Department of Production &
Management Engineering

Technical University of Crete

École doctoral décision informatique
mathématiques organization

Université Paris Dauphine

An integrated marketing system for the optimal product line design problem, in a competitive reaction context, based on the qualitative consumer behavior analysis

by

Stelios Tsafarakis

Submitted for the partial fulfillment of the requirements for the degree of
Doctor of Philosophy

February 2010

Η διατριβή του Στέλιου Τσαφαράκη εγκρίνεται:

1. Ματσατσίνης Νικόλαος
2. Τσουκιάς Αλέξης
3. Γρηγορούδης Ευάγγελος
4. Πάσχος Ευάγγελος
5. Βλαχοπούλου Μάρω
6. Μάνθου – Φραγκοπούλου Βασιλική
7. Μυγδαλάς Αθανάσιος

Contents

LIST OF TABLES

LIST OF FIGURES

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τους δύο επιβλέποντές μου, Νικόλαο Ματσατσίνη καθηγητή του τμήματος Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης και Αλέξη Τσουκιά διευθυντή έρευνας CNRS στο Lamsade Paris Dauphine, για την καθοδήγηση τους κατά τη διάρκεια της διατριβής. Θέλω επίσης να ευχαριστήσω τους επίκουρους καθηγητές Ευάγγελο Γρηγορούδη, και Αναστάσιο Δουλάμη, καθώς και τον λέκτορα Γιάννη Μαρινάκη για τα πολύτιμα σχόλιά τους. Ένα μεγάλο ευχαριστώ χρωστάω επίσης στον μεταπτυχιακό φοιτητή Κώστα Παπαδάκη για τη βοήθειά του στην ανάπτυξη του συστήματος. Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου Καίτη και Σταύρο, καθώς και την αρραβωνιαστικιά μου Λίνα, για την διαρκή υποστήριξη που μου παρέχουν.

This work is part of the 03ED375 research project, implemented within the framework of the "Reinforcement Programme of Human Research Manpower" (PENED) and co-financed by the National and Community Funds (75% from E.U. - European Social Fund and 25% from the Greek Ministry of Development - General Secretariat of Research and Technology).

Σύντομο Βιογραφικό Σημείωμα

Ο Στέλιος Τσαφαράκης γεννήθηκε το 1977 στην Αθήνα. Το 1994 εισήχθη στη σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου από όπου αποφοίτησε το 2000. Το 2001 έλαβε το Master of Science στα Πληροφοριακά Συστήματα από το πανεπιστήμιο του Σαουθάμπτον της Μεγάλης Βρετανίας, και το 2007 έλαβε το Μεταπτυχιακό Δίπλωμα Ειδίκευσης στην Οργάνωση και Διοίκηση από το τμήμα Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης.

Από το 2006 έως το 2008 παρείχε επικουρικό έργο στη διεξαγωγή των μαθημάτων «Ηλεκτρονικό Επιχειρείν» και «Εισαγωγή στην Τεχνητή Νοημοσύνη», στο τμήμα Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης. Επίσης από το 2003 έως το 2009 υπήρξε διδάσκων των εργαστηριακών μαθημάτων «Πληροφορική ΙΙ», «Συστήματα Λήψης Μετρήσεων & Ελέγχου», «Μικροελεγκτές» και «Περιβαλλοντική Στατιστική» στο τμήμα Φυσικών Πόρων και Περιβάλλοντος του ΤΕΙ Κρήτης. Το 2009 εκλέχθηκε σε θέση ΕΤΕΠ στο τμήμα Αρχιτεκτόνων Μηχανικών του Πολυτεχνείου Κρήτης όπου υπηρετεί μέχρι σήμερα.

Έχει δημοσιεύσει εργασίες σε διεθνή επιστημονικά περιοδικά όπως το International Journal of Research in Marketing και το Journal of the Operational Research Society, ενώ έχει συμμετάσχει και σε διεθνή επιστημονικά συνέδρια με παρουσιάσεις εργασιών. Σχετικές με τη διδακτορική διατριβή εργασίες έχουν παρουσιαστεί μετά από κρίση στα: European Marketing Academy Conference (2008 και 2009), IEEE International Conference on Systems, Man, and Cybernetics (2008), IEEE International Engineering Management Conference (2008).

Περίληψη

Ο σχεδιασμός βέλτιστων προϊόντων είναι μια από τις πιο σημαντικές διαδικασίες για τη βιωσιμότητα μιας εταιρείας. Ο *Βέλτιστος Σχεδιασμός Γραμμής Προϊόντων* αποτελεί ένα ευρύ ερευνητικό πεδίο στο ποσοτικό μάρκετινγκ για πάνω από τριάντα χρόνια, το οποίο συνήθως μοντελοποιείται στα πλαίσια της *Συζυγούς Ανάλυσης*. Αντιμετωπίζοντας αυτό το NP-hard πρόβλημα βελτιστοποίησης, ένας μάνατζερ πρέπει να αποφασίσει σχετικά με μια σειρά από ζητήματα: την προσομοίωση της ανθρώπινης συμπεριφοράς επιλογής, τον αλγόριθμο βελτιστοποίησης που θα χρησιμοποιηθεί και τη μοντελοποίηση των πιθανών αντιδράσεων των ανταγωνιστών. Η εφαρμογή μιας αποδοτικής μεθόδου έχει σημαντικές επιπτώσεις για έναν μάρκετινγκ μάνατζερ, αφού η σχεδίαση μιας μη αποτελεσματικής γραμμής προϊόντων μπορεί να αποβεί στην κατάκτηση μικρότερου του αναμενομένου μεριδίου αγοράς, ή ακόμα και στον κανιβαλισμό των υφιστάμενων προϊόντων της εταιρείας. Ο μάνατζερ θα πρέπει να συγκρίνει προσεχτικά τα διαφορετικά μοντέλα επιλογής, αλγόριθμους βελτιστοποίησης και μοντέλα θεωρίας παιγνίων, και να επιλέξει αυτά που ταιριάζουν καλύτερα στις ανάγκες της εταιρείας. Αυτό αποτελεί μια πολύπλοκη διαδικασία, ιδιαίτερα για ένα μάρκετινγκ μάνατζερ ο οποίος συνήθως δεν διαθέτει εξειδικευμένες γνώσεις σχετικά με αλγορίθμους βελτιστοποίησης και θεωρία παιγνίων. Στα πλαίσια αυτά έχουν αναπτυχθεί διάφορα συστήματα μάρκετινγκ τα οποία υποστηρίζουν τους αποφασίζοντες σε αυτό το υψηλής πολυπλοκότητας πρόβλημα.

Τα υφιστάμενα συστήματα χρησιμοποιούν απλά ντετερμινιστικά μοντέλα επιλογής για την προσομοίωση της ανθρώπινης συμπεριφοράς επιλογής, για να μειώσουν την πολυπλοκότητα του προβλήματος. Τα μειονεκτήματα των ντετερμινιστικών μοντέλων επιλογής έχουν καταγραφεί λεπτομερώς στη βιβλιογραφία. Επιπλέον, όλα τα σχετικά συστήματα στοχεύουν στη βελτίωση της επίδοσης της μοναδικής καλύτερης λύσης, η οποία τελικά παρουσιάζεται στον αποφασίζοντα. Όμως ο σχεδιασμός προϊόντος είναι ένα πολύπλοκο, ήμι-δομημένο πρόβλημα το οποίο εμπίπτει τόσο στο πεδίο του μάρκετινγκ όσο και σε αυτό της μηχανικής. Στα πλαίσια αυτά, ενώ η βέλτιστη λύση είναι σημαντική για μια εταιρεία, αυτή μπορεί να είναι τεχνολογικά ανέφικτη ή το κόστος παραγωγής της να είναι απαγορευτικό. Έτσι, είναι εξίσου σημαντικό για έναν μάνατζερ η ύπαρξη ενός

αριθμού σχεδόν βέλτιστων λύσεων, τις οποίες θα μπορεί να αξιολογήσει χρησιμοποιώντας κάποια δευτερεύοντα κριτήρια όπως κόστος παραγωγής, στρατηγική προσέγγιση και τεχνολογικοί περιορισμοί. Επίσης, όλες οι υφιστάμενες προσεγγίσεις στο πρόβλημα θεωρούν την αγορά στατική, όπου οι υπάρχουσες εταιρείες δεν θα αντιδράσουν στην είσοδο των νέων προϊόντων. Όμως, λαμβάνοντας υπόψη μόνο τα υφιστάμενα ανταγωνιστικά προϊόντα γίνεται μια πολύ περιοριστική παραδοχή και η γραμμή που θα σχεδιαστεί μπορεί να αποδειχθεί βέλτιστη μόνο σε βραχυπρόθεσμο χρονικό ορίζοντα. Έχει πλέον καθιερωθεί η άποψη ότι σε μακροπρόθεσμο ορίζοντα, οι αλγόριθμοι βελτιστοποίησης πρέπει να εμπεριέχουν και τις κινήσεις των ανταγωνιστών, οι οποίοι πιθανόν να εισάγουν νέα προϊόντα ή να επανασχεδιάσουν τα υφιστάμενά τους, ως απάντηση στην είσοδο της νέας εταιρείας στην αγορά.

Στην παρούσα διατριβή προτείνεται μια ολοκληρωμένη προσέγγιση στο πρόβλημα του Βέλτιστου Σχεδιασμού Γραμμής Προϊόντων, η οποία συνδυάζει καινοτόμες μεθόδους για τα τρία υποπροβλήματα. Η προσομοίωση της ανθρώπινης συμπεριφοράς επιλογής υλοποιείται μέσω ενός μοντέλου προσομοίωσης αγοράς, το οποίο προσαρμόζει πιθανοτικά μοντέλα επιλογής σε ατομικό επίπεδο. Η Βελτιστοποίηση Σμήνους Σωματιδίων, ένας νέος αλγόριθμος πληθυσμού λύσεων εμπνευσμένος από τη φυσική νοημοσύνη, χρησιμοποιείται για την παροχή ενός συνόλου καλών σχεδόν βέλτιστων λύσεων, από τις οποίες ο αποφασίζων μπορεί να επιλέξει την περισσότερο προτιμητέα. Η δυναμική φύση του ανταγωνισμού μοντελοποιείται μέσω της ισορροπίας κατά Nash, όπου οι υφιστάμενες εταιρείες της αγοράς αντιδρούν επανασχεδιάζοντας τις γραμμές προϊόντων τους. Όλα τα προτεινόμενα μοντέλα αξιολογούνται σε σχέση με τις καλύτερες υπάρχουσες μεθόδους, με χρήση τόσο τεχνητών όσο και πραγματικών δεδομένων σχετικών με καταναλωτικές προτιμήσεις. Το πραγματικό σετ δεδομένων προήλθε από μια έρευνα αγοράς η οποία είχε ως στόχο τη μέτρηση των καταναλωτικών προτιμήσεων σχετικά με το φρέσκο γάλα. Η προτεινόμενη προσέγγιση είναι η πρώτη προσπάθεια ενιαίας αντιμετώπισης των τριών υποπροβλημάτων του βέλτιστου σχεδιασμού μέσω μιας ολοκληρωμένης μεθοδολογίας. Τέλος, αναπτύχθηκε ένα σύστημα μάρκετινγκ το οποίο ενσωματώνει την εν λόγω μεθοδολογία, μέσω μιας αποδοτικής και φιλικής προς το χρήστη διεπαφής.

Abstract

Designing optimal products is one of the most critical activities for a firm to stay competitive. The *Optimal Product Line Design* constitutes a wide area of research in quantitative marketing for over thirty years, which is usually formulated in the context of *Conjoint Analysis*. Dealing with this NP-hard combinatorial optimization problem, a manager must decide on a number of issues: how to simulate the consumer choice process, which optimization algorithm to apply, and how to model the possible retaliatory actions from competitors. The application of an effective approach has several important practical implications for marketing managers, since a bad designed product line may result in a lower than expected market share, or may even cannibalize the firm's existing products. The manager should carefully compare the different alternative choice models, optimization algorithms, game theoretic approaches and choose those that better fit the company's requirements. This constitutes a quite complex task, especially for marketing managers who usually do not have special knowledge concerning optimization algorithms and game theory. In this context, a number of marketing systems have been developed, assisting a manager in this problem of high complexity.

The marketing systems that have been developed so far use simple deterministic choice models for simulating the human choice process in order to reduce the problem's complexity. The limitations of deterministic choice rules regarding the effectiveness of customer choice behavior simulation are well documented in the literature.

Furthermore, all marketing systems aim at improving the performance of a single best solution, which is finally provided to the decision maker. Product design however is a complex and not well formalized discipline that draws upon both marketing and engineering fields. In this context, while it is important for a firm to obtain the optimal solution, this product line configuration may not be technologically feasible, or the production cost may be prohibitive. Hence, it is just as critical for the managers to be provided with a wide range of near-optimal product profiles, in order to assess them

using a number of secondary criteria such as production costs, strategic fit, and technological considerations.

Moreover, all the approaches that have been applied to the Optimal Product Line Design problem assume a static market, where the incumbent firms will not respond to the introduction of one or more new products. However, considering only competitors' current products constitutes a very restrictive assumption, and the product designs derived from such static optimization approaches might proved to be optimal only for the short term. It is now becoming well known that in the longer run, optimization algorithms should take into account the retaliatory actions of the incumbent firms, which may launch new products or redesign their existing ones, as a response to the entrance of a new firm to the market.

In the present thesis, an integrated approach for dealing with the Optimal Product Line Design problem is presented, which combines state of the art methods for the three properties of the problem. The simulation of customer choice behavior is implemented through an innovative market simulation model that individually calibrates probabilistic choice rules. Particle Swarm Optimization, a new population-based algorithm inspired from natural intelligence, is used to provide a set of good near-optimal solutions, from which the decision maker will be able to select the most beneficial one. The concept of Nash equilibrium is employed for modeling the dynamic nature of competition, where each incumbent firm responds to competitive moves by redesigning its product line. All the methods and algorithms proposed are evaluated against the current state of the art approaches, using both simulated and real data regarding consumer preferences. The real data set was obtained from a market survey, the purpose of which was the measurement of customer preferences concerning milk. The proposed approach is the first attempt to integrate the three most important properties of the problem into a single methodology. A marketing system is developed, which integrates the underlying algorithms of the methodology under an efficient and friendly interface.

1 Introduction

Nowadays the economic environment where firms operate has become more competitive than ever. The globalization of the markets, the shorter product life cycles, and the rapid technology development, put high pressure on companies' profitability. In this context new product development constitutes one of the most critical success factors for the viability of a company. Firms that delay in introducing new products or redesigning their existing ones are exposing themselves to great risks, since their current offerings are vulnerable to changing consumer expectations and needs, as well as to increased competition. Whereas the continuous new product launching is crucial for the survival of a company, such procedures entail great risks, since the assets required for the development of a new product are usually high. Under such circumstances, a new product introduction that does not reach the company's expectations may have a catastrophic effect on its profitability. The commercial failure of the Edsel model cost Ford $350 million, while the Concorde aircraft did not recover its investment (Kotler and Keller, 2009). New products continue to fail at a disturbing rate; recent studies put the rate at 95 percent in the United States and 90 percent in Europe (Kotler and Armstrong, 2009).

In order to minimize the associated risks, managers try to assess a new concept's penetration to the market at its early design stage before it enters the production stage. This constitutes a wide area of research in quantitative marketing for over thirty years, known as the *Optimal Product (Line) Design* problem, which is usually formulated in the context of *Conjoint Analysis*. Here, the products are represented by a number of characteristics (*attributes*); a laptop, for example, may be represented by the attributes

monitor, memory etc. Each attribute can take a number of specific levels; the monitor may be 20" or 24", the memory may be 2GB or 4GB etc. The customer preferences concerning laptops can be measured with the use of Conjoint Analysis, which estimates a utility value for each attribute level called *partworth*. The combination of a consumer's partworths for a certain product gives the utility value he assigns to the product. Given a number of competing products, a customer's partwoths can be used for estimating the utility of each product, which can then be converted to choice probability for each product through a *choice model*, which simulates the customer choice behavior. The product choice probabilities can then be aggregated for calculating hypothetical market shares.

In the Optimal Product Line Design problem, a company is interested in designing a number (*line*) of products, the introduction of which to the market will optimize a specific objective (usually market share maximization). The problem's input is the buyers' partworths and the configuration of the competitive products that form the market. An optimization algorithm is used for the optimization of the firm's objective. The problem's complexity depends on the number of products in the line, the number of attributes that form a product, and the number of levels in each attribute. In real world applications, as the number of attributes and levels increases, the number of candidate solutions (product profiles) can grow uncontrollable large, making the managerial task for selecting the appropriate product configuration (combination of attribute levels) practically infeasible. Actually the Optimal Product Line Design problem has been proved to be NP-hard, which means that there is no algorithm that can find and verify the global optimal solution in polynomial time (Papadimitriou & Steiglitz, 1983). In this context, a number of different heuristic approaches have been applied to solve the problem from 1974 until today, the most important being Dynamic Programming (Kohli and Sukumar 1990), Beam Search (Nair, Thakur and Wen 1995), Genetic Algorithms (Alexouda and Paparrizos, 2001; Steiner and Hruschka 2003; Balakrishnan *et al.*, 2004) and Lagrangian Relaxation with Branch and Bound (Camm *et al.*, 2006; Belloni *et al.*, 2008).

1.1 Motivation and Objectives

Whereas the optimization part of the problem has been extensively studied, with the use of state of the art algorithms that provide near-optimal solutions in larger and larger solution spaces, other critical factors that affect the quality of the final solution have received little attention in the related literature. These factors are the choice model used to simulate the consumer choice process, and the behavior of the competing firms after the introduction of the new products to the market. Previous studies use simple choice models in order to reduce the problem's complexity. The *first choice rule* (where each consumer is assumed to deterministically select the product with the highest utility) is usually employed, since it permits the formulation of the problem as a liner program. The limitations of the first choice rule regarding the effectiveness of customer choice behavior simulation are well documented in the literature (Elrod, 1989). Probabilistic choice models provide a better representation of the human choice process than deterministic choice models (Kaul and Rao, 1995), but increase the problem's complexity since the problem becomes non-linear. For this reason, only basic probabilistic choice models, like the MultiNomial Logit (MNL) (McFadden, 1974) or the Bradley-Terry-Luce (BTL) (Bradley, and Terry, 1952; Luce, 1959) have been used in the Optimal Product Line Design problem (Chen and Hausman 2000; Steiner and Hruschka 2003). Nowadays, however, simulating the purchasing behavior of customers is implemented through more sophisticated models. Actually, current state of the art approaches are focusing on *market simulation* models. Recent approaches like the ALPHA rule (Krieger *et al.*, 2004), or the VOICE model (Krieger and Green, 2002), consider the behavior of the entire group of customers that form the market, rather than independently simulating the behavior of each individual. In the Optimal Product Line Design literature, researchers use simple models for simulating the customer choice behavior in order to keep the problem's complexity low. Actually, no approach has utilize market simulation models yet.

Furthermore, the comparison of the different algorithms concerns only the approximation of the optimal solution, whereas marketing practitioners who work on real problems are interested in a number of other more qualitative issues. Except for

Genetic Algorithms, all the approaches developed so far aim at improving the performance of a single best solution, which is finally provided to the decision maker. Product design however is a complex and not well formalized discipline that draws upon both marketing and engineering fields. In this context, while it is important for a firm to obtain the optimal solution concerning customer preferences, this product line configuration may not be technologically feasible, or the production cost may be prohibitive. Hence, it is just as critical for the managers to be provided with a wide range of near-optimal product profiles, in order to assess them using a number of secondary criteria such as production costs, strategic fit, and technological considerations. As Balakrishnan *et al.* (2004) state, the "single best solution" approach denies the decision makers a preferred list of solutions to discuss and select from, but rather tends to have one imposed on them.

Moreover, all the approaches that have been applied to the Optimal Product Line Design problem assume a static market, where the incumbent firms will not respond to the introduction of one or more new products. However, considering only competitors' current products constitutes a very restrictive assumption, and the product designs derived from such static optimization approaches might proved to be optimal only for the short term. It is now becoming well known that in the longer run, optimization algorithms should take into account the retaliatory actions of the incumbent firms, which may launch new products or redesign their existing ones, as a response to the entrance of a new firm to the market. Actually, only two approaches have incorporated competitive reactions using game theory in the conjoint analysis context (Choi and DeSarbo, 1993; Green and Krieger, 1997). However, both approaches solve only the single product design problem, using traditional single-best optimization approaches.

Dealing with this problem of high complexity, a manager must decide on a number of issues: how to simulate the consumer choice process, which optimization algorithm to apply, and how to model the possible retaliatory actions from competitors. The application of an effective approach has several important practical implications for marketing managers, since a bad designed product line may result in a lower than expected market share, or may even cannibalize the firm's existing products. The manager should carefully compare the different alternative choice models, optimization

algorithms, game theoretic approaches and choose those that better fit the company's requirements. This constitutes a quite complex task, especially for marketing managers who usually do not have special knowledge concerning optimization algorithms and game theory. In this context, a number of marketing systems have been developed, assisting a manager in such tricky decisions.

Genetic Algorithms constitute the most advanced optimization method that has been incorporated into a marketing system that deals with the problem (Alexouda, 2005). However the specific marketing system has been implemented in such a way that provides the decision maker with only a single best solution. Hence, it fails to capitalize on the Genetic Algorithm's main advantage, which is the capability to provide a wide range of good solutions. Furthermore, the system uses the first choice rule for simulating the purchasing behavior of customers, while the competition is considered static, and competitors' responses are not incorporated to the system.

The objective of the present thesis is twofold. Firstly, an integrated approach for dealing with the Optimal Product Line Design problem will be developed, which will combine state of the art methods for the three properties of the problem. The simulation of customer choice behavior will be implemented through an innovative market simulation model that individually calibrates probabilistic choice rules. Particle Swarm Optimization, a new population-based algorithm inspired from natural intelligence, will be used to provide a set of good near-optimal solutions, from which the decision maker will be able to select the most beneficial one. The concept of Nash equilibrium will be employed for modeling the dynamic nature of competition, where each incumbent firm responds to competitive moves by redesigning its product line. All the methods and algorithms proposed will be evaluated against the current state of the art approaches, using both simulated and real data regarding consumer preferences (partworths). The real data set was obtained from a market survey, the purpose of which was the measurement of customer preferences concerning milk. The proposed approach is the first attempt to integrate the three most important properties of the problem into a single methodology. The second objective of the thesis is to make this methodology easy to use for a typical marketing manager. For this reason a marketing system will be

developed, which will integrate the underlying algorithms under an efficient and friendly interface.

The rest of the thesis is organized into seven chapters as follows. Chapter 2 provides a brief description of the problem and the critical factors that affect the solution's quality. In chapter 3, the related literature is reviewed, the performance of the algorithms that have been applied to the problem is compared, and the pros and cons of the relevant marketing systems are discussed. The market survey is described in chapter 4. The proposed market simulation model is theoretically validated in chapter 5, and its performance is compared to that of the state of the art approach. In chapter 6 Particle Swarm Optimization is applied to the Optimal Product Line Design problem, and its performance is evaluated against that of Genetic Algorithms regarding a number of variables of interest. Competitive reactions are formulated in the context of Nash equilibrium in chapter 7. Chapter 8 presents the system that incorporates the proposed methodology. Finally, chapter 9 provides an overview of the main conclusions of the study, while its limitations are addressed and future research areas are suggested.

