branch on
variables x_3 and x_8 to satisfy respondent 1, and branch on variables x_3 and x_7 to
satisfy respondent 2. But logic test II would suggest that the node be fathomed. This is
because the aggregate constraint (cluster \{1,2\}) cannot be satisfied after branching on x_1
and so \( Z_{inc} \) cannot be improved by further branching. Suppose a backtracking step is
taken by fixing variable $x_1 = 0$ and then variable $x_2$ is branched on by fixing it at value

1. Now, logic test I would suggest fathoming the node under consideration. This is because further branching cannot satisfy respondent 1. But, logic test II cannot fathom the node. The above example shows that both logic tests I and II can be beneficial as pruning methods, although logic test II is empirically found to be more effective in pruning the search tree.

Finally, note that the aggregate constraint is “tighter” than the individual constraints (in the above example). The ratio of the maximum possible coverage

$$\left( \sum_{k \in A} \max (\mu_{ij}) \right)$$

over the hurdle value ($h_1$) is 1.1 for respondent 1 and 1.6 for respondent

2. The ratio of the maximum possible coverage

$$\left( \sum_{k \in A} \max (\nu_{ij}) \right)$$

over the hurdle value ($c_1$) is 1.05 for the aggregate constraint. As mentioned earlier, a heuristic algorithm that forms disjoint sets of respondents from the set $R$ is explained in the next sub-section.

4.5.3 The heuristic algorithm used to aggregate constraints

The heuristic algorithm identifies distinct pairs of constraints to aggregate, so that the resulting aggregate constraints are “tight”, as defined in sub-section 4.5.2. Suppose that there are $N$ simple constraints to begin with, where $N$ is assumed to be an even number for ease of explanation. There are $^N\text{C}_2$ distinct ways to combine $N$ simple constraints to form aggregate constraints, where each aggregate constraint is composed of two simple constraints. To keep the run-times low, the best $^N\text{C}_2$ aggregate constraints from among the $^N\text{C}_2$ aggregate constraints are to be chosen efficiently.

An aggregate constraint can be formed from two simple constraints as explained below. The left-hand side value of an aggregate constraint is generated by a vector
addition of the left-hand sides of two simple constraints. A scalar addition of the right-hand side values of two simple constraints generates the right-hand side value of an aggregate constraint.

For each aggregate constraint that could be formed from $^N\!C_2$ distinct combinations of simple constraints, the ratio of the maximum coverage possible on the left-hand side over its right-hand side value is computed. This ratio is referred to as the ‘tightness ratio’. The objective is to form $^N\!C_2$ aggregate constraints so that the “tightness ratio” associated with each of them is as small as possible. One option is to search through each of the $^N\!C_2$ combinations sequentially to find the best set of two simple constraints to aggregate, then again searching through each of the $^N\!C_2 - ^N\!C_2$ combinations sequentially to find the next best set of two simple constraints to aggregate, and so on until all $^N\!C_2$ sets of two simple constraints are identified. But this search procedure is empirically found to be very time consuming, particularly if the value of $N$ is greater than 1000. To speed up the search procedure, a heuristic algorithm is devised as described next.

For each aggregate constraint that could be formed from $^N\!C_2$ distinct combinations of simple constraints, the ‘tightness ratio’ is computed and stored in an appropriate position in a square matrix. The number of rows or columns of this matrix are equal to the number of simple constraints. In the first iteration, the average value of the entries in each column of the matrix is computed. Then, the simple constraint (column), say $p$, with the lowest average value is identified, and the simple constraint (row), say $q$, with the lowest value in column $p$ is identified. An aggregate constraint is formed using simple constraints $p$ and $q$. In the second iteration, the average value of the
entries in each of the remaining (N-2) columns (ignoring rows and columns p and q) of the matrix is recomputed. The above process is repeated to identify the next best set of simple constraints to aggregate. Continuing in this fashion, all the $\frac{N}{2}$ aggregate constraints are generated.

The pseudo-code for the heuristic algorithm is presented after some additional notation is defined below.

$S_1$ – A set with an even number of elements that each represents a constraint,

$S_E$ – A set of elements that each represents an aggregate constraint, which are formed from a unique pair of constraints belonging to the set $S_1$, so that $|S_E| = 0.5 \cdot |S_1|$,

$\tilde{\mu}_{ij}$ - A matrix of partworths, where $i \in S_1$ and $j \in L$,

$\tilde{h}_i$ - A vector of hurdle values, where $i \in S_1$,

$\tilde{u}_{ij}$ - A matrix of aggregate partworths, where $i \in S_E$ and $j \in L$,

$\tilde{c}_i$ - A vector of aggregate hurdle values, where $i \in S_E$,

$\epsilon_{pq}$ - A matrix which stores the ratio $\sum_{k \in A} \max_{j \in L} (\tilde{u}_{ij})$ over $\tilde{c}_i$, where $j \in L$, i corresponds to an aggregate constraint formed from constraints p and q, where $p \in S_1$, $q \in S_1$ and $p \neq q$,

$\Phi$ - A null set, and

$\omega$ - A set that records the elements of $S_1$ that are already used to form aggregate constraints at a given iteration (step 3) in the heuristic algorithm.

Step 1 – Set $\omega = \Phi$. Set $\epsilon_{pp} = 0$, $p \in S_1$.

Step 2 – For all $p \in S_1$ and $q \in S_1$ such that $q > p$, perform the following operation.
Let $\varepsilon_{pq} = \varepsilon_{qp} = \left( \sum_{k \in A} \max_{j \in L_k} (\bar{\mu}_{pq} + \bar{\mu}_{qp}) \right) / \left( \bar{h}_p + \bar{h}_q \right)$.

Step 3 – For all $i \in S_E$, perform the following sequence of operations.

- Let $p^*$ be such that $\sum_{q \in S_i, q \neq 0} \varepsilon_{pq} = \min_{p \in S_i, p \neq 0} \left( \sum_{q \in S_i, q \neq 0} \varepsilon_{pq} \right)$. Increment $\omega = \omega \cup p^*$. And, let $q^*$ be such that $\varepsilon_{pq} = \min_{p \in S_i, p \neq 0} \left( \varepsilon_{pq} \right)$. Increment $\omega = \omega \cup q^*$.

- Let $\bar{v}_{ij} = \bar{\mu}_{pq} + \bar{\mu}_{qj}$, $j \in L$, and $\bar{c}_i = \bar{h}_p + \bar{h}_q$.

We repeat the three steps above and continue the process of forming aggregate constraints as long as both the following conditions are satisfied.

1. The number of aggregate constraints is greater than the number of respondents who were not satisfied by the best incumbent solution found thus far. That is, $|S_E| \geq |R| - Z_{inc}$, and

2. Aggregating constraints again lowers the ratio of $\sum_{i \in S_E} \left( \sum_{k \in A} \max_{j \in L_k} (\bar{v}_{ij}) / \bar{c}_i \right)$ over $|S_E|$, between two consecutive iterations.

Note that each aggregate constraint in the set $S_E$ is created from two distinct constraints belonging to the set $S_i$. Also, the matrix $\bar{v}$ and vector $\bar{c}$ (with new notation $\bar{v}$ and $c$ respectively) are used in step 4 of phase II of the exact algorithm (to be described in section 4.6).

A summary of the three steps used in the heuristic algorithm is given below. In step 1, the set $\omega$ is initialized to the null set and the diagonal elements of the $\varepsilon$ matrix are initialized to values of zero. In step 2, two distinct constraints (say $p$ and $q$) from the set
S₁ are used to compute the ratio of \( \frac{\sum_{k \in A} \max_{j \in L_k} (\bar{\mu}_{pj} + \bar{\mu}_{qj})}{(\bar{h}_p + \bar{h}_q)} \). This value is stored in the matrix elements \( \varepsilon_{pq} \) and \( \varepsilon_{qp} \). This procedure is repeated for all pairs of distinct constraints from the set S₁. In step 3, the pair of constraints \( (p^*, q^*) \) with the lowest ‘tightness ratio’ is identified and an aggregate constraint is formed. In the same manner the rest of the available constraints are paired up to form the other aggregate constraints.

4.6 Phase II of the Exact Algorithm

Phase II of the exact algorithm employs a branch-and-bound method with logic-based bounding rules to solve a given instance of the share-of-choice problem to optimality. A flow diagram of Phase II of the exact algorithm is shown in Figure 4.3.

We first define the notation used to describe phase II of the exact algorithm, some of which are re-stated for convenience.

**Decision Variables:**

\( x_j = 1 \) if level \( j \) is chosen, 0 if not, where \( j \in L \),

\( y_i = 1 \) if the hurdle level of respondent \( i \) is exceeded, 0 if not, where \( i \in R \),

**Sets:**

\( \bar{A} \) - The attribute set \( A \) that is ordered by the branching order for the algorithm,

\( \Psi \) - This set records the indices of attributes that are branched on, when at a given node in the search tree,
Ω - This set records the indices of respondents from the set R whose hurdle values are not yet exceeded \((i \mid b_i > 0)\), when at a given node in the search tree, but the potential for exceeding the hurdle by further branching exists,

Φ - A null set,

\(S_k\) - This set is used to store the row-indices of constraints (respondents) from the \(\mu\) matrix that are either satisfied or cannot be satisfied by further branching, when attribute \(k\) is branched on in the search tree, where \(k \in \overline{A}\),

\(\omega\) - This set records the indices of aggregate constraints from the set \(W\) whose aggregate hurdle values are not yet exceeded \((i \mid d_i > 0)\), when at a given node in the search tree, but the potential for exceeding the aggregate hurdle by further branching exists,

\(T_k\) - This set is used to store the row-indices of aggregate constraints from the \(\upsilon\) matrix that are either satisfied or cannot be satisfied by further branching, when attribute \(k\) is branched on in the search tree, where \(k \in \overline{A}\),

\(W\) – A set whose elements represent aggregate constraints,

**Parameters:**

\(\eta\) - It records the node we are currently at in the search tree,

\(b_i\) – This vector element stores the hurdle value remaining to be covered at a given node in the search tree, where \(b_i = h_i - \sum_{k \in \Psi} \sum_{j \in L_k} \mu_{ij} x_j, i \in R\),

\(\sigma_i\) - This vector element stores the total number of respondents from the set \(R\) whose hurdle values are exceeded when \(i\) attributes are branched on, where \(i \in \overline{A}\) and \(i = |\Psi|\),

\(\beta\) - A matrix defined as \(\beta_{ik} = \max_{j \in L_k} (\mu_{ij})\), where \(i \in R\) and \(k \in \overline{A}\),

129
\( v_{kj} \) - The matrix of aggregate partworths, where \( v_{kj} = \sum_{i \in W_k} \mu_{ij}, k \in W \) and \( j \in L \),

\( c_k \) – An element of the vector that stores the aggregate hurdle values, where

\[
c_k = \sum_{i \in W_k} h_i, \quad k \in W,
\]

\( d_i \) – An element of the vector that stores the aggregate hurdle values remaining to be covered at a given node, that is, \( d_i = c_i - \sum_{k \in \Psi; j \in L_k} \sum_{i \in W_k} v_{ij} x_j, i \in W \),

\( \Gamma_i \) - This vector element stores the total number of aggregate constraints from the set \( W \) whose aggregate hurdle values cannot be satisfied when \( i \) attributes are branched on, where \( i \in \overline{\Gamma} \) and \( i = |\Psi| \),

\( \gamma \) - A matrix defined as \( \gamma_{ik} = \max_{j \in L_k} (v_{ij}) \), where \( i \in W \) and \( k \in \overline{\Gamma} \).

The pseudo-code for Phase II of the algorithm is given below.

Step 1(a) - Let \( \Psi = \Phi \). Let \( \Omega = R \). Let \( S_k = \Phi, \quad k \in \overline{\Gamma} \). Let \( \sigma_i = 0, \quad i \in \overline{\Gamma} \). Let \( b_i = h_i, \quad i \in R \), \( y_i = 0, \quad i \in R \) and \( x_j = -1, \quad j \in L \). Set \( \eta = 1 \). If \( Z_{\text{inc}} \geq 0.5*|R| \) then, go to step 1(b), else go to step 2.

Step 1(b) - Let \( \omega = W \). Let \( T_k = \Phi, \quad k \in \overline{\Gamma} \). Let \( \Gamma_i = 0, \quad i \in \overline{\Gamma} \). Let \( d_i = c_i, \quad i \in W \).

Go to step 2.

Step 2 - Let \( j^* \) be such that \( j^* \in L_\eta \) and \( x_{j^*} = -1 \). If variable \( x_{j^*} \) exists, go to step 3.

Else, go to step 4.

Step 3 - Fix \( x_{j^*} = 1 \). Increment \( \Psi = \Psi \cup \eta \). Go to step 6.

Step 4 – If \( \eta = 1 \), terminate the algorithm. Else, go to step 5.

Step 5 – Reset \( x_j = -1, \quad j \in L_\eta \). Decrement \( \eta = \eta - 1 \). Go to step 14(b).
Step 6 – If \( Z_{\text{inc}} \geq 0.5 \cdot |R| \), go to step 7. Else, go to step 9.

Step 7 - For all \( i \in \omega \), decrement \( d_i = d_i - \nu_j \). If \( d_i \leq 0 \) then decrement \( \omega = \omega - i \) and increment \( T_\eta = T_\eta \cup i \). For all \( i \in \omega \), if \( (d_i - \sum_{k,K} \gamma_{ik}) > 0 \), then decrement \( \omega = \omega - i \), increment \( T_\eta = T_\eta \cup i \) and increment \( \Gamma_\eta = \Gamma_\eta + 1 \). If \( \eta \neq 1 \), then update \( \Gamma_\eta = \Gamma_\eta + \Gamma_{\eta - 1} \).

Go to step 8.

Step 8 - If \( \Gamma_\eta \geq \left( |R| - Z_{\text{inc}} \right) \), then go to step 14(a), else go to step 9.

Step 9 - For all \( i \in \Omega \), decrement \( b_i = b_i - \mu_j \). If \( b_i \leq 0 \), then decrement \( \Omega = \Omega - i \), increment \( S_\eta = S_\eta \cup i \) and increment \( \sigma_\eta = \sigma_\eta + 1 \). For all \( i \in \Omega \), if \( (b_i - \sum_{k,K} \beta_{ik}) > 0 \), then increment \( S_\eta = S_\eta \cup i \) and decrement \( \Omega = \Omega - i \). If \( \eta \neq 1 \), then update \( \sigma_\eta = \sigma_\eta + \sigma_{\eta - 1} \). Go to step 10.

Step 10 - If \( |\Psi| = |\bar{A}| \) and \( \sigma_\eta > Z_{\text{inc}} \), then go to step 11(a). Else, go to step 12.

Step 11(a) – If \( Z_{\text{inc}} < 0.5 \cdot |R| \) and \( \sigma_\eta \geq 0.5 \cdot |R| \), then go to step 11(b). Else, update \( Z_{\text{inc}} = \sigma_\eta \). If \( Z_{\text{inc}} = |R| \), then terminate the algorithm. Else, go to step 14(b).

Step 11(b) - Update \( Z_{\text{inc}} = \sigma_\eta \). If \( Z_{\text{inc}} = |R| \), then terminate the algorithm. Else, initialize the aggregate data parameters and sets as in step 1(b). For each x-variable that is fixed at value 1, update sets and aggregate data parameters as shown in step 7 by following the branching order. Then, go to step 14(b).

Step 12 - If \( (\sigma_\eta + |\Omega|) \leq Z_{\text{inc}} \), then go to step 14(b), else go to step 13.

Step 13 – We increment \( \eta = \eta + 1 \) and return to step 2.
Step 14(a) – Find \( \hat{j} \) such that \( \hat{j} \in L_\eta \) and \( x_\hat{j} = 1 \). Now, fix \( x_\hat{j} = 0 \) and decrement \( \Psi = \Psi - \eta \). Increment \( \omega = \omega \cup T_\eta \). Reset \( T_\eta = \Phi \) and \( \Gamma_\eta = 0 \). Then, increment \( d_i = d_i + v_{ij}, \ i \in \omega \). Return to step 2.

Step 14(b) – Find \( \hat{j} \) such that \( \hat{j} \in L_\eta \) and \( x_\hat{j} = 1 \). Now, fix \( x_\hat{j} = 0 \) and decrement \( \Psi = \Psi - \eta \). Increment \( \Omega = \Omega \cup S_\eta \). Reset \( S_\eta = \Phi \) and \( \sigma_\eta = 0 \). Then, increment \( b_i = b_i + \mu_{ij}, \ i \in \Omega \). If \( Z_{inc} \geq 0.5 \cdot |R| \), then increment \( \omega = \omega \cup T_\eta \). Reset \( T_\eta = \Phi \) and \( \Gamma_\eta = 0 \). Then, increment \( d_i = d_i + v_{ij}, \ i \in \omega \). Return to step 2.

A summary of the steps involved in Phase II of the exact algorithm is given below. In steps 1(a) and 1(b), the values of the various sets, decision variables and parameters used are initialized. The first child-node is to be branched on and \( \eta \) which represents a node (level) in the search tree is set equal to 1. In step 2, a free variable is sought in node \( \eta \). In step 3, if a free variable \( x_j \) is available for branching, it is branched on, the set \( \Psi \) is updated accordingly, and the algorithm proceeds to step 6. But, if no free variable is available, then there are two courses of action. (i) Suppose attribute \( \eta \) is not the first element in set \( \overline{\Lambda} \), i.e., \( \eta \neq 1 \). Then, proceed to step 5 where the \( x \)-variables belonging to node \( \eta \) are reset to value -1 (free variables). Perform a backtracking step to the node \( \eta - 1 \) and go to step 14(b). (ii) Suppose that \( \eta = 1 \). Then, the exact algorithm can be terminated because all the \( x \)-variables in node \( \eta \) are fixed at value 0. This means that the search tree has been explicitly or implicitly enumerated for all possible values of the \( x \)-variables belonging to attribute \( \eta = 1 \), which happens to be the starting point for the branch-and-bound procedure.
In step 6, a check is made if aggregate constraints are available for pruning the search tree using logic-test II. Step 7 involves algebraic operations on the set of aggregate constraints and related sets and parameters. These operations are performed to account for branching on variable $x_j$. In step 8, an attempt to fathom the node using logic-test II is made. If the node can be fathomed, go to step 14(a). Else, go to step 9. Step 9 involves algebraic operations on constraints from the set $R$ and related sets and parameters. This is to account for branching on variable $x_j$. In step 10, a check is made if all the attributes have been branched on. That is, if $|\Psi| = |\bar{A}|$. If true, a feasible solution is available and if the current feasible solution is better than the incumbent solution, go to step 11(a). Else, go to step 12. In step 11(a), if the current feasible solution is greater than or equal to 50% of $|R|$ and $Z_{inc}$ is strictly less than $|R|$, go to step 11(b). Else, update the value of $Z_{inc}$. If $Z_{inc}$ is equal to the maximum possible objective function value $|R|$, the algorithm can be terminated since a better solution cannot be found. Else, go to step 14(b). Update the value of $Z_{inc}$ in step 11(b). If $Z_{inc}$ is equal to the maximum possible objective function value $|R|$, the algorithm can be terminated. Else, generate the aggregate data needed for logic-test II and update sets and data parameters to account for all the $x$-variables branched on at the current node. Then, go to step 14(b). In step 12, logic-test I is used to fathom the current node. If the node is fathomed, go to step 14(b). Else, go to step 13 to branch on the next level $\eta+1$, and return to step 2 to continue the branching process.
Step 14 is split into steps 14(a) and 14(b). The following operations are common to both these steps. Perform the backtracking step after identifying the variable \( x_j \) in attribute \( \eta \) that has been fixed at value 1. Fix the value of \( x_j \) to 0 and remove element \( \eta \) from the set \( \Psi \). The difference between the two steps is the following. After branching on the variable \( x_j \), if the node is fathomed using logic-test II (in step 8), only data pertaining to the aggregate constraints and its related sets and parameters needs to be updated. In step 14(a), these data updates are performed and the algorithm returns to step 2. But, suppose fathoming of a node occurs in step 5 (when backtracking takes us to the previous level in the search tree) or in step 11 (when all attributes are branched on) or in step 12 (due to successful fathoming through logic-test I). In such a case, all the data pertaining to the simple and aggregate constraints, along with the related sets and parameters needs to be updated. In step 14(b), all these data updates are performed and the algorithm returns to step 2 to resume the branching process.

We summarize the main sections of this chapter thus. A binary integer programming formulation of the share-of-choice problem was presented. Then, the limitations of LP-based branch-and-bound method in solving the share-of-choice problem to optimality and the potential for using logic-based bounds in the branch-and-bound tree were discussed. Next, the implicit enumeration scheme that forms the basis of the search tree used in the exact algorithm was presented. Phase I of the exact algorithm was next discussed in detail, in the following order. Three pre-processing steps were performed on the input data. A greedy algorithm to solve the share-of-choice problem was stated and the greedy solution was retained as an incumbent solution. A Lagrangian relaxation of the share-of-choice problem was defined and its solution obtained from using the SGO
procedure. The incumbent solution was updated if a better solution was found during the SGO procedure. The branching rule was presented. Then, the demerit of checking for column dominance in the constraint matrix during the branch-and-bound stage (Phase II) of the exact algorithm was given. Next, two logic-based pruning methods used to fathom a node in the search tree were described. Also, a heuristic algorithm that forms aggregate constraints from simple constraints was stated. Finally, a complete description of phase II (branch-and-bound) of the exact algorithm was given. In the next chapter, algorithmic approaches for product line design will be described.
The product line design (PLD) problem is one of the key challenges faced by marketing managers. Unlike the single product design problem, the PLD problem requires us to determine an optimal set of products to offer in a market comprising consumers from many market segments and product lines offered by several competitors. Dobson and Kalish (1988) state “The PLD problem is faced by firms introducing products into new markets, by firms introducing products into existing markets, or by firms modifying their product line”. Morgan et al. (2001) define the PLD problem as “involving the determination a mix of products that can compete in the marketplace and earn profits for the firm”. Krieger et al. (2004) present a case study related to the credit card business. Many competitive strategies were analyzed and line extensions discussed. With regard to line extensions, the objective was to draw business only from competitor’s products.

Dobson and Kalish (1988) make the following observation. A firm can produce an ideal product for each consumer segment, charge a premium price for all such products and as a result make a big profit. But, this would require the firm to support a large product line. Consequently, the total cost of maintaining the product line would shoot up. Also, there may be a cannibalization of the products within the product line due to which consumers could shift to lower priced products and cause the overall profits to fall. The objective of the PLD problem therefore becomes one of determining an optimal size for the product line as well as the prices to charge for each product in the product line.

Dobson and Kalish (1993) provide the following insight, which we quote. “A profit seeking firm will typically provide products of quality lower than the ideal
(maximum welfare) to the less lucrative segments, in order to minimize the effects of cannibalization on the more lucrative segments. The firm assigns the wealthier segments to their ideal products and charges these segments less than their reservation price for these products, in order to prevent them from switching to lower priced products”.

Some examples of PLD issues that managers have to deal with are given below.

1. How many versions of Personal Computers should Dell Computers offer?

2. How many flavors and sizes of Frito Lay’s potato chip bags should be in the product line?

3. How many models of cars should Toyota offer its customers?

4. How many types of bank accounts should Bank of America offer its customers?


1. What should the product line size be and how should the product line be composed?

2. What products need to be added and deleted from the existing product line?

3. Is there substitutability and complementarity among products within the product line?

4. What are the relationships between the product line offered by the firm and the product lines offered by the competitors?