2 The optimal product line design problem

The goal in the optimal product line design problem is the design of a set of products, the introduction of which to the market will maximize an objective of the firm (usually market share). This requires the proper modeling of customer preferences concerning the various product features. In particular, each product is represented as a bundle of attributes (features) which can take specific levels. A personal computer for example, consists of the attributes monitor, processor, hard disk, memory etc., the levels of which are illustrated in Table 2.1. Customers select the levels of the attributes according to their preferences; a civil engineer, for example, will probably choose a large monitor, whereas a mathematician may select a fast processor. Through conducting market surveys, companies can reveal the customer preferences concerning the various product attributes. This is usually done with the application of Conjoint Analysis, which estimates values (called part-worths) for each attribute level, at the individual, segment, or aggregate market level. An example is given in Table 2.1, where the preferences of 2 customers concerning the features of a personal computer are represented as part-worths for each attribute level.

Table 2.1: Part-worths for each attribute level of a personal computer

Attributes	**Levels**	**Partworths**	
		Customer1	***Customer2***
Monitor	17”	0.8	0.1
	19”	0.2	0.3
	20”	0.3	0.4
	24”	0.5	0.9
Processor	Single-core 3,8 GHz	0.1	0.2
	Core-2 2,6 GHz	0.3	0.3
	Core-4 2Ghz	0.9	0.5
Hard disk	200 GB	0.4	0.2
	500 GB	0.6	0.3
	750 GB	0.7	0.5
	1 T	0.4	0.8
Memory	2 GB	0.2	0.1
	4 GB	0.4	0.3
	6 GB	0.9	0.4
Mouse	Cable	0.3	0.1
	Wireless	0.4	0.9
Camera	Embedded	0.3	0.8
	No camera	0.2	0.2

Each customer is assumed to evaluate all product attributes in a simultaneous compensative manner and implicitly assign a utility value to each competing alternative, which is usually represented as the sum of the part-worths of the corresponding attribute levels that comprise the product (linear-additive parth-worth model). The higher the product's utility, the higher the probability to be chosen. Suppose that the two customers whose part-worths are presented in Table 2.1, have to select between PC1 (17", core-4 2GHz, HD 750 GB, 6 GB RAM, cable mouse, no camera) and PC2 (24", Single-core 3,8 GHz, HD 200 GB, 6 GB RAM, wireless mouse, embedded camera). Customer 1 will probably choose PC1 (utility=3.8) over PC2 (utility=2.5), whereas Customer 2 will probably choose PC2 (utility=3.4) over PC1 (utility=1.8). The utilities are converted to choice probabilities for each product through the use of choice models, and are then aggregated across the whole customer base to provide hypothetical market shares. If the part-worths for a population of consumers are known, the introduction of different product configurations (combinations of attribute levels) to the market can be simulated and conditional market shares can be estimated. With the use of optimization algorithms the products that maximize the firm's objective can be found, given the customer preferences and the configuration of the competitive products in the market. An example could be a car manufacturer company that is interested in introducing 3 new car models in different categories (Sport, SUV, Station Wagon) that will provide it with the highest possible volume sales. Next, the different properties of the optimal product line design problem are described.

2.1.1 Preference measurement

Before starting a market simulation the consumers' preferences must be elicited through a market survey. Different procedures can be used such as personal interviews, questionnaires, or computer programs. The products that participate into the survey are assumed to consist of a combination of attributes (quantitative and/or qualitative), each taking a number of distinct levels. Each customer evaluates the competitive products or hypothetical product profiles comprising of different combinations of attribute levels. Various evaluation processes can be used such as ranking or pairwise comparisons of

product profiles, rating of attribute levels or importance weights etc. The preference structure of each individual or market segment is revealed through the use of Conjoint Analysis, the output of which is the values that a consumer assigns to every level of each attribute, the well known part-worths in marketing literature. The (usually linear) combination of the part-worths that correspond to the product's attributes gives its total utility value that an individual expects to obtain from the product.

Customer preferences are estimated at one of three levels of data aggregation. At the individual level, a unique set of part-worths is estimated for each customer. At the segment level, the market is assumed to comprise a number of homogeneous segments, and consumers belonging to the same segment follow the same preference structure. Finally, at the aggregate level a single set of part-worths is calculated for the entire customer population, pooling across the data collected for the whole consumer base.

2.1.2 Choice modeling

Choice modeling is the process of simulating the behavior of a consumer who has to select among a set of alternatives. This is usually implemented through the use of a *choice model*. A choice model is the underlying process by which a customer integrates information to choose a product from a set of competing products. A number of choice models have been developed with varying assumptions and purposes, which differ in the underlying logic structure that drives them (Manrai, 1995). The choice model represents the consumer's purchasing pattern by relating preference to choice. It is a mathematical model which converts the product utilities that an individual assigns to the set of alternatives under consideration, to choice probabilities for each alternative. Choice models can be either deterministic or probabilistic. The *first choice* (or *maximum utility*) *rule* is a deterministic model, which assumes that the individual will always purchase the product with the highest utility. In this case the highest utility alternative receives probability of choice equal to 1, while the rest of the alternatives get a zero probability. The main weakness of the first choice rule is that it displays information only about the product with the maximum utility, while the relative customer preference for the

remaining products is not reflected. Consider a scenario where three consumers have to choose among three brands and assign them the utilities shown in Table 2.2.

Table 2.2: Customer utilities for a 3-product scenario

	Brand A	Brand B	Brand C
Customer 1	0.8	0.7	0.2
Customer 2	0.9	0.8	0.1
Customer 3	0.3	0.8	0.9

The first choice rule exaggerates the share of brand A (66.67%), while underestimating brand B (0%) which, although it receives high utility values from all customers, will never be selected because it is not ranked first. Since the maximum utility model allocates all the choice probability to the first product, regardless of the extend its utility differs from the others, standard errors of the predicted choice shares tend to be higher than the other choice models, especially in small sample sizes. The first choice model is mainly of historical interest, since its deterministic rule fails to adequately represent actual human choice behavior.

It is widely accepted that the human purchasing process reflects a lot of randomness in the real world. A customer will not always buy the brand that perceives best due to out of stock occasions, high search costs, buyer confusion, variety seeking etc. Instead of allocating all choice likelihood to a single brand, probabilistic models distribute choice probabilities among all products, in proportion to their utility value. They incorporate the relative differences in utility values to the choice likelihoods, and allow even the worst alternative to receive a probability of choice. Probabilistic choice models (also known as *share of preference* models) are divided into *constant utility* and *random utility* models. Constant utility models assume that the product utilities are constant and capture the stochastic nature of human behavior, by assuming a level of uncertainty in

the decision rule. The most popular constant utility probabilistic model is the BTL: $P_{ij}=\frac{U_{ij}}{\sum_{j=1}^{n} U_{ij}}$,

where P_{ij} is the probability that consumer i selects product j, U_{ij} is the utility he assigns to product j, and n is the number of competing products. The Pessemier's model (1971) is an extension of BTL, where product utilities are subject to an exponential transformation, which uniformly controls the flatness or steepness of choice probabilities, while preserving the original rank order of preferences:

$$P_{ij}=\frac{U_{ij}^{a}}{\sum_{j=1}^{n} U_{ij}^{a}}$$

where a is a user specified exponent. Small values of a have a flattening effect, overestimating the choice likelihoods of low-utility brands. Large values of a have a sharpening effect, and as a approaches infinity the model turns into a first choice rule. The Lesourne (1977) model uses an exponent of 2: $P_{ij}=\frac{U_{ij}^{2}}{\sum_{j=1}^{n} U_{ij}^{2}}$.

In contrast to models like BTL that assume constant utilities and apply probabilistic choice rules, *random utility* models assume that the consumer always chooses the alternative with the highest utility (U), which consists of two parts. A deterministic component (V) specified as a function of the measured product attributes and individual preferences, and a stochastic term (e) that represents the unmeasured variation in preferences (Baltas and Doyle, 2001): $U = V + e$.

The MNL model assumes independently identically distributed errors (stochastic terms) across the customer population, according to the double exponential distribution. Under this assumption and the principle of utility maximization, the probability that customer i chooses product j is:

$$P_{ij}=\frac{e^{U_{ij}}}{\sum_{j=1}^{n} e^{U_{ij}}}$$

Brice (1997) demonstrated that share of preference models (BTL, MNL) generally predict actual market shares better than the more extreme First Choice model.

2.1.3 Optimization criteria

The first criterion introduced was the maximization of a company's market share, also known as *share of choices* (Shocker and Srinivasan, 1974), which remains the most frequently used objective until today. Later, two more criteria were presented, the seller's welfare (Green et al., 1981) and the buyer's welfare (Green and Krieger, 1988). In the latter, no competition is assumed, and the aim is the maximization of the sum of the utilities that products under design offer to all customers. This is the least frequently used criterion, which mainly concerns product and services offered by public organizations. In the seller's welfare, the goal is the maximization of a firm's profit. This is the most complicated criterion since it requires the incorporation of the marginal return that the firm obtains from each attribute level into the objective function.

2.1.4 Number of products to be designed

The optimal product design problem (one product to be designed) was first formulated by Zufryden (1977). Eight years later Green and Krieger (1985) introduced the optimal product line design problem (two or more products to be designed), which is the main focus of the specific research area today.

2.1.5 Procedure steps

The optimal product line design problem can be formulated either as a one-step or a two-step approach. In the latter, a reference set of candidate alternatives is first defined, and the items that optimize a certain criterion are selected next with the use of a specific algorithm (Green and Krieger, 1985). The problem here is to decide on the size of the reference set of products, and the way that it will be constructed in order to include all potential good solutions. Nowadays, the increase in computers' speed, as well as the development of advanced optimization algorithms, has enabled the design of the items that comprise the line directly from part-worth data in a one-step approach (Green and Krieger, 1988).

2.1.6 Attribute Levels Optimization

In real world applications, as the number of attributes and levels increases, the number of different product profiles raises exponentially, making the selection of the appropriate combination of attribute levels a very complex managerial task. For example in a product category with 7 attributes each taking 6 different levels, the number of possible product profiles is 279,936, while for designing a line of 3 products the number of candidate solutions is over 10^{15}. The exponential increase in the number of candidate solutions with the increase in the number of attributes and levels is illustrated in Table 2.3 (Alexouda, 2004).

Table 2.3: The number of possible solutions (products and product lines) of different problem sizes (source: Alexouda, 2004)

Products in line	*Attributes*	*Levels*	*Number of possible products*	*Number of possible product lines*
2	3	4	64	2016
2	4	3	81	3240
2	4	4	256	32,640
2	5	3	243	29,403
3	3	4	64	41,664
3	4	3	81	85,320
3	4	4	256	2,763,520
3	5	3	243	2,362,041
2	5	4	1024	523,776
2	5	5	3125	4,881,250
2	5	6	7776	30,229,200
2	6	4	4096	8,386,560
2	6	5	15,625	122,062,500
2	6	6	46,656	1,088,367,840
2	7	4	16,384	134,209,536
2	7	5	78,125	3,051,718,750
2	7	6	279,936	39,181,942,080

2	8	4	65,536	2,147,450,880
2	8	5	390,625	76,293,750,000
2	8	6	1,679,616	1,410,554,113,920
3	5	4	1024	178,433,024
3	5	5	3125	5,081,381,250
3	5	6	7776	78,333,933,600
3	6	4	4096	11,444,858,880
3	6	5	15,625	635,660,812,500
3	6	6	46,656	16,925,571,069,120
3	7	4	16,384	732,873,539,584
3	7	5	78,125	79,469,807,968,750
3	7	6	279,936	3,656,119,258,074,240
3	8	4	65,536	46,910,348,656,640
3	8	5	390,625	9,934,031,168,750,000
3	8	6	1,679,616	789,728,812,499,209,000

Kohli and Krishnamurti (1989) proved that the share of choices for the single product design problem is NP-hard, which means that the complete enumeration of the solution space is practically infeasible in tractable time. Kohli and Sukumar (1990) proved the same for the buyer's welfare and the seller's welfare, also for the single product design. In this context many different heuristic approaches have been applied to the problem, the most significant of which are presented in Section 3.1.

2.2 Problem formulation

The formulation of the problem depends on the employed choice model and the selected optimization criterion.

2.2.1 Deterministic choice rules

The most common approach found in the literature is the share of choices problem for the optimal product line design using the first choice rule.

2.2.1.1 Share of choices

Here, each individual is assumed to have an existing favorite product called *status quo*. The problem can be formulated as a 0-1 integer program, with the use of the following parameters (Kohli and Sukumar, 1990):

$\Omega = \{1,2,\ldots, K\}$ is the set of K attributes that comprise the product.

$\Phi_k = \{1,2,\ldots, J_k\}$ is the set of J_k levels of attribute k.

$\Psi = \{1,2,\ldots M\}$ is the set of products to be designed.

$\theta = \{1,2,\ldots I\}$ is the set of I customers.

w_{ijk}= is the part-worth that customer $i \in \theta$ assigns to level $j \in \Phi_k$ of attribute $k \in \Omega$.

j^*_{ki} = is the level of attribute $k \in \Omega$ of customer's $i \in \theta$ status quo product.

$c_{ijk} = w_{ijk} - w_{ij^*k}$ is the relative difference in the part-worth that customer $i \in \theta$ assigns between level j and level j^* of attribute $k \in \Omega$.

Since the firm may already offer a number of products, the set of customers whose current status quo product is offered by a competitor is indexed as $\theta' \subset \theta$. In this way the company aims at gaining the maximum possible number of clients from its competitors, without cannibalizing its existing product line. Three decision variables are also used:

$$x_{jkm} = \begin{cases} 1, \text{ if the level of product's m attribute k is j,} \\ 0, \text{ otherwise} \end{cases}$$

$$x_{im} = \begin{cases} 1, \text{ if product's m utility for customer i is less than his status quo,} \\ 0, \text{ otherwise} \end{cases}$$

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

In this context the share of choices problem in the product line design using a deterministic rule is formulated as follows:

$$\min \sum_{i \in \theta'} x_i \tag{1}$$

subject to

$$\sum_{j \in \Phi_\kappa} x_{jkm} = 1, \qquad k \in \Omega,\ m \in \Psi, \tag{2}$$

$$\sum_{k \in \Omega} \sum_{j \in \Phi_\kappa} c_{ijk} x_{jkm} + y_{im} > 0, \qquad i \in \theta',\ m \in \Psi, \tag{3}$$

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

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

Constraint (2) requires each product in the line to be assigned exactly one level of each attribute. Constraint (3) restricts x_{im} to be 1 only if customer i prefers his status quo to product m. Constraint (4) forces x_i to be 1 only if $x_{im} = 1$ for all $m \in \Psi$, that is if customer i prefers his status quo to all products in the line. Constraint (5) represents the binary restrictions regarding the problem's decision variables. The objective function (1) minimizes the number of instances for which $x_i = 1$, and hence minimizes the number of customers who decide to be loyal to their status quo products (which is equivalent to

maximizing the number of customers who switch from their status quo to a product from the company's line).

2.2.1.2 Buyer's welfare

In this case no status quo product is assumed for the customer (buyer), who will select the item from the offered line that maximizes his utility. The following decision variable is used:

$$x_{ijkm} = \begin{cases} 1, \text{ if level j of attribute k appears in product m, and buyer i,} \\ 0, \text{ otherwise} \end{cases}$$

The problem can be formulated as a 0-1 integer program as follows (Kohli & Sukumar, 1990):

$$\max \sum_{i\in\theta}\sum_{m\in\Psi}\sum_{k\in\Omega}\sum_{j\in\Phi_\kappa} w_{ijk} x_{ijkm} \tag{6}$$

subject to

$$\sum_{j\in\Phi_\kappa}\sum_{m\in\Psi} x_{ijkm} = 1, \quad i\in\theta,\ k\in\Omega, \tag{7}$$

$$\sum_{j\in\Phi_\kappa} x_{ijkm} - \sum_{j\in\Phi_{\kappa'}} x_{ijk'm} = 0, \quad k'>k, \quad k,k'\in\Omega,\ i\in\theta,\ m\in\Psi, \tag{8}$$

$$x_{ijkm} + x_{i'j'km} \le 1, \quad i'>i,\ j'>j,\ i,i'\in\theta,\ j,j'\in\Phi\kappa, k\in\Omega,\ m\in\Psi, \tag{9}$$

$$x_{ijkm} = 0, 1 \text{ integer}, \ i\in\theta,\ j\in\Phi_\kappa,\ k\in\Omega,\ m\in\Psi \tag{10}$$

Constraint (7) requires that, across products, only one level of an attribute be associated with a specific buyer. Constraint (8) requires that, across attributes, the level assigned to buyer $i\in\theta$ must correspond to the same product. Constraint (9) requires that for all buyers assigned to a specific product, the same level of an attribute must be specified. Together, these three constraints require that each buyer be assigned one of the

products in the line. The objective function (6) selects the products (attribute levels combination) to maximize the total utility across buyers.

2.2.1.3 Seller's welfare

Kohli and Sukumar (1990) provide a detailed description of the seller's welfare problem, where the firm wants to maximize the marginal utility obtained by the introduction of a line of M new products. The seller may already offer some products in the market, and competition is represented through the existence of a current status quo product for each customer. If customer $i \in \theta$ selects a product in which level $j \in \Phi_\kappa$ of attribute $k \in \Omega$ appears, the seller is assumed to obtain a utility value u_{ijk}. The seller's marginal return obtained from level $j \in \Phi_\kappa$ of attribute $k \in \Omega$ is:

$d_{ijk} = u_{ijk} - u_{ij^*k}$, if customer $i \in \theta$ switches from a product offered by the seller

$d_{ijk} = u_{ijk}$ if customer $i \in \theta$ switches from a product offered by a competitor

The problem can be formulated as a 0-1 integer program as follows:

$$\max \sum_{i\in\theta}\sum_{m\in\Psi}\sum_{k\in\Omega}\sum_{j\in\Phi_\kappa} d_{ijk}\, x_{ijkm}\, y_i \tag{11}$$

subject to

$$\sum_{m\in\Psi}\sum_{k\in\Omega}\sum_{j\in\Phi_\kappa} w_{ijk}\,(x_{ijkm} - x_{i'jkm}) \geq 0 \quad , \quad i \neq i',\ i \in \theta, \tag{12}$$

$$y_i \sum_{m\in\Psi}\sum_{k\in\Omega}\sum_{j\in\Phi_\kappa} w_{ijk}\, x_{ijkm} \geq y_i u_i^* \quad , \quad i \in \theta \tag{13}$$

$y_i = 0, 1$ integer, and (7), (8), (9), (10).

Constraints (7)-(10) require, as in the buyer's welfare problem, that a specific product is assigned to each customer, and that each product in the line be assigned exactly one level of each attribute. Constraint (12) requires that each customer is assigned to the product that maximizes his utility. Constraint (13) requires that the seller obtains a return from customer i only if the utility of the new item assigned to the customer is

higher than the utility of his status quo product. The objective function (11) selects the products to maximize the seller's total return from the products in the line.

2.2.2 Probabilistic choice models

When probabilistic choice models are used, the market is assumed to consist of N competitive products with known configurations, including the M candidate items for the firm's line:

$\Xi = \{1,2,\ldots N\}$ is the set of products that comprise the market.

2.2.2.1 Share of choices

As before $\Psi \subset \Xi$ is the set of products to be designed. Customers do not have a status quo product, and do not deterministically choose the highest utility product. Instead, it is assumed that each of the N alternatives has a certain probability to be selected, which is calculated with the use of a probabilistic choice model. Using BTL for example, the probability that customer i will choose product m is estimated as follows:

$$P_{im} = {U_{im}}\Big/{\sum_{n\in\Xi} U_{in}}, \qquad i\in\theta,\ m\in\Psi,\ n\in\Xi, \tag{14}$$

where U_{im} the utility that customer i assigns to product m (sum of its part-worths):

$$u_{im} = \sum_{k\in\Omega}\sum_{j\in\Phi_\kappa} w_{ijk}\, x_{jkm}, \quad i\in\theta,\ j\in\Phi_\kappa,\ k\in\Omega,\ m\in\Psi.$$

In this context the problem is formulated as the following non-linear program:

$$\max \sum_{m\in\Psi}\sum_{i\in\theta} P_{im} \tag{15}$$

subject to

$x_{jkm} = 0, 1$ integer, and (2).

The objective function (15) maximizes the market share of the m products (probability to be purchased) of the company's line.

2.2.2.2 Seller's welfare

Green and Krieger (1992) presented the seller's welfare problem, in an application of the SIMOPT program to pharmaceutical products. In order for the company's profit to be maximized, variable (depending on attribute levels) and fixed costs for each product must be included in the objective function. The variable cost per unit for a product m is given by the following linear additive function:

$$c_m^{(\text{var})} = \sum_{k \in \Omega} \sum_{j \in \Phi_\kappa} c_{jk}^{(\text{var})} x_{jkm} \quad , \quad j \in \Phi_\kappa,\ k \in \Omega,\ m \in \Psi,$$

where

$c_{jk}^{(\text{var})}$ the variable cost of attribute's k level j for the seller.

A similar function is used for the fixed cost of product m:

$$c_m^{(fix)} = \sum_{k \in \Omega} \sum_{j \in \Phi_\kappa} c_{jk}^{(fix)} x_{jkm} \quad , \quad j \in \Phi\kappa,\ k \in \Omega,\ m \in \Psi.$$

If p_m denotes item's m price, the problem is formulated as the following non-linear program:

$$\max \sum_{m \in \Psi} \left[(p_m - c_{jk}^{(\text{var})}) \sum_{i \in \theta} P_{im} I - c_m^{(fix)} \right] \tag{16}$$

subject to

x_{jkm} = 0, 1 integer, and (2).

The objective function (16) maximizes the total seller's profit obtained from the introduction of a line of M products.

3 Previous Approaches

As discussed in the previous chapter the Optimal Product Line Design problem has been proved to be NP-hard, hence there is no algorithm that can find and verify the global optimal solution in polynomial time. From its establishment in 1974 until today, a number of optimization approaches have been applied to the problem in order to provide (near) optimal solutions in tractable time. Some of the proposed algorithms have been incorporated to intelligent marketing systems, which assist a manager in this problem of high complexity. Most approaches employ the deterministic first choice rule due to its simple form. The probabilistic approaches use either the BTL or the MNL, with the exception of Krieger and Green (2002) who use the Pessemier model with a different exponent for each customer. Almost all approaches maximize the firm's market share, and only a few deal with the other two objectives. Two-step approaches as well as single-product optimization approaches are mostly reported in the first studies, whereas the last few years researchers have focused on one-step approaches that optimize product lines. With the exception of only two studies, all other approaches consider the competition to be static. That is, the rivals do not alter their current products or introduce new ones after the firm optimizes its product line. On the contrary, Choi and DeSarbo (1993), and Green and Krieger (1997) propose a dynamic market, where each competitor responds by optimizing its product line until a Nash equilibrium is reached. Seven algorithms have been incorporated into marketing systems. Table 3.1 summarizes the 24 most important approaches that have been reported in the literature.

Table 3.1: Approaches applied to the optimal product (line) design problem

Paper	Choice rule	Objective	Steps	Algorithm	Products	Rival	System
Shocker & Srinivasan 1974	Deterministic	Share, Profit	One	Gradient search, Grid search	Single	Static	
Zufryden 1977	Deterministic	Share	One	Mathematical programming	Single	Static	ZIPMAP
Green, Carroll & Goldberg 1981	Deterministic, Probabilistic	Share, Profit	One	Response Surface methods	Single	Static	QUALIN
Green & Krieger 1985	Deterministic	Share	Two	Greedy Heuristic, Interchange Heurist	Line	Static	DESOP, LINEOP
Kohli & Krishnamusti 1987	Deterministic	Share	One	Dynamic Programming	Single	Static	
Green & Krieger 1988	Probabilistic	Share, Profit, Buyers welfare	One	Divide & Conquer	Line	Static	SIMOPT
McBride & Zufryden 1988	Deterministic	Share	Two	Mathematical programming	Line	Static	
Sudharshan, May & Gruca 1988	Deterministic, Probabilistic	Share	One	Non linear programming	Line	Static	DIFFSTRAT

Green, Krieger & Zelnio 1989	Probabilistic	Share	Two	Coordinate Ascent	Line	Static	PROSIT
Kohli & Sukumar 1990	Deterministic	Share, Profit, Buyers welfare	One	Dynamic Programming	Line	Static	
Dobson & Kalish 1993	Deterministic	Share, Profit, Buyers welfare	Two	Greedy Heuristic	Line	Static	
Choi & DeSarbo 1993	Deterministic	Profit	One	Branch and Bound	Single	Dynamic	
Nair, Thakur & Wen 1995	Deterministic	Share, Profit	One	Beam Search	Line	Static	
Balakrishnan & Jacob 1996	Deterministic	Share, Buyers welfare	One	Genetic algorithm	Single	Static	
Green & Krieger 1997	Deterministic	Profit	One	Divide & Conquer	Single	Dynamic	
Chen & Hausman 2000	Probabilistic	Profit	Two	Non linear programming	Line	Static	

Alexouda & Paparrizos 2001	Deterministic	Profit	One	Genetic algorithm	Line	Static	MDSS
Shi, Olafsson & Chen 2001	Deterministic	Share	One	Nested Partitions	Line	Static	
Krieger & Green 2002	Probabilistic	Share	One	Greedy Heuristic	Single	Static	
Steiner & Hruschka 2003	Probabilistic	Profit	One	Genetic algorithm	Line	Static	
Alexouda 2004	Deterministic	Share	One	Genetic algorithm	Line	Static	MDSS
Balakrishnan, Gupta & Jacob 2004	Deterministic	Share	One	Genetic algorithm	Line	Static	
Camm, Cochran, Curry & Kannan 2006	Deterministic	Share	One	Branch and Bound with Lagrangian relaxation	Single	Static	
Belloni, Freund, Selove & Simester 2008	Deterministic	Profit	One	Branch and Bound with Lagrangian relaxation	Line	Static	

3.1 Optimization algorithms applied to the problem

In this section the most important algorithms that have been applied to the Optimal Product Line Design problem are reviewed and evaluated. Emphasis is placed on Genetic Algorithms, since it is the only population based algorithm that has been applied to the problem, and it constitutes the most advanced optimization algorithm that has been incorporated into a marketing system.

3.1.1 Greedy Heuristic

Introduced by Green and Krieger (1985), this heuristic proceeds in two steps. At the first step a "good" set of reference products is created. The second step begins by choosing the best alternative among the candidate products. Then, the second alternative is selected from the reference set, which optimizes the objective function provided that the first product is already included in the market. The procedure iterates by adding one product at a time until the desired number of products in the line has been reached. In another paper, Green and Krieger (1987) describe the "best in heuristic" for developing the set of reference products. Initially the product profile that maximizes the utility $u1_{max}$ of customer 1 is found through complete enumeration of the attribute levels. If customer's 2 utility for customer's 1 best product is within a user specified fraction ε of $u2_{max}$, then customer's 2 best product is not added to the set; otherwise it is. As the method proceeds through the group of customers, all of the products currently on the set are tested to see if any are within ε of uk_{max} for customer k, and the previous rule is applied. The process is usually repeated through randomized ordering of the customers, and different values of ε, depending on the desired size of the set. Local optimality is not guaranteed, as it depends on the first product added to the line.

3.1.2 Interchange Heuristic

In the same paper, Green and Krieger (1985) introduced another method where initially, a product line is randomly selected and its objective value is estimated. Next, each alternative from the reference set is checked to see whether there exists a product in the line, the replacement of which by the specific alternative will improve the line's value. If this condition holds, the alternative is added, and the product that is removed is the one

that results in the maximum improvement of the line's value. The process is repeated until no further improvement is possible. The authors recommend the use of the solution provided by the Greedy Heuristic, as the initial product line. The Interchange Heuristic guarantees local optimality, where the local neighborhood includes all solutions that differ from the existing by one product.

3.1.3 Divide and Conquer

In this approach, developed by Green and Krieger (1988), the set of attributes *K* that comprise the product line is divided into two equal subsets *K1* and *K2*. First, the levels of attributes belonging to *K1* that are good approximations of the optimal solution are estimated. The authors suggest averaging the part-worths within each level of each attribute, and selecting for each attribute the level with the highest average. In each iteration, the values of the attributes belonging to the one subset are held fixed, while the values of the other subset are optimized through an exhaustive search. If the search space is too large for completely enumerating half of the attributes, the set of attributes can be divided into more subgroups, at the risk of finding a worst solution. Local optimality is guaranteed, where the local neighborhood depends on the number of subsets.

3.1.4 Coordinate Ascent

Green *et al.* (1989), propose a heuristic that can be considered as a Coordinate Ascent implementation. A product line is initially formed at random and evaluated. The algorithm then iterates through each product attribute in a random order, and assesses each possible level. The altering of an attribute's level is acceptable if it improves the solution's quality. Only a single attribute change is assessed at a time (one opt version), and the algorithm terminates when no further improvement is possible. Local optimality is guaranteed, with the local neighborhood including all solutions that differ from the existing one by a single attribute.

3.1.5 Dynamic Programming

Kohli and Krishnamusti (1987), and Kohli and Sukumar (1990) use a dynamic programming heuristic for solving the optimal product and product line design problems respectively. Here, the product (line) is built one attribute at a time. Initially, for each level of attribute *B*, the best level of attribute *A* is identified, forming in this way a number of partial product profiles, equal to attribute's *B* number of levels. Next, for each level of attribute *C*, the best partial profile (consisting of attributes *A* and *B*) that was built in the previous step is identified. The method proceeds until all product(s) attributes have been considered. Finally, the product (line) that optimizes the desired criterion is selected among the full profiles constructed. The quality of the final solution is highly dependent to the order in which the attributes are considered, thus multiple runs of the heuristic using different attribute orderings are recommended. No local optimality is guaranteed.

3.1.6 Beam Search

Nair *et al.* (1995) solved the product line design problem using Beam Search. BS is a breadth-first process with no backtracking, where at any level of the search only the *b* (Bean Width) most promising nodes are further explored in the search tree. The method begins with *K* relative part-worth matrices C(k) (with elements $c_{ij} = w_{ij} - w_{ij^*}$), and initializes work matrices $A_1(\bullet)$ based on C. At each stage l (layer), matrices $E_1(\bullet)$ of combined levels are formed, by combining two matrices $A_l(\bullet)$ at a time in the given order. Then, the *b* most promising combinations of levels are selected to form columns in new matrices $A_{l+1}(\bullet)$ in the next layer, where it remains approximately half of the number of matrices in the previous layer. In this way, unpromising attribute levels are iteratively pruned, until a single work matrix remains. This final matrix consists of *b* columns, each containing a full product profile. These are the candidate alternatives for the first product in the line. For the first of the *b* alternatives, the data set is reduced by removing the customers who prefer this product over their status quo. The previous process is repeated for finding one second-product in the line, and iterated until *M* products are build that form a complete product line. The same procedure is repeated, until *b* complete product lines are designed, from which the one that gives the best value in the objective function is selected. The final solution depends on the way of pairing the different attribute combinations at each layer. The authors suggest a best-worst pairing, which gives better results than the random one. No local optimality is guaranteed.

3.1.7 Nested Partitions

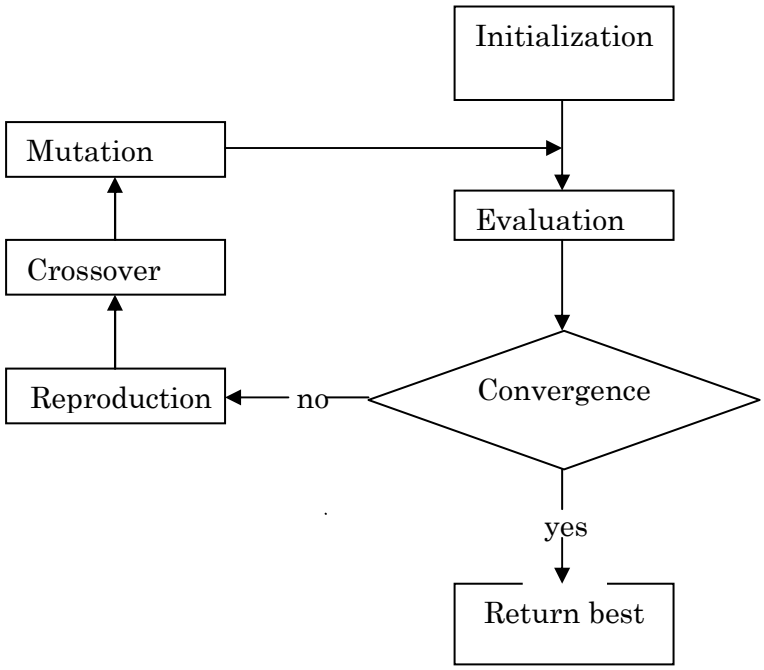
In the Nested Partitions implementation (Shi *et al.*, 2001), a region is defined by a partial product line profile, for example all products that contain a specific attribute level. In each iteration a subset of the feasible region is considered the most promising, which is further partitioned into a fixed number of subregions, by determining the level of one more attribute, and aggregating what remains of the feasible region into one surrounding region. In each iteration therefore, the feasible region is covered by disjoint subregions. The surrounding region and each of the subregions are sampled using a random sampling scheme, through which random levels are assigned to the remaining attributes. The randomly selected product profiles are evaluated, in order for an index to be estimated that determines which region becomes the most promising in the next iteration. This region is then nested within the last one. If the surrounding region is found to be more promising than any of the regions under consideration, the method backtracks to a larger region using a fixed backtracking rule. NP combines global search through partitioning and sampling, and local search through calculation of the promising index. The method can incorporate other heuristics to improve its performance. The authors tried a Greedy Heuristic, as well as a Dynamic Programming into the sampling step, and a Genetic Algorithm into the selection of the promising region. The results of their study indicate that the incorporation of each of the three heuristics is beneficial, with GA giving the best performance.

3.1.8 Genetic Algorithms

Genetic Algorithms are optimization techniques that were first introduced by Holland (1975). They are based on the principle of "natural selection" proposed by Darwin a long time ago, and constitute a special case of Evolutionary Programming algorithms. In accordance with Biology science, GAs represent each solution as a chromosome that consists of genes (variables), which can take a number of values called alleles. A typical GA works as illustrated in Figure 3.1. Initially a set of chromosomes (population) is generated. If prior knowledge about the problem exists, it is used to create possible "good" chromosomes; else the initial population is generated at random. Next, the problem's objective function is applied to every chromosome of the population, in order

for its fitness (performance) to be evaluated. The chromosomes that will be reproduced to the next generation are then selected according to their fitness score, that is, the higher the chromosome's fitness the higher the probability that it will be copied to the subsequent generation. Reproduction ensures that the chromosomes with the best performance will survive to the future generations, a process called "survival of the fittest", so that high quality solutions will not be lost or altered.

Figure 3.1: Genetic Algorithm flowchart



A mating procedure follows, where two parents are chosen to produce two offspring with a probability p_c, through the application of a crossover operator. The logic behind crossover is that a chromosome may contain some "good" features (genes) that are highly valued. If two chromosomes (parents) exchange their good features then there is a great possibility that they will produce high performance chromosomes (offspring) that will combine their good features. The expectation is that from generation to generation, crossover will produce new higher quality chromosomes. Subsequently, each of the newly formed chromosomes is selected with a probability p_m to be *mutated*. Here one of its genes is chosen randomly and its value is altered to a new one randomly generated. Mutation produces new chromosomes that would never be created through crossover. In this way, entirely new solutions are produced in each generation, enabling the algorithm to search new paths and escape from possible local minima. Whereas reproduction

reduces the diversity of the population, mutation maintains a certain degree of heterogeneity of solutions, which is necessary to avoid premature convergence of the evolutionary process. However, mutation rates must be kept low, in order to prevent the disturbance of the search process that would lead to some kind of random search. Finally, if the convergence criterion has been met, the algorithm stops and the best solution so far is returned; else it continues from the population's evaluation step.

3.1.8.1 Type of problems solved

GAs were first applied to the optimal product design problem by Balakrishnan and Jacob (1996), who dealt with the share of choices and the buyer's welfare problem, by employing the first choice rule. The authors provide a number of advantages that leaded them to use this approach. The search is implemented from a set of points (equal to the size of the population) rather than a single point, increasing in this way the method's exploratory capability. GAs do not require additional knowledge, such as the differentiability of the function; instead they use the objective function directly. GAs do not work with the parameters themselves but with a direct encoding of them, which make them especially suited for discontinuous, high-dimensional, and multimodal problem domains, like the optimal product design. Later, Alexouda and Paparrizos (2001) applied GAs to the seller's welfare problem for the optimal product line design, while Alexouda (2004), as well as Balakrishnan *et al.* (2004) dealt with the share of choices problem. All three approaches employed the first choice rule. The only approach that uses probabilistic choice rules is that of Steiner and Hruschka (2003), who dealt with the seller's welfare problem.

3.1.8.2 Problem representation

Except for Balakrishnan et al. (2004), all other approaches adopted a binary representation scheme. In Balakrishnan and Jacob (1996), each product is represented by a chromosome, which is divided into K substrings that correspond to the product's attributes. Each substring consists of J_k (the number of attribute's k levels) genes that take values (alleles) 0 or 1. Hence the length of a chromosome is $P=\sum_{k\in\Omega} j_k$. A value of 1 denotes the presence of the specific level in the corresponding attribute, and a value of 0 its absence. This representation has the restriction that exactly one gene must take the

value of 1 in each substring. For example, it is assumed that a personal computer consists of the attributes processor (Single-core 3,8 GHz, Core-2 2,6 GHz, Core-4 2Ghz), monitor (17", 19", 20", 24"), and hard disk (200 GB, 500 GB, 750 GB). Then a Core-2 2,6 GHz with 20" monitor and 750 GB hard disk will be represented by the chromosome C={010 0010 001}. In Alexouda and Paparrizos (2001), Steiner and Hruschka (2003), and Alexouda (2004), a chromosome corresponds to a line of products. Each chromosome is composed of M^*K substrings that represent the product's attributes, each consisting of J_κ genes that take values 0 or 1. As before, a value of 1 denotes the presence of the specific level in the corresponding attribute, and a value of 0 its absence. The restriction that exactly one gene must take the value of 1 in each substring also holds here. The length of each chromosome is $P=M^*\sum_{k\in\Omega} j_k$. Referring to the personal computer example, the chromosome D={010 0010 001|100 0001 010} represents a line of two products; a Core-2 2,6 GHz with 20" monitor and 750 GB hard disk, and a single-core 3,8 GHz with 24" monitor and 500 GB hard disk.

Balakrishnan *et al.* (2004) use an integer representation scheme, where a chromosome corresponds to a line of products, a gene to an attribute, and the gene's values to attribute levels. Hence, each chromosome is of length M^*K, and is divided into M substrings, each representing a product in the line. Within each substring, gene k can take J_κ different values. The line of the two products described by chromosome D above, is represented in this case by chromosome E={233|142}. Here, the authors raise an issue concerning the principal of minimal redundancy, according to which each member of the space being searched should be represented by one chromosome only (Radcliffe, 1991). The integer representation scheme does not adhere to this principle, since the same line of products can be represented by $M!$ different chromosomes. The previous PC product line, for instance, can also be represented by the chromosome E'={142|233} (the two products exchange their positions). This could cause inefficiencies in the search process, as the crossover between two identical products (E and E') may result in two completely different sets of offspring. On the other hand, it may prove to be an advantage, as more members of the search space will probably be explored. In order to alleviate this concern, they adopt an alternative representation scheme where the substrings (products) in a chromosome are arranged in lexicographic order. That is, product 111 is before 112 which is before 121 etc. In this encoding, called sorted representation, the chromosome E would not exist. They tested both the sorted and the unsorted representations.

3.1.8.3 Genetic Algorithm's parameters

Balakrishnan and Jacob (1996) modeled the problem with the use of matrices. The GA population (number of chromosomes) has a size of N, and is stored in the matrix POP_{N*P}. Customers' preferences (part-worths for each attribute level) are maintained in the matrix $BETA_{I*P}$. The utilities that each of the I customers assigns to each of the N products (represented by chromosomes) are estimated in each generation, and stored in the matrix PRODUTIL= BETA*POP^T. For the share of choices problem the utility of each customer's status quo product is maintained in the matrix STATQUO. The chromosome n is evaluated through the comparison of the n-th column in PRODUTIL with the corresponding in STATQUO. The fitness of the chromosome is the number of times that PRODUTIL(i,n)>STATQUO(i,n), i=1…I, that is the number of customers that prefer the new product to their status quo. For the buyer's welfare problem the fitness of the chromosome n is the sum of elements of the n-th column in PRODUTIL, that is the aggregate utility value for the whole set of customers.

3.1.8.3.1 Initialization of the population

All five approaches initialize the GA population in a totally random manner. Furthermore, Alexouda and Paparrizos (2001), Alexouda (2004), and Balakrishnan *et al.* (2004), also assess the performance of a hybrid strategy in respect to the initialization of the population. Before running the GA, a Beam Search heuristic is applied and the best solution found is seeded into the genetic algorithm's initial population, while the remaining N-1 chromosomes are randomly generated. The population size is set to 100 (Balakrishnan and Jacob, 1996), 150 (Alexouda and Paparrizos, 2001; Steiner and Hruschka, 2003), 180 (Alexouda, 2004), or 400 (Balakrishnan et al., 2004).

3.1.8.3.2 Reproduction

Except for Steiner and Hruschka (2003), all other approaches adopt an elitist strategy for the process of reproduction, where the F fittest chromosomes are copied intact into the next generation. Such an approach ensures that the best chromosomes will survive to the subsequent generations. The value of F ranges from $4N/10$ (Alexouda and Paparrizos, 2001; Alexouda, 2004), to $N/2$ (Balakrishnan and Jacob, 1996; Balakrishnan *et al.*, 2004). Steiner and Hruschka (2003) employ a binary tournament selection procedure, where $N/2$ pairs of chromosomes are randomly selected with replacement,

and from each pair only the chromosome with the higher fitness value survives to the succeeding generation. This is a semi-random process, which ensures that the chromosomes with higher fitness values have more probabilities to survive.

3.1.8.3.3 Crossover

In the approaches that adopt a binary representation scheme, the unit of interest in the crossover procedure is the substring, in order for feasible solutions to be produced. In Steiner and Hruschka (2003) for example, who use one-point crossover with probability p_c=0.9 and random selection of the cross site, the crossover of the two parents

A = {010 0010 001 | 100 0001 010} and

B = {100 0100 010 | 010 0010 100},

after the second substring will generate the two offspring

A' = {010 0010 010 | 010 0010 100} and

B' = {100 0100 001 | 100 0001 010}.

Except for the above approach, the other ones employ a uniform crossover with the probability p_c taking the values 0.4 (Alexouda and Paparrizos, 2001), 0.45 (Alexouda, 2004) and 0.5 (Balakrishnan and Jacob, 1996). In the approach that employs an integer representation scheme, the unit of interest in crossover is the gene. If for instance, the two parents

S={122 | 323} and

T={141 | 421},

exchange their second and sixth genes, this will generate the offspring

S'={142 | 321} and

T'={121 | 423}.

When the sorted representation is used, the offspring are sorted in lexicographic order after the crossover operation. According to Radcliffe (1991), a *forma* specifies at certain chromosome's positions (called defining positions) particular values that all its instances must contain. That is, if a chromosome η is an instance of a forma β, then η and β both

contain the same values at the specified positions. Chromosomes S and T, for example, both belong to the forma:

β = 1** *2*,

where the * denotes a "don't care" value. The principle of respect defines that the crossover of two chromosomes that belong to same forma must produce offspring also belonging to the same forma. Whereas in the unsorted representation the crossover is "respectful", the property does not hold in the sorted representation, due to the ordering of the attributes after the crossover.

3.1.8.3.4 Mutation

Except for the one with the integer representation scheme, in all other approaches the mutation operator is applied at the substring level. Chromosomes are randomly selected (without replacement) with a probability p_m (mutation rate). An attribute (substring) of the selected chromosome is randomly picked and its level is altered. If, for instance, chromosome A is chosen to be mutated at the second substring, a potential mutated chromosome will be A"={010 1000 001 | 100 0001 010}. In Balakrishnan *et al.* (2004), the mutation takes place at the gene level, while two different mutation operators are used. Except for the standard mutation operator, a hybridized one is employed, which uses as a mutator chromosome the best solution found by the Beam Search heuristic. Whenever a chromosome is selected for mutation, a gene is randomly selected and its value is either randomly changed using the standard mutation operator, or altered to the value contained in the specific attribute of the mutator chromosome. In this way the good attribute values of the BS best solution will be copied to the GA population. On the other hand, this may result in premature convergence to the alleles of the mutator string. In order to avoid this, the two mutator operators have equal probability to be applied. The mutation rate takes a wide range of values: 0.05 (Steiner and Hruschka, 2003), 0.1 (Alexouda, 2004), 0.2 (Alexouda and Paparrizos, 2001), 0.3 (Balakrishnan and Jacob, 1996), or 0.4 (Balakrishnan *et al.*, 2004).

3.1.8.3.5 Stopping criterion

From the entire set of chromosomes only the N fittest are maintained to the next generation, and the algorithm iterates until a stopping criterion is met. Balakrishnan

and Jacob (1996), Steiner and Hruschka, (2003), and Balakrishnan *et al.* (2004) employ a moving average rule, where the algorithm terminates when the percentage change in the average fitness of the best three chromosomes over the five previous generations is less than 0.2% (convergence rate). In the other two approaches the procedure terminates when the best solution does not improve in the last 10 (Alexouda and Paparrizos, 2001), or 20 (Alexouda, 2004) generations.

3.1.8.4 Performance evaluation

3.1.8.4.1 Genetic Algorithm vs. Dynamic Programming

Balakrishnan and Jacob (1996) compared the results of their approach and the Dynamic Programming approach (Kohli and Krishnamusti, 1987) with the complete enumeration solutions in 192 data sets, in both the share of choices and buyer's welfare problems. A full factorial experimental design was generated using the factors and levels presented in Table 3.2.