5. How should the prices for the various products be decided?

6. What objective function (buyer’s welfare, seller’s welfare, market share, profits, etc.) should be used in a PLD model?
7. How are fixed and variable costs of producing the various products accounted for in the PLD model?

8. How does the choice of a product line affect manufacturing cost and efficiency?

9. How can marketing and manufacturing coordination be achieved in PLD?

The answers to the above questions depend on the nature of the product, the type of market, objective of the firm etc. The PLD models discussed in this chapter attempt to address many such issues. In this chapter, a variety of mathematical models proposed in the literature to solve the PLD problem are described.

The chapter is organized as follows. In section 5.1, PLD models that incorporate three types of objective functions called the buyer’s welfare, seller’s welfare, and the share-of-choice problem respectively are discussed. There are six sub-sections within section 5.1 that each reviews a specific model and/or solution method to solve problems with the three aforementioned objective function types. The six solution methods are a greedy heuristic, an X-system solver, a dynamic programming heuristic, a beam search heuristic, an ideal point heuristic and a nested partitions heuristic. In section 5.2, a share-of-choice model that includes price and budget is reviewed. In section 5.3, a model for the product line and price selection problem is reviewed. In section 5.4, models for positioning and pricing a product line are reviewed. In section 5.5, a model to maximize profit by considering manufacturing synergies is reviewed, and is followed by a discussion on product platforms. The chapter is concluded with a summary.

5.1 Buyer’s Welfare, Seller’s Welfare and Share-of-choice Models

The objective of the buyer’s welfare problem is to maximize the total sum of the buyer’s utilities of the sub-set of products (product line) that are chosen from a reference set of
products. The objective of the seller’s welfare problem is to maximize the total sum of the seller’s utilities of the sub-set of products (product line) that are chosen from a reference set of products. The share-of-choice problem is to maximize the number of respondents in the study who can choose at least one product from the product line that has a higher utility than their respective status-quo utilities. For each of the three objectives discussed above, the buyers are assumed to choose a single product from the product line with which they associate the highest utility. In most of the models, there is a requirement that a product be chosen by a consumer only if its utility exceeds the utility of the consumer’s status quo product. The various heuristics proposed in the literature are discussed below. Krieger et al. (2004) recommend a two-stage approach to PLD. In the first stage, a set of reference products is chosen from each segment using the first choice rule. That is, the chosen products have utility close to that of the highest utility product for most respondents. In the second stage, a product line is selected from the set of reference products after considering the presence of competitive products.

5.1.1 Greedy Heuristic

Green and Krieger (1985) propose models and heuristics for solving the buyer’s and seller’s version of the product line selection problem using preference data obtained from conjoint analysis. A two-step approach to PLD is taken as discussed above. In the first step, a reference set of product designs is created. In the next step, a specified number of products required in the product line are selected from the reference set of products. Given a specification of the number of products required in the line, the authors describe a model for obtaining a product line that maximizes an objective function made up of buyer preference data. A reference set of products and buyer preference data for these
products are assumed to be available. Besides new product profiles, the products in the reference set N could include any number of status quo products that are currently being used by a sub-set of respondents in the study. The buyer is assumed to choose the product for which he or she has the maximum utility. The notation is defined below.

**Sets:**

N – The reference set of products,

R - The set of respondents in the study,

**Parameters:**

\[ U_{ij} = \text{The utility of respondent } i \text{ (normalized within individuals) for reference product } j, \]

where \( i \in R \) and \( j \in N \),

K – The desired number of products in the product line,

**Decision Variables:**

\[ y_j = 1 \text{ if product } j \text{ is chosen in the product line, } 0 \text{ if not, where } j \in N \]

\[ x_{ij} = 1 \text{ if respondent } i \text{ chooses product } j, 0 \text{ if not, where } i \in R \text{ and } j \in N \],

The math programming formulation for the buyer’s welfare problem is given below.

\[
\begin{align*}
\text{Max} & \quad \sum_{i \in R} \sum_{j \in N} U_{ij} x_{ij} \\
\text{Subject to} : & \quad \sum_{j \in N} y_j = K \\
& \quad x_{ij} \leq y_j, i \in R \text{ and } j \in N \\
& \quad x_{ij} = \{0,1\}, i \in R \text{ and } j \in N \\
& \quad y_j = \{0,1\}, j \in N
\end{align*}
\]

(5.1) (5.2) (5.3) (5.4) (5.5)

Solving the buyer’s welfare problem (of even a moderate size) to optimality using a branch and bound optimization procedure is reported to take a large amount of computing
time. The authors suggest a greedy-and-interchange (GI) heuristic to solve the buyer’s welfare problem. We first describe the greedy part. Until the required number of products is chosen, at every stage a new product that is not already selected is identified and added to the product line. The product chosen at each stage is the one that maximizes objective function (5.1) with the partial product line determined at that stage. Once the greedy solution is completely identified, the incumbent solution is set equal to the greedy solution, and then the interchange heuristic is used to improve upon the incumbent solution. During the interchange heuristic, a product from the product line is replaced by each of the products that are not in the product line, one at a time in a sequential manner. The resulting objective function values are recorded. If the best objective function value found in the interchange process is better than the incumbent solution, the newly added product becomes part of the product line, the incumbent solution is updated, and the replaced product is kept out of the product line. We continue this interchange process by replacing another product in the current solution by those that do not appear in the current solution. We continue this iterative process until no improvement can be found after all possible replacements are tested. Based on a result from an earlier work, the authors show that the GI heuristic solution value is always better than 63\% of the optimal solution value. The authors also discuss the use of a Lagrangian relaxation technique in solving the buyer’s welfare problem to obtain an upper bound on the optimal solution value, but do not provide any technical details or model formulation.

The model formulation of the seller’s welfare problem is described next. In this model, the buyers are assumed to choose a product from the product line that maximizes
their individual utilities, provided it exceeds the utility of their respective status quo products. The buyer’s individual choices are identified after the product line is identified. We define some additional notation needed to formulate the model.

Parameters:

\[ v_{ij} = \text{The utility to seller if respondent } i \text{ chooses reference product } j, \text{ where } i \in R \text{ and } j \in N, \]

\[ U_i = \text{The utility of the status quo product for respondent } i, \text{ where } i \in R, \]

\[ \mu_{ij} = 1 \text{ if } U_{ij} \geq U_i, \text{ else } \mu_{ij} = 0, \text{ where } i \in R \text{ and } j \in N, \]

The math programming formulation for the seller’s welfare problem is given below.

\[
\begin{align*}
\text{Max} & \quad \sum_{i \in R} \sum_{j \in N} v_{ij} \mu_{ij} x_{ij} \\
\text{Subject to:} & \quad \sum_{j \in N} y_j = K \\
& \quad x_{ij} \mu_{ij} \leq y_j, \ i \in R \text{ and } j \in N \\
& \quad x_{ij} \in \{0,1\}, \ i \in R \text{ and } j \in N \\
& \quad y_j \in \{0,1\}, \ j \in N
\end{align*}
\]

The seller’s welfare problems of a moderate size are also difficult to solve to optimality. Therefore, the authors suggest a greedy heuristic to solve the seller’s welfare problem. Products are chosen one at a time until the required number of products is chosen. At every stage a new product that is not already selected is identified and added to the product line. The product chosen at each stage is the one that maximizes the incremental contribution to the objective function (5.6).

It is assumed that the seller’s utilities can be estimated accurately. The greedy heuristic does not perform well in terms of finding the optimal solution, particularly if the
respondents in the study belong to many market segments. If the buyer’s utilities are positively correlated with the seller’s utilities, the greedy heuristic is expected to perform well. A detailed example of a homeowner’s insurance policy is given.

5.1.2 The X-System Solver

Mcbride and Zufriden (1988) extend the work of Green and Krieger (1985) on the PLD problem. They present a new model formulation to represent the seller’s welfare problem. A solver called the X-system is used to solve the model to optimality. The seller’s utility for a product is said to be based on several factors such as frequency of purchases, purchase quantities, dollar value of purchases, and profitability of the product over a time horizon. The following assumptions are made in the model.

1. Consumer utilities are obtained using the conjoint analysis methodology.
2. Consumers choose a single product with which they associate the highest utility from among the products offered in the product line. The highest utility value obtained must exceed the utility value of the consumer’s status quo product, or else the consumer will stay with his or her status quo product.
3. As in Green and Krieger (1985), a reference set of products and buyer preference data for these products is available and the product line is chosen from the reference set.

We define the notation used as follows.

Sets:

N – The reference set of products,
R - The set of respondents in the study,

Parameters:
$U_j = \text{The utility of respondent } i \text{ for product } j, \text{ where } i \in R \text{ and } j \in N,$

$v_{ij} = \text{The utility to seller if respondent } i \text{ chooses product } j, \text{ where } i \in R \text{ and } j \in N,$

$U_i = \text{The utility of the status quo product for respondent } i, \text{ where } i \in R,$

$K = \text{The desired number of products in the product line},$

*Decision Variables:*

$y_j = 1 \text{ if the product } j \text{ is chosen in the product line, } 0 \text{ if not, where } j \in N$

$x_{ij} = 1 \text{ if respondent } i \text{ chooses product } j, 0 \text{ if not, where } i \in R \text{ and } j \in N,$

The math programming formulation for the seller’s welfare problem is given below.

\[
\text{Max } \sum_{i \in R} \sum_{j \in N} v_{ij} x_{ij} \quad (5.11)
\]

Subject to:

\[
\sum_{j \in N} x_{ij} \leq 1, \ i \in R \quad (5.12)
\]

\[
x_{ij} \leq y_j, \ i \in R \text{ and } j \in N \quad (5.13)
\]

\[
y_j U_{ij} \geq y_m U_{im} - M(1 - x_{ij}), i \in R, j \in N, m \in N \text{ and } m \neq j \quad (5.14)
\]

\[
\sum_{j \in N} y_j \leq K \quad (5.15)
\]

\[
x_{ij} = \{0,1\}, \ i \in R \text{ and } j \in N \quad (5.16)
\]

\[
y_j = \{0,1\}, \ j \in N \quad (5.17)
\]

Two pre-processing steps are applied to the input data as shown below.

1. If $U_{ij} < U_i$, then fix $x_{ij} = 0$, where $i \in R$ and $j \in N$.

2. If $U_{ij} < U_i$ for all $i \in R$, then fix $y_j = 0$.

Two of the constraint sets are rewritten to reduce the number of constraints contained in them. Constraint set (5.13) is rewritten as $\sum_{i \in R} x_{ij} \leq |R| * y_j, \ j \in N$ - (5.13a). Constraint set
(5.14) is rewritten as $y_m + x_{ij} \leq 1, i \in R, j \in N, m \in R, m \neq j,$ and $U_{im} > U_{ij} - (5.14a)$. But, it is not clear if constraint set (5.14a) can substitute for constraint set (5.14).

The problem defined by (5.11) – (5.17) is referred to as P1. P1 is reported to be difficult to solve in a reasonable amount of time. So, the authors consider solving a special case of P1 when $v_{ij} = v_i, j \in N$ for all $i \in R$. This special problem is referred to as P2. The authors cite examples to show real world situations where product lines consist of products that have equal profit margins and hence have similar utilities for the seller. A few examples of such product lines are chewing gums, air fresheners, pet foods and cough drops. Problem P2 is simpler and the solution technique is explained next.

Consider a problem P3 that is defined to be problem P2 minus constraint set (5.14a). The constraint matrix of P3 has a special structure. It is evident from the definition of problem P3 that its objective function value serves as an upper bound on the optimal solution for P2. It so happens that the optimal objective function value for P3 is equal to that for P2. Only the optimal values of $y_j$ and $x_{ij}$ variables may differ between the two problems. The optimal solution to P3 guarantees that consumers pick acceptable products but does not guarantee that consumers will pick products with which they associate the highest utility. But, P2 guarantees both the aforementioned conditions. Since the objective function coefficients are all the same (for P2 or P3), it is possible to reassign products (from within the chosen product line) to consumers one at a time, so that they are assigned products with which they associate the highest utility. This reassignment of products enforces constraint set (5.14a). The optimal values of $y_j$ and $x_{ij}$ variables are reset after the reassignment. That is, once an optimal solution to P3 is obtained, constraint (5.14a) is enforced and an optimal solution to P2 is obtained.
If the objective function of P3 is changed to \( \text{Max} \sum_{i \in R} \sum_{j \in N} U_{ij} x_{ij} \), we have the buyer’s welfare problem as described in Green and Krieger (1985). Since P3 can be easily solved, the authors expect that the buyer’s welfare problem can also be solved easily using a similar approach.

A set of test problems is generated to study the performance of X-system in solving P2. The individual consumer utilities and their status quo utilities are generated from a uniform distribution. The seller’s utilities are fixed at a constant value of 1. The number of products in the product line is fixed at 10. The number of consumers is set at levels 100, 200 and 300. The number of products in the reference set is set at levels 16, 32, 64, 128, 256 and 512. It is reported that the X-system is very effective in solving all the test problems of type P2.

Next, a set of test problems is generated to study the performance of X-system in solving P1. As before, the individual consumer utilities and their status quo utilities are generated from a uniform distribution. The seller’s utilities are generated from a uniform distribution. The number of products in the product line is fixed at 10. The number of consumers is set at levels 50, 75 and 100, and the number products in the reference set is set at levels 16 and 32. It is shown that the X-system is not effective in solving test problems of type P1 except for small problem instances. This is because the number of constraints for a given problem is much more in the case of P1 than in P2 and it becomes difficult to prove optimality. The authors suggest reducing the number of consumers in problem P1 using a cluster analysis of partworths to create homogeneous market segments, and then fitting an average utility function to each cluster of consumers. This way, the burden on the optimization algorithm could be reduced.
A sensitivity analysis (using surface response methodology) is conducted to test the influence of the number of respondents, the computing speed and the number of reference products on the run time of the X-system in solving the problems of type P2. All the three factors are reported to significantly influence run times. Based on a statistical analysis, the independent variables are reported to explain 94% of variation in the dependent variable. No sensitivity analysis is conducted for problems of type P1 because of the difficulty in solving them to optimality.

The authors show that it is possible to extend the formulation of P1 to include new constraints that enforce additional rules. For example, if fixed costs \( C_j \) are to be considered for each product in the product line, the objective function could be changed as follows: \[
\text{Max } \sum_{i \in R} \sum_{j \in N} v_{ij} x_{ij} - \sum_{j \in N} y_j C_j.
\] Other examples of additional constraints suggested include (1) either-or type of constraints when two or more products cannot appear together in the product line, (2) constraints to include at most ‘p’ products of a certain product class in the product line consisting of ‘m’ products \( (p \leq m) \), and (3) constraints that enforce all or nothing type of rules. That is, a given set of products is included in the product line or else none of the products in the given set are included in the product line.

The following topics are suggested for future research. (1) A methodology to select the set of N reference products, (2) Alternative approaches to single product choice models, (3) Dynamic formulations of the PLD problem that consider time-varying consumer and seller utilities, (4) Integrating the two step procedure of identifying a reference set of products and then selecting a product line from it to a single step model,
and (5) Incorporating product promotion, distribution and competitive environment into the model.

5.1.3 Dynamic Programming Heuristic

Kohli and Sukumar (1990) extend their dynamic programming (DP) heuristic for single product design to PLD and selection. The authors argue that it is better to construct product lines directly from partworths data than use the two-step procedure suggested by Green and Krieger (1985). This is because the optimality of the product line depends on the quality of product designs in the reference set. A fixed set of competitive products is assumed in the PLD model. The PLD and selection problem is solved for three different objective functions. 1) To maximize total buyer’s welfare (utility). 2) To maximize share-of-choice. 3) To maximize seller’s welfare (revenue). The following notation is used.

**Sets:**

- Ω – The set of attributes,
- \( \Phi_k \) - The set of levels of attribute k, where \( k \in \Omega \),
- θ - The set of respondents in the study,
- Ψ – The set of products (the product line) to be chosen,

**Parameters:**

- M – The desired number of products in the product line,
- \( w_{ijk} \) = The partworth of respondent i for level j of attribute k, where \( i \in \theta, j \in \Phi_k \), and \( k \in \Omega \),

**Decision Variables:**
\( x_{ijkm} = 1 \) if level \( j \) of attribute \( k \) is assigned to product \( m \) and respondent \( i \), 0 if not; where \( i \in \Theta, j \in \Phi_k, k \in \Omega, \) and \( m \in \Psi \).

The model for the buyer’s welfare problem is given below.

\[
\text{Max} \quad \sum_{i \in \Theta} \sum_{m \in \Psi} \sum_{k \in \Omega} \sum_{\Phi_k} w_{ijkm} x_{ijkm} \tag{5.18}
\]

Subject to:

\[
\sum_{j \in \Phi_k} x_{ijkm} = 1, \quad i \in \Theta \quad \text{and} \quad k \in \Omega \tag{5.19}
\]

\[
\sum_{j \in \Phi_k} x_{ijkm} - \sum_{j \in \Phi_k} x_{ijkm} = 0, \quad k' > k, (k, k') \in \Omega, \quad i \in \Theta \quad \text{and} \quad m \in \Psi \tag{5.20}
\]

\[
x_{ijkm} + x_{ij'km} \leq 1, \quad i' > i, j' > j, (i, i') \in \Theta, (j, j') \in \Phi_k, \quad k \in \Omega \quad \text{and} \quad m \in \Psi \tag{5.21}
\]

\[
x_{ijkm} = \{0, 1\}, \quad i \in \Theta, \quad m \in \Psi, \quad k \in \Omega \quad \text{and} \quad j \in \Phi_k \tag{5.22}
\]

The additional model formulation of the share-of-choice problem is given after the following notation is defined.

**Parameters:**

- \( a_i \) - The weight associated with respondent \( i \), where \( i \in \Theta \),

- \( w_{j^*k} \) - The partworth of respondent \( i \) for level \( j^* \) from attribute \( k \) of his/her status-quo product, where \( i \in \Theta, j^* \in \Phi_k, \) and \( k \in \Omega, \)

- \( C_{ijk} \) - The partworth of level \( j \) from attribute \( k \) relative to the partworth of level \( j^* \) from attribute \( k \) for respondent \( i \), given as \( C_{ijk} = w_{ij} - w_{j^*k} \); the \( w_{ij} \) values are normalized to sum to 1, and hence \( -1 < \sum_{k \in \Omega} \sum_{\Phi_k} C_{ijk} < 1, \quad i \in \Theta \),

**Decision Variables:**

\( x_{ijkm} = 1 \) if level \( j \) of attribute \( k \) is assigned to product \( m \), 0 if not; where \( j \in \Phi_k, k \in \Omega, \) and \( m \in \Psi \),
\( x_{im} = 1 \) if product \( m \) cannot provide respondent \( i \) greater utility than his or her status-quo utility, 0 otherwise; where \( i \in \Theta \) and \( m \in \Psi \),

\( x_i = 1 \) if respondent \( i \) cannot be satisfied by any of the products in the product line \( \Psi \), 0 otherwise; where \( i \in \Theta \).

Min \( \sum_{i \in \Theta} a_i x_i \) \hspace{1cm} (5.23)

Subject to:

\[ \sum_{j \in \Phi_k} x_{jkm} = 1, k \in \Omega \text{ and } m \in \Psi \] \hspace{1cm} (5.24)

\[ \sum_{k \in \Omega} \sum_{j \in \Phi_k} C_{ijk} x_{jkm} + x_{im} > 0, i \in \Theta \text{ and } m \in \Psi \] \hspace{1cm} (5.25)

\[ x_i - \sum_{m \in \Psi} x_{im} \geq 1 - M, i \in \Theta \] \hspace{1cm} (5.26)

\[ x_{jkm}, x_{im}, x_i \in \{0, 1\}, i \in \Theta, m \in \Psi, k \in \Omega \text{ and } j \in \Phi_k \] \hspace{1cm} (5.27)

The formulation for the seller’s welfare problem is given after the following additional notation is defined.

Parameters:

\( \nu_{ijk} \) = The partworth reflecting the seller’s return if respondent \( i \) purchases a product with level \( j \) of attribute \( k \), where \( i \in \Theta \), \( j \in \Phi_k \), and \( k \in \Omega \),

\( d_{ijk} \) = The marginal return that the seller associates with respondent \( i \), level \( j \) of attribute \( k \), where \( i \in \Theta \), \( j \in \Phi_k \) and \( k \in \Omega \); \( d_{ijk} = \nu_{ijk} - \nu_{ij'k} \) if respondent \( i \) switches from a status-quo product offered by the seller, and \( d_{ijk} = \nu_{ijk} \) if respondent \( i \) switches from a status-quo product offered by a competitor.

\( U_i \) = The utility of the status-quo product of respondent \( i \), where \( i \in \Theta \).

Decision Variables:
\[ x_{ijkm} = 1 \text{ if level } j \text{ of attribute } k \text{ is assigned to product } m \text{ and respondent } i, \ 0 \text{ if not; where} \]
\[ i \in \Theta, \ j \in \Phi_k, \ k \in \Omega, \text{ and } m \in \Psi, \]
\[ y_i = 1 \text{ if a product assigned to respondent } i \text{ has a higher utility that his or her status-quo utility, } 0 \text{ otherwise,} \]

\[
\text{Max } \sum_{i=0}^{\Theta} \sum_{m \in \Psi} \sum_{k \in \Omega} \sum_{j \in \Phi_k} d_{ijk} x_{ijkm} y_i
\]

Subject to:
\[
\sum_{m \in \Psi} \sum_{k \in \Omega} \sum_{j \in \Phi_k} \sum_{i \in \Theta} w_{ijk} (x_{ijkm} - x_{ijkm}) \geq 0, \ i \neq i', \ i \in \Theta \quad (5.29)
\]
\[
y_i \sum_{m \in \Psi} \sum_{k \in \Omega} \sum_{j \in \Phi_k} \sum_{i \in \Theta} w_{ijk} x_{ijkm} \geq y_i U_1, \ i \in \Theta \quad (5.30)
\]
\[
\sum_{j \in \Phi_k} x_{ijkm} = 1, \ i \in \Theta \text{ and } k \in \Omega \quad (5.31)
\]
\[
\sum_{j \in \Phi_k} x_{ijkm} - \sum_{j \in \Phi_k} x_{ijkm'} = 0, \ k' > k, (k, k') \in \Omega, \ i \in \Theta \text{ and } m \in \Psi \quad (5.32)
\]
\[
x_{ijkm} + x_{ij'km} \leq 1, \ i' > i, j' > j, (i, i') \in \Theta, (j, j') \in \Phi_k, \ k \in \Omega \text{ and } m \in \Psi \quad (5.33)
\]
\[
x_{ijkm}, y_i = \{0,1\}, \ i \in \Theta, m \in \Psi, k \in \Omega \text{ and } j \in \Phi_k \quad (5.34)
\]

The buyer’s welfare problem and the share-of-choice problem are modeled as binary integer programs with linear objective function and constraints, whereas the seller’s welfare problem is modeled as a binary integer program with a non-linear objective function and both linear and non-linear constraints. The authors report that each of these problems is NP-Hard. All these three model formulations employ a large number of binary integer variables and a large number of constraints. Hence, real-world sized problems are difficult to solve to optimality.

Hence, the authors extend the DP heuristic for single product design as described in chapter 3 (sub-section 3.3.2) to solve the PLD problem. We skip a detailed description of the DP heuristic and provide a short summary instead, since the approach for PLD practically mirrors the approach for single product design with a few changes. The
attributes are stages and the attribute levels are states. All possible combinations of columns in the first attribute set and the second attribute set are identified and new aggregate-columns are generated by adding every pair of such combination of columns. The resulting aggregate-columns comprise a set and each such column is evaluated using an evaluation function depending upon the objective function (buyer’s welfare, seller’s welfare or share-of-choice). The best partial product profiles are selected and the associated aggregate-columns are retained. In the next step, all possible combinations of aggregate-columns in the newly created set and the third attribute set are identified and new aggregate-columns are generated by adding every pair of such combinations of columns. The best partial profiles are again selected and retained using an evaluation function, as described before. This process continues until we have considered all the attribute sets. From the last set of complete product profiles, the best set of products is identified and retained as the final solution. The authors provide a step-by-step example to illustrate the DP heuristic. To eliminate infeasible product profiles in the final solution, the authors suggest the same remedy as explained in sub-section 3.3.2.

The evaluation functions are different for the three objective function types. For the buyer’s welfare problem, the partial product profiles that are selected correspond to those with the maximum incremental contribution to the objective of maximizing the buyer’s utilities. For the share-of-choice problem, the partial product profiles that are selected correspond to those that have the maximum incremental number of respondents with a positive relative partworths utility. For the seller’s welfare problem, the partial product profiles that are selected maximize the seller’s incremental return, given that a
respondent picks an available partial profile with the highest positive relative partworths utility.

A Monte Carlo simulation is designed to study the performance of the DP heuristic. A 3x3x3x3 experimental design, with (4,5,6) attributes, (2,3,4) levels per attribute, (50,100,150) respondents and (2,3,4) product line sizes as the design factors are used to generate test problems. Four replicates are solved for each problem. The solution quality and run-times is compared to those of the greedy heuristic suggested by Green and Krieger (1985), and to the optimal solution computed using brute force enumeration. The DP heuristic is shown to be better than the greedy heuristic in closely approximating the optimal solutions and in faster run-times. An application of the DP heuristic is explained using a consumer durable product as example. The authors recommend that the DP heuristic be used to perform sensitivity analysis of the robustness of the final solution.

5.1.4 Beam Search Heuristic

Nair et al. (1995) have proposed a beam search (BS) heuristic to solve the PLD and selection problem. The BS heuristic is a one-step procedure like the DP heuristic discussed in sub-section 5.1.3. That is, the product line is constructed directly from the partworths data. A fixed set of competitive products is assumed to be available. The BS heuristic is used to solve the buyer’s welfare problem, the share-of-choice problem and the seller’s welfare problem. The mathematical model formulations for the three aforementioned problems are adopted from Kohli and Sukumar (1990).

The BS heuristic is a breadth first search procedure without backtracking. The BS heuristic is controlled by a beam width factor $b$ that limits the number of nodes selected
at each level of the search tree and thereby helps prune inferior branches. Therefore, the BS heuristic cannot guarantee an optimal solution. The selection of b nodes at each level of the search tree is based on certain evaluation functions that identify the potential of each node in finding a good solution. Each of the b nodes can be branched into further nodes in the next level of the search tree. Note that if a node is not branched into all possible child nodes but limited to a factor ‘f’, the method is referred to as a filtered beam search. Since the BS heuristic does not branch on all nodes, it runs faster than the complete enumeration method.

Each of the three types of problems can be solved using the BS heuristic. At first, pairs of attribute matrices are combined to create single aggregate matrices at the first level of the search tree. In case of the seller’s welfare problem, the attribute matrix is modified to also include seller’s return. The BS heuristic works as follows. Let A(1) and A(2) be two attribute matrices. All distinct pairs of columns, one each from matrices A(1) and A(2), are added to one another to create a set of aggregate columns. The resulting set of aggregate columns are stored in an aggregate matrix, say E(1). All other distinct pairs of attribute matrices are combined in this manner at the current level of the search tree. If there are an odd number of attribute matrices to begin with, we ignore any one of the attribute matrices during the aggregation process and carry it over to the next level of the search tree. In each of the set of aggregate matrices, the best b columns are identified through evaluation functions and retained, while the rest are culled. If there are less than b columns in the aggregate matrix, all are chosen. These aggregate matrices are carried over to the next level of the search tree. Note that at the next level, we will roughly have about half the matrices we have in the current level, depending on whether
the current level has an odd or even number of matrices. This aggregation process is continued until in the end we have a single aggregate matrix consisting of columns that represent complete product profiles. The best b product profiles (columns) are selected using an evaluation function and they represent the first products in the b product lines.

Next, the remaining products in each product line are to be identified. It is desired that each incremental product added to the product line complements the products already chosen. Therefore, the resulting product lines are expected to cater to the choice of the largest number of respondents, and chances for product duplication within a product line is reduced. Consider the case where we have to identify the second product in the first product line. Identify the respondents (rows) in the attribute matrices for whom the utility from the first product exceeds their status-quo utilities, and remove these rows from further consideration (after some minor modification in the case of the buyer’s problem). The second product is the single best product profile found by applying the aggregation procedure on the reduced attribute matrices. A similar approach is used to find all the remaining products in the first product line. The same solution procedure is repeated for each of the remaining (b-1) product lines to identify the remaining products.

The primary evaluation function used to select the best column is to find the column with the maximum number of positive elements in it. In case of a tie, the column with the maximum number of non-negative entries is chosen. In case of further ties, the column with the maximum sum of the positive entries is chosen. If the tie is still not resolved, one of the tied columns is arbitrarily chosen.
To study the performance of the BS heuristic, a 3x3x3x3 experimental design, with (4,5,6) attributes, (2,3,4) levels per attribute, (50,100,150) respondents and (2,3,4) product line sizes as the design factors are used to generate problems (exactly as in the DP heuristic case). Either 5 or 10 replicates are solved for each problem depending on its size. The solution quality and run-times is compared to those obtained from using the DP heuristic and to the optimal solution computed using brute force enumeration. The BS heuristic solutions are reported to be superior to the DP heuristic solutions using six performance measures as the basis. The measures are 1) Better average performance in approximating the optimal solution, 2) Lower standard deviation of solution values with respect to the optimal solution over all replicates of a problem, 3) Better hit rate (finding the optimal solution), 4) Better solution found more number of times within each problem type, 5) Better quality (in terms of objective function value) of product lines, and 6) faster computing times. An application of the BS heuristic is illustrated using a consumer durable product as example.

It is suggested that the BS heuristic be used to perform sensitivity analysis of the robustness of the final solution, such as the number of levels of each attribute to offer, the number of products to include in the product line etc. It is difficult to miss similarities between the BS and DP heuristics. Both heuristics work by combining attribute matrices and both use the same type of evaluation functions to select good aggregate columns. The difference is that the DP heuristic starts with one attribute matrix and adds one matrix from each of the remaining attribute sets in a sequential manner. On the other hand, the BS heuristic combines two matrices at a time at each level of the search tree. According to the authors, the BS heuristic produces superior solutions and faster run
times than the DP heuristic because the products chosen are complementary in the former case and not necessarily so in the latter case.

5.1.5 Ideal Point Heuristic

Easton and Pullman (2001) propose an ideal point heuristic to solve the seller’s welfare problem. The focus of their study is a generic service product. Using the example of a restaurant facility, service quality attributes such as reliability, responsiveness, assurance and empathy are explored in detail. The objective is to maximize the seller’s profits, not the market share of the product. The authors make the point that price and the technical features of a product are objective and easily understood by a customer. But, customer perceptions about the service features are usually more subjective and difficult for the marketer to understand. The service features are generally as important as the technical features in terms of their impact on market share and profits. The authors point out two drawbacks of the models previously proposed for the seller’s welfare problem. The prior models assume that profits vary linearly with the cost of providing service, and that attractive product profiles are associated with high market shares. Both these assumptions are restrictive and often fail in providing us with good quality solutions to real-world problems.

To address the above drawbacks, the seller’s welfare problem is modeled as follows. The notation is used is defined below.

*Parameters:*

- \( T \) – Number of periods in the planning horizon,
- \( M_t \) - Estimated total market size for period \( t \),
- \( \beta_{ijk} \) - Respondent \( i \)'s partworth for attribute \( k \) at level \( j \),
\( V_x \) - Average unit variable cost associated with product profile \( x \),

\( \pi_x \) - Expected market share for product profile \( x \),

\( L_x \) - Labor expense associated with product profile \( x \), over the planning horizon,

\( F_x \) - Fixed costs associated with product profile \( x \), over the planning horizon,

*Decision Variables:*

\( X \) – The set of product profiles that are evaluated,

The model to maximize the seller’s profit is given as follows:

\[
\text{Max} \quad (P_x - V_x)\pi_x \sum_{i=1}^{T} M_i - L_x - F_x
\]  

(5.35)

The value of \( L_x \) is obtained by solving a separate problem. The value of \( \pi_x \) is computed using consumer utilities (partworths) and a suitable choice rule. The computational complexity of model (5.35) is NP-hard. The following ideal point heuristic is proposed to solve model (5.35).

The ideal point heuristic consists of three phases. In the first phase, conjoint analysis technique is used to obtain consumer’s partworths for all attribute-levels. In the second phase, the partworth data is combined with marketing and operational data to estimate market share and profit for each of the product profiles that are ranked by consumers. In the final phase, the expected profit from each product profile is computed. The seller’s utility for a product profile is set equal to the profit generated by it. The overall utility for a product profile is then decomposed to provide individual attribute-level partworths. The solution to the product design problem is obtained by identifying the attribute-level providing the highest profit for each attribute. A numerical example is given to explain the implementation details of the ideal point heuristic.
To test the performance of the ideal point heuristic, 54 test problems are simulated. Four factors form part of the experimental design. Factor 1 has (3, 6, 8) attributes each with three levels. Factor 2 has market shares based on BTL share of utility and share of first choice rules. Factor 3 has consumer preference of seeking quality, being price conscious, or being unstructured. Factor 4 has cost relationships of type linear, multiplicative, step and interaction between attribute level choice and service delivery cost. The ideal point heuristic is reported to find solutions within 98% of the optimal solution value. The performance of the ideal point heuristic is compared to the performance of (1) a methodology in which a sub-set of product profiles is generated based on their projected market share and which are then optimized to find the best solution, and (2) a methodology with linear cost assumption. The methodology in (1) is reported to provide solutions that are on average 59% of optimal and the methodology in (2) is reported to provide solutions that are on average 92% of optimal. Hence, the authors conclude that the ideal point heuristic is superior to the other two methods.

5.1.6 Nested Partitions Heuristic

Shi, Olafsson and Chen (2001) proposed the nested partitions (NP) heuristic to solve the single product design problem for the share-of-choice objective. In the same paper, they extend the NP heuristic to solve PLD problems with the share-of-choice objective. The NP/GA/GS heuristic (refer sub-section 3.3.5) is claimed by the authors to be the best for solving the single product design problem and they also recommend it for solving the PLD problem. To solve the PLD problem, the single product design problem is solved in a sequential manner using the NP/GA/GS heuristic, for as many times as there are products in the product line. As in the case of the BS heuristic proposed by Nair et al.
(1995), after the first product is selected, the subsequent products are solved over a reduced data set. That is, the customers who are satisfied by the first product will not be considered while determining the second product, and the customers who are satisfied by either the first or second products will not be considered while determining the third product, and so on.

The performance of the NP/GA/GS heuristic in solving the PLD problem is tested on the 7 test problems that were used to test the performance of the said heuristic for the single product design problem. The number of products in the product line is fixed at 2, 3, and 4. A total of 21 problems are solved with 10 replications for each problem. The NP/GA/GS heuristic and the BS heuristic are tested on these problems. With respect to solution quality, and former is shown to completely dominate the latter.

5.2 Share-of-choice Model with Price and Budget

Thakur et al. (2000) propose a new model and solution method for PLD. The model incorporates prices, budget constraints and minimum acceptable thresholds for the utility of a chosen level from each attribute with the share-of-choice objective function. The model overcomes two restrictions found in the earlier models discussed in section 5.1. In the first restriction, a consumer is only concerned with meeting his preferences with regard to purchasing a product, irrespective of price of the product and his budget. In the next restriction, a consumer is expected to accept a product comprising levels with low utility value that are compensated for by levels with high utility value. The proposed model overcomes the two restrictions and permits a study of the effect of price of a product on its market share. A simple example is given to illustrate the benefits of using the proposed model. Conjoint analysis technique is used for obtaining data pertaining to
consumer preferences and prices. The budget and threshold limits are gathered by asking each consumer a set of direct questions.

The following notation is defined.

Sets:

K – The set of attributes,

L_k - The set of levels of attribute k, where k ∈ K,

R - The set of respondents in the study,

M – The set of products from which the product line is chosen,

Parameters:

w_{ijk} - The partworth of respondent i for level j of attribute k, where i ∈ R, j ∈ L_k, and k ∈ K,

w_{ijk}\_k - The partworth of respondent i for level j\_k from attribute k of his/her status-quo product, where i ∈ R, j\_k ∈ L_k, and k ∈ K,

c_{ijk} - The partworth of level j from attribute k relative to the partworth of level j\_k from attribute k for respondent i, given as c_{ijk} = w_{ijk} - w_{ijk}\_k,

p_{jk} - The price charged by seller for level j of attribute k, where j ∈ L_k and k ∈ K,

T_{ik} - The minimum threshold of respondent i for attribute k, where i ∈ R and k ∈ K,

t_{ik} - The relative threshold of respondent i for attribute k, given as t_{ik} = T_{ik} - w_{ijk}\_k,

b_i - The budget of respondent i, where i ∈ R,

Q – A large positive number,

Decision Variables:
$x_{jkm} = 1$ if level $j$ of attribute $k$ is chosen for product $m$, 0 if not; where $j \in L_k$, $k \in K$ and $m \in M$,

$y_{im} = 1$ if respondent $i$ is a customer of product $m$, 0 if not; where $i \in R$ and $m \in M$,

$x_i = 1$ if respondent $i$ is a customer of one of the products in the product line, 0 if not; where $i \in R$.

The mathematical model for the share-of-choice problem is described below.

$$\text{Max} \quad \sum_{i \in R} x_i$$

Subject to:

$$\sum_{j \in L_k} x_{jkm} = 1, m \in M \text{ and } k \in K$$

$$\sum_{j \in L_k} \sum_{k \in K} c_{ijk} x_{jkm} + (1 - y_{im})Q \geq 0, i \in R \text{ and } m \in M$$

$$\sum_{j \in L_k} \sum_{k \in K} p_{jk} x_{jkm} - (1 - y_{im})Q \leq b_i, i \in R \text{ and } m \in M$$

$$\sum_{j \in L_k} c_{ijk} x_{jkm} + (1 - y_{im})Q \geq t_{ik}, i \in R, k \in K \text{ and } m \in M$$

$$\sum_{m \in M} y_{im} \geq x_i, i \in R$$

$$x_{jkm}, x_i, y_{im} = \{0, 1\}, i \in R, j \in L_k, k \in K \text{ and } m \in M$$

The model formulation in (5.36) – (5.42) ensures that a consumer will choose a product from the product line if 1) the sum of relative preference values is positive, 2) the total price of the product does not exceed his or her budget (otherwise, the consumer will stick to his or her status quo product), and 3) the minimum threshold for each attribute is exceeded. The objective function (5.36) maximizes the number of respondents who prefer at least one of the products in the product line to their respective status quo products. The problem (5.36) – (5.42) is NP-hard. So, the following heuristic method is proposed.
The heuristic is based on the beam search (BS) method, Nair et al. (1995). The BS method is already explained in sub-section 5.1.4. Only a few changes are needed to solve problem (5.36) – (5.42). The BS heuristic is modified to satisfy the budget and threshold constraints for all products in the product line. The evaluation function is defined as the ratio of the number of positive entries in the combined (aggregate) column to the price of the combined (aggregate) column. The b columns with the highest such ratios are retained and carried over from one iteration to the next. The authors trace the various steps in the BS heuristic algorithm from start to finish through a trivial example. Also, a genetic algorithm (GA) based heuristic is presented to solve problem (5.36) – (5.42). The GA methodology is already described in sub-section 3.3.4 of chapter 3, and hence will not be discussed again.

To study the performance of the BS heuristic, a 3x3x3x3 experimental design, with (4,5,6) attributes, (2,3,4) levels per attribute, (50,100,150) respondents and (2,3,4) product line sizes as the design factors are used to generate test problems. The partworth values are generated from a uniform distribution between zero and one and then to quote the authors “normalized within individuals”. The threshold values were similarly generated. The prices were generated from a uniform distribution between 10 and 50. The budget for each respondent and the status quo products were also randomly generated. Ten replicates are solved for each problem (five in the case of larger problems) using the BS heuristic and a complete enumeration. A sub-set of the test problems with 100 respondents is solved using the GA heuristic. The BS heuristic is shown to perform better than the GA heuristic in the following performance measures. The measures are 1) Better average performance in approximating the optimal solution,
2) Lower standard deviation of solution values with respect to the optimal solution over all replicates of a problem, 3) Better hit rate (finding the optimal solution), 4) Better quality of product lines with respect to share-of-choice objective, and 5) faster solution times. A real world application of PLD is introduced and the PLD problem is then solved using the BS heuristic. The BS heuristic approach is explained using an example.

5.3 Model for the product line and price selection problem

Chen and Hausman (2000) propose a model formulation of the optimal product line and price selection problem that uses choice-based conjoint (CBC) analysis data. The authors show that the mathematical properties of the model help in solving realistic problem instances efficiently. The CBC method involves aggregating individual-level conjoint data and estimating consumer choice in terms of a multinomial logit model. CBC models are not expected to perform well in predicting share-of-choice at the individual level. The two main reasons are the aggregation of individual-level conjoint data and the property of independence from irrelevant alternatives (IIA). The IIA property is already discussed in sub-section 1.4.2 of chapter 1. The authors suggest that their model should only be used in situations when all the attribute sets are monotonic.

The objective of the product line and price selection problem is to maximize the expected total profit. The model assumes the several conditions that are given below. 1) A set of unique candidate product profiles is available, 2) the total population size of the potential market is available, 3) a potential customer is expected to purchase at most one unit of a particular product from the product line, 4) the purchasing probabilities of customers in the market are independent and identically distributed across customers, 5)
the production capacity is unlimited, and 6) there are no fixed costs associated with introducing new products.

The notation used in the model is defined below.

*Indices:*

\[ i = \text{Index for candidate products } 1, 2, \ldots, m, \]
\[ j = \text{Index for attribute levels of candidate products } 1, 2, \ldots, n, \]
\[ r = \text{Index for discretized price levels } 1, 2, \ldots, r, \]

*Decision Variables:*

\[ x_{ir} = 1 \text{ when candidate product } i \text{ is offered at price } s_{irs} \text{ in the product line, } 0 \text{ otherwise,} \]

*Parameters:*

\[ z_{ij} = \text{Utility value of attribute level } j \text{ of product } i, \]
\[ c_i = \text{Unit variable cost of candidate product } i, \]
\[ s_{ir} = \text{Price of product } i \text{ at discretized level } r, \]
\[ \nu_i = \text{Customer’s scaled expected utility for product } i; \text{ a function of } z_{ij} \text{ and } c_i, \]
\[ N = \text{Total population size of potential customers,} \]
\[ C = \text{A constant defined as } \sum_{i \in Q} e^{\nu_i}, \text{ where the set } Q \text{ consists of all competing products,} \]
\[ X_L = \text{Minimum number of products to be introduced in the product line, } X_L \geq 1, \]
\[ X_U = \text{Maximum number of products to be introduced in the product line, } X_U \leq m, \]
\[ e_k = [1 \ldots 1]_{1 \times k}, \text{ A vector of ones,} \]
\[ m_i = \text{Total number of discretized prices for product } i, \]
\[ \hat{m} = \sum_{i=1}^{m} m_i \]
The model formulation is a non-linear binary integer program as shown below.

\[
x_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T, \quad \bar{x} = [x_1, x_2, \ldots, x_m]^T;
\]

\[
u_i = [e^{v_{i1}}, e^{v_{i2}}, \ldots, e^{v_{im}}]^T, \quad \bar{u} = [u_1, u_2, \ldots, u_m]^T;
\]

\[
k = [k_1, k_2, \ldots, k_m]^T, \quad k_i = [Nu_{i1}(s_{i1} - c_i), Nu_{i2}(s_{i2} - c_i), \ldots, Nu_{im}(s_{im} - c_i)]^T;
\]

The model formulation is a non-linear binary integer program as shown below.

\[
\text{Max } \frac{k^T \bar{x}}{\bar{u}^T \bar{x} + C} \quad (5.43)
\]

subject to:

\[
X_L \leq e_{m}^T \bar{x} \leq X_U \quad (5.44)
\]

\[
e_{m_i}^T x_i \leq 1, \text{ for all } i \quad (5.45)
\]

\[
x_{ir} \in \{0, 1\} \text{ for all } i, r \quad (5.46)
\]

The constraint set (5.46) is relaxed as $0 \leq x_{ir} \leq 1, \text{ for all } i, r \ (5.47)$. The revised model is

\[
\text{Max } \frac{k^T \bar{x}}{\bar{u}^T \bar{x} + C} \quad (5.43)
\]

subject to:

\[
X_L \leq e_{m}^T \bar{x} \leq X_U \quad (5.44)
\]

\[
e_{m_i}^T x_i \leq 1, \text{ for all } i \quad (5.45)
\]

\[
0 \leq x_{ir} \leq 1, \text{ for all } i, r \quad (5.47)
\]

The model (5.42) – (5.46) includes large numbers of binary integer variables and constraints, and solving it to optimality is difficult. The authors show that the relaxed model (5.42) – (5.47) has unique mathematical properties and hence an optimal solution is obtained easily. To solve the relaxed model, standard non-linear programming solvers could be employed. The authors note that their model should not be used to predict individual purchase decisions. Rather, the model should be used to predict market share for products offered in a product line.
5.4 Models for positioning and pricing a product line

Dobson and Kalish (1988) propose a model to position and price a product line so that profit is maximized. Several assumptions are made in the model, which are listed below.

1) The seller is a monopolist and there is no competitive response. 2) There is no uncertainty in the preference of consumers. 3) Product performance is not an issue. 4) The market is composed of many customer segments of various sizes and the consumers within a segment are homogeneous. The sizes and preferences of the segments, and the cost data are available to the firm. 5) The firm faces both fixed and variable costs. 6) As in Green and Krieger (1985), a reference set of products is assumed to be available from which the product line is composed. 7) Utility is measured in dollar terms for each product. Consumers select a single product that provides the maximum value for money. That is, consumers choose the product that maximizes the difference between the value of the product and the actual price of the product.

The input data for the model requires three types of information. 1) How many segments are in the market and what kind of products do these segments desire? 2) How much will it cost to produce each type of product? 3) What set of products are currently offered by the competitors and what new products are they planning to introduce in the near future? A conjoint analysis study involving a representative sample of consumers should answer question 1). To answer question 2), basic cost information can be obtained from within the firm. Using the basic costs, the cost of various product designs can be estimated. To avoid answering question 3), no competitor products are assumed in the model. The authors feel that in real world situations, complete information on the input data is generally not available. In such situations, firms introduce one or two
products, collect more information, introduce a few more products, collect further information and carry on in a sequential manner until a desired number of products have been introduced. In the model described next, all products in the product line are introduced at the outset. The output of the model lists the product line, their prices and the market segments served by each product. We first define the following notation.