Table 3.2: Factors and levels used in the experiment

Factor	Levels			
Number of attributes	4	6	8	
Number of attribute levels	2	3	4	5
Number of customers	100	200	300	400

The part-worths were randomly generated following a normal distribution, and normalized within each customer to sum to 1. Random was also the generation of each customer's status quo product. Four replications were performed in each case resulting in a total of 192 data sets. In the share of choices problem, the average best solution provided by GA was 99.13% of the optimal product profile found by complete enumeration, while the same value for the DP was 96.67%. GA also achieved a tighter standard deviation (0.016) than that of DP (0.031). In the buyer's welfare problem the

respective values were 99.92% for the GA with 0.0028 std, and 98.76% for the DP with 0.0165 std. The number of times that the optimal solution was found (hit rate) was 123 for the GA and 51 for the DP in the share of choices, and 175 for the GA and 82 for the DP in the buyer's welfare. The performance of GA was also compared with that of DP in two larger problems of sizes 326,592 and 870,912, where an exhaustive search was infeasible in tractable time. The data sets consisted of 200 customers, and 9 attributes that take (9,8,7,6,2,2,3,3,3) or (9,8,8,7,6,2,2,3,3) levels, while ten replications for each data set were performed. GA showed a better, worse, and equal performance compared to DP in 11, 3, and 6 data sets for the share of choices, and in 8, 3, and 9 data sets respectively for the buyer's welfare.

3.1.8.4.2 Genetic Algorithm vs. Greedy Heuristic

Steiner and Hruschka (2003) compared the results of their approach and the Greedy Heuristic approach (Green and Krieger 1985) with the complete enumeration solutions, in the seller's welfare problem. A factorial experimental design was generated using the factors and levels presented in Table 3.3.

Table 3.3: Factors and levels used in the experiment

Factor	Levels		
Number of attributes	3	4	5
Number of attribute levels	2	3	4
Number of products in the line	2	3	4
Number of competing firms	1	2	3

From the 81 different cases a subset of 69 was considered. Four replications were performed under each case, resulting in a total of 276 problems solved, where customer part-worths, attribute level costs, and competitive products configuration were randomly generated. The value of the solution found by GA was never less than 96.66% of the optimal (minimum performance ratio), while the corresponding value for the GH was

87.22%. The optimal solution was found in 234 cases by the GA, and in 202 cases by the GH, which corresponds to a hit ratio of 84.78% and 73.19% respectively. The solution found by GA was strictly better than that found by GH in 66 cases, and strictly worse in only 25.

3.1.8.4.3 Genetic Algorithm vs. Beam Search

Alexouda and Paparrizos (2001), Alexouda (2004), and Balakrishnan *et al.* (2004) compared the performance of GA with that of BS, which was considered the state of the art approach of the time. The first two approaches make a comparison of the two methods with a full search method in the seller's welfare and share of choices problems respectively. Eight small problems were solved using different values for the number of products in the line (2, 3), number of attributes (3, 4, 5, 6, 7, 8), and number of levels (3, 4, 5, 6). Ten replications were performed in each case, while the number of customers was kept constant to 100. The results are shown in Table 3.4.

Table 3.4: Results of the comparison of the two methods

	Seller's welfare	Share of choices
GA found optimal	73.75%	77.50%
BS found optimal	41.25%	45%
GA outperforms BS	53.75%	33.75%
BS outperforms GA	12.50%	12.50%
GA/optimal	0.9958	0.9951
BS/optimal	0.9806	0.9882

Furthermore, they compared the performance of a GA with completely random initialization (GA1), a GA where the initial population is seeded with the best BS solution (GA2), and a BS heuristic, in problems with larger sizes where complete

enumeration is unfeasible. The number of customers was set to either 100 or 150 (Table 3.5).

Table 3.5: Results of the comparison of the three methods

	Seller's welfare		Share of choices	
	I=100	*I=150*	*I=100*	*I=150*
GA1 outperforms BS	93.88%	93.33%	47.92%	53.33%
BS outperforms GA1	6.11%	5.83%	33.33%	31.25%
GA2 outperforms BS	86.66%	80.83%	40%	43.33%
GA1 outperforms GA2	-	-	31.67%	35%
GA2 outperforms GA1	-	-	45.83	43.33%
GA1/ BS	1.0962	1.0794	-	-
GA2/ BS	1.0853	1.0702	-	-

Balakrishnan *et al.* (2004) defined eight different types of GA and hybrid GA procedures (Table 3.6).

Table 3.6: Genetic Algorithm techniques defined

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

A 2x2 full factorial experimental design was employed using the factors number of products in the line (4 or 7), and number of attributes (7 or 9), with respective attribute levels (6 3 7 4 5 3 3) and (7 3 5 5 6 3 3 7 5), while the number of customers was 200. Two replications were performed in each case. The values of GA parameters are illustrated in Table 3.7.

Table 3.7: Values of the Genetic Algorithm parameters

Parameter	Value
Mutation rate	0.04
Population size	400
Number of attributes to crossover (N=4, K=7)	10
Number of attributes to crossover (N=4, K=9)	17
Number of attributes to crossover (N=7, K=7)	12
Number of attributes to crossover (N=7, K=9)	21
Number of generations	500

After experimentation it was found that a mutation rate less than 0.04 resulted in a premature convergence to suboptimal solutions, while higher values did not offered a substantial improvement. In addition, higher number of attributes to crossover was more beneficial in problems with smaller number of products in the line, as compared to problems with larger product lines. The results are presented in Table 3.8.

Table 3.8: Results of the comparison of the 10 methods

Method	Best solution found (percentage of cases)	Average approximation of best solution
GASM	12.5%	94.44%
GASSM	12.5%	94.21%
GAHM	12.5%	94.16%
GASHM	12.5%	94.15%
GASMBS	25%	94%
GASSMBS	0	93.35%
GAHMBS	0	92.82%
GASHMBS	0	92.32%
BS	0	89.53%
CPLEX	50%	82.68%

Another full factorial design (2x2x2) was employed, in order to assess the impact of the number of products in line (4 or 7), the number of attributes (7 or 9), and the presence or absence of attribute importance, to the following variables of interest:

- The best GA solution.
- The ratio of the best GA solution to the best BS solution.
- The number of unique chromosomes in the final population:
    - With the best fitness.
    - With fitness within the 5% of the best solution.
    - With fitness between the 5% and 10% of the best solution.
- The worst chromosome in the final population.
- The average fitness in the final population.
- The standard deviation of chromosomes' fitness in the final population.
- The number of generation at which the best solution was found.

Two product lines are considered different when at least one product exists in the one but not in the other, while two products are considered to be different if they differ in the level of at least one attribute. Ten replications were performed in each case resulting in a total of 80 data sets. The eight GA instances, as well as the BS heuristic, were run 10 times for each data set, hence 6400 different GA runs were performed. The results showed that GA techniques performed better or equally well as compared to BS in the 6140 cases (95.93%), performed strictly better in 5300 (82.81%), and underperformed in 260 (4.07%). The best GA solution reached a maximum difference of 12.75% with that of the BS, and was on average 2.6% better. The maximum difference reached when the BS solution was better was 6.1%. The hybridized GA methods always produced solutions at least as good as the BS solution, and in the 80.2 % of cases produced strictly better solutions.

An interesting finding is that GA techniques which employ the unsorted representation, the standard mutation, and do not seed initial population with the best BS solution, showed the best average performance. A possible reason is the fact that the sorted representation scheme does not adhere to the principle of respect regarding the crossover operation. In addition, the incorporation of the best BS solution into the initial GA population, as well as the hybrid mutation operator probably make the algorithm converge to an area of solutions around the seeded BS solution, which in some cases may be suboptimal. Some loss in diversity of the final population may also be exhibited, as the integrated techniques displayed the worst results in respect to the number of unique chromosomes in the final population. Furthermore, integrated techniques suffer from premature convergence, as they tend to produce the best solution earlier, and result in the lowest standard deviation of chromosomes' fitness in the final population. Particularly, GA techniques without any hybridization (GASM, GASSM) provided final solutions at least as good as that of the hybridization techniques in 52.37% of cases on average, and strictly better on 35.12%. This indicates that the integration with the BS heuristic does not improve the quality of the solution. The number of products and number of attributes significantly affect ($p<0.0001$) the best GA solution, the ratio of the best GA solution to the best BS solution, all three measures of unique chromosomes in the final population, the standard deviation of chromosomes' fitness in the final population, and the number of generation at which the best solution was found; all in the positive direction. Finally, the presence of attribute importance has a statistically significant impact on the best GA solution, and the ratio of the best GA solution to the best BS solution.

3.1.8.5 Sensitivity Analysis

Balakrishnan and Jacob (1996) conducted a sensitivity analysis of the GA performance to changes in the values of its parameters, employing both the share of choices and the buyer's welfare criterion. A full factorial experimental design was generated using the factors and levels presented in Table 3.9.

Table 3.9: Factors and levels included in the experiment

Factor	Levels				
Mutation rate	0	0.01	0.1	0.25	0.3
Attributes participating in the crossover	0	K/4	K/2	3K/4	
Population size	50	100	200		
Degree of improvement in stopping rule	2%	0.2%			

The product category was assumed to consist of 8 attributes, each taking 5 levels, while the number of customers was set to 400. For each of the two problems a total of 120 GA runs were performed. In the share of choices, the average best solution provided by GA was 96.8% of the optimal product profile found by complete enumeration, and was found after 7.35 iterations (generations) on average. Hence GA reaches a near optimal solution by evaluating only the one fourth of the percent of the total number of possible solutions, which for the specific problem is 390625. Analyses of variance were performed to assess the impact of the four parameters to the quality of the solution. A main effects model had an R^2 of 0.504 and was statistically significant ($p<0.05$). Larger population sizes result in higher fitness of the best chromosome. As the number of attributes participating in the crossover increase, the quality of the solution also increases. As it was expected the tightening of the convergence parameter from 2% to 0.2% improves the fitness of the best solution. Whereas mutation rate had no significant main effect ($p=0.175$), the best algorithm's performance was achieved at the highest mutation rate.

Similar results concerning the parameters' impact in the solution's quality were exhibited in the buyer's welfare problem, where a main effects model had an R^2 of 0.724 and was statistically significant ($p<0.05$). The average best solution provided by GA was 97.9% of the optimal product profile found by complete enumeration, and was found after 8.48 iterations on average.

Steiner and Hruschka (2002) in another paper studied the sensitivity of the approximation of the optimal solutions with regard to varying parameter values for different problem sizes. A 12x5x3 factorial experiment was designed with 12 values of population size in the range [30, 250] at increments of 20 chromosomes, 5 different crossover probabilities (0.6, 0.7, 0.8, 0.9, 1), and 3 values of mutation rate (0, 0.01, 0.05). The size of the search space varied from 12650 to 10586800 feasible product lines, depending on the number of products in the line (2, 3, 4), number of attributes (2, 3, 4), and number of levels (4, 5). The recommended GA parameter values depending on the problem size after more than 1500 test runs are illustrated in Table 3.10.

Table 3.10: Recommended GA parameter values

Problem size	**12650**	**79800**	**161700**	**3921225**	**10586800**
Population size	130	150	230	250	250
Approximation of optimal	99.5%	99%	98.3%	99.2%	97.5%
Crossover probability	1	0.9	1	1	1
Approximation of optimal	99%	98.4%	97.6%	98.6%	96.8%
Mutation rate	0.05	0.05	0.05	0.01	0.01
Approximation of optimal	98.9%	98.8%	97.7%	98.5%	96.8%

3.1.9 Lagrangian Relaxation with Branch and Bound

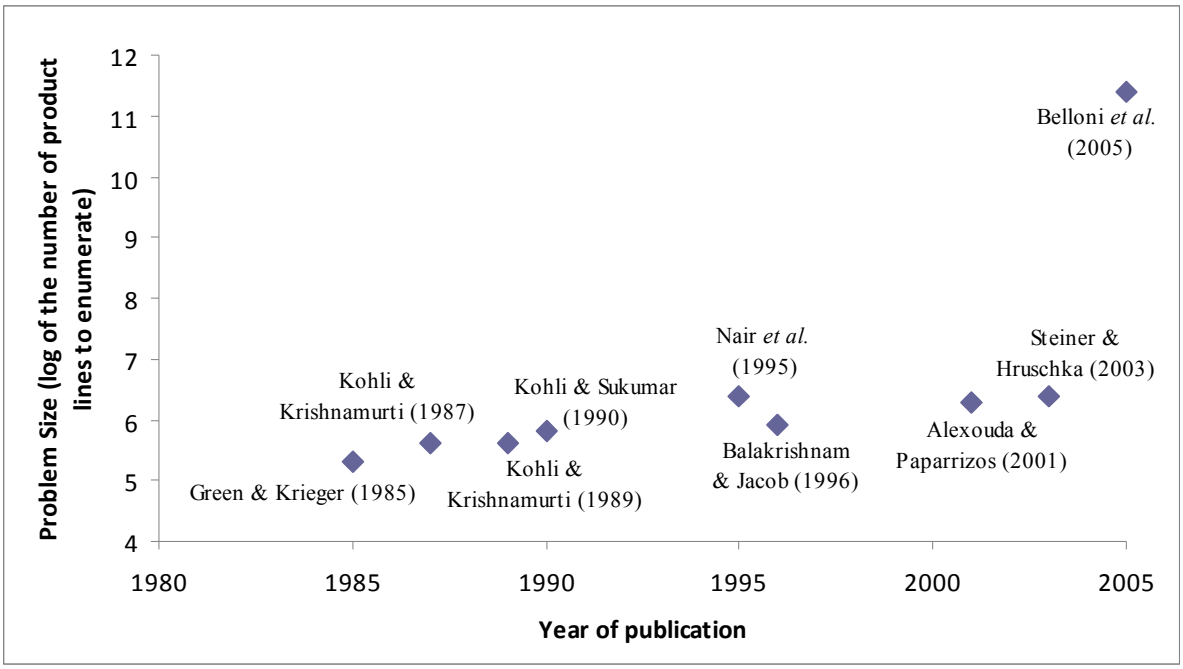
Camm *et al.* (2006) introduced a computationally efficient algorithm that guarantees global optimality in the share of choices problem for designing a single product. They developed an exact method that uses Lagrangian Relaxation with Branch and Bound for finding provable optimal solutions to large scale problems, using a deterministic choice rule. Branch and Bound (Land and Doig, 1960) constitutes an optimization algorithm mainly used in discrete and combinatorial problems, which attempts to discard large subsets of the entire set of feasible solutions without enumeration, by proving that the global optimal solution cannot be contained in them. This procedure requires the estimation of lower and upper bounds of the objective being optimized, so that the search is limited to promising regions only. When the lower bound exceeds the upper bound in a certain branch, then this branch is excluded from further search. In order to calculate upper bounds the authors use Lagrangian Relaxation, a method that "relaxes" hard problem constraints in order to create another problem that is less complex than the initial. The constraints are moved into the objective function and a penalty is added to the fitness of the solution if they are violated. The upper bounds provide an indication of the quality of any feasible solution compared to the (unknown) optimal. The lower bounds are created using heuristics that generate feasible solutions. The proposed method is initialized with the use of a greedy algorithm that finds a feasible solution. Next, a lagrangian dual problem is defined, by relaxing constraint (3), and the subgradient optimization procedure of Downs and Camm (1996) is used for the estimation of the values of the associated lagrangian multipliers. They use this lagrangian problem as a quick attempt to improve on the initial greedy solution. The search tree is initialized with the use of the best solution between the greedy and lagrangian generated one, and a depth first strategy is employed. The algorithm branches on constraints (2) in ascending order with respect to their cardinality (number of levels within attribute). In this way, each level of the search tree corresponds to an attribute. The authors use several logic rules to develop and prune the search tree, in order to significantly decrease the number of variables on which they branch, thereby reducing the time required to solve problems to verifiable optimality. The algorithm found and verified the global optimum solution to 1 real and 32 simulated problems with as many as 32 attributes and 112 levels. The required time ranged from 1.4 seconds to 40 minutes, depending on the problem complexity.

Belloni *et al.* (2008) proposed a Lagrangian Relaxation with Branch and Bound method for identifying global optimal solutions in the seller's welfare problem for designing a line of products, using a deterministic choice rule. As the authors mention, the lagrangian relaxation itself is not a practical algorithm, and most managers would consider it too complicated and computationally intensive for implementation and practical use. However, they use it to compute guaranteed optimal solutions, which are then used to benchmark the solutions generated by other heuristic algorithms. Heuristics are used to generate a feasible solution that has a fitness value (profit) of *f*. If it is shown that any feasible solution which includes a certain product generates a fitness value of less than *f*, then all solutions that contain the particular product are excluded from further search. Lagrangian relaxation is employed for the estimation of an upper bound on the fitness score that can be generated by a given set of solutions. The constraint relaxed is that each consumer can purchase exactly one product. Hence, for any solution in which the consumer selects more than one product, a penalty is subtracted from the fitness of that solution. Similarly, when a consumer chooses less than one product, a reward is added to the solution's fitness. The method seeks for the tightest possible upper bounds by varying the penalties which are applied to the objective function when a solution does not satisfy the relaxed constraints. Finding tight upper bounds helps ruling out portions of the feasible set as fast as possible. The algorithm was applied to 12 simulated problems, as well as 2 versions of a real world problem. The full problem had almost $5*10^{15}$ feasible solutions and the truncated problem had over 147 billion feasible solutions. With a computer that evaluates 30,000 solutions per second, it would take 57 days to completely enumerate the truncated problem, and over 5,000 years to exhaustively search the full problem. The method solved in about 24 hours the truncated and in approximately one week the full problem.

3.1.10 Comparison of the algorithms

Belloni *et al.* (2005) measured the complexity of the problems evaluated in previous studies from 1985 to 2005 using the log of the number of feasible product lines (Figure 3.2).

Figure 3.2: Size of problems solved (source: Belloni *et al.*, 2005)



Belloni et al. (2008) compared the performance of 9 different algorithms both in actual and simulated data sets. The real problem had over $4.9*10^{15}$ feasible solutions, and the lagrangian relaxation with branch and bound took over a week to find the global optimum. Except for algorithms' performance, they report a subjective assessment of relative difficulty, where "medium" or "high" level of difficulty denotes methods that require some problem-specific fine tuning of parameter values. Table 3.11 illustrates the results for ten trials of each method.

Table 3.11: Comparison of methods on the actual data set (source: Belloni *et al.*, 2008)

Method	Average performance (%)	Best performance as % of the optimal	CPU time	Subjective difficulty
Lagrangian relaxation with branch and bound	100	-	1 week	Very high
Coordinate ascent	98.0	98.6	5.4 sec	Low
Genetic algorithm	99.0	100	16.5 sec	Medium
Simulated annealing	100	100	128.7 sec	Medium
Divide and conquer	99.6	100	12.5 sec	Low
Greedy heuristic	98.4	98.4	3.5 sec	Low
Product swapping	99.9	99.9	14.1 sec	Low
Dynamic programming	94.4	97.4	5.5 sec	High
Bean search	93.9	98.6	1.9 sec	High
Nested partitions	96.7	98.4	8.4 sec	High

As the authors comment, among the more practical methods, the genetic algorithm, simulated annealing, divide and conquer, and product swapping perform best, reaching solutions that are on average within 1% of the optimum. The methods' performance was also evaluated using 12 simulated data sets. Table 3.12 presents the results for 10 problem instances for each data set.

In the simulated data sets the genetic algorithm and the simulated annealing manage to accomplish at least as good performance as on the actual data set, whereas the divide and conquer, and the product swapping perform slightly worse. The simulated data sets enable the extraction of more general conclusions about the algorithms performance than the single real data set. Hence, the genetic algorithm and the simulating annealing seem to be the best methods to be applied to the optimal product line design problem, since they provide excellent performance as well as the highest stability among all data sets. Simulated annealing always reaches the global optimum (but cannot guarantee it) with a small cost in time (more than two minutes), while genetic algorithm finds or comes very close to the global optimum, requiring much less time (11-16 sec).

Table 3.12: Comparison of methods on the simulated data sets (source: Belloni *et al.*, 2008)

Method	Average performance (%)	Finds optimal solution (%)	Finds solution >95% of optimal (%)	Average CPU time (sec)
Lagrangian relaxation with branch and bound	100	100	100	659.4
Coordinate ascent	96.0	15.8	65.8	0.6
Genetic algorithm	99.9	81.7	100	11.8
Simulated annealing	100	100	100	131.8
Divide and conquer	98.7	45.8	97.5	0.7
Greedy heuristic	97.5	23.3	82.5	0.2
Product swapping	98.5	39.2	95.8	0.8
Dynamic programming	96.3	10.0	70.8	0.9
Bean search	99.1	46.7	99.2	0.4
Nested partitions	93.9	4.2	44.2	2.2

3.2 Programs and Systems

In this section the programs and systems that deal with the optimal product (line) design problem are presented. All systems have been developed using one or more of the algorithms discussed in the previous section.

3.2.1 DESOP-LINEOP

DESOP and LINEOP are the two modules that comprise the program developed by Green and Krieger (1985), which was the first that dealt with the optimal product line design problem. The choice rule is deterministic, the objective is the maximization of market share, and the approach proceeds in two steps. In the first step, a reference set of promising products is constructed through the use of DESOP. The input is a matrix containing the part-worths of the I customers for each level of each attribute as well as a matrix containing the configuration of each customer's status quo product. The program accepts up to 400 customers and 20 attributes, each taking up to 9 levels, while the total number of levels must not exceed 80. The customers whose status quo product has higher utility than the best possible product profile are removed. The user is provided with summary descriptive data regarding the frequency with which each attribute displays the highest part-worth, and he is able to remove a subset of levels or fix an attribute at a certain level. Using the best in heuristic, the program generates the reference set of products, as well as an IxM matrix with the utilities each customer assigns to each of the candidate products. This utilities matrix along with the status quo matrix are entered at the second step to the LINEOP, which selects the products from the reference set that will comprise the product line. The program accepts up to 64 candidate products and produces a line of a maximum length of 30, using either the Greedy or the Interchange heurist.

3.2.2 SIMOPT

SIMOPT (Green and Krieger, 1988) solves all three problems, directly from part-worth data in a one step approach, using the Divide and Conquer heuristic. The user can specify the subset compositions of the heuristic, which, according to authors, should be formed so as to minimize the correlation of part-worths across subsets of attributes.

Attributes that are more closely related to each other should be assigned to the same subset. Except for the customer part-worths matrix, the set of competitive product profiles is also required, as the system uses probabilistic choice rules. Furthermore, the user may optionally provide importance weights for each customer (reflecting the frequency and/or the amount of purchase), background attributes or demographic weights for use in market segment selection and market share forecasting. When the Seller's welfare is selected, costs/return data measured at the individual-attribute level are required. The system provides the user with the capability to perform a sensitivity analysis, in order to observe how market shares (or return) change for all competitors as one varies the levels within each attribute in turn. Since in practice a manager will not probably be interested just in maximizing market share or return, but needs to have a picture of the tradeoff between them, SIMOPT also supports a Pareto frontier analysis. The user is provided with all the undominated profiles with respect to return and share, and can simulate giving up an amount of the one objective for an increase in the other.