**Sets:**

I – The set of segments,

J – The set of reference products (that includes a dummy product \( y_0 \)),

**Parameters:**

\( u_{ij} \) - The reservation price of segment \( i \) for product \( j \),

\( q_i \) - The number of individuals represented by segment \( i \),

\( c_j \) - The unit cost to produce product \( j \),

\( f_j \) - The fixed cost incurred if product \( j \) is offered,

**Decision Variables:**

\( p_j \) - The price of product \( j \),

\( y_j = 1 \) if product \( j \) is offered, 0 otherwise,

\( x_{ij} = 1 \) if segment \( i \) is assigned to product \( j \), 0 otherwise.

\[
\text{Max } \sum_{i \in I} \sum_{j \in J} q_i (p_j - c_j)x_{ij} - \sum_{j \in J} f_j y_j
\]

Subject to:

\[
\sum_{j \in J} x_{ij} = 1, \ i \in I
\]
\[ x_{ij} \leq y_j, i \in I \text{ and } j \in J \quad (5.50) \]
\[ \sum_{k \in J} (u_{ik} - p_k)x_{ik} \geq (u_{ij} - p_j)y_j, i \in I \text{ and } j \in J \quad (5.51) \]
\[ p_0 = 0 \quad (5.52) \]
\[ x_{ij}, y_j = \{0,1\}, i \in I \text{ and } j \in J \quad (5.53) \]

We refer to the problem defined by (5.48) – (5.53) as P. P is a mixed integer non-linear problem. Objective function (5.48) maximizes the net profit. Constraint set (5.49) ensures that each consumer segment is assigned a single product. Constraint set (5.50) ensures that fixed cost is deducted in the objective function for each product offered. Constraint set (5.51) ensures that each consumer segment chooses the product that provides it with the maximum utility when compared to all other products. Constraint (5.52) handles the special case when a segment is not assigned a product at all. The dummy product \( y_0 \) is an element of the set J, has cost \( c_0 = 0 \), fixed cost \( f_0 = 0 \) and reservation price \( u_{i0} = 0, i \in I \).

The \( u_{ij} \) represents the reservation price that segment i has for product j. In other words, \( u_{ij} \) represents the value (in monetary terms) that segment i has for product j. The objective function in problem P can be modified to transform problem P to the maximize buyer’s welfare problem. To effect this transformation, the \( p_j \) variables in objective function (5.48) need to be replaced with \( u_{ij} \) variables. The objective function for the buyer’s welfare problem is: \[ \text{Max} \sum_{i=1}^{\infty} \sum_{j \in J} q_{ij} (u_{ij} - c_j)x_{ij} - \sum_{j \in J} f_j y_j \quad - (5.54). \]

The buyer’s welfare problem and problem P are both NP-hard. The following heuristic is suggested to solve problem P.
The heuristic decomposes problem P into an assignment problem and a pricing sub-problem. In the assignment problem segments are assigned to products and in the pricing sub-problem, the prices of the products are determined so that the segment assignments to products made earlier are satisfied. A detailed explanation of the decomposition scheme can be found in Dobson and Kalish (1988). We summarize the decomposition approach without going into the mathematical details. In the first phase, the assignment of segments to products is made using a suitable heuristic from among a few that are suggested. The second phase is to solve the pricing sub-problem. The pricing sub-problem (a linear program) is shown to have a special form, the dual of which can be decomposed into a set of shortest path problems on a network. Since the shortest path problem can be solved efficiently, it means that given a feasible assignment of segments to products, the pricing sub-problem can also be solved efficiently. In an iteration of the heuristic, the assignment and pricing problems are solved one after the other. The heuristic moves to the next iteration by modifying the assignment of segments to products so that higher prices can be charged for certain products, thereby improving profit. The objective of the reassignment of segments among products is to prevent high-paying segments from switching to low-profit products. The heuristic terminates when no improvement in profit can be found between two successive iterations.

In order to evaluate the heuristic algorithm, many small problem instances are randomly generated. The problems are of three types. The first type assumes that all the segments have similar preferences when choosing a product. The second type assumes that each segment has an ideal product and that reservation prices decrease with deviations from the ideal point. The third type assumes an unstructured market where
each segment has a uniform random distribution of preferences for the products. Each of
the three types of test problems considers five segments and four products. The variable
and fixed costs are randomly generated. Test problems are solved for two cases. In the
first case fixed costs are ignored, and in the second case fixed costs are considered. For
each problem type, 40 randomly generated test problems are created. Hence, a total of
240 test problems are solved.

The test results reveal that 1) the heuristic reports profits that are close to the
optimal solution, and 2) the heuristic reports better profits than randomly generated
solutions in more than 90% of the test problems. While summarizing their work, the
authors suggest the following areas for future research. 1) Continue work in the area of
dollar metric utility measurement, 2) Validate the model with actual behavior, 3)
Incorporate competition in the model, 4) Refine the heuristic algorithm and investigate
optimization opportunities, 5) Incorporate uncertainty about cost and demand in the
model, and 6) Study a real application using the model.

Dobson and Kalish (1993) propose an extension of the product line positioning
and pricing model discussed in Dobson and Kalish (1988). The model formulations for
the buyer’s welfare and the profit maximization problems remain unchanged. In the
present work, it is shown that the buyer’s welfare problem is equivalent to the
uncapacitated plant location (UPL) problem. This equivalence was also shown in Green

In order to solve the buyer’s welfare problem, heuristics for solving the UPL
problem can be suitably adapted and used. A bunch of heuristics (greedy, greedy
interchange, reverse greedy and reverse greedy interchange) is evaluated on a set of very
large problems. The simulation design considers three factors. Factor 1 is the three types of problems discussed in Dobson and Kalish (1988), factor 2 is (10, 20, 30, 80) products, and factor 3 is (20, 60, 200, 800) segments. All the four heuristics report solutions that are close to optimal. But, the greedy interchange heuristic consistently finds the best solution.

Two heuristics are suggested to solve the profit maximization problem. They are termed as the greedy (PG) and reverse greedy (PRG) heuristics. The PRG heuristic is proposed in Dobson and Kalish (1988). It begins with the buyer’s welfare solution and finds prices that maximize profit. Next, an iteration process is started. At each iteration, one segment is reassigned a product or removed from the current solution. The maximum profit prices are recomputed using the pricing sub-problem (the decomposition of problem P into an assignment problem and a pricing sub-problem was discussed earlier in this section). The segment chosen for reassignment is the one that results in maximum incremental profit. The PRG heuristic moves to the next iteration and continues this process until no further improvement in the profit objective is observed between two successive iterations. The second heuristic is called the PG heuristic and works as follows. We start with an empty set of products. At a given iteration of the PG heuristic, one additional product is added to the set currently offered. The product added is the one that provides the maximum incremental profit. The prices of all existing products are recomputed to maximize total profit. The PG heuristic moves to the next iteration and the process continues until termination occurs when adding a product reduces the overall profit.
The two profit maximization heuristics are evaluated on the set of test problems that were used to evaluate the buyer’s welfare heuristics. The PG heuristic outperforms the PRG heuristic with respect to solution quality and solution times. The solution is reported in the form of a ratio of the heuristic objective function value over the lowest upper bound (the optimal solutions are not found due to the large test problem sizes). While using the PG heuristic, the best solution ratio is 0.92 and the worst is 0.68. So, it appears that the heuristic solution may be far from the optimal solution. While the PG heuristic solves the biggest test problems in under a minute, the PRG heuristic could not solve the larger test problems even after several hours of computing time. On the basis of test results, the solution quality from using the PG heuristic (or the PRG heuristic) seems to improve as the problem size increases. The authors are not able to assign a reason for this behavior.

In order to provide managerial insights into the two objectives of maximizing buyer’s welfare and maximizing profit, three example problems are generated, one of each type that were discussed earlier in the section. The problems have 60 customers and 20 products each. The three problem instances are solved for each of the two objective functions and the solutions are illustrated on a graph. Several managerial issues are discussed through the use of these three examples.

The authors offer the following suggestions for using their models as a decision support system. 1) Consumer utilities can be obtained by doing a conjoint analysis study or by directly measuring dollar-based utilities, 2) Variable and fixed costs must be separated, 3) Fixed cost must be separated into a one-time sunk cost and a recurring cost. Examples of the former include research and development cost, cost of special
equipment, cost of training workers in the new technology etc., and examples of the latter include inventory holding cost, handling cost, setup cost etc., 4) Fixed costs for the existing products and new products are likely to be different and must be measured correctly, 5) Handling competitor products is a complex issue and must be done with care. However, by simulating ‘what if’ kind of scenarios, the decision support system can be tweaked to provide valuable insights to management.

The authors suggest the following directions for future research. 1) Develop a tighter upper bound for the PG heuristic used to solve the profit maximization problem, 2) analyze different competitive scenarios, 3) introduce cost and demand uncertainty, and a sequential product entry strategy, 4) introduce structure to costs and utilities, and 5) introduce probabilistic product choice rules.

Yoo and Ohta (1995) propose an optimal pricing and product-planning model based on conjoint analysis that maximizes profit. Consumers provide rank order data of preferences to a full profiles conjoint study. The reservation price of each consumer’s most preferred product is estimated from the conjoint data. An optimization model is presented along with an example of an automobile product (car). The solution procedure for solving the model is not provided in sufficient detail. Also, no experiment is designed to study the effectiveness of the solution procedure in solving real world sized test problems. Therefore, we do not summarize the model.

5.5 Model to maximize profit using manufacturing synergies

Morgan et al. (2001) propose a model to maximize profit of a product line while considering synergies in the manufacture of various products within the product line. It is relevant to quote the views of Morgan et al. (2001) in this regard. “Firms across
industries increasingly recognize the importance of providing variety to the marketplace while simultaneously identifying and exploiting synergies among products in manufacturing. The recent focus on product modularity observed in many industries, such as the computer industry, is driven in part by the resulting synergies in manufacturing”.

The following assumptions are made in the model. (1) Marketing managers identify diverse consumer preferences and develop marketing strategies to satisfy demand from differentiated consumer segments. The consumer utilities are generated using a conjoint analysis procedure. (2) The products within a product category share common manufacturing resources such as manpower, equipment and facility. (3) Products that are in the same manufacturing class have similar manufacturing requirements (synergy) such as parts, set-ups or systems. (4) Customers choose products with which they associate the highest utility. (5) Every product in the product line is produced in each of the production cycles over the planning horizon. Total manufacturing class set-up costs are minimized by producing products from the same manufacturing class consecutively. We next define the notation and present the product line design problem (PLDP) formulation.

Sets:

M – The set of consumer segments,

N – The set of products considered by the firm,

N’ - The union of set N with the set of all the competitor products,

L – The set of product families (manufacturing classes),

Parameters:

T – The planning time horizon,
\( d_i \) - The demand from consumer segment \( i \), where \( i \in M \),

\( u_{ij} \) - The utility of segment \( i \) for product \( j \), where \( i \in M \) and \( j \in N' \),

\( q_{jk} = 1 \) if product \( j \) is a member of manufacturing class \( k \), 0 otherwise; where \( j \in N \) and \( k \in L \),

\( p_j \) - The unit price charged for product \( j \), where \( j \in N \),

\( c_j \) - The unit variable production cost of product \( j \), where \( j \in N \),

\( h_j \) - The unit inventory holding cost over the planning horizon, where \( j \in N \),

\( S_k \) - The manufacturing class set-up cost incurred if any product \( j \) for which \( q_{jk} = 1 \) is assigned to the product line, where \( j \in N \) and \( k \in L \),

\( s_j \) - The individual product set-up cost, where \( j \in N \),

**Decision Variables:**

\( y_j = 1 \) if product \( j \) is offered, 0 otherwise; where \( j \in N \),

\( x_{ij} = 1 \) if product \( j \) is chosen by segment \( i \), 0 otherwise; where \( i \in M \) and \( j \in N' \),

\( T \) - The number of production cycles in the planning horizon,

\( r_k = 1 \) if class \( k \) set-up cost will be incurred, 0 otherwise; where \( k \in L \).

\[
\text{Max} \quad \sum_{j \in N} (p_j - c_j) \sum_{i \in M} x_{ij} d_i - \left( \frac{1}{2T} \right) \sum_{j \in N} \sum_{i \in M} x_{ij} d_i h_j - T \sum_{k \in L} r_k S_k - T \sum_{j \in N} y_j s_j \\
\text{Subject to:} \\
\sum_{j \in N'} x_{ij} = 1, \quad i \in M \quad (5.56) \\
\sum_{i \in M} x_{ij} \leq |M| y_j, \quad j \in N \quad (5.57)
\]
The objective function (5.55) maximizes the net profit from the product line.

Constraint set (5.56) ensures that each consumer segment chooses exactly one product. Constraint set (5.57) ensures that only products included in the product line satisfy demands from any consumer segment. Constraint set (5.58) ensures that each consumer segment chooses its most desirable product from those available. Constraint set (5.59) ensures that common manufacturing set-up cost is incurred for a family of products if any member of the product family is included in the product line. Constraint set (5.60) ensures non-negativity of production frequency variable $T$, and also enforces binary integer restriction on the rest of the variables in the PLDP model. The PLDP problem is shown to be NP-hard. The authors solve the PLDP model to optimality using implicit enumeration for problem instances with up to 20 products and 40 consumer segments. The authors suggest that heuristics should be devised to solve larger problems.

The authors provide a small example to explain the input requirements for the model and then illustrate the impact of the various manufacturing cost implications on the composition of the optimal product line. The impact of manufacturing cost and synergies on the product line characteristics such as its breadth (PLB), profitability (PLP), manufacturing class breadth (MCB) and target market share (TMS) are studied by designing an experiment. Five factors are chosen in the experimental design. Factor 1 is the number of candidate products, set at values 10, 15 and 20. Factor 2 is the number of consumer segments, set at values 10, 20 and 40. Factor 3 is the number of manufacturing
classes, set at values 2, 5 and 8. Factors 4 and 5 are multipliers that are used to calculate the unit holding cost and the total product set-up cost respectively. Factors 4 and 5 are both set at values 1 and 3. For each of the 324 problem settings that are possible, 10 replications were generated, yielding a total of 3240 test problems. The unit profit margin of each product was drawn from a uniform distribution on the interval [10,20]. Utility values were also randomly generated, but the details are not given. Products were randomly assigned to each of the manufacturing classes.

Regression analysis was used to examine the main effects of the five experimental factors on the four dependent variables PLP, PLB, MCB and TMS. We present some of the managerial insights that were reported.

1. In case of PLP, the holding and set-up costs were found to have the biggest impact on profits, with higher cost manufacturing environment leading to lesser profits. As the number of candidate products (from which the product line is chosen) increases, profits go up. Also, as the degree of synergy among products in each manufacturing class increases, the profits go up.

2. In case of PLB, it is found that higher holding and set-up costs limit PLB since the cost of each additional product increases total cost. Higher level of synergies across products leads to increase in PLB. As the number of products (from which the product line is chosen) increases, the PLB increases slightly since more cost effective combinations of products end up in the product line. As the number of consumer segments increase, PLB increases and a more diverse mix of consumer segments can be served.
3. In case of MCB, as the level of synergy across products in each manufacturing class increases, the optimal product line tends to belong to fewer manufacturing classes. That is, MCB decreases so that the optimal product line can exploit those synergies.

4. The results for TMS were found to be similar to the results for PLP.

The authors also compare the PLDP model with two alternative models that either consider or do not consider fixed costs respectively. The main effects of the five experimental factors on the four dependent variables PLP, PLB, MCB and TMS obtained from the PLDP model and from both the alternative models are compared. The PLDP model is shown to be better than the two other alternative models.

The computational results and analysis support the theory that profitability is improved when synergies among products are exploited in the PLD problem, particularly when the synergy levels are high and the manufacturing costs are also high. The following extensions of the PLDP model are suggested. (1) Capacity constraints and lot-sizing issues may be included in a new model. (2) The PLDP model is a single period model that may be extended to a multi-period model. (3) The PLDP model may be modified to include uncertainty of demand for each consumer segment.

The authors make the following concluding statement. “By integrating marketing inputs with more detailed manufacturing cost information, including manufacturing synergies attained through coordinated product design efforts, the model captures the trade-off between the benefits derived by providing variety to the marketplace, and the cost savings that can be realized by selecting a mix of products that can be produced efficiently within a firm’s manufacturing environment”.

179
A product platform is defined as a design, technology, or set of subsystems and interfaces shared by one or more product families, Moore et al. (1999). The authors show how conjoint analysis can be used to help design product platforms by bringing demand-side forecasting methods and supply-side cost estimates together. Product platforms serve as the foundation for multiple derivative products. The advantages of product platforms are (1) They save engineering, manufacturing and purchasing costs, (2) They minimize manufacturing complexity while retaining product variety in the market, and (3) The speed to market the derivative products is enhanced.

We summarize the chapter as follows. In the first section, we defined the buyer’s welfare, seller’s welfare and share-of-choice problems. Six different approaches proposed in the literature to solve one or more of these three problem types are discussed. Some of these models take a two-step approach to PLD, while the other models build a model directly using partworths data. All these models are solved using heuristics that do not guarantee an optimal solution. In the second section, a PLD model that explicitly considers price and budget constraints is described. The model is reported to be difficult to solve to optimality and hence is solved using a heuristic. In the third section, a model to simultaneously determine the product line and its price to maximize overall profit is described. The model uses input data obtained from the CBC method and is easily solved because the model has some special properties. In the fourth section, a positioning and pricing model is described. The model is solved to determine the physical attributes of the products chosen and the selling prices of the chosen products. In the fifth and final section, a model that maximizes profits while considering manufacturing synergies is
discussed. This model seeks to coordinate the marketing and operations functions within a firm to optimize profits.

We have presented a wide variety of PLD models in this chapter. In the next chapter, we discuss extensions of the exact algorithm introduced in chapter 4 to handle different types of additional constraints. We also discuss the potential for using the exact algorithm to solve the PLD problem using some of the ideas presented in this chapter.
Chapter 6.0: Extensions of the exact algorithm

In this chapter we discuss extensions of the exact algorithm proposed in chapter 4. The exact algorithm is designed to solve the share-of-choice problem for single product design. The share-of-choice problem formulation may be modified to include additional constraints and a weighted objective function, as suggested in Kohli and Krishnamurti (1987, 1989), and Nair et al (1995). To solve the modified share-of-choice problem, the exact algorithm needs some modification that will be explained in section 6.1.

Several approaches to solve the PLD problem were discussed in chapter 5. Green and Krieger (1985) and Mcbride and Zufryden (1988) suggested a two-step approach to solve the PLD problem. In the first step, a reference set of products is created and in the second step, the product line is chosen from the reference set. The exact algorithm can also be used to generate a set of good product designs that serve as a reference set of products. To solve the PLD problem, Nair et al (1995), and Shi et al (2001) solved the single product design problem sequentially, for as many times as the desired number of products in a product line. In section 6.2, we discuss the uses of the exact algorithm both in obtaining a set of reference products and in solving the PLD problem.

6.1 Extensions of the exact algorithm for single product design

The extensions of the exact algorithm involve the following factors.

(a) To consider a weighted objective function,

(b) To prevent infeasible product profiles from appearing in the optimal solution,

(c) To consider consumer budget constraints, and

(d) To prevent unacceptable attribute-levels from being chosen by a consumer even if the total utility from the end product exceeds his or her status quo utility.
The following researchers have suggested the above extensions. Kohli and Krishnamurti (1987, 1989) suggest that (a) respondents may be assigned weights based on their purchase frequency and/or purchase volumes, and that (b) based on factors such as consumer preferences, technological feasibility, manufacturing costs etc., all the infeasible product profiles could be identified and prevented from appearing in the final solution. Thakur, Nair, Wen and Tarasewich (2000) assume that (c) a consumer will purchase a product only if its price is within his or her budget, and that (d) a consumer will not accept products that comprise one or more unacceptable levels of attributes.

Consider the extension suggested in (a). The objective function of the share-of-choice problem formulated in section 4.1 of chapter 4 is to be modified. Weights are assigned to all the y-variables. The relevant notation from chapter 4 is repeated, with the definition of new notation.

**Sets:**

R - The set of respondents = \{1,2,3,...,p\}

A - The set of attributes = \{1,2,3,...,q\}

L - The set of all the attribute-levels = \{1,2,3,...,r\}

L_k - The set of levels belonging to attribute k, k ∈ A

**Parameters:**

μ_{ij} - The partworth of respondent i for level j, where i ∈ R and j ∈ L

h_i - The hurdle value for respondent i, where i ∈ R

a_i - The weight of respondent i, where i ∈ R

**Decision Variables:**
\( x_j = 1 \) if level \( j \) is chosen, 0 if not, where \( j \in L \)

\( y_i = 1 \) if the hurdle level of respondent \( i \) is exceeded, 0 if not, where \( i \in R \)

We note that the relationship between the sets \( L_k, k \in A \), and the set \( L \) is as defined in section 4.1 of chapter 4. The formulation of the modified share-of-choice problem, with a weighted objective function is stated below.

\[
\text{Max } \sum_{i \in R} a_i y_i \tag{6.1}
\]

Subject to:

\[
\sum_{j \in L} \mu_{ij} x_j - h_i y_i \geq 0, \quad i \in R \tag{6.2}
\]

\[
\sum_{j \in L_k} x_j = 1, \quad k \in A \tag{6.3}
\]

\( x_j = (0,1), \quad j \in L \tag{6.4} \)

\( y_i = (0,1), \quad i \in R \tag{6.5} \)

The extensions (b), (c) and (d) are not directly considered in the above model formulation. But, during the traversal of the search tree, the exact algorithm uses constraints derived from these three extensions to prune the search tree. When we use the exact algorithm to solve model (6.1) – (6.5), it is ensured that extensions (b), (c) and (d) are satisfied whenever we compute the objective function value of a partial solution (for bounding purposes), or a complete solution (for obtaining a better incumbent solution). Next, we elaborate on how constraints derived from extensions (b), (c) and (d) are enforced in the exact algorithm.

In extension (b), infeasible product profiles are to be prevented from appearing in the final solution. We assume that the data on infeasible product profiles is available in one form or the other. Every time we obtain a feasible and superior solution, for example, during Phase II (branch-and-bound stage) of the exact algorithm, we run
through a list of the given infeasible product profiles and determine if the current solution is acceptable or not.

In extension (c), a consumer can purchase a product only if its price is within his or her budget. We assume that the firm provides us with the prices of each attribute-level, and that the budget limits can be gathered from each consumer. Consider the following notation, some of which are redefined from section 4.6 of chapter 4.

\[ p_{jk} \] - The price charged by the firm for level \( j \) of attribute \( k \), where \( j \in L_k \), \( k \in A \) and \( p_{jk} \geq 0 \),

\[ B_i \] - The budget of respondent \( i \), where \( i \in R \) and \( B_i \geq 0 \),

\[ \Psi \] - The set of attributes that are branched on when at a node in the search tree,

\[ \Omega \] - The sub-set of respondents from the set \( R \) whose hurdle values are not yet exceeded \(( i \mid b_i > 0 )\) when at a given node in the search tree, but the potential for exceeding the hurdle after further branching exists.

The constraints derived from this extension are implemented in the search tree as shown below. Every time we branch on a node during Phase II of the exact algorithm, we check if

\[ \sum_{k \in \Psi} \sum_{j \in L_k} p_{jk} x_j \leq B_i \]

for each constraint \( i \), where \( i \in \Omega \). The constraints \( i \), that violate the above test are recorded and not allowed to contribute to the objective function value computed at that node.

In extension (d), a consumer should not accept products comprising unacceptable attribute-levels. The unacceptable attribute-levels have utility values that are lower than the consumer’s threshold limit for that attribute. We assume that the threshold limits can be gathered from each consumer. Consider the notation given below.
$T_{ik}$ - The minimum threshold of respondent $i$ for attribute $k$, where $i \in R$ and $k \in K$.

The constraints derived from extension (d) are implemented in the same way as constraints derived from extension (c) are. Suppose we branch on a node, say variable $x_j : j' \in L_k, k \in A$, during Phase II of the exact algorithm. We check if $\mu_{ij'} \geq T_{ik}$ for each constraint $i$, where $i \in \Omega$. The constraints $i$, that violate the above test are recorded and not allowed to contribute to the objective function value computed at that node.

Next, we discuss the modifications to be made in the exact algorithm to solve model (6.1) – (6.5). Four modifications need to be made in the exact algorithm. They are 1) the rule for determining the branching order, 2) the greedy algorithm used to solve for a good incumbent solution, 3) the Lagrangian relaxation problem used to solve for a better incumbent solution, and 4) the rules for fathoming a node using the two logic tests.

At first, consider modification 1). The algorithm for determining the (original) branching order is described in sub-section 4.4.4 of chapter 4, with notation. The ‘a’ vector is defined earlier in this section. Only step 2 needs to be revised as shown below.

Step 2 - For all $R_i \in \omega$, identify $\beta_{ik} = \max_{k \in A, k \in A} (a_i \beta_{ik})$ and increment $S_{ik} = S_{ik} \cup i$.

Next, consider modification 2). The (original) greedy algorithm is described in sub-section 4.4.2 of chapter 4, with notation. Only steps 2 and 6 need to be modified, as shown below.

Step 2: Let $j'$ be such that $\sum_{i \in R, \omega} a_i * \min(\mu_{ij'}, b_i) = \max_{j \in L_k, k \in A, k \in \Psi} \left\{ \sum_{i \in R, \omega} a_i * \min(\mu_{ij}, b_i) \right\}$.

Step 6: Let $P$ represent the end product profile. We check product $P$ for infeasibility. If $P$ is found to be infeasible, then set $Z_{\text{greedy}} = Z_{\text{inc}} = 0$ and stop. Else, continue to the next
check. For all \(i \in \Omega\), check if \(\sum_{j \in P} p_{jk} > B_i\), and if true, remove element \(i\) from the set \(\Omega\).

Continue to the next check. For all \(i \in \Omega\), and for each \(j \in P\), check if \(\mu_{ij} \geq T_{ik}\), where \(k : j \in L_k\). If there is at least one violation, remove element \(i\) from the set \(\Omega\). Finally, set

\[Z_{inc} = Z_{greedy} = \sum_{i \in \Omega} a_i\] and stop.

Next, consider modification 3). A Lagrangian relaxation of the problem defined in (6.1) – (6.5) is formulated. There is only a slight change from the Lagrangian relaxation formulation in sub-section 4.4.3 of chapter 4, and we use the same notation. Let LSOC refer to the Lagrangian relaxed (modified) share-of-choice problem. After simplification, the maximization objective of LSOC \((\lambda_i, i \in R)\) is given as follows.

\[
\begin{align*}
\text{Max} & \quad \sum_{i \in R} (a_i - \lambda_i h_i) y_i + \sum_{j \in L} (\sum_{i \in R} \mu_{ij} \lambda_i) x_j \\
\text{s.t.} & \quad \sum_{j \in L_k} x_j = 1, \ k \in A \\
& \quad x_j = (0,1), \ j \in L \\
& \quad y_i = (0,1), \ i \in R 
\end{align*}
\]

(6.6) – (6.9)

The subgradient optimization procedure (SGO) is used to solve problem (6.6) – (6.9), as done in sub-section 4.4.3 of chapter 4. The only changes are in steps 2 and 4 of the SGO procedure. We state these two steps below, after the necessary modifications are made.

Step 2 – Set \(x_{j^*} = 1\), where \(j^*\) is such that \(\sum_{i \in R} \mu_{ij} \lambda_i = \max_{j \in L_k} \left(\sum_{i \in R} \mu_{ij} \lambda_i\right)\), \(k \in A\).

If \((a_i - \lambda_i h_i) > 0\) then, set \(y_i = 1\), else set \(y_i = 0\), where \(i \in R\). Set \(x_j = 0\), \(j \in L\).

Compute \(Z_{soc} = \sum_{i \in R} (a_i - \lambda_i h_i) y_i + \sum_{j \in L} (\sum_{i \in R} \mu_{ij} \lambda_i) x_j\). Go to Step 3.
Step 4 - Set $Z_{\text{int}} = 0$. For all $i \in R$, if $(\sum_{j \in L} \mu_{ij} x_j) - h_i \geq 0$ then, increment $Z_{\text{int}} = Z_{\text{int}} + a_i$.

Is $Z_{\text{int}} > Z_{\text{inc}}$? If false, go to step 5. Else, re-compute the value of $Z_{\text{int}}$ by subjecting it to constraints derived from extensions (b), (c) and (d), as explained below. Let $P$ represent a feasible product profile, obtained in the current iteration of the SGO procedure. We check product $P$ for infeasibility. If $P$ is found to be infeasible, then go to step 5. Else, continue to the next check. For all $i \in R$, check if $\sum_{j \in P} p_{i,j} > B_i$, and if true, add element $i$ to a set $\Omega$, that is initialized to an empty set at the beginning of each iteration. We continue to the next check. For all $i \in R$, check if $\mu_{ij} \geq T_k$, for each $j \in P$, where $k : j \in L_k$. If there is at least one violation for element $i$, add it to the set $\Omega$. We go to the last step and re-compute the value of $Z_{\text{int}}$. Reset $Z_{\text{int}} = Z_{\text{int}} - \sum_{i \in \Omega} a_i$. If $Z_{\text{int}} > Z_{\text{inc}}$ then update $Z_{\text{inc}} = Z_{\text{int}}$ and go to step 5.

Finally, consider modification 4). At first, the logic test I described in sub-section 4.5.1 of chapter 4 is modified. The following result from sub-section 4.5.1 is stated for ready reference. At the node under consideration, let $\tau = \{i \mid b_i \leq 0\}$, $i \in R$, $\omega = \{i \mid \max_{j \in L_k} (\mu_{ij}) < b_i\}$, $i \in R$ and $\rho = \{i \mid i \notin \tau \text{ and } i \notin \omega\}$, $i \in R$. The upper bound calculation at the node is modified to be equal to $\sum_{i \in \tau} a_i + \sum_{i \in \rho} a_i$. If $\sum_{i \in \tau} a_i + \sum_{i \in \rho} a_i \leq Z_{\text{inc}}$ then the node may be fathomed.

Next, the logic test II described in sub-section 4.5.2 of chapter 4 is modified. The following result from sub-section 4.5.2 is stated for ready reference. At the node under
consideration, let \( \omega = \left\{ i \mid \sum_{k \in A} \max_{j \in I_k} \left( u_{ij} \right) < d_i \right\}, \ i \in W \). The upper bound calculation at the node is modified as \( \sum_{i \in \mathcal{R}} a_i - \sum_{i \in \mathcal{W}} \min(a_k) \). Hence, if \( \sum_{i \in \mathcal{R}} a_i - \sum_{i \in \mathcal{W}} \min(a_k) \leq Z_{inc} \) then the node may be fathomed. To be able to fathom a node using logic test II, the best possible reduction in the objective function value using the aggregated constraints, must be greater than or equal to the difference between the maximum possible objective function value and the best incumbent solution value. That is, \( \sum_{i \in \mathcal{W}} \min(a_k) \geq \sum_{i \in \mathcal{R}} a_i - Z_{inc} \). Or else, \( \sum_{i \in \mathcal{R}} a_i - \sum_{i \in \mathcal{W}} \min(a_k) \) will always be greater than \( Z_{inc} \) and the fathoming test for logic test II will always fail. This ends the discussion on the four modifications to the exact algorithm.

We have not provided a modified pseudo-code for Phase II of the exact algorithm in this section. But, all the necessary details are given to be able to rewrite it. The quality of solutions obtained using the modified exact algorithm is expected to behave in the following manner (as compared to the original algorithm).

1. The quality of the incumbent solution obtained from using the greedy algorithm and the Lagrangian relaxed (modified) share-of-choice problem are expected to be poor. This is because only extension (a) is considered when solving the model for a solution. After the additional constraints from extensions (b), (c) and (d) are imposed on this solution, the objective function value of is likely to reduce considerably. It is worthwhile to explore alternative methods that yield a better incumbent solution.
2. In Phase II of the modified exact algorithm, more computational work is involved in obtaining an upper bound at each node, because additional constraints from extensions (b), (c) and (d) have to be considered. Also, the number of nodes explored is likely to be smaller because the additional constraints result in tighter bounds at each node.

3. Logic test I is expected to do a better job than logic test II in fathoming nodes in the search tree. Logic test I is based on the simple (original) constraints. If a simple constraint becomes infeasible at a node, its associated weight is deducted from the upper bound. But, the rule for logic test II is different as it is based on the aggregate constraints. We note that the objective of aggregating constraints is to obtain an aggregate constraint that is “tighter”, see sub-section 4.5.2 in chapter 4 for details. There may be many aggregate constraints that become infeasible at a node in the search tree, but the upper bound may still be too high to be able to fathom the node. For example, consider an aggregate constraint obtained by aggregating two or more simple constraints. Suppose this aggregate constraint is determined to be infeasible at a given node. The best reduction to the maximum possible objective function value is calculated as follows. The best reduction is equal to the minimum weight from amongst all the weights corresponding to the simple constraints comprising the aggregate constraint. If the minimum value is much smaller than the rest of the weights associated with the aggregate constraint, the upper bound is expected to be weak, and hence fathoming the node may be more difficult.
6.2 Extensions of the exact algorithm for product line design

This section is divided into two sub-sections that both discuss extensions of the exact algorithm for PLD. In sub-section 6.2.1, we discuss the potential use of the exact algorithm in obtaining a set of reference products that serves as input data in the second stage of the two-step PLD approach. In sub-section 6.2.2, we discuss the use of the exact algorithm in building a product line directly, by solving the single product design problem for as many times as the required number of products.

6.2.1 Generating a reference set of products

A reference set of products is defined as a set of good product designs, from which a subset of products may be chosen to represent a product line. In the PLD literature, Green and Krieger (1985, 1989), Mcbride and Zufryden (1988), Chen and Hausman (2000), Dobson and Kalish (1988, 1993), and Morgan et al (2001), recommend a two-step procedure to solve the PLD problem. In the first step, a reference set of products is generated, and in the second step, a product line is chosen from the reference set. Mcbride and Zufryden (1988) mention that the determination of a reference set of good product profiles is one of the important areas of future research. Green and Krieger (1989) suggest that a reference set of about sixty products is sufficient to provide adequate buyer heterogeneity. That is, there will be at least one product with a total utility close to within a few percentage points of each buyer’s highest utility product.

The exact algorithm can be used to generate a reference set of products. A few potential methods are described below. One or more of these methods can be used, although there is potential for methods that yield better results.
1. The exact algorithm is optimized over the set of respondents belonging to each buyer segment, resulting in one optimal product design for each segment. Suppose it is known that there are M segments. A set of M product profiles can be identified and added to the reference set of products. If more number of products is desired in the reference set, then the best 3 or 5 product profiles for each segment can be identified by sequentially solving the exact algorithm. The product profiles that are obtained from the previous runs can be prevented from appearing again in the current solution by making minor modifications to the exact algorithm.

2. Segments that are homogeneous with regard to buyer preferences (based on partworths data) can be identified using certain similarity measures. Suppose that N segments are identified. The exact algorithm is used to optimize over the set of respondents from each of these homogeneous market segments. As described in the previous method, a set of N or more product profiles can be identified and added to the reference set.

3. The exact algorithm is optimized over sets of respondents belonging to more than one segment. The product profiles are identified and added to the reference set. The product profiles in the reference set may consist of alternate optimal solutions that are only slightly different from each other. For example, two products may differ only with respect to the levels chosen for a single attribute. To reflect buyer heterogeneity, it is necessary to choose product profiles that are maximally different from one another, even if they have the same objective function value. An avenue for future
research is to develop methods that yield a set of reference products that are well
differentiated from one another.

We note that the best-in heuristic can be used to obtain a set of reference products,
as described in Green et al (2004). The best-in heuristic is summarized below. Consider
the set of respondents, \( k = \{1 \text{ to } K\} \). First, find the product profile that has maximal
utility for respondent 1, referred to as \( U(1) \). Next, consider respondent 2. If respondent
2’s utility for respondent (buyer) 1’s best product is within a user-supplied fraction \( \varepsilon \) of
\( U(2) \), then respondent 2’s best product is not added to the reference product list.
Otherwise, it is. When considering respondent 3, all the products in the reference product
list are checked to see if any are within \( \varepsilon \) of \( U(3) \) and the preceding rule is applied. The
heuristic proceeds through the entire list of respondents in this fashion.

Various values of \( \varepsilon \) can be tried depending on how large one wants the reference
set to be. Also, it is suggested that the above process be repeated several times through
randomized orderings of the respondents. It is also suggested that each respondent’s best
product utility be better that the status quo utility associated with the respondent’s most
preferred current product.

6.2.2 Building a product line using the exact algorithm

Kohli and Sukumar (1990), Nair et al (1995), and Shi et al (2001) advocate a single step
procedure for PLD, in which a product line is created directly using partworths data.
The single product design problem is also solved directly from partworths data. Hence,
Nair et al (1995), and Shi et al (2001), solve a single product design problem sequentially
to build a complete product line. Green and Krieger (1989) also discuss sequentially
obtained product lines, but no algorithmic details are provided.
As shown by Nair et al (1995), and Shi et al (2001), we can solve the single product design problem (using the exact algorithm) sequentially to build a product line. The sequential method is described as follows. Using the partworths data, solve the share-of-choice problem using the exact algorithm to find an optimal product. Identify the respondents for whom the utility from this product (the first product) exceeds their status-quo utilities, and then remove these respondents from further consideration when solving for the next (second) product. The next chosen product corresponds to the optimal solution found by using the exact algorithm to solve over the set of respondents not satisfied by the products that are already chosen. In other words, the respondents who are satisfied by the first product will not be considered while determining the second product, and the respondents who are satisfied by either the first or second products will not be considered while determining the third product, and so on. The above process is repeated until the required numbers of products are chosen for the product line. Note that the modified exact algorithm proposed in this chapter, can be used to solve single product design problems with objective functions that maximize either share-of-choice or seller’s return. Also, it is found that the buyer’s welfare problem can be solved to optimality quite easily, as explained in Shi et al (2001).

Nair et al (1995) point out the following advantage of using the above sequential method for solving the PLD problem. Each incremental product added to the product line is complementary to products already chosen. Therefore, the resulting product lines are expected to cater to the choice of the largest number of respondents. Also, the chance for product duplication within a product line is reduced. The sequential method has the
following disadvantage. An optimal solution cannot be guaranteed, although the solution obtained can serve as an incumbent solution in an optimization algorithm.

In chapter 5, we described more comprehensive models for PLD that were proposed in the literature. These models capture additional factors such as price of the products, budget of the respondents who belong to many market segments, manufacturing synergies among the products, etc. Hence, the sequential method that uses the exact algorithm to solve for a product line seems very simplistic. Yet, users may find some value in the sequential method. Finally, it is up to the user to choose the appropriate PLD model. In the next chapter, we discuss a simulation design that generates a set of test problems that mimic real world problem characteristics using an appropriate experimental design.
Chapter 7.0: Test Problem Generation and Experimental Design

The algorithms published in the literature for solving the product design (share-of-choice) problem have all used numerically simulated data. But, the partworth and hurdle data used in those test problems were determined in a very elementary manner and may not be realistic. Our objective is to perform simulations that generate realistic data for the share-of-choice problem. The simulation design that we use for generating test problems was proposed by Curry (2000) and is also described in Camm et al. (2005). This chapter summarizes the simulation design detailed in the aforementioned papers. The test problem generation is discussed in section 7.1 and the experimental design underlying the test problem generation is discussed in section 7.2.

7.1 Test problem generation

We have chosen an elaborate simulation method to generate partworth utilities and hurdles at the respondent level. The main functions of the simulator are given below as described in Camm et al. (2005).

1. To produce partworth utilities for any number of respondents to a conjoint experiment,
2. To permit latent segments to exist in the market (latent segments are groups of respondents with similar utility functions),
3. To control the degree of heterogeneity between segments,
4. To control the degree of homogeneity within segment, by segments,
5. To respect degrees-of-freedom constraints within attribute within respondent, and
6. To provide a parameterized model to create consistent status-quo products and corresponding hurdle utilities.
A master experimental design (discussed in the next section) program repeatedly calls the simulator to create partworths and hurdles for products, Camm et al. (2005). The simulated products are designed to have either 20 or 32 attributes. Within each of these attributes, a variety of combinations of 2, 3, 4, and 5 levels are used so that the total number of levels in a product is either 70 or 112. The numbers of attributes and levels of the products that are simulated are much larger than many reported commercial applications, as evident from the CA applications discussed in chapter 2. It is important to note that problems with larger numbers of attributes and levels are not easy to design even with the most efficient fractional designs, bridging designs, and Hierarchical Bayes methods available today (Allenby et al. 1995, Lenk et al. 1996).

A flow chart illustrating the simulation process is given in Camm et al. (2005). We briefly describe the simulation process as follows. In the first stage, the respondent’s partworths are modeled as random Beta-distributed deviations from the within group mean. In other words, the partworth for a given respondent is obtained as the mean group partworth (the respondent belongs to the group) plus the random deviation for that respondent. The within group partworth mean on a given attribute represents a ‘data seed’. The data seeds are set to maximize differences between segments during the simulation. The random deviations of partworths are modeled as Beta random variables to be able to restrict the partworth range, just as the partworths are estimated in actual practice. The partworths for attribute \( k \) have \( (L_k - 1) \) degrees of freedom, where \( L_k \) represents the number of levels in an attribute. The random deviations are constrained to sum to zero for each attribute and for each respondent. This is done by coordinating the family of Beta densities used on a given attribute so that the shape of each Beta
corresponds to the position of its level. For example, extreme levels are truncated while more moderate levels take on a smoother shape like bell-shaped, rectangular or U-shaped. Simulating partworths using the simulation method described above generates realistic partworths. The use of Beta distribution gives us the ability to model the shape of partworth probability distributions as desired.

In the next stage, we simulate the hurdle values for each respondent. The hurdle values simulated in the literature assume the existence of status-quo products for each respondent (consumer). But, consumers frequently use an imaginary ideal product while making conjoint evaluations. Such an ideal product may be made up of attribute levels or combinations of attribute levels that are not available in any existing product. To handle the above condition, our simulation provides a parameterized choice model for each respondent. When simulating hurdles for respondents, we control the size of the competitive set (500 or 1000 in the experimental design) and instantiate values on all attributes for every product member of this set. Since the size of the competitive set is quite large, it ensures that some products in the set are not currently offered in the market. Therefore, there exists a probability that some consumers are using an ideal product as the basis for evaluations in the conjoint task. The partworth values of the levels of an ideal product are determined through interpolation, according to each simulated respondent's set of partworth utilities.

Consumer choice of a product from members of the competitive set is simulated as follows. The existing products are ranked in descending order according to the simulated respondent's overall utility function. From this ranked order, the simulator controls the depth of search the respondent uses when choosing a product. Each
simulated respondent is associated with a reservation utility. Products with overall utility greater than a simulated respondent’s reservation utility are accepted into what is termed as the simulated respondent’s acceptable set. The sizes of the acceptable sets are kept as small as possible. Within the acceptable set, each simulated respondent randomly chooses one product (one draw from an equiprobable multinomial distribution). The overall utility of this choice is the simulated respondent’s hurdle utility. We wish to highlight the following two important features of the simulator, Camm et al. (2005).

1. The simulator does not assume either the existence of a finite number of products in the market nor consumer ownership of or desire for a particular existing product.

2. The simulator permits levels selected for the conjoint experiment to differ from those of products that actually exist in the market. For example, a continuous attribute such as percent cocoa content can be varied continuously in a product (such as a chocolate), but must be represented by a finite number of levels in a conjoint experiment.

7.2 Experimental Design

We use a balanced $2^5$ factorial design with three replications for each of the thirty-two experimental treatments. The design factors and their levels are as follows.

1. *Number of Respondents* (600 or 1200),

2. *Number of Levels* (70 or 112),

3. *Size of Competitive Set* (500 or 1000),

4. *Number of Segments* (2 or 4), and

5. *Relative Hurdle Difficulty* (low or high).
In the next chapter, several models will be discussed for analyzing the results data. The statistical analysis will show which factor main effects and interactions are significant. With regard to the experimental design and statistical analyses, we found the discussion on factorial designs in Berger and Maurer (2002) to be informative. We also found it useful to go over an introduction to one-way and two-way ANOVA tests in Levine et al. (2005).

For conducting the statistical analyses, the General Linear Model (GLM) Univariate procedure is used. We use the GLM procedure available in SPSS (2005) for analysis. The following information about the GLM Univariate procedure is obtained from the user’s manual of SPSS (2005).

1. The GLM Univariate procedure provides regression analysis and analysis of variance for one dependent variable by one or more factors and/or variables.

2. The GLM Univariate procedure allows us to model the value of a dependent scale variable based on its relationship to categorical and scale predictors. Using the GLM procedure, we can test null hypotheses about the effects of other variables on the means of various groupings of a single dependent variable.

3. The GLM Univariate procedure is based on the General Linear Model procedure, in which factors and covariates are assumed to have a linear relationship to the dependent variable.

4. We can investigate interactions between factors as well as the effects of individual factors, some of which may be random. The effects of covariates and covariate interactions with factors can also be included. For regression analysis, the independent variables are specified as covariates.
Type I sums of squares is chosen for the GLM Univariate Procedure. The Type I sum-of-squares method is also known as the hierarchical decomposition of the sum-of-squares method. Under our balanced design it is an orthogonal decomposition, and the sums of squares in the model add up to the total sum of squares. For the purposes of testing hypotheses concerning parameter estimates, the GLM Univariate procedure makes the following assumptions.

1. The values of errors are independent of each other and the variables in the model,
2. The variability of errors is constant across cells, and
3. The errors have a normal distribution with a mean of 0.

In this chapter we have discussed test problem generation and the experimental design. The next chapter will present the computational results of these test problems and statistical analysis of those results.
Chapter 8.0: Computational Results and Statistical Analysis

In this chapter we present the computational results of two real world problems and the set of simulated test problems. We then conduct several statistical analyses of the computational results and discuss the findings. Each problem is solved independently using the exact algorithm and a commercial solver. The commercial solver is ILOG’s Cplex, version 8.0, (ILOG, (2004)) optimization software operating within the modeling software AMPL®, version 7.0, Fourer et al. (2002). The exact algorithm uses a logic-based branch-and-bound method, while the Cplex 8.0 Solver uses a linear programming-based branch-and-bound method. Sections 8.1 and 8.2 present the computational results and statistical analysis for the real world problems and the set of simulated test problems respectively.