3.2.3 GENESYS

Balakrishnan and Jacob (1995) developed the GENEtic algorithms based decision support SYStem, which uses the triangulation methodology to increase the confidence in the quality of the obtained solution for the single product design problem. According to it the solution obtained with a certain method is considered "good", if it is in the ball park of the solution obtained through a maximally different heuristic. Using complete enumeration for small problems, Genetic Algorithm, and Dynamic Programming, GENESYS enables the user to avoid the solutions that are caught in local optima. The system consists of a menu driven user interface, where the user can select a single heuristic or the triangulation approach, as well as whether the share of choices or buyer's welfare problem will be solved. Customer part-worths and status quo products are stored in a database, and the three solution methods are stored in a model base. The DP implementation is as in Kohli and Krishnamusti (1987), and the GA as in Balakrishnan and Jacob (1996).

3.2.4 MDSS

Alexouda (2004) developed a Marketing Decision Support System for solving all the three problems in the optimal product line design, using a deterministic choice rule. The system employs a one-step approach through a GA implementation (Alexouda and Paparizzos, 2001). Borland C+ Builder 3 has been used for the construction of the system, which consists of a database where the seller's return data, as well as the customer part-worths and status quo products are stored, a model base that contains the GA implementation for each of the three problems as well as a complete enumeration method for small problems, and a graphical user interface. Emphasis has been placed on the friendliness of the user interface, which is menu-driven with common easy-of-use features like grid formats, navigators for grids, and pop-up menus. Tools are available that provide an easy to understand visible way to present options to the user, as well as shortcuts that perform actions quickly. Except for the attribute optimization, the system also offers a market simulation module that provides the user with the capability to perform what-if analysis, and assess the likely degree of success of different product line configurations to the market.

3.2.5 Advanced Simulation Module

ASM is a commercial system that was launched by Sawtooth Software in January 2003. All three problems of the optimal product line design are supported, as well as a market simulation module. The user can select between a deterministic and a probabilistic choice rule, as well as among five different optimization methods: Complete Enumeration, Grid Search, Gradient Search, Stochastic Search, and Genetic Algorithms. Grid Search is similar to the Coordinate Ascent approach by Green *et al.* (1989). In the Gradient Search, a combination of attributes to be altered simultaneously is found, through a Steepest Ascent method that locates the top of a peak in a response surface. An initial solution is generated randomly or specified by the user. Each attribute is changed (one at a time) and the resulting gain or loss in the objective is measured. Then, the direction for changing all attributes simultaneously that results in the largest improvement per unit change is decided. This is the direction of locally Steepest Ascent for the response surface, called Gradient. A line search is conducted next, beginning from the existing solution and moving in the direction specified by the gradient. The first move is very small, and each subsequent move is twice as far from the starting point. The results from the final three points are used to fit a quadratic

curve to the response surface, and the point that maximizes the quadratic function is located. The response surface is evaluated at that point, and the solution is retained if it is better than the previous best. When no improvement is achieved from one iteration to the next, the algorithm terminates. In Stochastic Search one attribute is randomly altered at a time and if it results in an objective's improvement the change is acceptable. The process iterates for a prespecified number of times. The authors recommend using either Grid or Stochastic Search from different starting points. If the same solution is always obtained then this is probably the global optimum. Otherwise the search domain should be reduced using the experience obtained, in order to conduct a complete enumeration. When continuous attributes exist (e.g. price), the Gradient Search is the most appropriate. Genetic Algorithms should be used when conditions limit the capabilities of the other methods, for instance when the response surface is very irregular with multiple peaks.

3.2.6 Discussion

Marketing systems that deal with the optimal product line design problem have evolved considerably among the past 25 years. Among the five systems presented, four are purely academic and only one is a commercial product. A lot of work has been done since the launch of the first program (Green and Krieger, 1985), which could only solve problems of limited size (not more than 400 customers and 80 attribute levels in total). However, among the algorithms that achieved the highest performance in the problem, only GAs have been incorporated into marketing systems. Whereas GAs have been used to systems that solve both the Single Product (GENESYS) and the Product Line Design problem (MDSS and ASM), all systems provide the decision maker only with a single best solution. As mentioned before, the manager that will make the final decision is usually interested in having a range of good solutions, so that he can select the one which satisfies a number of subjective criteria.

4 The Market Research

The market research surveys that companies conduct nowadays are mostly descriptive, focusing on the collection of demographic consumer data. Whereas providing valuable information for product promotion and advertising decisions, demographic data cannot support the firm's decision makers in new product design, development and positioning applications. Such decisions require tools that will assist product managers in estimating consumer preferences with regard to the various product characteristics. This kind of information can be further used for predicting the customer's purchasing behavior, and design products that will maximize the company's profits. As discussed in the previous chapter, in order for the customer preferences concerning a product category to be measured, the product is described in terms of a set of attributes. Furthermore, every product attribute is broken down into a number of levels. A statistical technique that enables the estimation of a numerical value (called part-worth*) for each attribute level is* Conjoint Analysis *(Green and Rao, 1971).*

4.1 Conjoint Analysis

Conjoint Analysis was developed by Paul Green, a professor of the Wharton School of the University of Pennsylvania, and has its origins in Mathematical Psychology. Orme (2006) divides Conjoint Analysis into 3 different types: a) Conjoint Value Analysis, b) Adaptive Conjoint Analysis, and c) Choice-based Conjoint Analysis.

Conjoint Value Analysis is the traditional full profile approach, where each product profile is represented by the entire set of attributes. The first step in this approach is the

determination of the set of attributes that will be used for the product representation. The number of attributes must be kept as small as possible, in order to minimize the respondent's burden. Next the levels for each attribute are decided. The following step is the construction of the product profiles that the respondents will evaluate. If the number of possible profiles is relatively small (up to 18) then a *full factorial design* may be employed, where the respondents evaluate all the possible combinations of attribute levels. When, however, the number of possible profiles is large, a *fractional factorial design* must be employed, where the respondents evaluate only a limited number of representative product profiles. The analyst must also decide the way that the profiles will be provided to the respondent: cards where products are described according to their attribute levels, visual products presented in personal computers, or even the products themselves in their natural form. The next step is the determination of the way that the respondents will evaluate the profiles. A rating scale from 1 to 10 or to 100 can be used for each profile, or the respondent can rank the profiles from the most to the least preferred. A market research survey is usually conducted for data collection, where respondents provide information either by answering questions or by filling in questionnaires. Specially designed computer programs can also be used for collecting data from distance. The method that will be used for data analysis depends on the way information was collected. In Conjoint Value Analysis linear regression or dummy variable regression are typically used.

4.2 The survey

In order for the models proposed in the present thesis to be tested with the use of real world data, a market survey was conducted. The survey concerns the consumer preferences with regard to fresh milk in the Greek retailing sector. The Greek retailing market in milk is composed of four large players and a number of other small companies that only have a local distribution of their products in 1-4 prefectures. More than 30 products are offered to the market that belong to three main categories:

1. fresh cow milk,
2. high pasteurized cow milk,
3. fresh goat milk.

These products vary with regard to the size, the package and a number of other attributes. One of the small players of the specific market is a company located in the island of Crete in the prefecture of Chania (company ALPHA from now on). ALPHA is currently offering only a limited number of products to the specific market, which all belong to the category of fresh goat milk. ALPHA will be used as a case study of a niche player who wishes to become a large player in the market, through the design of a new line of products. Using the product line design approach introduced in this thesis, ALPHA will redesign its fresh goat milk products, while entering the market of fresh and high pasteurized cow milk. Since the data needed for applying the proposed models are the customer partworths concerning the milk product attributes, a conjoint market survey was conducted.

The design of the survey was made in cooperation with the company's marketing managers. Specifically, four attributes were identified as having impact to a customer's purchasing decision: the type of the milk, the percentage of fat included in the milk, the product's size, and the product's packaging. The main types of milk offered in the market are the three that have already been described. Two levels were specified for the fat percentage: 1.5% and 3.5%. Four product sizes are available in the market: 0.5 liter, 1 liter, 1.5 liter, and 2 liter. Finally, fresh milk is offered in two different types of package: paper and plastic. The 4 attributes with the corresponding levels are illustrated in Table 4.1.

Table 4.1: Attribute and levels included in the study

Attribute	Levels			
Size (*l*)	0.5	1	1.5	2
Milk type	Fresh	High pasteurized	Goat	
Fat	1,5%	3,5%		
Package	Paper	Plastic		

The partworths for each attribute level will be estimated with the application of Conjoint Value Analysis. Ranking a list of hypothetical full profile milk products was chosen as the customer preference elicitation process, since it provides the respondents with the minimum possible burden. The number of possible product profiles is 48, which constitutes a very large number for a full factorial approach. Hence, a fractional factorial approach is adopted, where 16 hypothetical were created (Table 4.2) with the use of the Orthogonal Design option of the statistical program SPSS 16.

Table 4.2: The hypothetical milk profiles

Milk type	*Fat*	*Size (l)*	*Package*
Goat	3.5%	0.5	Plastic
High pasteurized	3.5%	1	Plastic
High pasteurized	1.5%	1.5	Plastic
Fresh	1.5%	0.5	Paper
Fresh	1.5%	1	Paper
Goat	3.5%	1.5	Paper
Fresh	3.5%	1.5	Paper
Fresh	1.5%	2	Plastic
High pasteurized	1.5%	0.5	Paper
Fresh	3.5%	0.5	Plastic
Fresh	3.5%	2	Paper
Goat	1.5%	1	Paper
Fresh	3.5%	1	Plastic
Fresh	1.5%	1.5	Plastic
High pasteurized	3.5%	2	Paper
Goat	1.5%	2	Plastic

Six more hypothetical milk profiles were created (Table 4.3), where four of the profiles were designed to be efficient choice sets that have statistically optimal utility and level

balance (Huber and Zwerina, 1996), while the other two had all attributes duplicated except for one.

Table 4.3: The profiles for the holdout task

Milk type	*Fat*	*Size (l)*	*Package*
Fresh	3.5%	0.5	Plastic
High pasteurized	1.5%	1	Plastic
Goat	3.5%	1.5	Paper
Fresh	1.5%	2	Paper
Fresh	3.5%	1	Plastic
High pasteurized	3.5%	1	Plastic

Except for the conjoint data, each respondent provided personal information (demographic characteristics) and information concerning his milk usage patterns, by answering the set of questions illustrated in Table 4.4.

Table 4.4: The demographics part of the questionnaire

1. What is your sex?	☐ Male ☐ Female
2. What is your age?	…….
3. What is your family status?	☐ Single ☐ Married without children ☐ Married with children
4. What is your education level?	☐ Primary school ☐ High school ☐ University
5. Occupational status?	☐ Unemployed ☐ Free lancer ☐ Public servant ☐ Private servant ☐ Student ☐ Pupil
6. How often do you buy milk?	☐ 4-7 times a week ☐ 1-3 times a week ☐ 1-3 times a month ☐ less than once a month
7. How do you use milk *(multiple choice)?*	☐ Consumption ☐ Cooking

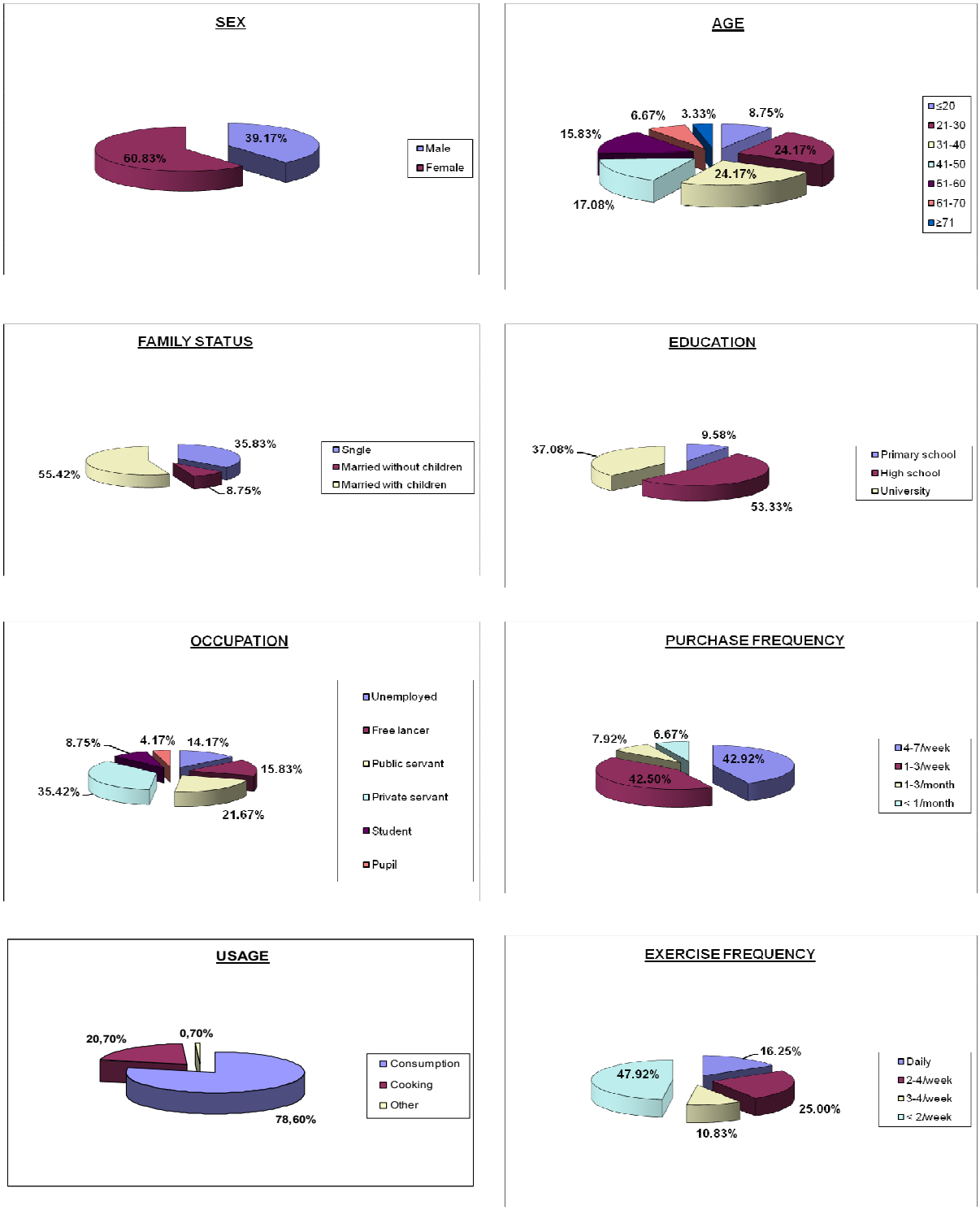
	☐ Other
8. How often do you exercise?	☐ Daily ☐ 2-4 times a week ☐ 3-4 times a month ☐ less than 3 times a month
9. Are you concerned about nutrition components?	☐ Not at all ☐ Little ☐ Moderately ☐ Very

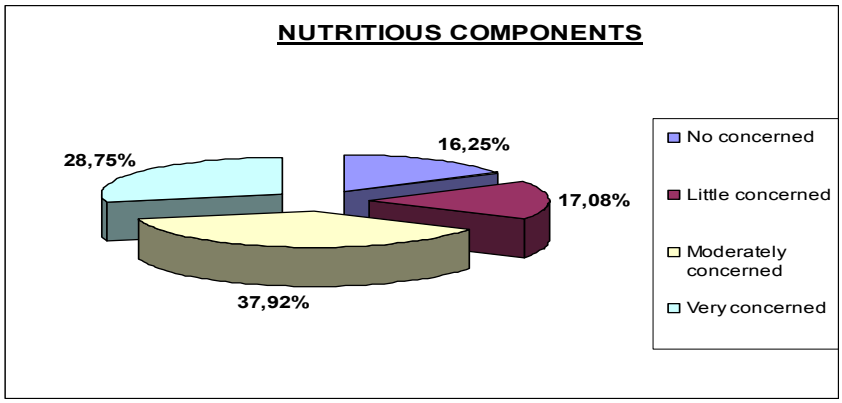
Data collection was completed in different super-markets in the city of Chania within the period April-December 2008. The target group was frequent milk buyers. A total of 482 consumers were interviewed with the use of a simple random sampling procedure. Each respondent completed an anonymous questionnaire, which consisted of the Tables 4.2, 4.3, and 4.4. After filling in Table 4.4, each respondent ranked the profiles shown in Table 4.2 from the most to the least preferred, in order for the partworths for each product attribute to be estimated. Finally, each respondent completed a holdout task by selecting one among the 6 alternative milk profiles of Table 4.3. The data from the holdout task will be used for the validation of the market simulation model that is presented in the next chapter.

4.3 The results of the survey

The data set was cleaned and the sample was reduced to 480 respondents. The following figures provide for illustrative purposes a brief description of the results concerning Table 4.4.

Figure 4.1: Survey results





The results indicate that the customer sample is representative of the average human being (before the data collection there was a concern that most subjects would be undergraduates). The data above will be used after the design of ALPHA's product line, for posititioning and advertising purposes, and thus will not be further analyzed in the present thesis. The data from Table 4.2 were used for the estimation of part-worths for the 480 respondents with the use of Conjoint Analysis provided by the SPSS 16 software. These data along with the respondents' holdout choices will be used as the input to the models introduced in the subsequent chapters.

5 Market Simulation Model

In this chapter an innovative market simulation model is developed, that will be used for simulating the choice process of the group of customers each time a company optimizes its product line.

5.1 Introduction

More and more companies today try to forecast the market penetration of a new concept before it enters the production stage. In this context, marketing managers include consumer preference modeling in the early stages of new product development. Market Simulations assist a manager in developing an effective marketing strategy through performing what-if questions. The manager can simulate market behaviour on new product introductions, product line extensions, or product modifications. He can implement scenarios of introducing different product configurations to the market, and predict conditional choice shares, estimate the direct and cross elasticity of price changes, or form the logical guide to strategic simulations that anticipate competitive responses (Orme, 2006). Customer brand switching behavior can be revealed, enabling the firm to design new products that take share from its competitors without cannibalizing its existing product line.

One of the most popular approaches among marketing practitioners is the simulation of market behavior on new product introductions using conjoint data. Consumer preferences for the various product attributes are estimated through conjoint analysis, and are used to predict the hypothetical market shares that different product configurations might gain. In this way, conjoint market simulation assists managers in reducing the uncertainty when designing new products for a specific market.

The effectiveness of a market simulation depends on the level of accuracy in the product market shares prediction. This requires the proper modeling of human choice behavior, a task of high complexity, since it involves the integration of sophisticated theoretical assumptions into mathematical models (choice models). As Baier and Gaul (2001) note, the determination of an adequate choice model is the most cumbersome task in market simulation situations. Allenby *et al.* (2005) stress the complexity of human behavior, and propose the adjustment of choice models to better predict it. Popular choice models usually applied in market simulations are the BTL, and the MNL. Despite their extensive usage, these models fail to represent similarity among alternatives in the choice rule, suffering from the well known *Independence from Irrelevant Alternatives* bias. As a result they tend to overestimate the market shares of similar products. The first choice rule on the other hand, where the consumer is assumed to deterministically select the most preferred product, overestimates the choice shares of the highest rated alternatives.

A number of approaches have been developed that overcome the above limitations, whereas other models focus on the optimization of market shares prediction. However, an integrated approach that efficiently addresses both issues has not been proposed yet. In this chapter, a market simulation model will be developed that effectively combines the satisfaction of the critical theoretical properties that a market simulation should reflect, with high predictive accuracy on market shares estimation. It will be shown that calibrating choice models using data from the products' multiattribute analysis by each customer, can substantially improve the effectiveness of market simulators.

5.2 Market Simulations

Before starting a market simulation the consumers' preferences must be elicited, usually with the use of Conjoint Analysis. The partworths estimated through Conjoint Analysis form the product utility values for each individual. Product utility values are then converted to choice probabilities through the use of a choice model like the BTL or the MNL. Next, a market scenario is formulated where n customers have to select one product among m alternatives. For each customer i, a vector of product choice probabilities $[P_{i1}, P_{i2,...,} P_{im}]$ is calculated, and the total choice likelihood for a product j results from the aggregation of its choice probabilities across the whole customer base:

$CP_j = \sum_{i=1}^{n} P_{ij}$. Finally, the simulated market share for each product j is estimated: $MS_j = 100 * \mathrm{CP}_j \Big/ \sum_{k=1}^{m} \mathrm{CP}_k$ %.

Numerous simulations may be conducted by altering the configuration of one or more products and observe the relative change in market shares. In this way, the outcome of different competitive market strategies may be anticipated. Market simulations can capture cross-elasticity effects between different brands or attributes, answer what-if questions about new product launches, product modifications, or product line extensions given a current competitive environment, reveal price/sales elasticities and guide pricing strategy (Orme, 2006).

5.3 Critical properties of market simulations

Two critical properties enable a market simulation to track the complexity of market behavior; *differential impact* and *differential substitution* (Orme and Huber, 2000).

5.3.1 Differential Impact

When a current feature is modified or a new one is added, the selection probabilities of the items in the choice set will be differentially impacted. People who like speed prefer sport cars, whereas a person who likes luxury and safety might prefer a limousine. Hence, if the horsepower of a sport car is increased, its sales can be significantly raised, since potential sport cars buyers give high importance to the car's acceleration. On the contrary, the same action will probably have small impact on the sales volume of a limousine, as candidate limousine buyers do not consider the car's acceleration as an important attribute. While such complex patterns of interactions can be represented as cross terms in an aggregate-level model, Huber *et al.* (2001) state that the modeling of interaction terms in the utility function is complicated, as their number can grow uncontrollable large, leading to overfitting or misspecification problems. They recommend the aggregation of heterogeneous individuals, each following a different preference model, as a more efficient way to represent the differential impact that a certain feature may have on specific brands. This constitutes a more stable modeling approach than the curve-fitting exercise of the cross term, and has the additional

advantage of being more managerial actionable. In addition, the heterogeneity account permits the reflection of idiosyncratic individual preferences in market simulations, in the context of a simple additive model. Allenby *et al*. (2005) also favor the use of a main-effects model for each individual, since the interactions between the parameters reflecting the alternatives and the characteristics of the respondents could result in unmanageably large aggregate models. They argue that exploring the impact of alternative scenarios on a product's market share is more accurate with respondent-level parameters. Simulating markets at the *individual level* enables managers to identify the critical marginal consumers, who are most likely to change their selection, and may also help them to design new product offerings, which will exceed the consumer's purchasing threshold.

5.3.2 Differential Substitution

Differential substitution refers to the assumption that when a new product enters a market it gains share mostly from the similar products, rather than from the dissimilar ones. This is also known as the "similarity hypothesis" (Tversky, 1972). This property is very important, since the two popular choice models, BTL and MNL, do not exhibit differential substitution. Instead, they follow the *proportionality* assumption, according to which the ratio of the shares of any two products of the market is independent from the rest products. The outcome of this property, also known as *Independence from Irrelevant Alternatives* (IIA), is that a new product takes share from all existing products in proportion to their current shares (no similarity effect). The limitation of the IIA bias is explained through the red/blue bus paradox (Ben-Akiva and Lerman, 1985), where two alternative means of transportation, a car and a red bus, are equally valued by an individual, thus they have 50% likelihood to be chosen. Now we assume that another bus which differs only in color (blue) from the existing one (thus offering the same utility to the individual) enters the choice set. Since the three alternatives have the same utility value, a proportional model will predict 33.3% final share for each of them. However, in reality it is expected that the new product (blue bus) will take share mostly from products that are similar to it (red bus), instead of dissimilar ones (car). In such a case the car's choice likelihood will probably remain close to 50%, and the two buses will share the rest 50%. The market shares of the alternatives closest to the launched one are usually reduced, due to the greater *substitution* that occurs between them (negative

similarity effect). The IIA bias prevents the model from postulating any pattern of differential substitutability among alternatives, hence the cross-elasticity of the probability of product i with respect to a change in U_k is the same for all i with $i \neq k$ (Baltas and Doyle, 2001).