8.1 Results and statistical analysis of real world problems

In this section, we compare the performance of the exact algorithm and the Cplex 8.0 Solver independently on two real world problems, referred as problems A and B respectively. The two solution approaches are compared on the basis of solution time and solution quality. The model (4.1) - (4.5) described in chapter 4 is solved. A marketing research firm based in the Midwestern United States provided Problem A, while Professor Paul Green (Green, 1999) provided Problem B. Problems A and B are comparable to the size of real world problems discussed in the literature in terms of numbers of attributes, levels and respondents. All results were obtained using a Personal Computer with Pentium 4 processor (1.4 GHz) and 512 MB of RAM. The exact algorithm is coded in C. Table 8.1 shows computational results for Problems A and B.
We find that the CPlex 8.0 Solver was unable to prove optimality within the three hours limit given for each of the two problems. Yet, the quality of the best solution at 100% and 99.5% of optimal coverage for the two problems respectively is excellent. On the other hand, the exact algorithm found and verified the global optimal solution to Problem A in 1.4 seconds and to Problem B in 7.8 seconds. We also find that the greedy incumbent solution captures 95% and 99.5% of the optimal coverage, while the subgradient optimization solution captures 98.6% and 100% of the optimal coverage for the two problems A and B respectively. In Problem A, the greedy solution is improved by the subgradient optimization procedure, but the optimal solution was found later during the branch-and-bound stage. In problem B, the subgradient procedure again improves the greedy solution and this improved solution is proven to be optimal during the branch-and-bound stage.

The greedy algorithm and subgradient procedure both yield high quality solutions to problems A and B. This may be because the sample of respondents in each of the two problems is homogeneous with regard to their preferences. Solution time and solution quality of the greedy solution and the subgradient optimization solution are expected to deteriorate with heterogeneous preferences of the respondents and with increased numbers of attributes and levels. The solution times for the two problems are also quite small when using the exact algorithm. In the next section, stringent tests of the exact algorithm and the CPlex 8.0 Solver on simulated data are conducted by varying the five experimental design factor levels mentioned in the previous chapter.
8.2 Results and statistical analysis of simulated test problems

The solution times required by the exact algorithm to find provably optimal solutions for the set of test problems at each combination of design factor levels are provided in Table 8.2. The performance of components of the exact algorithm such as the greedy solution and the subgradient optimization solution are provided for each problem. The performance of CPLEX 8.0 Solver in solving each of these test problems within a time limit of three hours is also provided.

-----------------------------
Table 8.2 inserted here
-----------------------------

8.2.1 Performance of the CPLEX 8.0 Solver

We report the following performance statistics pertaining to the CPLEX 8.0 Solver. The optimal solution for 37 of the 96 test problems could not be found within the 3-hour time limit. Although the optimal solution was found for the remaining 59 problems, the optimality of only 54 of these could be proven within the 3-hour time limit. This means that for 42 problems, the 3-hour time limit was reached. As a measure of solution quality, the ratio of the CPLEX solution over the optimal solution was obtained for each problem. The average coverage (ratio) over all the 96 problems was found to be 98.02%, which is excellent. But convergence to an optimal solution is not guaranteed for many problems within 3 hours, as mentioned before. The coverage (ratio) corresponding to the worst performance of the CPLEX 8.0 Solver is found to be about 80%.

The CPLEX 8.0 Solver found the optimal solution faster than the exact algorithm for 25 problems. To find out if the time difference is significantly large, we produced the graph in Figure 8.1.
We find that for 11 problems the time difference is less than 1 second, for 6 problems the time difference is between 1 and 10 seconds, for 7 problems the time difference is between 10 and 50 seconds, and for only one problem the time difference is greater than 50 seconds. The highest time difference is 1001.5 seconds. The superior performance of the CPLEX 8.0 Solver on these problems is because the optimal solution for all these problems is very close to total coverage. This fact enables the branch-and-bound procedure to prune the search tree faster. The exact algorithm also has the same advantage, but the pruning of the search tree is done using only two simple logic-based rules. Whereas, a commercial solver like CPLEX 8.0 Solver can be expected to employ advanced (state of the art) methods to prune the search tree.

Since the CPLEX runtimes to find a provably optimal solution are not available for 42 problems, a standard statistical analysis is not performed. Instead, the following analysis is performed to study the impact of each factor on the quality of CPLEX coverage. At first, we choose a factor and identify all the problems with that factor at its low level. We compute the average of the CPLEX coverage for all such problems. Note that CPLEX coverage is computed as the ratio of CPLEX solution over the optimal solution. Next, we identify all the problems with the same factor at its high level. We compute the average of the CPLEX coverage for all such problems. The above procedure is then repeated for all other factors.

A comparison of the average CPLEX coverage at the low and high levels of a factor helps us understand the influence of that factor in determining the CPLEX solution quality.
(ignoring factor interaction effects). The graph of the average CPLEX coverage for each factor at its low and high levels is shown in Figure 8.2.

Figure 8.2 inserted here

We find that the factors Number of Respondents and Size of Competitive Set do not influence the average CPLEX coverage much. The factor Number of Levels seems to have a minor influence since the average CPLEX coverage is better when there are a larger number of levels. This improvement in coverage might be because more number of levels increases the number of ways (combinatorial) to satisfy a respondent’s hurdle utility. The factors Number of Segments and Relative Hurdle Difficulty also seem to have a minor influence since the average CPLEX coverage falls when either of the factor levels is at their high values. A higher Number of Segments causes the partworths to be more heterogeneous between respondents, and hence it is difficult to satisfy respondents with a single product design. A higher Relative Hurdle Difficulty directly causes higher hurdle utility values, and it is more difficult to satisfy respondents with a single product design.

8.2.2 Performance of the Exact Algorithm

In chapter 4, we stated that the exact algorithm explicitly or implicitly enumerates all the \( \prod_{k \in A} |L_k| \) possible solutions to the share-of-choice problem. This means that the runtime complexity of the exact algorithm is nonlinear (multiplicative). The various models that we use in this section to analyze the test results, assume a linear relationship between the dependent variable and a set of independent factor variables. Therefore, a log-transform is performed in each of the models by taking a natural logarithm of the values corresponding to the dependent variable. Besides the five experimental design factors,
two covariate variables *Optimal Proportional Coverage* and *Proportional Gap Between Greedy and Optimal Coverage* are collected for use in the models to be discussed next. For *Optimal Proportional Coverage*, we compute the ratio of the optimal solution over the sample size, expressed as a percentage. For *Proportional Gap Between Greedy and Optimal Coverage*, we compute the ratio of the difference between optimal and greedy solutions over the optimal solution, expressed as a percentage.

We first conduct analyses of Model 1 and Model 2, each of which has logarithm of solution time as the dependent variable. Model 1 uses only those factors known to the analyst prior to running the exact algorithm. That is, it attempts to estimate solution times from known problem characteristics. Factors such as the *Number of Respondents, Number of Levels, Number of Segments* and *Size of Competitive Set* are known to an analyst in a conjoint study, and hence are included in Model 1. But the factor *Relative Hurdle Difficulty* is determined by the simulation and is not known to the analyst prior to running the exact algorithm. Also, *Optimal Proportional Coverage* and *Proportional Gap Between Greedy and Optimal Coverage* are collected after the exact algorithm is run. Hence, *Relative Hurdle Difficulty* and the two covariates are not included in Model 1. Model 2 uses all factors controlled and/or recorded from the simulations. Hence, all five factors and the two covariates are included in Model 2. We note that Model 1 is nested within Model 2. In Model 1, all the factor main effects and all the factor-factor interactions are included. For Model 2, we have considered two different cases, which are referred as Models 2(a) and 2(b). In Model 2(a), we consider all factor and covariate main effects and all factor-factor interactions. In Model 2(b), we consider all factor and
covariate main effects, and all 2-way and 3-way interactions between factors and covariates.

The result of the GLM Univariate analysis of Model 1 is given in Table 8.3.

----------------------------
Table 8.3 inserted here
----------------------------

We find that Model 1 has $R^2 = .801$, adjusted $R^2 = .764$, $F = 21.523$ and p-value $< 0.001$. At the 95% confidence level, we conclude that there is significant relationship between the dependent variable and the four factors. The main effects for factors *Number of Respondents*, *Number of Levels* and *Number of Segments* are all highly significant. The interaction between the factors *Number of Levels* and *Number of Segments* is also highly significant.

The result of the GLM Univariate analysis of Model 2(a) is given in Table 8.4.

----------------------------
Table 8.4 inserted here
----------------------------

We find that Model 2(a) has $R^2 = .949$, adjusted $R^2 = .922$, $F = 35.228$ and p-value $< 0.001$. The $R^2$ value is higher than in Model 1 as expected, since we have included more explanatory variables in Model 2(a). At the 95% confidence level, we conclude that there is a significant relationship between the dependent variable and the set of factors and covariates. Except for the factor *Size of Competitive Set*, the rest of the four factors and the two covariates are highly significant. Many of the interactions are significant, but the highly significant interactions are between

1. *Number of Levels* and *Number of Segments*, and

2. *Relative Hurdle Difficulty*, *Number of Levels* and *Number of Segments*. 
The result of the GLM Univariate analysis of Model 2(b) is given in Table 8.5.

We find that Model 2(b) has $R^2 = .979$, adjusted $R^2 = .940$, $F = 25.466$ and p-value $< 0.001$. The $R^2$ value is higher than in Model 2(a). It may be because we have also included interactions between factors and covariates. At the 95% confidence level, we conclude that there is a significant relationship between the dependent variable and the set of factors and covariates. Except for the factor \textit{Size of Competitive Set}, the rest of the four factors and the two covariates are highly significant. Many of the interactions are significant, but the highly significant interactions are between

1. \textit{Number of Levels} and \textit{Number of Segments}, and
2. \textit{Number of Levels} and \textit{Optimal Proportional Coverage}.

As we have seen, each of the models does a good job of explaining the values taken by the dependent variable. Therefore, we focus on the main effects and their associated marginal means. The marginal means are computed as the ratio of solution time for the high level vs. the low level on each factor. The marginal means provide a measure of the relative impact on solution time of an easier problem (factor at low level) when compared to a more difficult problem (factor at high level), averaging over all other factors. The graph of the marginal means for each factor is shown in Figure 8.3.

These results suggest the existence of three groups of factors, with minor, moderate and major impact respectively on the solution time. The factor \textit{Size of
Competitive Set (marginal mean of 1.16) has a minor impact on solution time. Two of the factors have a moderate influence on solution time. They are Number of Respondents (3.88) and Relative Hurdle Difficulty (6.15). The two factors with a major impact are Number of Levels (102.75) and Number of Segments (269.55). For example, it takes about 3.88 times longer on average to solve problems with 1200 respondents (high level) than for problems with 600 respondents (low level). On the other hand, it takes 269 times longer on average to solve problems with 4 segments (high level) than with 2 segments (low level).

We now discuss the implications of these results. The factor Size of Competitive Set has a negligible (minor) impact on solution time. A consumer may process large amounts of information to compare products before making a purchase decision. The solution time for the exact algorithm is expected to increase by only a small margin. The factor Number of Respondents moderately impacts solution time. Using larger representative samples is not expected to appreciably slow the exact algorithm. The factor Relative Hurdle Difficulty also moderately impacts solution time. Some consumers have higher expectations (total utility) from a new product and hence have relatively higher hurdle difficulty than other consumers. The solution times are expected to be appreciably higher for consumers with high hurdle difficulty than for consumers with low hurdle difficulty.

The two factors with major impact on solution time are Number of Levels and Number of Segments. Many product categories consist of a high Number of Levels. The solution times are expected to be very much higher (about a hundred-fold increase) for products with high Number of Levels than for products with low Number of Levels. For
problems with 112 levels, averaging over all other design factors, solution times are still reasonable at about 46 minutes on our PC. The *Number of Segments* has the biggest impact on solution time. This factor accounts for the overall heterogeneity in utility functions among respondents in a study who may come from different market segments in a population. We find a 269-fold increase in solution times for problems with 4 segments than for those with 2 segments. In fact, the most difficult problems are those in which the factors *Relative Hurdle Difficulty*, *Number of Levels* and *Number of Segments* are all at their high levels. Among the eight problems with 112 levels and 4 segments, those with low *Relative Hurdle Difficulty* average about 25 minutes to solve while those with high *Relative Hurdle Difficulty* average about 2 hours and 37 minutes to solve. It is close to a 6.3:1 ratio, with the most difficult problem requiring about 7 ¾ hours.

A couple of relevant statistics are given below. The ratio of the average solution times for problems in which the two factors *Number of Levels* and *Number of Segments* are at their high levels, to the average solution times for problems in which the said factors are at their low levels, is found to be 534.56. The ratio of the average solution times for problems in which the three factors *Relative Hurdle Difficulty*, *Number of Levels* and *Number of Segments* are at their high levels, to the average solution times for problems in which the said factors are at their low levels, is found to be 23,523.65. These two results are supported by the results from models 1, 2(a) and 2(b).

Next, we perform the following statistical analyses using two models, referred as models 3 and 4 respectively. Model 3 is used to study how well the greedy solution approaches the optimal solution. Model 4 is used to study how problem characteristics (factor settings) affect the overall coverage of the optimal solution. For Model 3, the
dependent variable is the logarithm of percent coverage of the greedy solution relative to the optimal solution (Greedy Proportional Coverage). For Model 4, the dependent variable is the logarithm of percent coverage of the optimal solution relative to the total sample (Optimal Proportional Coverage). Both Models 3 and 4 use the five design factors and all their interactions as independent variables.

The result of the GLM Univariate analysis of Model 3 is given in Table 8.6.

```
Table 8.6 inserted here
```

We find that Model 3 has $R^2 = .918$, adjusted $R^2 = .879$, $F = 23.173$ and $p$-value $< 0.001$. At the 95% confidence level, we conclude that there is significant relationship between the dependent variable and the five factors. The main effects for factors Number of Levels, Number of Segments and Relative Hurdle Difficulty are all highly significant. Many of the interactions are significant, but the highly significant interactions are between

1. Number of Levels and Number of Segments,
2. Number of Levels and Relative Hurdle Difficulty,
3. Relative Hurdle Difficulty and Number of Segments,
4. Number of Levels, Number of Segments and Relative Hurdle Difficulty.

The result of the GLM Univariate analysis of Model 4 is given in Table 8.7.

```
Table 8.7 inserted here
```

We find that Model 4 has $R^2 = .987$, adjusted $R^2 = .981$, $F = 155.948$ and $p$-value $< 0.001$. At the 95% confidence level, we conclude that there is significant relationship
between the dependent variable and the five factors. The main effects for factors *Number of Levels*, *Size of Competitive Set*, *Number of Segments* and *Relative Hurdle Difficulty* are all highly significant. The variance explained by the factor *Size of Competitive Set* is quite small, and hence we ignore its significance. Many of the interactions are significant, but the highly significant interactions are between

1. *Number of Levels* and *Number of Segments*,
2. *Size of Competitive Set* and *Relative Hurdle Difficulty*,
3. *Size of Competitive Set* and *Number of Segments*,
4. *Number of Segments* and *Relative Hurdle Difficulty*,
5. *Size of Competitive Set*, *Number of Segments* and *Relative Hurdle Difficulty*.

Again, the variance explained by all the interaction terms involving the factor *Size of Competitive Set* are quite small, and hence we ignore their significance. We find that the results for Models 3 and 4 are quite similar. The significant impact of *Relative Hurdle Difficulty* is as expected. This is because we can manipulate coverage from zero (by using impossibly high hurdles) to the total sample (by using naively permissive hurdles). Thus, the factor *Relative Hurdle Difficulty* influences both the performance of the greedy algorithm and the maximum possible coverage in a sample. From Table 8.2, we find that for many problems the greedy solution matches the optimal solution, representing the best case. In the worst case, the greedy solution is as low as 50.13% of the optimal. In the best case, the optimal solution achieves 100% coverage in some problems. In the worst case, it achieved as low as 51.17% coverage of the sample. The significant impact of *Number of Segments* is also as expected. As the number of segments increases, coverage (greedy or optimal) will steadily decrease because no single
product profile can satisfy increasingly diverse viewpoints. It is not clear what the impact of factor *Number of Levels* would be.

The following analysis is performed to study the impact of each factor on *Greedy Proportional Coverage*. We compute the average *Greedy Proportional Coverage* for each factor at its low level and high level respectively. The average is computed over all problems in which the factor is fixed at a specific level. A comparison of the average *Greedy Proportional Coverage* at low and high levels of a factor helps us understand the influence of that factor in determining the quality of the greedy solution (ignoring factor interaction effects). The graph of the average *Greedy Proportional Coverage* for each factor is shown in Figure 8.4.

---

Figure 8.4 inserted here
---

We find that the factors *Number of Respondents* and *Size of Competitive Set* do not influence the *Greedy Proportional Coverage* much. The factor *Number of Levels* has a significant influence since the average *Greedy Proportional Coverage* is better when *Number of Levels* is at its high value. This improvement in coverage might be because more number of levels increases the number of ways (combinatorial) to satisfy a respondent’s hurdle utility. The factors *Number of Segments* and *Relative Hurdle Difficulty* also have a significant influence since the average *Greedy Proportional Coverage* falls substantially when either of the factor levels is at their high values.

We repeat the above analysis to study the impact of each factor on *Optimal Proportional Coverage*. As explained before, we compute the average *Optimal Proportional Coverage* for each factor at its low level and high level respectively. The
graph of the average *Optimal Proportional Coverage* for each factor is shown in Figure 8.5.

We find that the factors *Number of Respondents* and *Size of Competitive Set* do not influence the *Optimal Proportional Coverage* much. The factor *Number of Levels* has a significant influence since the average *Optimal Proportional Coverage* is better when *Number of Levels* is at its high value. The factors *Number of Segments* and *Relative Hurdle Difficulty* also have a significant influence since the average *Optimal Proportional Coverage* falls substantially when either of the factor levels is at their high values. The explanation for the above observations is the same as given earlier for *Greedy Proportional Coverage* results.

The exact algorithm uses a greedy heuristic followed by a subgradient optimization procedure to generate a good incumbent solution. Finally, the branch-and-bound phase of the exact algorithm finds the optimal solution (if not found already) and proves optimality of the solution. It would be interesting to know how often each of these solution methods generates the optimal solution. Figure 8.6 illustrates how often the greedy solution turns out to be optimal.

Whether the factors *Number of Respondents* and *Size of Competitive Set* are set at low or high levels, the greedy solutions are optimal roughly the same number of times. The greedy solutions are optimal significantly more number of times when the factors
Number of Levels, Number of Segments and Relative Hurdle Difficulty are set at low levels than when they are set at high levels. For instance, the factor Number of Segments seems to affect the performance of the greedy heuristic most. We find that none of the greedy solutions are optimal when Number of Segments is set at high level, while close to 73% of greedy solutions are optimal when Number of Segments is set at low level.

Figure 8.7 illustrates how often the subgradient optimization solution turns out to be optimal. We note that the subgradient optimization method is run after the greedy heuristic in the exact algorithm. Hence, we check the subgradient solution for optimality only if the greedy heuristic had failed to find the optimal solution.

The subgradient solutions are optimal significantly more number of times when the factors Number of Respondents and Size of Competitive Set are set at their low levels. This might be because lesser number of respondents means that the problems are of smaller size, and smaller size of competitive set means that the hurdle values are easier to satisfy. The subgradient solutions are optimal significantly more number of times when the factors Number of Levels, Number of Segments and Relative Hurdle Difficulty are set at high levels than when they are set at low levels. This unexpected result might be because the greedy heuristic has already found optimal solutions for a large number of problems when these three factors were at their low levels.

Figure 8.8 illustrates how often the optimal solution was found during the branch-and-bound phase of the exact algorithm. We wish to reiterate that it is the branch-and-bound phase of the exact algorithm that proves optimality for each problem.
For the factor *Number of Levels*, significantly more number of optimal solutions was found when at low level. This is likely because the subgradient optimization procedure already found a large number of optimal solutions when factor *Number of Levels* was at high level. Also, significantly more number of optimal solutions was found during the branch-and-bound phase when each of the remaining four factors was at their high levels. The significance of the factor *Number of Segments* needs to be highlighted. We find in the branch-and-bound phase that only about 10% of optimal solutions are found when *Number of Segments* is set at low level, while close to 71% of optimal solutions are found when *Number of Segments* is set at high level. This is because the greedy heuristic already found most of the optimal solutions for problems in which *Number of Segments* is at low level.

We report the following performance statistics pertaining to the greedy heuristic, the subgradient optimization procedure and the branch-and-bound phase of the exact algorithm. The greedy solution for 35 of the 96 test problems was found to be optimal. As a measure of solution quality, the coverage ratio of the greedy solution over the optimal solution was obtained for each problem. The average coverage ratio over all the 96 problems was found to be 93.92%. The subgradient optimization method found the optimal solution for 22 problems in which the greedy solution was sub-optimal. Further, the subgradient optimization method improved the greedy solution in 35 problems, yet the improved solution was not optimal. The branch-and-bound phase of the exact algorithm had to find the optimal solution for 39 problems and prove optimality for all 96
problems. Four problems out of the 96 test problems needed more than 3 hours for the exact algorithm to find a provably optimal solution.

We conclude this chapter with a summary. We presented the computational results for two real world problems and the set of 96 simulated test problems. We then performed statistical analyses on the computational results and discussed the statistical results. The performance of CPLEX 8.0 Solver was compared with that of the exact algorithm over all the solved problems. The next chapter is the final chapter of this dissertation, and we present the conclusions and future directions for our research.
Chapter 9.0: Conclusions and Potential Extensions

We reviewed the conjoint analysis methodology and underscored its importance in designing superior products. We also presented practical applications of conjoint analysis in the industry. Several heuristic procedures proposed in the literature to solve the share-of-choice problem were discussed. We then proposed an exact branch-and-bound algorithm to solve the share-of-choice problem. We have demonstrated that the exact algorithm can find provably optimal solutions within a reasonable amount of time on two real world problems and 96 carefully simulated problems (three replicates for each of thirty-two experimental treatments). The simulated problems have as many as 32 attributes (with up to 112 levels) and 1200 respondents. We have isolated certain problem characteristics to which the exact algorithm’s performance is most sensitive and the statistical analysis shows that the exact algorithm will effectively solve large problem instances to optimality.

We also showed that the exact algorithm is quite flexible in terms of dealing with a weighted objective function and several types of additional constraints. We discussed several models for product line design that appear in the literature. We showed how our exact algorithm could be used to find a reference set of products that serve as input data for the product line design problem. We also showed how the exact algorithm could be used to solve the product line design problem by sequentially solving the single product design problem.

In terms of future extensions of our research, efforts could be made to focus on modeling and solving single product design problems that include continuous attributes for which the optimal value may lie between levels included in the conjoint study design.
These problems should also explicitly take manufacturing costs into consideration. Modeling and solving product line design problem to optimality is another area where efforts need to be made.
References


Huber, J. (1997). What we have learned from 20 years of conjoint research: When to use self-explicated, graded pairs, full profiles or choice experiments, Fuqua School of Business, *Sawtooth Software Research Paper Series*.


Appendices (Tables & Figures)
### Table 1.1: A set of attributes and levels for designing a Personal Computer

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Access Memory (RAM)</td>
<td>128 MB&lt;br&gt;256 MB&lt;br&gt;512 MB&lt;br&gt;1 GB</td>
</tr>
<tr>
<td>Hard Disk Capacity</td>
<td>30 GB&lt;br&gt;60 GB&lt;br&gt;80 GB</td>
</tr>
<tr>
<td>Processor Make/Speed</td>
<td>Pentium IV 2.4 GHz&lt;br&gt;Pentium IV 3.2 GHz&lt;br&gt;Pentium IV 4.0 GHz</td>
</tr>
<tr>
<td>Monitor Size</td>
<td>17 &quot;&lt;br&gt;19 &quot;&lt;br&gt;21 &quot;</td>
</tr>
<tr>
<td>Flat Screen Monitor</td>
<td>Yes&lt;br&gt;No</td>
</tr>
<tr>
<td>DVD Option</td>
<td>Yes&lt;br&gt;No</td>
</tr>
</tbody>
</table>

Table 1.2: Sports car in the mid-price range (attributes and levels of conjoint stimuli), Source: Green et al. 2004.
Figure 1.1: Alternative approaches to measuring preference structures, Source Green and Srinivasan (1990).

How likely are you to purchase this computer? (Use a scale from 0 to 100, where 0 = "not at all likely" and 100 means "definitely would purchase.")

<table>
<thead>
<tr>
<th>IBM</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 GHz processor</td>
</tr>
<tr>
<td>512 MB RAM</td>
</tr>
<tr>
<td>17-inch monitor</td>
</tr>
<tr>
<td>$1,199</td>
</tr>
</tbody>
</table>

Your Score: __________

Table 1.3: Traditional Conjoint: Card-Sort Method (5 attributes)
Source – Pedagogical material from sawtoothsoftware.com
Using a 100-pt scale where 0 means definitely would NOT and 100 means definitely WOULD

How likely are you to purchase…

1997 Honda Accord
Automatic transmission
No antilock brakes
Driver and passenger airbag
Blue exterior/Black interior
50,000 mile warranty
Leather seats
optional trim package
3-year loan
5.9% APR financing
CD-player
No cruise control
Power windows/locks
Remote alarm system
$18,900

Your Answer:___________

Table 1.4: Traditional Conjoint: Card-Sort Method (15 attributes)
Source – Pedagogical material from sawtoothsoftware.com

Table 1.5: ACA Survey - Self-Explicated “Priors” Section,
Source – Pedagogical material from sawtoothsoftware.com
If these automobiles were identical in all other ways, which would you prefer?

<table>
<thead>
<tr>
<th>Two-Door Sedan</th>
<th>Four-Door Sedan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Made in USA</td>
<td>Made in Japan</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strongly Prefer Left</th>
<th>Somewhat Prefer Left</th>
<th>Indifferent</th>
<th>Somewhat Prefer Right</th>
<th>Strongly Prefer Right</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1.6: ACA Survey - Conjoint “Pairs” trade-offs (Preferences), Source – Pedagogical material from sawtoothsoftware.com

If two automobiles were acceptable in all other ways, how important would this difference be to you?

<table>
<thead>
<tr>
<th>Red —instead of— Black</th>
<th>Not At All Important</th>
<th>Somewhat Important</th>
<th>Very Important</th>
<th>Extremely Important</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1.7: ACA Survey - Conjoint “Pairs” trade-offs (Importances), Source – Pedagogical material from sawtoothsoftware.com
Table 1.8: ACA Survey – Calibration Concept to obtain purchase likelihood scores, Source – Pedagogical material from sawtoothsoftware.com

If you were in the market to purchase a PC today, and these were your only alternatives, which would you choose?

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dell</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 GHz Processor</td>
<td>3 GHz Processor</td>
<td>2 GHz Processor</td>
<td>None: If these were my only choices, I'd defer my purchase.</td>
</tr>
<tr>
<td>512 MB RAM</td>
<td>1 GB RAM</td>
<td>512 MB RAM</td>
<td></td>
</tr>
<tr>
<td>21-inch Monitor</td>
<td>17-inch Monitor</td>
<td>15-inch Monitor</td>
<td></td>
</tr>
<tr>
<td>$1,399</td>
<td>$1,199</td>
<td>$1,099</td>
<td></td>
</tr>
</tbody>
</table>

Table 1.9: Choice Based Conjoint Question, Source – Pedagogical material from sawtoothsoftware.com
AT&T’s first cellular telephone – Chicago based study of 1000 drivers’ reactions to cell phone features of the new “honey-comb” relay system

Ford Fairlane – conjoint analysis used in redesign of Ford to reflect general automotive downsizing objectives

IBM RISC 6000 workstation – conjoint used to measure potential buyer reactions to variations in performance and reliability features of a new workstation

Squibb’s captopril antihypertensive – six-country study of physicians’ evaluations of Capoten’s efficacy and safety features

Tagamet (SKF) and Zantac (Glaxo) ulcer drugs – competitive pricing and analysis of demand elasticities

US Navy benefit packages for reenlistment – conjoint used to develop menu of new reenlistment plans based on individual differences in types of duties, health needs and sign-over bonuses

Fedex new services study – trade-off study of customer reactions to new methods for tracking delayed and lost letters and packages

Polaroid’s instant camera design – study of consumer reactions to new features and camera design aesthetics

Marriott time-share units – development of optimal interior and exterior decors, services and price

Japanese cable TV – conjoint survey of Japanese consumer’s trade-offs among services and prices of satellite TV

Table 2.1: A partial list of conjoint applications, Source: Green et al. (2001)
### Consumer Non-durables:
1. Bar soaps  
2. Hair shampoos  
3. Carpet cleaners  
4. Synthetic fiber garments  
5. Gasoline pricing  
6. Panty-hose  
7. Lawn chemicals

### Other Products:
1. Automotive styling  
2. Automobile and truck tires  
3. Car batteries  
4. Ethical drugs  
5. Toasters/ovens  
6. Cameras  
7. Apartment design

### Financial Services:
1. Branch bank services  
2. Auto insurance policies  
3. Health insurance policies  
4. Credit-card features  
5. Consumer discount cards  
6. Auto retailing facilities  
7. High-tech maintenance service

### Other Services:
1. Car-rental agencies  
2. Telephone services and pricing  
3. Employment agencies  
4. Information-retrieval services  
5. Medical laboratories  
6. Hotel design

### Industrial Goods:
1. Copying machines  
2. Printing equipment  
3. Facsimile transmissions  
4. Data transmission  
5. Portable computer terminals  
6. Personal computer design

### Transportation:
1. Domestic airlines  
2. Transcontinental airlines  
3. Passenger train operations  
4. Freight train operations  
5. International air transportation association  
6. Electric car design

---

Table 2.2: A sample list of conjoint applications, Source: Green and Krieger (1993)
Figure 3.1: First choice and logit rules - Simulation methods compared,
Source: Pedagogical material from sawtoothsoftware.com
Figure 4.1: Traversing the search tree

Root Node ($Z_{inc} = 6$)

$L_0$

Forward Step

$L_{11}$

$x_1 = 1,$
$x_2 = -1$

Forward Step

$L_{11}'$

$x_1 = 0,$
$x_2 = -1$

Backtrack Step

$L_{12}$

$x_1 = 0,$
$x_2 = 1,$
$Z_{inc} = 8$

Backtrack Step

$L_{12}'$

$x_1 = 0,$
$x_2 = 0$

Forward Step

$L_{21}$

$x_3 = 1,$
$x_4 = -1,$
$Z_{inc} = 6$

Forward Step

$L_{21}'$

$x_3 = 0,$
$x_4 = -1$

$L_{22}$

$x_3 = 0,$
$x_4 = 1,$
$Z_{inc} = 8$

$L_{22}'$

$x_3 = 0,$
$x_4 = 0$

Fathomed node
Generate a feasible solution to the SOC problem using a greedy algorithm and set $Z_{inc} = Z_{greedy}$.

Solve a lagrangian relaxation of the SOC problem and update $Z_{inc}$ if an improved feasible solution (to the SOC problem) is obtained during sub-gradient optimization.

Is $Z_{inc} = |R|$?

- Yes: Stop
- No: Determine branching order for Phase II of the algorithm

To Phase II

Figure 4.2: A flow diagram of Phase I of the exact algorithm.

Note: Objective function value of the best incumbent solution is $Z_{inc}$ and that of the solution obtained from the greedy algorithm is $Z_{greedy}$.
Let $\eta = 1$ and $x_j = -1$, $j \in L$.

Is a free variable $x_j : x_j = -1$ and $j^* \in L_{\eta}$ available for branching?

Branch on $x_j$ and fix $x_j = 1$.

Let $\eta = \eta + 1$

Is $\eta = 1$?

Is $\eta = |A|$?

Is $Z_{\text{inc}} = |R|$?

Update $Z_{\text{inc}} = Z_{\text{cur}}$

Is $Z_{\text{cur}} > Z_{\text{inc}}$?

Is $Z_{\text{inc}} \geq 0.5|\bar{R}|$?

Fathom node based on logic test I?

Fathom node based on logic test II?

Backtrack by identifying $x_j : x_j = 1$ and $\hat{j} \in L_{\eta}$.

Fix $x_i = 0$.

Reset $x_j = -1$, $j \in L_{\eta}$ and let $\eta = \eta - 1$.

Stop

Update $Z_{\text{inc}} = Z_{\text{cur}}$

Yes

Yes

No

No

Yes

Figure 4.3: Flow diagram of Phase II of the exact algorithm.

Note: Objective function value of the best incumbent solution is $Z_{\text{inc}}$ and that of the current feasible solution is $Z_{\text{cur}}$.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Number of Attributes</th>
<th>Number of Levels **</th>
<th>Number of Respondents</th>
<th>Exact Algorithm</th>
<th>CPLEX 8.0 Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Greedy Solution Value</td>
<td>CPLEX Solver Solution Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Subgradient Optimization Solution Value</td>
<td>CPLEX Solver Solution Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Optimal Solution Value</td>
<td>CPLEX Solver Solution Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Total Solution Time (in seconds)</td>
<td>CPLEX Solver Solution Time (in seconds)</td>
</tr>
<tr>
<td>A</td>
<td>15</td>
<td>39</td>
<td>321</td>
<td>211</td>
<td>222 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>219</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>222</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>15</td>
<td>50</td>
<td>1131</td>
<td>683</td>
<td>683 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>686</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>686</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.8</td>
<td></td>
</tr>
</tbody>
</table>

* - The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached;

** - The number of levels are 13@2 + 4@9 = 39 and 10@3 + 5@4 = 50 respectively.

Table 8.1: Results for two real world problems
<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Experimental Design Factors</th>
<th>Exact Algorithm</th>
<th>CPlex 8.0 Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Respondents</td>
<td>Number of Levels</td>
<td>Size of Competitor Set</td>
</tr>
<tr>
<td>1</td>
<td>600</td>
<td>70</td>
<td>500</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>70</td>
<td>500</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>600</td>
<td>70</td>
<td>500</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>600</td>
<td>70</td>
<td>500</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* - The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached; **- The 3 hour run-time limit was exceeded;

Table 8.2: Computational results for the simulated test problems
<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Experimental Design Factors</th>
<th>Exact Algorithm</th>
<th>CPLEX 8.0 Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Respondents</td>
<td>Number of Levels</td>
<td>Size of Competitor Set</td>
</tr>
<tr>
<td>21</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>22</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>23</td>
<td>600</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>24</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>25</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>26</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>27</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>28</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>29</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>30</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>31</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>32</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>33</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>34</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>35</td>
<td>600</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>36</td>
<td>600</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>37</td>
<td>600</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>38</td>
<td>600</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>39</td>
<td>600</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>40</td>
<td>600</td>
<td>112</td>
<td>1000</td>
</tr>
</tbody>
</table>

* The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached; **- The 3 hour run-time limit was exceeded; 

Table 8.2 (continued): Computational results for the simulated test problems
<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Number of Respondents</th>
<th>Number of Levels</th>
<th>Size of Competitor Set</th>
<th>Number of Segments</th>
<th>Hurdle Difficulty</th>
<th>Replicate</th>
<th>Greedy Solution Value</th>
<th>Subgradient Optimization Solution Value</th>
<th>Optimal Solution Value</th>
<th>Total Solution Time (in seconds)</th>
<th>CPLEX Solver Solution Value</th>
<th>CPLEX Solver Solution Time (in seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>600</td>
<td>112</td>
<td>1000</td>
<td>2</td>
<td>High</td>
<td>2</td>
<td>600</td>
<td>600</td>
<td>600</td>
<td>0.5</td>
<td>600</td>
<td>0.8</td>
</tr>
<tr>
<td>42</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>596</td>
<td>600</td>
<td>600</td>
<td>4.3</td>
<td>600</td>
<td>6.1</td>
</tr>
<tr>
<td>43</td>
<td>600</td>
<td>112</td>
<td>1000</td>
<td>4</td>
<td>Low</td>
<td>1</td>
<td>585</td>
<td>590</td>
<td>591</td>
<td>1017.5</td>
<td>591</td>
<td>5472.2</td>
</tr>
<tr>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>566</td>
<td>577</td>
<td>587</td>
<td>533.4</td>
<td>587</td>
<td>10800</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>577</td>
<td>583</td>
<td>587</td>
<td>683.9</td>
<td>586</td>
<td>10800</td>
</tr>
<tr>
<td>46</td>
<td>600</td>
<td>112</td>
<td>1000</td>
<td>4</td>
<td>High</td>
<td>1</td>
<td>398</td>
<td>407</td>
<td>427</td>
<td>4425.7</td>
<td>424</td>
<td>10800</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>351</td>
<td>351</td>
<td>408</td>
<td>3577.8</td>
<td>397</td>
<td>10800</td>
</tr>
<tr>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>356</td>
<td>397</td>
<td>430</td>
<td>4853.1</td>
<td>417</td>
<td>10800</td>
</tr>
<tr>
<td>49</td>
<td>1200</td>
<td>70</td>
<td>500</td>
<td>2</td>
<td>Low</td>
<td>1</td>
<td>1200</td>
<td>1200</td>
<td>1200</td>
<td>0.5</td>
<td>1200</td>
<td>0.4</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>1200</td>
<td>1200</td>
<td>1200</td>
<td>0.5</td>
<td>1200</td>
<td>0.6</td>
</tr>
<tr>
<td>51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>1200</td>
<td>1200</td>
<td>1200</td>
<td>0.5</td>
<td>1200</td>
<td>0.3</td>
</tr>
<tr>
<td>52</td>
<td>1200</td>
<td>70</td>
<td>500</td>
<td>2</td>
<td>High</td>
<td>1</td>
<td>1187</td>
<td>1187</td>
<td>1187</td>
<td>43.5</td>
<td>1187</td>
<td>42.3</td>
</tr>
<tr>
<td>53</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>1196</td>
<td>1196</td>
<td>1196</td>
<td>26.7</td>
<td>1196</td>
<td>10.2</td>
</tr>
<tr>
<td>54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>1197</td>
<td>1197</td>
<td>1198</td>
<td>32.4</td>
<td>1198</td>
<td>4.5</td>
</tr>
<tr>
<td>55</td>
<td>1200</td>
<td>70</td>
<td>500</td>
<td>4</td>
<td>Low</td>
<td>1</td>
<td>936</td>
<td>962</td>
<td>962</td>
<td>69.9</td>
<td>956</td>
<td>10800</td>
</tr>
<tr>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>864</td>
<td>895</td>
<td>952</td>
<td>79.4</td>
<td>933</td>
<td>10800</td>
</tr>
<tr>
<td>57</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>848</td>
<td>875</td>
<td>892</td>
<td>77.2</td>
<td>880</td>
<td>10800</td>
</tr>
<tr>
<td>58</td>
<td>1200</td>
<td>70</td>
<td>500</td>
<td>4</td>
<td>High</td>
<td>1</td>
<td>507</td>
<td>658</td>
<td>706</td>
<td>57</td>
<td>650</td>
<td>10800</td>
</tr>
<tr>
<td>59</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>532</td>
<td>707</td>
<td>761</td>
<td>61.5</td>
<td>665</td>
<td>10800</td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>470</td>
<td>694</td>
<td>708</td>
<td>37.6</td>
<td>566</td>
<td>10800</td>
</tr>
</tbody>
</table>

* - The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached; ** - The 3 hour run-time limit was exceeded;

Table 8.2 (continued): Computational results for the simulated test problems
<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Experimental Design Factors</th>
<th>Exact Algorithm</th>
<th>CPLEX 8.0 Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Respondents</td>
<td>Number of Levels</td>
<td>Size of Competitor Set</td>
</tr>
<tr>
<td>61</td>
<td>1200</td>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>62</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* - The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached; ** - The 3 hour run-time limit was exceeded;
<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Experimental Design Factors</th>
<th>Exact Algorithm</th>
<th>CPLEX 8.0 Solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Respondents</td>
<td>Number of Levels</td>
<td>Size of Competitor Set</td>
</tr>
<tr>
<td>81</td>
<td>1200</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>82</td>
<td>1200</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>83</td>
<td>1200</td>
<td>112</td>
<td>500</td>
</tr>
<tr>
<td>84</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>86</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>87</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>88</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>89</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>90</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>91</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>92</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>93</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>94</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>95</td>
<td>1200</td>
<td>112</td>
<td>1000</td>
</tr>
<tr>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* - The best solution found (optimality not proven) when the run-time limit of 10800 seconds (3 hours) was reached; **- The 3 hour run-time limit was exceeded;

Table 8.2 (continued): Computational results for the simulated test problems
### Model 1

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected Model</td>
<td>839.360</td>
<td></td>
<td>15</td>
<td>55.957</td>
<td>21.523</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>Intercept</td>
<td>951.571</td>
<td>8.626</td>
<td>1</td>
<td>951.571</td>
<td>365.999</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>R</td>
<td>30.905</td>
<td>-1.160</td>
<td>1</td>
<td>30.905</td>
<td>11.887</td>
<td>.001</td>
</tr>
<tr>
<td>L</td>
<td>84.208</td>
<td>-4.593</td>
<td>1</td>
<td>84.208</td>
<td>32.389</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>C</td>
<td>2.089</td>
<td>-.109</td>
<td>1</td>
<td>2.089</td>
<td>.803</td>
<td>.373</td>
</tr>
<tr>
<td>S</td>
<td>590.314</td>
<td>-7.06</td>
<td>1</td>
<td>590.314</td>
<td>227.051</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxL</td>
<td>.331</td>
<td>.188</td>
<td>1</td>
<td>.331</td>
<td>.127</td>
<td>.722</td>
</tr>
<tr>
<td>RxC</td>
<td>.044</td>
<td>-1.425</td>
<td>1</td>
<td>.044</td>
<td>.017</td>
<td>.897</td>
</tr>
<tr>
<td>LxC</td>
<td>.952</td>
<td>.202</td>
<td>1</td>
<td>.952</td>
<td>.366</td>
<td>.547</td>
</tr>
<tr>
<td>RxS</td>
<td>1.627</td>
<td>-.736</td>
<td>1</td>
<td>1.627</td>
<td>.626</td>
<td>.431</td>
</tr>
<tr>
<td>LxS</td>
<td>110.037</td>
<td>4.340</td>
<td>1</td>
<td>110.037</td>
<td>42.323</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxS</td>
<td>.044</td>
<td>-1.322</td>
<td>1</td>
<td>.044</td>
<td>.017</td>
<td>.897</td>
</tr>
<tr>
<td>RxLxC</td>
<td>1.306</td>
<td>1.533</td>
<td>1</td>
<td>1.306</td>
<td>.502</td>
<td>.481</td>
</tr>
<tr>
<td>RxLxS</td>
<td>3.111</td>
<td>1.026</td>
<td>1</td>
<td>3.111</td>
<td>1.196</td>
<td>.277</td>
</tr>
<tr>
<td>RxCxS</td>
<td>3.320</td>
<td>3.954</td>
<td>1</td>
<td>3.320</td>
<td>1.277</td>
<td>.262</td>
</tr>
<tr>
<td>LxCxS</td>
<td>1.951</td>
<td>1.326</td>
<td>1</td>
<td>1.951</td>
<td>.750</td>
<td>.389</td>
</tr>
<tr>
<td>RxLxCxS</td>
<td>9.122</td>
<td>-4.932</td>
<td>1</td>
<td>9.122</td>
<td>3.509</td>
<td>.065</td>
</tr>
<tr>
<td>Error</td>
<td>207.994</td>
<td></td>
<td>80</td>
<td>2.600</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1998.925</td>
<td></td>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>1047.354</td>
<td></td>
<td>95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ R^2 = .801 \text{ (Adjusted } R^2 = .764) \]

R – Number of Respondents; L – Number of Levels; C – Size of Competitive Set; S – Number of Segments;

---

Table 8.3: Model 1 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time
Table 8.4: Model 2(a) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time (continued to the next page)
## Model 2(a)

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LxCxSxH</td>
<td>.635</td>
<td>-3.200</td>
<td>1</td>
<td>.635</td>
<td>.743</td>
<td>.392</td>
</tr>
<tr>
<td>RxLxCxSxH</td>
<td>1.376</td>
<td>3.942</td>
<td>1</td>
<td>1.376</td>
<td>1.609</td>
<td>.209</td>
</tr>
<tr>
<td>Error</td>
<td>53.029</td>
<td></td>
<td>62</td>
<td>.855</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1998.925</td>
<td></td>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>1047.354</td>
<td></td>
<td>95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R² = .949 (Adjusted R² = .922)

Factors:
- R – Number of Respondents;
- L – Number of Levels;
- C – Size of Competitive Set;
- S – Number of Segments;
- H – Hurdle Difficulty;

Covariates:
- \( OC \) – Optimal Proportional Coverage;
- \( OGG \) = Gap between Optimal and Greedy;

\[
OC = \left( \frac{\text{Optimal Solution}}{\text{Sample Size}} \right) \times 100;
\]

\[
OGG = \left( \frac{(\text{Optimal Solution} - \text{Greedy Solution})}{\text{Optimal Solution}} \right) \times 100;
\]