While representing real world purchasing situations more effectively than proportional models, choice models that reflect substitutability have shown low predictive accuracy due to the "attraction effect", according to which the introduction of a new product raises the attractiveness for the category it belongs to. This assumption is also known as "share inflation" or the "rich supply" effect, where the existence of similar alternatives increases their desirability (positive similarity effect). According to Huber and Puto (1983) the addition of a new alternative to a customer's choice set, initially results in a global attraction effect, where a general shift of preferences occurs toward the new item. Next, a local substitution effect takes place, where the new product takes shares mainly from similar items in the set. This explains the relatively good predictive performance that proportional models show, as the two effects counterbalance each other.

5.4 Previous Approaches

As far as predictive accuracy is concerned, the state of the art market simulation approach, which is widely used in commercial applications, is the SIMOPT (SIMulation and OPTimization) product-positioning model (Krieger *et al.*, 2004). In this model market behavior is simulated through the calibration of the exponent a of the Pessemier's model:

$$P_{ij} = U_{ij}^{a} \Big/ \sum_{j=1}^{n} U_{ij}^{a}$$

This is implemented with the use of the *ALPHA rule* as follows: Assuming that "external" market shares are known, the 'optimal' a is calculated so that the simulated choice shares are as close as possible to the external. The model can be applied to individual level conjoint data and exhibit the differential impact property. On the other hand, the SIMOPT approach does not display differential substitution, since the Pessemier's model is an extension of BTL model that also suffers from the IIA bias. The fact that the same choice model is used for the whole customer sample constitutes

another limitation of the ALPHA rule, since findings from consumer behavior research indicate that every individual follows a different choice pattern.

Matsatsinis and Samaras (2000), propose the selection of a different choice model for each consumer, through the study of the distribution of the total utilities he assigns to the set of products. Particularly, they consider the distribution's *Range* ($r=U_{max}-U_{min}$), *Kurtosis* (k) and *Skewness* (s), for selecting the choice model that better describes each consumer's purchasing pattern. The values of these three coefficients constitute the input that triggers a total of 27 *if-then* rules, which comprise a knowledge base containing 8 different brand choice models (Matsatsinis and Siskos, 1999). Two of the models used arise from the calibration at the individual level of the Pessemier and the MNL models, with the use of r as the exponent. Their approach displays differential impact, since customer preferences are estimated at the individual level using the UTASTAR (Siskos and Yannacopoulos, 1985) preference disaggregation method. Differential substitution is not adequately reflected, since six of the choice models used are subject to the IIA property.

The VOICE decision support model (Krieger and Green, 2002) optimizes a firm's market share for a specified product/service, taking as input the customer's stated: a) attribute performance scores for each product, b) attribute importance ratings, and c) constant sum probabilities of choosing each product. A grid search heuristic is used which, at the individual level, modifies the importance ratings and estimates an "optimal" exponent a for the Pessemier model that, together, best reproduce the vector of constant sum likelihoods. Since each consumer is dealt with individually, there is a great possibility that the previous procedure overfits to the constant sum probabilities provided by the consumer. To ameliorate this concern, they put a constraint to the extent that the modified attribute weights can differ from the customer supplied. While constituting an innovative approach, their model is quite hard to use in practice due to the large amount of data required by the respondents. Actually, psychologists have questioned the interviewee's ability to provide valid attribute weights, arguing that the results tend to be unstable and highly influenced by the decision context (Tversky and Simonson, 1993).

5.5 The proposed approach

The aim of the proposed approach is twofold. First, high performance in predicting actual product market shares has to be displayed. This will be achieved through the calibration of the choice model's exponent at the individual level. Second, compliance with the theoretical properties mentioned in Section 5.3 has to be accomplished. Differential impact, as well as differential substitution must be exhibited, through the reflection of both the substitutability and the attraction effect. A corrective method is developed for this purpose.

5.5.1 Calibration of the choice model

According to Orme and Johnson (2006), an appropriate and theoretically justified method for tuning simulated results to more closely fit real market shares is the adjustment of the exponent used in choice models. In the proposed model the approaches of SIMOPT and MARKEX are extend through the individual calibration of the Pessemier model, through the assumption that exponent a depends not only on the range of the product utilities distribution but also on its kurtosis and skewness. This assumption is based on studies that relate the values of the three coefficients with the level of difficulty a customer expresses in making a choice among a set of competing alternatives (Matsatsinis and Samaras, 2000; Tsafarakis *et al.*, 2008). Previous research on consumer behavior has related the value of the choice model's exponent to the expertise of the decision maker, or the level of customer's involvement into the purchasing decision. The proposed approach is more practical for a manager to apply it, since r, k, s are easy to measure coefficients. In order to find the relationship between exponent a and the three coefficients that better simulates market behavior, the performance of all the possible linear combinations among them will be assessed (Table 5.1).

Table 5.1: Variations of the choice model's exponent

Case	Exponent
1	$Rr_i + K\kappa_i + Ss_i$
2	$Rr_i + K\kappa_i$
3	$Rr_i + Ss_i$
4	$K\kappa_i + Ss_i$
5	Rr_i
6	$K\kappa_i$
7	Ss_i

In case 1 for example, the model's exponent for an individual i is given by the following equation:

$$a_i = Rr_i + K\kappa_i + Ss_i, \tag{17}$$

where the parameters R, K, and S have the same value across the customer sample, in order to prevent the model from individually overfitting.

An experiment using artificial data will be conducted, where the predictive accuracy of the ALPHA rule will be compared with the seven different approaches shown in Table 5.1. The data sets where the eight models will be tested consist of simulated part-worths for each respondent, as well as hypothetical market scenarios with different product configurations. The market is assumed to comprise 8 competitive products, each consisting of 9 attributes which can take 9 different levels. Initially, the part-worth functions of 5,000 hypothetical respondents are estimated. This large number of respondents represents the total target population, whose purchasing behavior is to be simulated. Part-worths for each attribute level are randomly drawn from a uniform distribution in the range [0, 1]. The market scenario is formulated next, through the random selection of the level of each of the 9 attributes for every product. Using the simulated part-worths in combination with the products' configuration, the utility value that every respondent assigns to each product is calculated. These error free utilities are used for the estimation of the "actual" market shares for each scenario, assuming that the individual will always choose the alternative with the highest utility value within the set. The sample of the population on which the models will be tested is then constructed, through the choice of 800 from the 5,000 respondents. In order for potential

errors derived from the procedure of measuring the customers' preferences to be simulated, error terms are added to the sample's "true" utilities. These error terms follow a normal distribution with zero mean and variance obtained from the following equation (Wittink and Cattin, 1981):

$$PEV=\frac{\sigma_\varepsilon^2}{\sigma_\varepsilon^2+\sigma_u^2} \Rightarrow \sigma_\varepsilon^2=\left(\frac{PEV}{1-PEV}\right)\sigma_u^2 \quad (18)$$

where σ_ε^2 is the variance of the error term, σ_u^2 is the variance of the distribution of the value of an alternative's utility across respondents, and *PEV* is the *Percentage of Error Variance* on preferences, which is set to 0.35. Next the part-worths are standardized by setting the lowest level of each attribute to zero, and rescaling the sum of the best attribute levels to unity.

Since the aim of the models is to predict with the highest possible accuracy the behavior of the market, they will be calibrated through the estimation of individual exponents a_i. The calibration is implemented through the calculation of the optimal *R, K,* and *S* parameters used in each case. The objective is that the market shares that the model simulates, should most closely resemble the real market shares. In case 1 for example, the optimization of the three parameters is an optimization problem formulated as follows:

$$find\ R, K, S\ that\ minimize\ f=\sum_j |RS_j - SS_j|\,|,\ \ j=1,2,\ldots,m,\quad R, K, S \in \Re$$

where *RSj* are the real shares and *SSj* are the shares simulated by the model.

The choice models' exponent usually takes real values greater than or equal to 1. The values of the *r* and *k* coefficients for the whole customer sample are positive, thus for the cases 2-7 a full search in the range [1, 300] is implemented with step 0.1 for the *R* and *K* as well as the *a* exponent of the ALPHA rule. The *s* coefficient takes both positive and negative values thus a full search is performed in the range [-150, 150] with step 0.1 for the *S* parameter. However, for case 1 the complexity for a full search of the three variables (*R*, *K* and *S*) is extremely high, requiring too much CPU time and computer memory to be implemented. Hence, two methods will be applied for solving the optimization problem: a stochastic logarithmic approach, and a genetic algorithm.

5.5.2 Stochastic Logarithmic Approach

The method consists of two main parts. In the first part, a single path of points is followed, beginning from a certain initial point. At each point of the path, only the set of its neighbors is examined, so that the next point in the path is selected towards the direction of the neighbor corresponding to the minimum value of *f*. At each iteration of the algorithm, the neighboring region is decreased until it reduces to a single point, which is selected as a solution of the optimization problem. The second part introduces a stochastic factor, which assigns probabilities to every neighbor point of the current examined point and then selects the next point among the neighboring ones by exploiting these probabilities. In the considered case, the probabilities assigned to each point are inversely proportional to criterion *f*. This way, different paths are examined depending on the probabilities and the randomness. Thus, in the second part, the experiment needs to be iterated many times and the best solution among all experiments is returned as the most appropriate. As a result, the algorithm is repeated in several experiments, say *m*, and within each experiment at *n* iterations.

The three variables *R*, *K*, and *S* are concatenated into a three dimensional vector, say **x**. It is now assumed that at the current *n*-th iteration of the algorithm and at the *m*-th experiment, a point $\mathbf{x_m}(n)$ has been selected. In the previous notation, the dependence of the point **x** on the iteration and experiment has been added. Then, the next point $\mathbf{x_m}(n+1)$ is then obtained by evaluating *f* among all its neighbors. The neighboring region is defined as:

$$N(\mathbf{x}_m(n),\delta(n)) = \{\mathbf{y} \in R^3 : \mathbf{y} = \mathbf{x}_m(n) + \delta(n)\mathbf{p},\ \mathbf{p} \in G^3\} \tag{19}$$

where $G=\{-1,0,1\}$ and $\delta(n)$ is an integer indicating the step size of the neighborhood region. At each iteration, the step size $\delta(n)$ is reduced such that $\delta(n+1)=\delta(n)/2$ until $\delta(n)=1$. Initially, a high value for $\delta(n)$ is chosen in order to cover the largest part of the space. This constitutes the first part of the algorithm.

In the second part, probabilities are assigned for each neighboring point of the current point $\mathbf{x_m}(n)$. In particular, if the three-dimensional neighbors of $\mathbf{x_m}(n)$ is denoted as $\mathbf{y_i}$, $i=1,\ldots,|N|$, where $|N|$ stands for the number of elements of the set *N* in (19), then

probabilities which are inversely proportional to the *f* can be calculated for each point $\mathbf{y}_i$,. That is, $p_i = 1 - \frac{f(\mathbf{y}_i)}{\sum_{j=1}^{|N|} f(\mathbf{y}_j)}, \quad i = 1,\ldots,|N|.$

A cumulative probability function is then constructed for all $\mathbf{y}_i$ i.e., $q_i = \sum_{j=1}^{i} p_j$, $i=1,\ldots,|N|$, with $q_0=0$. Using a given random number *r*, uniformly distributed in the range [0, 1], the next point at the *(n+1)-th* iteration $\mathbf{x}_m(n+1)$ is chosen among the neighbors $\mathbf{y}_i$ as follows:

$$\mathbf{x}_m(n+1) = \{\mathbf{y}_i \in N(\mathbf{x}_m(n), \delta(n)) : q_{i-1} < r \le q_i\}.$$

The algorithm is repeated until $\delta(n) = 1$ and for several experiments and the value that minimizes *f* over all iterations and experiments is selected as the most appropriate. That is, for one experiment, the optimal value $\bar{\mathbf{x}}_m = \underset{\text{for all iterations}}{\arg\min} f(\mathbf{x}_m(i))$ corresponding to the minimum criterion is returned. Then, the algorithm is repeated for many experiments and the best value over all experiments is returned, i.e., $\bar{\mathbf{x}} = \underset{\text{for all experiments}}{\arg\min} f(\bar{\mathbf{x}}_m)$ is selected.

5.5.3 Genetic Algorithm

Another possible solution for the above mentioned optimization problem is the use of an Evolution Program. In contrast to enumerative search techniques, such as Dynamic Programming, which may break down on complex problems of moderate size, evolutionary programs provide unique flexibility and robustness on such problems. For this reason, a Genetic Algorithm approach is adopted next. The approach seems to be very efficient for the particular optimization problem, given the size and dimensionality of the search space and the multimodal nature of the objective function. Possible solutions of the optimization problem, i.e., sets of parameters, are represented by chromosomes whose genetic material consists of frame numbers (indices). Chromosomes are thus represented by index vectors $\mathbf{x} = (x_1, x_2, x_3)$ following an integer number encoding scheme, that is, using integer numbers for the representation of chromosome elements (genes) $x_i, i = 1,\ldots,3$. The reason for selecting integer number (instead of binary) representation is that all genetic operators, such as crossover and mutation,

should only be applied to genes x_i, and not to arbitrary bits of their binary representation. An initial population of P chromosomes, $\mathbf{X}(0) = \{\mathbf{x}_1, \ldots, \mathbf{x}_P\}$ is then generated by selecting P sets of frames whose feature vectors reside in extreme locations of the feature vector trajectory, as described in the temporal variation approach. Since there exists some knowledge about the distribution of local optima, the above approach exploits the temporal relation of feature vectors and increases the possibility of locating sets of feature vectors with small correlation within the first few GA cycles.

The metric $f(\mathbf{x})$ is used as an objective function to estimate the performance of all chromosomes $\mathbf{x}_i, i = 1, \ldots, P$ in a given population. However, a fitness function is used to map objective values to fitness values, following a rank-based normalization scheme. In particular, chromosomes $\mathbf{x}_i$ are ranked in ascending order of $f(\mathbf{x})$, since the objective function is to be minimized. Let $rank(\mathbf{x}_i) \in \{1, \ldots, P\}$ be the rank of chromosome $\mathbf{x}_i, i = 1, \ldots, P$ ($rank$=1 corresponds to the best chromosome and $rank$=P to the worst). Defining an arbitrary fitness value F_B for the best chromosome, the fitness $F(\mathbf{x}_i)$ of the *i-th* chromosome is given by the linear function

$$F(\mathbf{x}_i) = F_B - [rank(\mathbf{x}_i) - 1]D, \quad i = 1, \ldots, P \tag{20}$$

where D is a decrement rate. The major advantage of the rank-based normalization is that, since fitness values are uniformly distributed, it prevents the generation of super chromosomes, avoiding premature convergence to local minima. Furthermore, by simply adjusting the two parameters F_B and D, it is very easy to control the selective pressure of the algorithm, effectively influencing its convergence speed to a global minimum.

After fitness values, $F(\mathbf{x}_i), i = 1, \ldots, P$, have been calculated for all members of the current population, parent selection is then applied so that a fitter chromosome gives a higher number of offspring and thus has a higher chance of survival in the next generation. The *roulette wheel selection procedure* (Goldberg, 1989) is used for parent selection, by assigning each chromosome a probability of selection proportional to its fitness value. The roulette wheel selection is one of the most popular methods, because it ensures that each chromosome has a growth rate proportional to its fitness value. Note also that due to rank-based normalization, selection probabilities remain constant between generations.

A set of new chromosomes (offspring) is then produced by mating the selected parent chromosomes and applying a crossover operator. The genetic material of the parents is combined in a random way in order to produce the genetic material of the offspring. A more general technique, that is employed in this application, is the uniform crossover, where each parent gene is considered to be a potential crossover point. This means that two parents

$$\mathbf{a}_0 = (a_1^0, a_2^0, \ldots, a_K^0) \text{ and } \mathbf{a}_1 = (a_1^1, a_2^1, \ldots, a_K^1)$$

generate the following two offspring:

$$\mathbf{a}_0' = (a_1^{s_1}, a_2^{s_2}, \ldots, a_K^{s_K}) \text{ and } \mathbf{a}_1' = (a_1^{1-s_1}, a_2^{1-s_2}, \ldots, a_K^{1-s_K})$$

where s_i, i=1,...,K are random numbers taking values 0 or 1 with equal probabilities, so that each component comes from the first or the second parent. In the examined problem, K=3.

Although single-point crossover is considered to be inferior to other techniques, no evidence has been reported in favor of uniform, multi-point or other types of crossover operators (such as arithmetical, segmented, or shuffle) (Michalewicz, 1994). Instead, this selection is heavily problem-dependent, and in the examined case uniform crossover has exhibited slightly better performance in the experiments.

The next step is to apply mutation to the newly created chromosomes, introducing random gene variations that are useful for restoring lost genetic material, or for producing new material that corresponds to new search areas. Uniform mutation is the most common mutation operator and is selected for the optimization problem under consideration. In particular, each offspring gene x_i is replaced by a randomly generated one x_i', with probability p_m. That is, a random number $r \in [0,1]$ is generated for each gene and replacement takes place if $r < p_m$; otherwise the gene remains intact. Other alternatives, such as non-uniform, boundary, or swap operators, are also possible. Non-uniform mutation is in general preferable in numerical optimization problems with respect to accuracy and convergence speed, but did not achieve better performance in the problem under consideration.

Once new chromosomes have been generated for a given population $\mathbf{X}(n)$, $n \geq 0$, the next generation population, $\mathbf{X}(n+1)$, is formed by inserting these new chromosomes into

$\mathbf{X}(n)$ and deleting an appropriate number of older chromosomes, so that each population consists of P members. The exact number of old chromosomes to be replaced by new ones defines the replacement strategy of the GA and greatly affects its convergence rate. An elitist strategy has been selected for replacement, where a small percentage of the best chromosomes is copied into the succeeding generation together with their offspring, improving the convergence speed of the algorithm.

Several GA cycles take place by repeating the procedures of fitness evaluation, parent selection, crossover and mutation, until the population converges to an optimal solution. The GA terminates when the best chromosome fitness remains constant for a large number of generations, indicating that further optimization is unlikely.

5.5.4 Stochastic Logarithmic Search vs. Genetic Algorithm

In order for the performance of the two methods to be evaluated, they are applied to the results of a market survey conducted in Paris, for the design of a new Cretan olive oil product (Siskos *et al.*, 2001). A total of 204 olive oil consumers tasted 6 different olive oils, rank them from the most to the least preferred, and rated them in 5 attributes: image (4 levels), color (3 levels), odour (3 levels), taste (3 levels), and package (4 levels). The customers' ranking data are used as external shares, assuming that a consumer always purchases the product that ranks first. The customers' marginal utilities (partworths) are derived from the application of UTASTAR. About 300,000 different values of the three parameters (R, K and S) are generated, for which the distribution of f is shown in Figure 5.1.

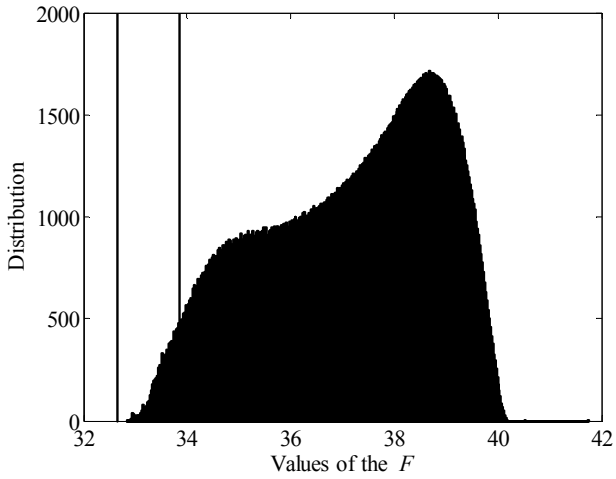


Figure 5.1: The distribution of the values of the f for more than 300,000 different values of the R, K and S.

The same figure presents the minimum value derived by the application of the Stochastic Logarithmic method (dotted vertical line) and the Genetic Algorithm method (solid vertical line). The value for the stochastic approach is 33.85, while for the genetic is 32.65. It can be observed that the genetic method gives much better results than the stochastic logarithmic approach. The genetic method is terminated after 2000 iteration cycles. Table 5.2 shows the relatively cost of the methods used.

Table 5.2: Comparison of the computational complexity of the two proposed algorithms

Method	Best Score Obtained	Average Computational Time (sec)
Stochastic Logarithmic Search	33.85	3.4
Genetic Algorithm	32.65	1

Additionally, the effect of different genetic parameters on the selection of the optimal value is depicted on Figure 5.2.

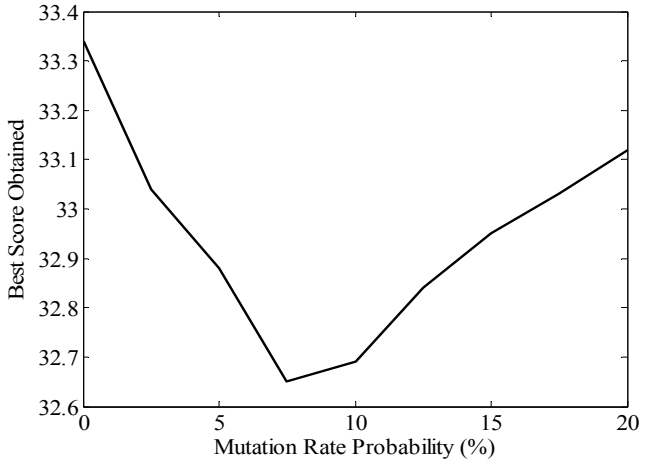


Figure 5.2: The best obtained values of *f* using the GA at different mutation rates and for 2000 iteration cycles and cross over probability of 20%

Particularly, Figure 5.2 illustrates the minimum value obtained for the same number of cycles (2000 in this case) using different mutation rate probabilities. The cross over selection probability is 20% in this case. As is observed the best results are derived for a mutation rate of about 7.5%. Similarly, the effect of cross over probability of the results is shown is Figure 5.3. Again, 2000 iteration cycles has been selected while the mutation rate is the best obtained, i.e., 7.5%. It is clear that a cross over probability of 20% gives the best performance.

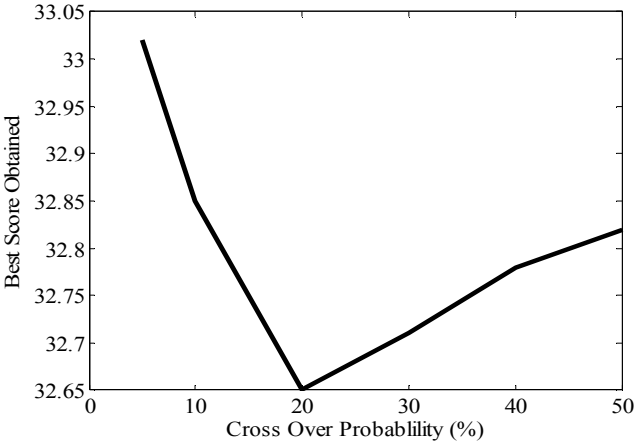


Figure 5.3: The best obtained values of f using the genetic algorithm at different cross-over rates and for 2000 iteration cycles and mutation of 20%.