Table 8.4: Model 2(a) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time
<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected Model</td>
<td>1024.921</td>
<td></td>
<td>61</td>
<td>16.802</td>
<td>25.466</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>Intercept</td>
<td>951.571</td>
<td>-191.575</td>
<td>1</td>
<td>951.571</td>
<td>1442.228</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>R</td>
<td>30.905</td>
<td>-29.831</td>
<td>1</td>
<td>30.905</td>
<td>46.840</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>L</td>
<td>84.208</td>
<td>40.242</td>
<td>1</td>
<td>84.208</td>
<td>127.628</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>C</td>
<td>2.089</td>
<td>211.287</td>
<td>1</td>
<td>2.089</td>
<td>3.166</td>
<td>.084</td>
</tr>
<tr>
<td>S</td>
<td>590.314</td>
<td>3711.318</td>
<td>1</td>
<td>590.314</td>
<td>894.697</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>H</td>
<td>83.442</td>
<td>173.862</td>
<td>1</td>
<td>83.442</td>
<td>126.467</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>OC</td>
<td>84.091</td>
<td>2.836</td>
<td>1</td>
<td>84.091</td>
<td>127.451</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>OGG</td>
<td>28.304</td>
<td>14.580</td>
<td>1</td>
<td>28.304</td>
<td>42.898</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxL</td>
<td>.366</td>
<td>7.042</td>
<td>1</td>
<td>.366</td>
<td>.555</td>
<td>.461</td>
</tr>
<tr>
<td>RxC</td>
<td>.161</td>
<td>1.590</td>
<td>1</td>
<td>.161</td>
<td>.243</td>
<td>.625</td>
</tr>
<tr>
<td>RxS</td>
<td>1.777</td>
<td>881.242</td>
<td>1</td>
<td>1.777</td>
<td>2.694</td>
<td>.110</td>
</tr>
<tr>
<td>RxH</td>
<td>.068</td>
<td>-9.908</td>
<td>1</td>
<td>.068</td>
<td>.102</td>
<td>.751</td>
</tr>
<tr>
<td>RxOC</td>
<td>6.008</td>
<td>.386</td>
<td>1</td>
<td>6.008</td>
<td>9.105</td>
<td>.005</td>
</tr>
<tr>
<td>RxOGG</td>
<td>1.390</td>
<td>1.455</td>
<td>1</td>
<td>1.390</td>
<td>2.107</td>
<td>.156</td>
</tr>
<tr>
<td>LxC</td>
<td>3.332</td>
<td>36.413</td>
<td>1</td>
<td>3.332</td>
<td>5.049</td>
<td>.031</td>
</tr>
<tr>
<td>LxS</td>
<td>26.085</td>
<td>-4342.879</td>
<td>1</td>
<td>26.085</td>
<td>39.535</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>LxH</td>
<td>.196</td>
<td>-347.115</td>
<td>1</td>
<td>.196</td>
<td>.297</td>
<td>.589</td>
</tr>
<tr>
<td>LxC</td>
<td>14.254</td>
<td>.019</td>
<td>1</td>
<td>14.254</td>
<td>21.604</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>LxOGG</td>
<td>.222</td>
<td>-9.120</td>
<td>1</td>
<td>.222</td>
<td>.336</td>
<td>.566</td>
</tr>
<tr>
<td>CxS</td>
<td>1.531</td>
<td>558.250</td>
<td>1</td>
<td>1.531</td>
<td>2.320</td>
<td>.137</td>
</tr>
<tr>
<td>CxH</td>
<td>.002</td>
<td>40.262</td>
<td>1</td>
<td>.002</td>
<td>.004</td>
<td>.953</td>
</tr>
<tr>
<td>CxOC</td>
<td>2.778</td>
<td>-2.975</td>
<td>1</td>
<td>2.778</td>
<td>4.210</td>
<td>.048</td>
</tr>
<tr>
<td>CxOGG</td>
<td>.519</td>
<td>-10.184</td>
<td>1</td>
<td>.519</td>
<td>.786</td>
<td>.381</td>
</tr>
<tr>
<td>SxH</td>
<td>2.476</td>
<td>84.279</td>
<td>1</td>
<td>2.476</td>
<td>3.752</td>
<td>.061</td>
</tr>
<tr>
<td>SxOC</td>
<td>8.704</td>
<td>-38.042</td>
<td>1</td>
<td>8.704</td>
<td>13.192</td>
<td>.001</td>
</tr>
<tr>
<td>SxOGG</td>
<td>5.772</td>
<td>-8584.333</td>
<td>1</td>
<td>5.772</td>
<td>8.749</td>
<td>.006</td>
</tr>
<tr>
<td>HxOC</td>
<td>1.498</td>
<td>-2.574</td>
<td>1</td>
<td>1.498</td>
<td>2.270</td>
<td>.141</td>
</tr>
<tr>
<td>HxOGG</td>
<td>.253</td>
<td>20.089</td>
<td>1</td>
<td>.253</td>
<td>.384</td>
<td>.540</td>
</tr>
<tr>
<td>OCxOGG</td>
<td>.196</td>
<td>-2.05</td>
<td>1</td>
<td>.196</td>
<td>.297</td>
<td>.589</td>
</tr>
</tbody>
</table>

Table 8.5: Model 2(b) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time (continued to the next page)
<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RxLxC</td>
<td>.556</td>
<td>-.181</td>
<td>1</td>
<td>.556</td>
<td>.842</td>
<td>.365</td>
</tr>
<tr>
<td>RxLxS</td>
<td>.267</td>
<td>-7.929</td>
<td>1</td>
<td>.267</td>
<td>.404</td>
<td>.529</td>
</tr>
<tr>
<td>RxCxH</td>
<td>1.690</td>
<td>4.537</td>
<td>1</td>
<td>1.690</td>
<td>2.562</td>
<td>.119</td>
</tr>
<tr>
<td>RxLxOC</td>
<td>.301</td>
<td>-.033</td>
<td>1</td>
<td>.301</td>
<td>.456</td>
<td>.504</td>
</tr>
<tr>
<td>RxLxOGG</td>
<td>.017</td>
<td>-.364</td>
<td>1</td>
<td>.017</td>
<td>.026</td>
<td>.872</td>
</tr>
<tr>
<td>RxCxS</td>
<td>2.024</td>
<td>3.033</td>
<td>1</td>
<td>2.024</td>
<td>3.068</td>
<td>.089</td>
</tr>
<tr>
<td>RxCxH</td>
<td>4.948</td>
<td>-.061</td>
<td>1</td>
<td>4.948</td>
<td>7.500</td>
<td>.010</td>
</tr>
<tr>
<td>RxCxOC</td>
<td>2.116</td>
<td>-.044</td>
<td>1</td>
<td>2.116</td>
<td>3.207</td>
<td>.082</td>
</tr>
<tr>
<td>RxCxOGG</td>
<td>1.030</td>
<td>.116</td>
<td>1</td>
<td>1.030</td>
<td>1.561</td>
<td>.220</td>
</tr>
<tr>
<td>RxSxH</td>
<td>.192</td>
<td>7.949</td>
<td>1</td>
<td>.192</td>
<td>.292</td>
<td>.593</td>
</tr>
<tr>
<td>RxSxOC</td>
<td>.269</td>
<td>-8.895</td>
<td>1</td>
<td>.269</td>
<td>.408</td>
<td>.527</td>
</tr>
<tr>
<td>RxSxOGG</td>
<td>6.428</td>
<td>-5.903</td>
<td>1</td>
<td>6.428</td>
<td>9.742</td>
<td>.004</td>
</tr>
<tr>
<td>RxHxOC</td>
<td>.072</td>
<td>.006</td>
<td>1</td>
<td>.072</td>
<td>.109</td>
<td>.744</td>
</tr>
<tr>
<td>RxHxOGG</td>
<td>.125</td>
<td>.422</td>
<td>1</td>
<td>.125</td>
<td>.189</td>
<td>.667</td>
</tr>
<tr>
<td>RxOCxOGG</td>
<td>.009</td>
<td>-.020</td>
<td>1</td>
<td>.009</td>
<td>.013</td>
<td>.910</td>
</tr>
<tr>
<td>LxCxS</td>
<td>.618</td>
<td>97.588</td>
<td>1</td>
<td>.618</td>
<td>.936</td>
<td>.340</td>
</tr>
<tr>
<td>LxCxH</td>
<td>3.657</td>
<td>3.724</td>
<td>1</td>
<td>3.657</td>
<td>5.543</td>
<td>.024</td>
</tr>
<tr>
<td>LxCxOC</td>
<td>.055</td>
<td>-1.377</td>
<td>1</td>
<td>.055</td>
<td>.083</td>
<td>.775</td>
</tr>
<tr>
<td>LxCxOGG</td>
<td>.003</td>
<td>1.535</td>
<td>1</td>
<td>.003</td>
<td>.005</td>
<td>.945</td>
</tr>
<tr>
<td>LxSxH</td>
<td>1.318</td>
<td>-72.976</td>
<td>1</td>
<td>1.318</td>
<td>1.998</td>
<td>.167</td>
</tr>
<tr>
<td>LxSxOC</td>
<td>1.777</td>
<td>43.071</td>
<td>1</td>
<td>1.777</td>
<td>2.693</td>
<td>.110</td>
</tr>
<tr>
<td>LxSxOGG</td>
<td>.170</td>
<td>10.738</td>
<td>1</td>
<td>.170</td>
<td>.257</td>
<td>.615</td>
</tr>
<tr>
<td>LxHxOC</td>
<td>.534</td>
<td>4.131</td>
<td>1</td>
<td>.534</td>
<td>.809</td>
<td>.375</td>
</tr>
<tr>
<td>LxHxOGG</td>
<td>.097</td>
<td>-6.969</td>
<td>1</td>
<td>.097</td>
<td>.147</td>
<td>.704</td>
</tr>
<tr>
<td>LxOCxOGG</td>
<td>.005</td>
<td>.105</td>
<td>1</td>
<td>.005</td>
<td>.008</td>
<td>.929</td>
</tr>
<tr>
<td>CxSxH</td>
<td>.158</td>
<td>-81.328</td>
<td>1</td>
<td>.158</td>
<td>.240</td>
<td>.627</td>
</tr>
<tr>
<td>CxSxOC</td>
<td>.510</td>
<td>-4.714</td>
<td>1</td>
<td>.510</td>
<td>.772</td>
<td>.386</td>
</tr>
<tr>
<td>CxSxOGG</td>
<td>.010</td>
<td>-1.837</td>
<td>1</td>
<td>.010</td>
<td>.015</td>
<td>.902</td>
</tr>
<tr>
<td>CxHxOC</td>
<td>1.074</td>
<td>.403</td>
<td>1</td>
<td>1.074</td>
<td>1.628</td>
<td>.211</td>
</tr>
<tr>
<td>CxHxOGG</td>
<td>.336</td>
<td>-2.128</td>
<td>1</td>
<td>.336</td>
<td>.509</td>
<td>.481</td>
</tr>
<tr>
<td>CxOCxOGG</td>
<td>6.321</td>
<td>.152</td>
<td>1</td>
<td>6.321</td>
<td>9.580</td>
<td>.004</td>
</tr>
</tbody>
</table>

Table 8.5: Model 2(b) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time (continued to the next page)
Model 2(b)

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SxHxOC</td>
<td>.000</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SxHxOGG</td>
<td>.000</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SxOCxOGG</td>
<td>6.493</td>
<td>85.980</td>
<td>1</td>
<td>6.493</td>
<td>9.841</td>
<td>.004</td>
</tr>
<tr>
<td>HxOCxOGG</td>
<td>.801</td>
<td>-.149</td>
<td>1</td>
<td>.801</td>
<td>1.214</td>
<td>.278</td>
</tr>
<tr>
<td>Error</td>
<td>22.433</td>
<td>34</td>
<td>.660</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1998.925</td>
<td>96</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>1047.354</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R² = .979 (Adjusted R² = .940)

Factors:
R – Number of Respondents; L – Number of Levels; C – Size of Competitive Set; S – Number of Segments; H – Hurdle Difficulty;

Covariates:
OC – Optimal Proportional Coverage; OGG = Gap between Optimal and Greedy;

\[
OC = \left( \frac{\text{Optimal Solution}}{\text{Sample Size}} \right) \times 100;
\]

\[
OGG = \left( \frac{\text{(Optimal Solution – Greedy Solution)}}{\text{Optimal Solution}} \right) \times 100;
\]

Table 8.5: Model 2(b) - GLM Univariate Analysis results: Dependent variable is the natural logarithm of solution time.
Model 3

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P.Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected Model</td>
<td>1.517</td>
<td></td>
<td>31</td>
<td>.049</td>
<td>23.173</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>Intercept</td>
<td>1974.004</td>
<td>4.471</td>
<td>1</td>
<td>1974.004</td>
<td>934739.305</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>R</td>
<td>.001</td>
<td>-.003</td>
<td>1</td>
<td>.001</td>
<td>.608</td>
<td>.439</td>
</tr>
<tr>
<td>L</td>
<td>.178</td>
<td>-.220</td>
<td>1</td>
<td>.178</td>
<td>84.501</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>C</td>
<td>.000</td>
<td>.103</td>
<td>1</td>
<td>.000</td>
<td>.172</td>
<td>.679</td>
</tr>
<tr>
<td>S</td>
<td>.465</td>
<td>.132</td>
<td>1</td>
<td>.465</td>
<td>220.036</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>H</td>
<td>.212</td>
<td>.106</td>
<td>1</td>
<td>.212</td>
<td>100.551</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxL</td>
<td>.000</td>
<td>.060</td>
<td>1</td>
<td>.000</td>
<td>.220</td>
<td>.641</td>
</tr>
<tr>
<td>RxC</td>
<td>.004</td>
<td>-.012</td>
<td>1</td>
<td>.004</td>
<td>1.782</td>
<td>.187</td>
</tr>
<tr>
<td>RxS</td>
<td>.001</td>
<td>-.001</td>
<td>1</td>
<td>.001</td>
<td>.465</td>
<td>.498</td>
</tr>
<tr>
<td>LxS</td>
<td>.184</td>
<td>.221</td>
<td>1</td>
<td>.184</td>
<td>86.969</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxS</td>
<td>.000</td>
<td>-.102</td>
<td>1</td>
<td>.000</td>
<td>.165</td>
<td>.686</td>
</tr>
<tr>
<td>RxH</td>
<td>.001</td>
<td>.009</td>
<td>1</td>
<td>.001</td>
<td>.373</td>
<td>.544</td>
</tr>
<tr>
<td>LxH</td>
<td>.083</td>
<td>.164</td>
<td>1</td>
<td>.083</td>
<td>39.532</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxH</td>
<td>.001</td>
<td>-.087</td>
<td>1</td>
<td>.001</td>
<td>.601</td>
<td>.441</td>
</tr>
<tr>
<td>SxH</td>
<td>.205</td>
<td>-.104</td>
<td>1</td>
<td>.205</td>
<td>97.019</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>LxC</td>
<td>.013</td>
<td>-.116</td>
<td>1</td>
<td>.013</td>
<td>5.936</td>
<td>.018</td>
</tr>
<tr>
<td>RxLxC</td>
<td>.002</td>
<td>-.182</td>
<td>1</td>
<td>.002</td>
<td>1.012</td>
<td>.318</td>
</tr>
<tr>
<td>RxLxS</td>
<td>.001</td>
<td>-.057</td>
<td>1</td>
<td>.001</td>
<td>.311</td>
<td>.579</td>
</tr>
<tr>
<td>RxCxS</td>
<td>.004</td>
<td>.012</td>
<td>1</td>
<td>.004</td>
<td>1.735</td>
<td>.193</td>
</tr>
<tr>
<td>LxCxS</td>
<td>.012</td>
<td>.115</td>
<td>1</td>
<td>.012</td>
<td>5.764</td>
<td>.019</td>
</tr>
<tr>
<td>RxLxH</td>
<td>.000</td>
<td>-.084</td>
<td>1</td>
<td>.000</td>
<td>.070</td>
<td>.792</td>
</tr>
<tr>
<td>RxCxH</td>
<td>.004</td>
<td>.000</td>
<td>1</td>
<td>.004</td>
<td>2.078</td>
<td>.154</td>
</tr>
<tr>
<td>RxSxH</td>
<td>.001</td>
<td>-.006</td>
<td>1</td>
<td>.001</td>
<td>.263</td>
<td>.610</td>
</tr>
<tr>
<td>LxSxH</td>
<td>.087</td>
<td>-.166</td>
<td>1</td>
<td>.087</td>
<td>41.226</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxSxH</td>
<td>.001</td>
<td>.086</td>
<td>1</td>
<td>.001</td>
<td>.615</td>
<td>.436</td>
</tr>
<tr>
<td>LxCxH</td>
<td>.020</td>
<td>.125</td>
<td>1</td>
<td>.020</td>
<td>9.682</td>
<td>.003</td>
</tr>
</tbody>
</table>

Table 8.6: Model 3 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the greedy solution over the optimal solution expressed as a percentage (continued to the next page)
Table 8.6: Model 3 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the greedy solution over the optimal solution expressed as a percentage

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P.Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RxLxCxS</td>
<td>.002</td>
<td>.182</td>
<td>1</td>
<td>.002</td>
<td>.994</td>
<td>.322</td>
</tr>
<tr>
<td>RxLxCxH</td>
<td>.004</td>
<td>.214</td>
<td>1</td>
<td>.004</td>
<td>2.052</td>
<td>.157</td>
</tr>
<tr>
<td>RxLxSxH</td>
<td>.000</td>
<td>.080</td>
<td>1</td>
<td>.000</td>
<td>.125</td>
<td>.725</td>
</tr>
<tr>
<td>RxCxSxH</td>
<td>.004</td>
<td>.000</td>
<td>1</td>
<td>.004</td>
<td>2.027</td>
<td>.159</td>
</tr>
<tr>
<td>LxCxSxH</td>
<td>.020</td>
<td>-.124</td>
<td>1</td>
<td>.020</td>
<td>9.462</td>
<td>.003</td>
</tr>
<tr>
<td>RxLxCxSxH</td>
<td>.004</td>
<td>-.214</td>
<td>1</td>
<td>.004</td>
<td>2.027</td>
<td>.159</td>
</tr>
<tr>
<td>Error</td>
<td>.135</td>
<td></td>
<td>64</td>
<td>.002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1975.656</td>
<td></td>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>1.652</td>
<td></td>
<td>95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R² = .918 (Adjusted R² = .879)

R – Number of Respondents; L – Number of Levels; C – Size of Competitive Set; S – Number of Segments; H – Hurdle Difficulty;
## Model 4

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected Model</td>
<td>3.527</td>
<td></td>
<td></td>
<td>.114</td>
<td>155.948</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>Intercept</td>
<td>1916.218</td>
<td>4.269</td>
<td>1</td>
<td>1916.218</td>
<td>2626455.248</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>R</td>
<td>.002</td>
<td>-.017</td>
<td>1</td>
<td>.002</td>
<td>3.181</td>
<td>.079</td>
</tr>
<tr>
<td>L</td>
<td>.389</td>
<td>-.272</td>
<td>1</td>
<td>.389</td>
<td>532.662</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>C</td>
<td>.037</td>
<td>.088</td>
<td>1</td>
<td>.037</td>
<td>50.893</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>S</td>
<td>1.778</td>
<td>.335</td>
<td>1</td>
<td>1.778</td>
<td>2436.723</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>H</td>
<td>.445</td>
<td>.316</td>
<td>1</td>
<td>.445</td>
<td>609.554</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxL</td>
<td>.000</td>
<td>.047</td>
<td>1</td>
<td>.000</td>
<td>.452</td>
<td>.504</td>
</tr>
<tr>
<td>RxC</td>
<td>.002</td>
<td>.074</td>
<td>1</td>
<td>.002</td>
<td>2.400</td>
<td>.126</td>
</tr>
<tr>
<td>LxC</td>
<td>.000</td>
<td>.016</td>
<td>1</td>
<td>.000</td>
<td>.560</td>
<td>.457</td>
</tr>
<tr>
<td>RsS</td>
<td>.002</td>
<td>.018</td>
<td>1</td>
<td>.002</td>
<td>3.317</td>
<td>.073</td>
</tr>
<tr>
<td>LxS</td>
<td>.377</td>
<td>.271</td>
<td>1</td>
<td>.377</td>
<td>516.393</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxS</td>
<td>.036</td>
<td>-.087</td>
<td>1</td>
<td>.036</td>
<td>49.076</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxH</td>
<td>.001</td>
<td>.017</td>
<td>1</td>
<td>.001</td>
<td>1.025</td>
<td>.315</td>
</tr>
<tr>
<td>LxH</td>
<td>&lt; .001</td>
<td>.000</td>
<td>1</td>
<td>&lt; .001</td>
<td>.066</td>
<td>.798</td>
</tr>
<tr>
<td>SxH</td>
<td>.429</td>
<td>-.314</td>
<td>1</td>
<td>.429</td>
<td>588.469</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>CxH</td>
<td>.011</td>
<td>-.076</td>
<td>1</td>
<td>.011</td>
<td>15.562</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>RxCxS</td>
<td>.001</td>
<td>-.076</td>
<td>1</td>
<td>.001</td>
<td>1.850</td>
<td>.179</td>
</tr>
<tr>
<td>LxCxS</td>
<td>.000</td>
<td>-.020</td>
<td>1</td>
<td>.000</td>
<td>.442</td>
<td>.508</td>
</tr>
<tr>
<td>RxLxH</td>
<td>&lt; .001</td>
<td>-.046</td>
<td>1</td>
<td>&lt; .001</td>
<td>.048</td>
<td>.827</td>
</tr>
<tr>
<td>RxLxS</td>
<td>.000</td>
<td>-.054</td>
<td>1</td>
<td>.000</td>
<td>.571</td>
<td>.453</td>
</tr>
<tr>
<td>RxLxC</td>
<td>&lt; .001</td>
<td>-.050</td>
<td>1</td>
<td>&lt; .001</td>
<td>.051</td>
<td>.821</td>
</tr>
<tr>
<td>RxCxH</td>
<td>.001</td>
<td>-.069</td>
<td>1</td>
<td>.001</td>
<td>.755</td>
<td>.388</td>
</tr>
<tr>
<td>LxCxH</td>
<td>.001</td>
<td>.016</td>
<td>1</td>
<td>.001</td>
<td>1.201</td>
<td>.277</td>
</tr>
<tr>
<td>RxSxH</td>
<td>.001</td>
<td>-.019</td>
<td>1</td>
<td>.001</td>
<td>1.103</td>
<td>.297</td>
</tr>
<tr>
<td>CxSxH</td>
<td>.011</td>
<td>.075</td>
<td>1</td>
<td>.011</td>
<td>14.565</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>LxSxH</td>
<td>&lt; .001</td>
<td>.001</td>
<td>1</td>
<td>&lt; .001</td>
<td>.010</td>
<td>.922</td>
</tr>
</tbody>
</table>

Table 8.7: Model 4 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the optimal solution over the sample size expressed as a percentage (continued to the next page)
### Table 8.7: Model 4 - GLM Univariate Analysis results: Dependent variable is the natural logarithm of ratio of the optimal solution over the sample size expressed as a percentage

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>Type I Sum of Squares</th>
<th>Parameter Estimate</th>
<th>Degrees of Freedom</th>
<th>Mean Square</th>
<th>F</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RxLxSxH</td>
<td>&lt; .001</td>
<td>.054</td>
<td>1</td>
<td>&lt; .001</td>
<td>.091</td>
<td>.763</td>
</tr>
<tr>
<td>RxLxCxS</td>
<td>.000</td>
<td>.062</td>
<td>1</td>
<td>.000</td>
<td>.241</td>
<td>.625</td>
</tr>
<tr>
<td>RxLxCxH</td>
<td>.000</td>
<td>.069</td>
<td>1</td>
<td>.000</td>
<td>.422</td>
<td>.518</td>
</tr>
<tr>
<td>RxCxSxH</td>
<td>.000</td>
<td>.070</td>
<td>1</td>
<td>.000</td>
<td>.463</td>
<td>.499</td>
</tr>
<tr>
<td>LxCxSxH</td>
<td>.001</td>
<td>-.012</td>
<td>1</td>
<td>.001</td>
<td>1.390</td>
<td>.243</td>
</tr>
<tr>
<td>RxLxCxSxH</td>
<td>.001</td>
<td>-.081</td>
<td>1</td>
<td>.001</td>
<td>.836</td>
<td>.364</td>
</tr>
<tr>
<td>Error</td>
<td>.047</td>
<td></td>
<td>64</td>
<td>.001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>1919.792</td>
<td></td>
<td>96</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>3.574</td>
<td></td>
<td>95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R² = .987 (Adjusted R² = .981)

R – Number of Respondents; L – Number of Levels; C – Size of Competitive Set; S – Number of Segments; H – Hurdle Difficulty;
Figure 8.1: Runtime difference between CPLEX 8.0 Solver and the exact algorithm
Figure 8.2: Ratio of Cplex Solution over Optimal Solution
Figure 8.3: Ratio of average runtimes at high level over average runtimes at low level for each factor.
Figure 8.4: Ratio of Greedy Solution over Optimal Solution
Figure 8.5: Ratio of Optimal Solution over Sample Size
Figure 8.6: Percentage of Greedy Solutions that are found to be Optimal
Figure 8.7: Percentage of Subgradient Optimal Solutions that are found to be

<table>
<thead>
<tr>
<th>Factor</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Respondents</td>
<td>27.08</td>
<td>18.75</td>
</tr>
<tr>
<td>No. of Levels</td>
<td>12.50</td>
<td>33.33</td>
</tr>
<tr>
<td>Size of Competitor Set</td>
<td>33.33</td>
<td>12.50</td>
</tr>
<tr>
<td>No. of Segments</td>
<td>16.67</td>
<td>29.17</td>
</tr>
<tr>
<td>Hurdle Difficulty</td>
<td>20.83</td>
<td>25.00</td>
</tr>
</tbody>
</table>
Figure 8.8: Percentage of Optimal Solutions found during the Branch-and-Bound Stage.