The full search approach demands a great computational complexity. The stochastic approach reduces this complexity but the fact that the method should be iterated using different experiments increases relatively the cost as well. Instead, the genetic method results in best performance while demanding the smaller computational cost. Hence, for the rest of study the method that will be used for optimizing the value of *R*, *K*, and *S*, is Genetic Algorithms.

5.5.5 Evaluation of the different approaches

In order for the performance of the eight approaches to be to assessed, 50 replications are executed. In each replication, random part-worths and product configurations are generated, while different data sets are used for model calibration and evaluation following the *Principle of Model Validation* (Elrod, 2001). According to this principle, if a model is estimated from a sample, and needs to be generalized to the total population, then different portions of the sample should be used for evaluation and estimation purposes. Thus, the models are calibrated with the use of 400 respondents, and are then applied to a different evaluation sample of 400 respondents. The calibration sample is only used for the estimation of the *R*, *K, S* and *a* parameters, whereas the models' performance is measured using the evaluation sample. The *Mean Absolute Error* between the simulated shares and the real market shares for each model is used as the main performance measure:

$$MAE=\sum_{i=1}^{m} \left| RS_i - SS_i \right| \tag{21}$$

where m is the number of products, RS_i are the real and SS_i are the simulated market share for product i. The mean value and the standard deviation of each of the 4 parameters across the 50 calibration sets are presented in Table 5.3:

Table 5.3: Parameters' mean values and the standard deviations across the 50 calibration sets

Model	Parameter	Mean	Std
1	R	9.6156	14.1424
	K	14.5159	21. 6499
	S	0.1066	0.4546
2	R	10.6386	16.6659
	K	16.8929	25.9584
3	R	19.3679	14.8433
	S	0.0558	0.3078
4	K	0.0047	0.0222
	S	34.3969	25.0943
5	R	22.576	15.0266
6	K	37.64	26.0959
7	S	39.544	101.6717
Alpha	A	128.28	116.2849

Table 5.4 shows the mean value of *MAE* for the 8 models both for the calibration and the evaluation sets across the 50 replications.

Table 5.4: Mean MAE values for the 8 models in the calibration and the evaluation sets*

Model	Calibration	Evaluation
1	$14.6612^{3,4,5,6,7,8}$	$16.8721^{2,3,4,5,6,7,8}$
2	14.9283^{7}	17.2429^{7}
3	15.0986^{7}	17.3465^{7}
4	15.0049^{7}	17.2852^{7}
5	15.0704^{7}	17.5573^{7}
6	14.9909^{7}	17.3041^{7}
7	36.5781	37.5036
8 (Alpha)	14.9603^{7}	17.2214^{7}

*Statistical significant differences between two means are denoted by superscript numbers attached to the superior mean ($p<0.1$).

The results indicate that the 1st approach outperforms the others, the 7th performs significantly worse and there are no significant differences among the others. The first two approaches exhibit a somewhat better calibration performance than the ALPHA rule, while only the 1st gives a slightly lower MAE in the evaluation sample; hence it is the one that is proposed for use.

5.5.6 The Corrective Method

Since the Pessemier model is subject to the IIA property, it will be adjusted for similarity in order to account for differential substitution between alternatives. This is implemented through the application of a corrective method to the results of the choice models (choice probabilities for each product) at the individual level. The degree of similarity S_{ij} among two products i and j is measured with the use of the attribute partworths:

$$S_{ij}=1-(\sum_k |U_{ik} - U_{jk}|)/2, \quad (22)$$

where U_{ik} is the partworth of attribute k for product i.

Initially the similarity matrix is created through the estimation of the similarity degree for every pair of products. The degree of similarity of product i to all other products, defined as its "total similarity" (Paffrath, 1997), is calculated by summing up the elements of column i. The correction is implemented through the division of the choice probability of each alternative by its corresponding total similarity. Finally, the corrected shares are normalized in order to sum up to 100%.

The effectiveness of the corrective method will be empirically assessed using six criteria established by Paffrath (1997) that all methods designed to minimize the IIA bias should met. According to the first criterion, the method should account for product similarity, so that when a new item is introduced to the market, it gains more share from the relatively similar items to it and less share from the relatively dissimilar ones. The application of the method should be at the individual level (criterion 2), and it must depend both on the customer's importance structure (criterion 3), and on all product's attributes (criterion 4). Slight product changes should only lead to slight changes in similarity and therefore only to minimal changes in its share (criterion 5). The

worsening of a product (e.g. an increase in its price) should not lead to an improvement of its share (criterion 6).

The proposed method is indeed applied to each consumer separately (criterion 2), and the part-worths that form the similarity degree (equation 7) depend both on the attribute importance weights (criterion 3), and the attribute values (criterion 4). In order for the compliance of the method with the rest criteria to be assessed, the method will be applied to the following case. Two products consisting of two attributes are evaluated by an individual (Table 5.5), and their choice probabilities are estimated using the BTL model.

Table 5.5: Product Evaluations

	Part-worth of Attribute 1	Part-worth of Attribute 2	Utility Value
Product 1	0.95	0.05	1
Product 2	0.05	0.95	1
Product 3	0.94	0.06	1

The two products have equal utility values, thus it is expected that the choice probability for each will be 50%. Now a third product, highly correlated to product 1, enters the choice set. The application of the BTL model without correction for similarity, predicts that product 3 gains equal share from the other two (proportionality), resulting in a 33% final share for each of the three items (IIA bias). If this was true in practice, the choice probability of product 2 could be minimized, through the entrance to the choice set of a number of very similar extensions of product 1. This is a fault simulation of reality, since a person who prefers, for example, driving to work, will probably take his car no matter how many different buses are available. The similarity matrix for the corrective method is created next (Table 5.6).

Table 5.6: The Similarity Matrix

	Product 1	**Product 2**	**Product 3**
Product 1	1	0.1	0.99
Product 2	0.1	1	0.11
Product 3	0.99	0.11	1
Total Similarity	2.09	1.21	2.1

The application of the corrective method results in 26.86% simulated share for product 1, 46.41% for product 2, and 26.73% for product 3. Although the three alternatives share the same utility value, product 3 gains more share (23.14%) from the almost identical product 1 (substitution), and only a small amount (3.59%) from the dissimilar product 2, showing that the similarity among products has been successfully incorporated into the choice process (criterion 1). Furthermore, the attraction effect has been effectively addressed, since the almost identical products 1 and 3 receive total share that exceeds the 50%.

Now the level of attribute 2 of product 1 is altered, and its part-worth changes from 0.05 to 0.01. The similarity degree of product 1 is reduced from 2.09 to 2.05, and the item becomes more dissimilar to the other two. The slight product modification resulted in only a small change in its similarity degree, due to the linear transformation function used, and therefore its share dropped insignificantly from 26.86% to 26.15% (criterion 5). Furthermore the increase in the product's share due to the corrective method (lower similarity), does not compensate the decrease due to its lower utility (0.96), thus the worsening of the product does not lead to an improvement in its share (criterion 6). This is an advantage of corrective methods that use linear transformation functions instead of convex (e.g., negative exponential), which may overestimate small product changes, resulting in large changes in choice shares. In such cases, a drop in an item's utility may result in an increased share, due to the excessive improvement caused by the decreased similarity.

5.6 A Monte Carlo study for the model's performance evaluation

A Monte Carlo experiment was designed in order for the predictive accuracy of the proposed approach to be to compared with that of the ALPHA rule, and 3 more

traditional choice models: the BTL, the Lesourne, and the MNL. Based on previous studies in market simulation models (Baier and Gaul, 2001) as well as the design of previous Monte Carlo studies for conjoint data (Vriens *et al.*, 1996), eight independent factors were specified as having potential impact on the performance of the models, each varying at two levels (Table 5.7).

Table 5.7: Factors included in the study

Factor		Levels	
1	Number of simulated respondents	100	500
2	Number of segments	2	4
3	Number of alternatives	4	8
4	Number of attributes	5	9
5	Number of levels	3	6
6	Segment heterogeneity	Homogeneous	Heterogeneous
7	Percentage of error variance on preferences	5%	35%
8	Similarity among alternatives	Dissimilar	Similar

5.6.1 Factors included in the study

The effect of the sample size on the models' performance will be evaluated using the levels 100 and 500, since the number of respondents reported in the majority of commercial uses of conjoint analysis, was as low as 30 and as high as 1000 with a mean of 268 (Wittink *et al.*, 1994). In order to determine whether the number of segments has an impact on the models' results, a two and a four segment situation is selected according to the related literature (Wedel and Steenkamp, 1989). The influence of the segments' heterogeneity will be assessed by setting the variance of the part-worths' distribution within each segment to either 0.05 for the homogeneous or 0.1 for the heterogeneous situation (Hagerty, 1985). The number of alternatives is set to 4 or 8 (Baier and Gaul, 2001), in order to test the implications of the size of the choice set (the number of products from which the customer selects) to the models' performance. Five and nine was arranged for the *low* and *high* number of attributes conditions respectively, as in Huber *et al.* (1993). For simplicity, equal number of levels was chosen for each attribute, within a range widely used in conjoint studies (i.e. 3 and 6). The 5%

and 35% percentages of error variance on preferences (Wedel and Steenkamp, 1989) will show the sensitivity of the models' predictive accuracy to the noise added when measuring customer preferences. Finally, the effect of the existence of similar alternatives in the choice set is investigated, in order to assess the models' tolerance with the IIA bias. A fractionated factorial design with 12 different profiles is employed (Table 5.8), as derived from Addelman's (1962) basic plans.

Table 5.8: The factor-level combinations

Profile	Respondents	Segments	Alternatives	Attributes	Levels	Variance	Error	Similarity
1	100	4	8	5	6	0.1	0.35	No
2	500	4	8	5	3	0.05	0.35	No
3	500	4	4	9	6	0.05	0.05	No
4	500	2	8	5	6	0.1	0.05	Yes
5	500	2	4	9	6	0.1	0.35	No
6	500	2	8	9	3	0.05	0.35	Yes
7	100	2	8	9	3	0.1	0.05	No
8	100	2	4	5	6	0.05	0.35	Yes
9	100	4	4	9	3	0.1	0.35	Yes
10	100	2	4	5	3	0.05	0.05	No
11	500	4	4	5	3	0.1	0.05	Yes
12	100	4	8	9	6	0.05	0.05	Yes

5.6.2 Data generation

Thirty replications were performed for each combination, resulting in a total of 360 synthetic data sets. The data sets consist of simulated part-worths for each respondent, as well as hypothetical market scenarios with different product configurations. Initially the part-worth functions of 5,000 hypothetical respondents are estimated, who belong to 2 or 4 equally sized segments. Part-worths for each attribute level are randomly drawn from a normal distribution, with a different mean for each segment selected within the range [0, 1]. The variance of the distribution within each segment is set to either 0.05 or

0.1. Next, the market scenario is formulated through the choice of the level of each attribute for every product. When no similar alternatives exist in the choice set, the selection of the attribute levels is totally random for all products. In the other case, it is assumed that there exist two groups of similar alternatives, consisting of 1 and 3, or 2 and 6 products, depending on the level of the factor *Number of alternatives*. Following Baier and Gaul (2001), the same level on the first 3 or 5 attributes is set for the alternatives within each group, whereas the remaining 2 or 4 attributes are assigned levels at random.

The utility value assigned by the respondents to each product is calculated with the use of the simulated part-worths along with the products' configuration. The "real" market shares are estimated for each scenario, with the use of the error free utilities and assuming a deterministic first choice rule for each individual. Then the sample of the population on which the models will be assessed is constructed, by selecting 200 or 1,000 from the 5,000 respondents (equal number from each segment). Error terms are added to the sample's "true" utilities, following a normal distribution with zero mean and 0.05 or 0.35 Percentage of Error Variance on preferences according to equation (18). The two models are calibrated with the use of 100 or 500 respondents, and are then applied to a different evaluation sample of 100 or 500 respondents respectively. The calibration sample is only used for the estimation of the R, K and S parameters of the proposed model and the a exponent of the ALPHA rule, whereas the performance of the five models is measured using the evaluation sample.

5.6.3 Results of the Monte Carlo study

The mean value and standard deviation of each of the 4 parameters (R, K, S and a) across all the 360 calibration sets are presented in Table 5.9.

Table 5.9: Parameters' values across all calibration sets

Proposed model						**ALPHA**	
R		***K***		***S***		***A***	
Mean	Std	Mean	Std	Mean	Std	Mean	Std
8.74	11.85	12.75	13.11	-0.08	6.8	115.6	106.8

As it is observed, the *K* parameter has the highest mean value and the lowest coefficient of variation (standard deviation to mean ratio), meaning that *kurtosis* is the coefficient with the highest impact, followed by the *range of the utilities*, whereas the effect of *skewnes* is rather limited. The mean values of *MAE* deriving from the calibration of the proposed approach and the ALPHA model are illustrated in Table 5.10, where the results with and without the application of the corrective method are given.

Table 5.10: Mean MAE values for the calibration of the 2 models with and without correction for similarity*

Proposed		**ALPHA**	
no correction	*with correction*	*no correction*	*with correction*
1	2	3	4
14.7126[2,3,4]	22.4189[4]	15.0141[2,4]	22.8941

*Statistical significant differences between two means are denoted by superscript numbers attached to the superior mean ($p<0.1$).

After the proposed model and the ALPHA model have been calibrated, they are applied to the corresponding evaluation data set, both with and without the corrective method, along with the three traditional models. In accordance with previous studies comparing the predictive accuracy of consumer choice models (Currim, 1982), the following prediction measures between the simulated and the real market shares are used:

- *Mean Absolute Error* ($MAE=\sum_j | RS_j - SS_j |$),
- *Mean Percentage Error* ($MPE=\sum_j | RS_j - SS_j / RS_j |$),
- and *Mean Square Error* ($MSE=\sum_j (RS_j - SS_j)^2$).

Table 5.11 presents the mean values of the three measures deriving from the application of the 5 models across the 360 evaluation sets.

Table 5.11: Mean values for the three error measures in the evaluation sets*

			MAE	MPE	MSE
Proposed	*no correction*	1	$15.80^{2,3,4,5,6,7}$	$2.37^{2,3,4,5,6,7}$	$7.32^{2,3,4,5,6,7}$
	with correction	2	$23.89^{4,5,6,7}$	$3.53^{4,5,6,7}$	$11.02^{4,5,6,7}$
ALPHA rule	*no correction*	3	$16.12^{2,4,5,6,7}$	$2.52^{2,4,5,6,7}$	$7.48^{2,4,5,6,7}$
	with correction	4	$24.38^{5,6,7}$	$3.61^{5,6,7}$	$11.26^{5,6,7}$
Bradley Terry Luce		5	55.96^{7}	9.35^{7}	25.49^{7}
Lesourne		6	$52.061^{5,7}$	$8.44^{5,7}$	$23.42^{5,7}$
MultiNomial Logit		7	59.57	9.81	27.24

*Statistical significant differences between two means are denoted by superscript numbers attached to the superior mean ($p<0.1$).

As it was expected the *MAE* of the proposed model for the calibration set is lower than that for the corresponding evaluation set, but the mean difference is relatively small (6.7%). This shows that the model avoids overfitting to the calibration set, and achieves high performance on the evaluation set too. The proposed approach outperforms the ALPHA rule in all cases, while the clear superiority of both approaches over the traditional models is obvious. The incorporation of the corrective method results in a 50% loss in prediction performance. This is reasonable, since the study is based on simulated data, and the real shares have been generated using the first choice rule, hence human subjectivity is not reflected. Yet, the mean error of the models with the corrective method remains less than half of the error derived from the traditional models. More clear conclusions about the performance of the corrective method will be drawn in the next section where real world data are used.

The mean values of *MAE* of each model under the different levels of the 8 factors included in the simulation are shown in Table 5.12.

Table 5.12: Mean MAE values under each of the factor levels for each model

Factor	Level	Proposed model		ALPHA rule		Bradley Terry Luce	Lesourne	Multi Nomial Logit
		no correction	*with correction*	*no correction*	*with correction*			
Respondents	100	16.5729	24.6496	16.7431	24.1859	54.7115**	50.1077*	58.221**
	500	15.3069*	23.9513**	15.5137*	23.0701**	57.2196	54.0143	60.9306
Segments	2	16.1085	25.0726	16.3272	25.6989	63.7151	58.5955	67.6043
	4	15. 9714	24.5284	16.0295**	24.9879	48.2160*	45.5265*	51.5472*
Alternatives	4	12.6780*	19.7930*	13.0711*	20.3518*	47.3152*	43.8257*	51.3155*
	8	19.1856	28.8080	19.2018	28.4220	64.6159	60.2963	67.8360
Attributes	5	17.6947	28.3923	17.8285	28.5579	57.8080	53.5855	62.3540
	9	14.1852*	20.2087*	14.4282*	20.2159*	54.1231**	50.5365**	56.7976*
Levels	3	17.0158	33.4907	17.1702	33.4801	57.3012	53.8031	60.8693
	6	14.8640*	15.1102*	15.0866*	15.2938*	54.6299**	50.3189**	58.2822**
Segment Heterogeneity	Homogeneous	15.9566	23.7555	16.1619	24.4939	61.2276	56.4835	64.6991
	Heterogeneous	15.9232	24.2071	16.0948	24.2183	50.7036*	47.6385*	54.4525*
Similarity of Alternatives	Similar	17.5827	25.3799	17.8328	24.8121	59.3911	56.1344	63.2165
	Dissimilar	14.2971*	23.9617**	14.4240*	23.2211**	52.5401*	47.9876*	55.9350*
Error Variance	5%	12.5447*	23.6590*	12.6940*	23.2025*	56.4006	52.7935	60.4176
	35%	19.3351	25.3985	19.5627	25.1148	55.5305	51.3285**	58.7340**

The difference between the two means, generated under the levels of the corresponding design factor, is significant:

* at the 0.01 level,

** at the 0.1 level.

As observed, the proposed approach does not underperform on any factor level. The two calibrated models perform better when applied to more respondents, managing to accomplish better calibration as the sample increases. On the contrary, traditional models exhibit a 6.5% error increase for 5 times more respondents, failing to take advantage of the larger data set and to improve their forecasting performance. If the number of segments is doubled, the traditional models display a 23% increase in the predictive accuracy. This factor, as well as the *segment heterogeneity*, does not impact the performance of the two calibrated models. Traditional models reflect a 17% difference in performance between the 2 different levels of segment heterogeneity. When the number of alternatives is doubled, the calibrated approaches exhibit a 50% loss in forecasting accuracy without the corrective method and a 46% with the incorporation of the method. A 20% increase in performance is displayed between the 5 and the 9 attribute situation, as well as the 3 and the 6 level situation. When there are similar alternatives in the choice set, the calibrated models encounter an 18% performance drop. The incorporation of the corrective method reduces the percentage of losses to only 5%. The same holds in case that 7 times more noise is added, where the calibrated models exhibit a 54% raise in the mean prediction error, while the addition of the corrective method diminishes the increase to only 7%.

5.7 A real world application

In order to test the performance of the model in real world conditions, the data from the market survey (chapter 4) were used. The data from Table 4.2 were used for the estimation at the individual level of the part-worths for every level of each attribute. Next, the customer sample was randomly divided into two equal groups of 240 respondents each. The proposed model and the ALPHA rule were calibrated, by estimating the values of the R, K, S and a parameters that best predict the holdout choices for the first group. Table 5.13 shows the *MAE* resulted from the models' calibration.

Table 5.13: Mean absolute error for the calibration of the 2 models with and without correction for similarity*

Proposed		**ALPHA**	
no correction	*with correction*	*no correction*	*with correction*
1	2	3	4
22.9168[3]	19.9566[1,3,4]	23.7051	20.8449[1,3]

*Statistical significant differences between two means are denoted by superscript numbers attached to the superior mean ($p<0.1$).

The parameters estimated (Table 5.14) were used in the second group, in order to assess the predictive accuracy of the five models.

Table 5.14: Parameters' values derived from models' calibration

Proposed model			**ALPHA**
R	*K*	*S*	*A*
1.47	5.64	-1.57	15.94

Table 5.15 presents the *MAE*, the *MPE* and the *MSE* for the application of the five models on the second group of respondents.

Table 5.15: The values for the three error measures on the evaluation group*

			AE	PE	LSE
Proposed model	no correction	1	23.74[2,4,5,6,7]	3.59[2,4,5,6,7]	11.05[2,4,5,6,7]
	with correction	2	20.81[1,3,4,5,6,7]	3.11[1,3,4,5,6,7]	9.86[1,3,4,5,6,7]
ALPHA rule	no correction	3	24.653[5,6,7]	3.817[5,6,7]	11.903[5,6,7]
	with correction	4	21.97[1,3,5,6,7]	3.24[1,3,5,6,7]	10.46[1,3,5,6,7]
Bradley Terry Luce		5	42.99209[7]	6.612[7]	20.033[7]
Lesourne		6	41.6241[5,7]	6.048[5,7]	20.816[5,7]
MultiNomial Logit		7	43.7552	6.914	21.537

*Statistical significant differences between two means are denoted by superscript numbers attached to the superior mean ($p<0.1$).

As the results indicate, the proposed approach outperforms the ALPHA rule and the traditional models in the real world application as well. The model achieves a better fit to real data, since the difference in *MAE* between calibration and evaluation is only 4%. The addition of the corrective method improves the performance of the calibrated models, resulting in a 14% mean reduction in error prediction.

6 Particle Swarm Optimization

In this chapter the optimization part of the Optimal Product Line Design problem is implemented with the use of Particle Swarm Optimization, a nature inspired intelligence technique introduced by Kennedy and Eberhart (1995) that has been successfully applied to a wide variety of optimization problems of high complexity. The performance of the proposed approach is benchmarked against Genetic Algorithms, the only population-based algorithm that has been currently applied to the problem. For simplicity reasons, the BTL choice model is employed to implement the comparison of the two algorithms.

6.1 Introduction

As discussed in a previous chapter, almost all approaches that have been applied to the Optimal Product Line Design problem aim at finding a better approximation of the global optimal solution. Whereas this may be a reasonable target in the Operational Research domain, marketing managers focus on other more practical issues. For example, when optimizing a firm's market share, the introduction of product line A may result in 25% market share for the firm. At the same time, a product line B may result in 24% market share but its production cost may be 40% less than that of product line A. Whereas product line A seems to be the best choice, achieving the highest performance concerning the optimization objective (market share), product line B constitutes a better option for the firm taking into account the significantly lower production cost. As long as single-objective optimization algorithms are used in the Optimal Product Line Design problem, such a drawback can only be alleviated with an algorithm that provides a number of alternative

good solutions to the problem. Genetic Algorithms (GA), which belong to the class of population-based algorithms, have been successfully applied to the optimal product line design problem, displaying high performance in the provision of both an optimal solution and a set of good near-optimal solutions.

In this chapter, a new population-based optimization algorithm called *Particle Swarm Optimization* (PSO) is proposed for solving the Optimal Product Line Design problem. Since this approach is the first to employ PSO on the specific problem, the PSO algorithm and its variants will be presented first, and then the algorithm's parameters will be fine tuned and the best approach will be selected. Finally, the effectiveness of the approach will be evaluated, through a comparison of the performance of PSO with that of GAs, regarding a number of variables of interest.

6.2 Original Algorithm

Particle swarm optimization is a population-based swarm intelligence algorithm. It was originally proposed by Kennedy and Eberhart (1995) as a simulation of the social behaviour of social organisms such as bird flocking and fish schooling. PSO uses the physical movements of the individuals in the swarm and has a flexible and well-balanced mechanism to enhance and adapt to the global and local exploration abilities. Most applications of PSO have concentrated on the optimization in continuous spaces, while recently some work has been done to the discrete optimization problem. Recent complete surveys for the Particle Swarm Optimization can be found in Banks *et al.* (2007, 2008) and Poli *et al.* (2007). The wide use of PSO, mainly during the last years, is due to the number of advantages that this method has compared to other optimization methods. Some of the key advantages are that this optimization method does not need the calculation of derivatives, that the knowledge of good solutions is retained by all particles and that particles in the swarm share information between them. Furthermore, PSO is less sensitive to the nature of the objective function, can be used for stochastic objective functions and can easily escape from local minima. Concerning its implementation, PSO can easily be programmed, has few

parameters to regulate and the assessment of the optimum is independent of the initial solution.

The PSO algorithm works as follows. First a set of P particles (population) is randomly initialized, where a particle is a solution to the problem. The size of the population (P) remains constant throughout the algorithm's iterations. The position of each particle is represented by a d-dimensional vector in problem space $s_i = (s_{i1}, s_{i2},..., s_{id})$, $i = 1, 2,..., P$, $s \in \Re$ and its performance is evaluated on the predefined fitness function. Thus, each particle is randomly placed in the d-dimensional space as a candidate solution. The velocity of the i-th particle $v_i = (v_{i1}, v_{i2},..., v_{id})$, $u \in \Re$ is defined as the change of its position. The flying direction of each particle is the dynamical interaction of individual and social flying experience. The algorithm completes the optimization through following the personal best solution of each particle and the global best value of the whole swarm, or the local best value of a part of the whole swarm depending of the population topology that is selected in the algorithm. Each particle adjusts its trajectory toward its own previous best position and the previous best position attained by any particle of the swarm, namely p_{id} and p_{gd}. The velocities and positions of particles are updated using the following formulas:

$$v_{id}(t+1) = v_{id}(t) + c_1 rand_1(p_{id} - s_{id}(t)) + c_2 rand_2(p_{gd} - s_{id}(t)) \tag{23}$$

$$s_{id}(t+1) = s_{id}(t) + v_{id}(t+1) \tag{24}$$

where t is the iteration counter; c_1 and c_2 are the acceleration coefficients; *rand1*, *rand2* are two random numbers in [0, 1]. The acceleration coefficients c_1 and c_2 control how far a particle will move in a single iteration. Typically, these are both set equal to a value of 2, although assigning different values to c_1 and c_2 sometimes leads to improved performance. Eberhart *et al.* (1996) proposed the limiting of the speed of each particle to a range [-vmax, vmax] in order to reduce the possibility of particle moving out of the problem's space. Usually a value ±4 is used. The newly formed particles are evaluated according to the objective function, and the algorithm iterates for a predetermined number of generations (iterations), or until a convergence criterion has been met. Finally, the best solution obtained across all generations is returned. The size of the population (P)

remains constant throughout the algorithm's iterations. A pseudocode of the Particle Swarm Optimization algorithm is presented in the following.

Initialization

Select the number of neighbourhoods

Select the number of particles for each neighbourhood

Generate the initial population of the particles

Evaluate the fitness of each particle according to the objective function

Keep the optimum solution of each particle

Keep the optimum particle of each neighbourhood

Keep the optimum particle of the whole swarm

Main Phase

Do until the maximum number of generations has been reached

Calculate the velocity of each particle according to function

Calculate the new position of each particle according to function

Evaluate the new fitness of each particle

Update the optimum solution of each particle

Update the optimum particle of each neighbourhood

Update the optimum particle of the whole swarm

Enddo

Return the best solution/set of solutions.

In the following sections, an analytical presentation of the main variants of the PSO algorithm is given.

6.2.1 Inertia Weight

An improvement of the initial algorithm was proposed by Shi and Eberhart (1998) which uses an inertia weight w. The inertia weight controls the impact of previous histories of velocities on current velocity. The particle adjusts its trajectory based on information about its previous best performance and the best performance of its neighbors. The inertia weight w is also used to control the convergence behavior of the PSO. The velocities of particles are updated, now, using the following formula:

$$v_{id}(t+1) = wv_{id}(t) + c_1 rand_1(p_{id} - s_{id}(t)) + c_2 rand_2(p_{gd} - s_{id}(t)) \quad (25)$$

In order to reduce this weight over the iterations, allowing the algorithm to exploit some specific areas, the inertia weight w is updated according to the following equation:

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad (26)$$

where w_{max}, w_{min} are the maximum and minimum values that the inertia weight can take, $iter$ is the current iteration (generation) of the algorithm and $iter_{max}$ is the maximum number of iterations.

6.2.2 Constriction Factor

Clerc and Kennedy, 2002 proposed a constriction factor in order to prevent explosion, to ensure convergence and to eliminate the parameter that restricts the velocities of the particles. The velocities of particles are updated, now, using the following formula:

$$v_{id}(t+1) = \chi(v_{id}(t) + c_1 rand_1(p_{id} - s_{id}(t)) + c_2 rand_2(p_{gd} - s_{id}(t))) \quad (27)$$

where:

$$\chi = \frac{2}{\left|2 - c - \sqrt{c^2 - 4c}\right|} \tag{28}$$

and:

$$c = c_1 + c_2, c > 4. \tag{29}$$

It should be noted that a number of theoretical studies have shown that the convergence behavior of PSO is sensitive to the values of the inertia weight, the acceleration coefficients and the constriction factor (Engelbrecht, 2007). The choice of values for PSO parameters that will ensure convergence to an equilibrium point is problem dependent.

6.2.3 Population topology

There are two kinds of population topology for the Particle Swarm Optimization, the *global best* (gbest) population topology and the *local best* (lbest) population topology (Engelbrecht, 2007). In the gbest PSO the neighbourhood for each particle is the entire swarm. The social network employed by the gbest PSO reflects the *star* topology, where all particles are interconnected. Thus, the velocities of each particle are updated based on the information obtained from the best particle of the whole swarm. In the lbest PSO each particle has a smaller neighborhood. In this case the network topology reflects to the *ring* topology, where each particle communicates with only *N* other members of the swarm. The communication is, usually, achieved with the indices of the particles. Thus, if the size of the neighbourhood is equal to 2 the selected neighbours for the particle *i* are the particles *i-1* and *i+1*. Thus, the velocities of each particle are updated based on the information obtained from the best particle of the neighborhood. The use of particle indices for the creation of the neighborhood is preferred because it is very difficult and computational expensive to calculate distances between all the particles in order to find the neighbours of each particle. Furthermore, if the indices are used then a particle may belong to more than one neighbourhood, having the possibility to spread a good solution in different neighborhoods. Usually the gbest PSO

converges faster than the lbest PSO. On the other hand the lbest PSO has larger diversity in the solutions and, thus, it is more difficult to being trapped in a local minimum.

6.2.4 Discrete Particle Swarm Optimization

The basic PSO algorithm and its variants have successfully operated for continuous optimization functions. In order to extend the application to discrete spaces, Kennedy and Eberhart (1997) proposed a discrete binary version of PSO where a particle moves in a state space restricted to zero and one on each dimension and where each v_i represents the probability of bit s_i taking the value 1. Thus, the particles' trajectories are defined as the changes in the probability and v_i is a measure of individual's current probability of taking 1. If the velocity is high it is more likely to choose 1, and lower values favour choosing 0. A sigmoid function is applied to transform the velocity from real number space to probability space:

$$sig(u_{id}) = 1/1 + \exp(-u_{id}) \tag{30}$$

In the binary version of PSO, the velocities of the particles are updated using the equations (23), (25) or (27) depended on which version of the PSO is used while the positions of the particles are updated using the following equation:

$$s_{id}(t+1) = \begin{cases} 1, \text{if } \text{rand}_3 < sig(v_{id}) \\ 0, \text{otherwise} \end{cases} \tag{31}$$

where $rand_3$ is a uniform random number in [0, 1].

6.3 The proposed approach

The customers' utility functions are estimated at the individual-level, and are converted to choice likelihoods through probabilistic choice rules. The *share of choices* problem will be solved, where the goal is the maximization of a firm's market share. The application of the

model to other problems (seller's return, buyer's welfare) is straightforward. In such a context the problem is formulated as follows.

6.3.1 Problem formulation

When probabilistic choice rules are used, the market is assumed to consist of N competitive products with known configurations, including the M candidate items for the firm's line. The parameters used to formulate the problem are described below:

$\Xi = \{1,2,\ldots N\}$ is the set of products that comprise the market.

$\Omega = \{1,2,\ldots,K\}$ is the set of K attributes that comprise the product.

$\Phi_k = \{1,2,\ldots,J_k\}$ is the set of J_k levels of attribute k.

$\Psi = \{1,2,\ldots,M\}$ is the set of products to be designed ($\Psi \subset \Xi$).

$\theta = \{1,2,\ldots,I\}$ is the set of I customers.

w_{ijk}= is the part-worth that customer $i \in \theta$ assigns to level $j \in \Phi_k$ of attribute $k \in \Omega$.

The following decision variable is also used:

$$x_{jkm} = \begin{cases} 1, \text{if the level of product's m attribute k is j,} \\ 0, \text{otherwise} \end{cases} \tag{32}$$

In contrast to the deterministic choice rule formulation, customers do not have a status quo product, and do not deterministically choose the highest utility alternative. Instead, each of the N alternatives has a certain probability to be selected, which depends on its utility value. Using the BTL model, the probability that customer i will choose product m is estimated as follows:

$$P_{im} = U_{im} \Big/ \sum_{n \in \Xi} U_{in}, \quad i \in \theta, m \in \Psi, n \in \Xi \tag{33}$$

where U_{im} the utility that customer i assigns to product m (sum of its part-worths):

$$u_{im} = \sum_{k \in \Omega} \sum_{j \in \Phi_k} w_{ijk} x_{jkm}, \qquad i \in \theta, j \in \Phi_k, k \in \Omega, m \in \Psi \tag{34}$$

In this context the problem is formulated as the following non-linear program:

$$\max \sum_{m \in \Psi} \sum_{i \in \theta} P_{im} \tag{35}$$

subject to

$$\sum_{j \in \Phi_k} x_{jkm} = 1, \qquad k \in \Omega, m \in \Psi \tag{36}$$

$$x_{jkm} = 0,1 \text{ integer} \tag{37}$$

Constraint (36) requires each product in the line to be assigned exactly one level of each attribute. The objective function (35) maximizes the market share of the *m* products (probability to be purchased) of the company's line.

6.3.2 Solution Representation

One of the most critical issues when developing a PSO algorithm is the solution representation. PSO has been employed in the optimization of various continuous nonlinear functions, but the applications of PSO on discrete problems are still limited. In this section, the potential formulations of the PSO algorithm for the Optimal Product Line Design are described, which is a combinatorial optimization problem. The goal is to construct a direct relationship between the PSO particles and the problem domain. Since PSO was initially developed for optimization in continuous search spaces, the mapping that yields the best results when converting from continuous domain to the discrete domain required by the problem will be explored. For this reason the performance of three integer and two binary mapping rules is compared.

6.3.2.1 Integer Representation

In an integer representation scheme, a search space of $S=M*K$ dimension is setup. Each dimension has a discrete set of possible values limited to the range [1, J_k]. For example, consider a problem where a line comprises two products, each consisting of three attributes, which can take three different levels. A possible solution would be:

2 1 1 | 3 3 2, (38)

where in the first product, level two appears in attribute one, and level one appears in attributes two and three, while in the second product, level three appears in attributes one and two, and level two appears in attribute three. In this case, the PSO population is represented as a PxS two-dimensional array consisting of P particles, each represented as a vector of S product attributes. Thus, a particle flies in an S-dimensional search space. An attribute is internally represented as an integer value indicating the selected level during the course of PSO.

In dealing with the *Task Assignment Problem*, Salman, Ahmad and Al-Madani (2002) map an n-task assignment instance into the corresponding n-coordinate particle position. The real values in the particles' positions are converted to integers by dropping the sign and the fractional part. This approach is employed through the further limitation of the integer value to the range [0, J_k -1], and it is given the name "Fix". Laskari, Parsopoulos and Vrahatis (2002) propose a PSO algorithm for Integer Programming, where the real values in the particles' positions are truncated to the nearest integer. In order for this approach to be applied to the product line design problem, the absolute value of the truncated particles' positions is taken, and the values are limited as before. This mapping is called "Trunc". Finally, a mapping will be tested where the values x in the particles' positions are first limited to the range [0, J_k-1] through the function $Y=x \bmod J_k$, which gives the remainder of the division of x by J_k. Y is then converted to integer through dropping its fractional part. This mapping is called "Mod". A line of two products for example are to be represented, each consisting of three attributes, which can take three levels (0, 1, 2). Table 6.1 illustrates the different product lines in which the same random particle corresponds, using the three integer mapping rules.

Table 6.1: Different integer mappings for a potential product line

	Product 1			Product 2		
Attributes	*1*	*2*	*3*	*1*	*2*	*3*
Particle	1.3	-2.1	0.6	-0.4	1.9	4.7
Fix	1	2	0	0	1	2
Trunc	1	2	1	0	2	2
Mod	1	0	0	2	1	1

As it is observed, when the three mapping rules are used, a specific particle corresponds to three different product lines. In the second attribute of product 1 for instance, the corresponding particle's value is -2.1. Under the Fix mapping, the sign is first dropped (-2.1→2.1), and then the fractional part (2.1→2). Under the Trunc mapping, the value is first truncated to the nearest integer (-2.1→-2), and then its absolute value is taken (-2→2). Under the Mod mapping, the remainder of the division of -2.1 by 3 (0.9) is taken first, and then the fractional part is dropped (0.9→0). In the third attribute of product 2, where the particle's value is 4.7, the Fix and Trunc mappings result in values 4 and 5 respectively, which are then both altered to a value of 2, since they exceed the range [0, 2]. The remainder of the division of 4.7 by 3 is 1.7, which results in a value of 1 if the fractional part is dropped (Mod mapping).

6.3.2.2 Binary Representation

In a binary representation scheme the solution (38) would be represented as: 010 100 100|001 001 010.

Here each particle dimension represents an attribute level. A value of 1 denotes that the specific level is assigned to the corresponding attribute. Therefore each particle flies in the $S=M*K*J_k$ dimension space. Since exactly one level must be assigned to each attribute, the particle is divided into parts, each describing a single attribute. Within each part exactly one dimension must take a value of 1 and all the others must take a value of 0. Tasgetiren *et al.* (2004) propose a heuristic rule called *Smallest Position Value* (SPV) to enable the continuous PSO algorithm to be applied to the Single Machine Total Weighted Tardiness

problem. In order for this rule to be applied to the product line design problem, it is modified as follows. Within each particle's part the dimension with the smallest real value takes a value of 1, and the rest take a value of 0. Liao, Tseng and Luarn (2007), apply to the Flowshop Scheduling problem a small modification of the discrete version of PSO for binary problems (6.2.4). The particle's velocity is converted to the change of probability, which is the chance of the binary variable taking the value 1. In the problem under investigation, the dimension with the highest velocity within each particle's part takes a value of 1, and the rest take a value of 0.

6.3.3 A comparison of the different mappings' performance

In order the most suitable mapping between the problem solution (line of products) and the particle to be found, the performance of the five different mappings is compared with the use of artificial data sets. This is implemented through the design of a fractional factorial experiment with five factors, each taking two levels (Table 6.2).

Table 6.2: Factors and levels used in the experiment

Factor	Levels	
Number of attributes	3	6
Number of attribute levels	4	7
Number of products in the line	2	4
Number of competing firms	3	5
Number of customers	100	500

The data sets where the five mappings will be tested consist of simulated part-worths for each customer, as well as hypothetical market scenarios with different configurations of competitive products. The market is assumed to consist of 3/5 competing firms, each offering 2/4 different products, which is also the number of products that our company plans to introduce. Each product consists of 3/6 attributes which can take 4/7 different levels. The individual-level part-worths for each attribute level, are randomly drawn from a

uniform distribution in the range [0, 1]. The part-worths are normalized within each customer, by setting the lowest level of each attribute to zero, and rescaling the sum of the best attribute levels to unity. In order for the market scenario to be formulated, each attribute level for each competitive product is randomly selected. Using the simulated part-worths along with the products' configuration, the utility value that each consumer assigns to each product is calculated. Finally, by adding the potential products under design and applying the choice model, the fitness of each possible solution is estimated. Eight combinations of the factors above were generated based on Addelman's (1962) basic plans. Twenty replications were performed for each of the eight profiles, resulting in a total of eight hundred runs of the algorithm for the five different mappings. The parameters used in the PSO algorithm (Table 6.3) are typical values found in the related literature (Kennedy & Eberhard, 2001).

Table 6.3: PSO parameters used in the experiment

Parameter	Value
Population size	50
c_1	2
c_2	2
w_{max}	0.9
w_{min}	0.1
X	0.728
Number of iterations	1000

Table 6.4 shows the average fitness among the twenty replications that each mapping achieved in each profile. Fitness values are represented as a percentage of the best value obtained from among all mappings.

Table 6.4: Mean fitness values for each mapping

Profile	Discr	Fix	SPV	Trunc	Mod
1	1	0.9952	0.9972	0.9948	0.9950
2	1	0.9952	0.9976	0.9948	0.9948
3	1	0.9925	0.9984	0.9937	0.9928
4	1	0.9970	0.9982	0.9976	0.9965
5	1	0.9873	0.9948	0.9873	0.9870
6	1	0.9953	0.9970	0.9958	0.9954
7	1	0.9918	0.9964	0.9912	0.9906
8	1	0.9964	0.9994	0.9964	0.9952
Mean	1	0.9938	0.9973	0.9939	0.9934

The superiority of the binary approaches over the integer ones is obvious. The *Discr* mapping strictly outperforms the other mappings in all profiles. The average values of the *mean*, *maximum*, and *minimum* number of iterations at which the best solution was found, as well as the time required by the algorithm to complete the one thousand iterations, are presented in Table 6.5.

Table 6.5. Values for iteration best fitness found and algorithm completion time

Measure		Discr	Fix	SPV	Trunc	Mod
Iteration best	Mean	569.1	16.2	488.4	73.8	39.2
fitness found	Max	701.9	23.7	690.3	268.7	95.6
	Min	404.7	9.5	289.3	11.2	10.1
*CPU Time**		56.9	8.2	47.1	24.7	31.3

* CPU Time is measured in seconds on a 2.4 core 2 duo PC with 4GB of RAM

As it is observed, integer mappings require less number of iterations to find the solution than do the binary mappings, but take more time to complete each iteration. The approaches that will be further tested are the *Discr*, as it achieves the higher fitness, and the *SPV*, as it achieves the second higher fitness in shorter computational time.

6.3.4 PSO configuration

This section evaluates the performance of the two binary mappings combined with different configurations of the PSO algorithm. Specifically the performance of the simple PSO will be compared with the PSO with inertia weight, the PSO with constriction factor, and the PSO with both inertia and constriction. The previous designed experiment is used here as well, and the results are illustrated in Table 6.6.

Table 6.6: A comparison of different PSO configurations under the two binary mappings

		Discr	Discr with Constriction	Discr with Inertia	Discr with Inertia & Constriction	SPV	SPV with Constriction	SPV with Inertia	SPV with Inertia & Constriction
Profile	1	0.9952	0.9948	0.9937	0.9933	0.9941	0.9976	0.9956	1
	2	0.9960	0.9952	0.9941	0.9945	0.9941	0.9980	0.9960	1
	3	0.9961	0.9972	0.9953	0.9945	0.9930	0.9992	0.9976	1
	4	0.9982	0.9976	0.9976	0.9976	0.9970	0.9988	0.9982	1
	5	0.9903	0.9857	0.9863	0.9869	0.9846	0.9960	0.9926	1
	6	0.9922	0.9914	0.9902	0.9902	0.9887	0.9953	0.9937	1
	7	0.9919	0.9884	0.9884	0.9890	0.9861	0.9942	0.9919	1
	8	0.9970	0.9969	0.9958	0.9958	0.9952	0.9982	0.9976	1
Mean Fitness*		0.9946	0.9934	0.9926	0.9927	0.9916	0.9971	0.9954	1
Iteration best fitness found	Mean	505.3	569.4	329.2	506.1	502.6	685.9	696.2	783.9
	Max	695	709.4	531.2	757.4	680.4	815.4	806.2	860.6
	Min	413.2	410.8	114	357.4	373.6	545.4	514.6	402.4
CPU Time**		49	55.1	31.9	48.9	48.5	65.9	66.9	75.3

*Fitness values are represented as a percentage of the best value obtained from among all PSO configurations

** CPU Time is measured in seconds on a 2.4 core 2 duo PC with 4GB of RAM

The use of the extra parameters helps the *SPV* approach to outperform the *Discr* approach, requiring however more iterations and computational time. The *SPV* with inertia and constriction strictly outperforms the other approaches in all profiles.

6.3.5 Population topology

In the present section the performance of the PSO algorithm with inertia and constriction under the SPV mapping will be assessed, with use of different population topologies. Specifically, the *gbest* topology and a number of different *lbest* topologies will be tested, with use of the same experiment as before. Here, a complete enumeration of the search space in each problem instance is also implemented, in order for the algorithm's approximation of the optimum solution to be to evaluated. The values of the PSO parameters are the same as in Table 6.3, except for the size of the population. Through the evaluation of the algorithm's performance using 9 different population sizes, from 20 to 90, it was found that a size of 60 gave the best results. Ten different topologies are tested; from 1 neighborhood comprising 60 particles (gbest topology), to 30 neighborhoods comprising of two particles each. Table 6.7 illustrates the mean values of the *best fitness as a percentage of the optimum*, the *iteration in which best fitness was found*, and the *computational time*, across the 160 runs for each topology.

Table 6.7: Performance evaluation of ten different PSO topologies

Topology	Number of neighborhoods	Neighborhood size	Best Fitness	Iteration best fitness found	CPU Time*
Gbest	1	60	0.9946	863	80.20
lbest2	2	30	0.9948	892.75	84.44
lbest3	3	20	0.9954	812.75	85.43
lbest4	4	15	0.9955	932.25	86.49
lbest5	5	12	0.9960	940.75	87.72
lbest6	6	10	0.9966	781.5	88.73
lbest10	10	6	0.9965	853.5	91.98
lbest12	12	5	0.9957	788.75	99.88
lbest15	15	4	0.9951	802.25	104.04
lbest30	30	2	0.9949	789.25	117.56

* CPU Time is measured in seconds on a 2.4 core 2 duo PC with 4GB of RAM

The first finding is that the *lbest* topology strictly outperforms the *gbest* topology, since the latter gives the worst mean value of the best fitness across the ten different topologies. As the number of neighborhoods increases the algorithm's performance also increases until the topology of 6 neighborhoods, each consisting of 10 particles. A decline in the algorithm's performance is observed from this point until the final topology of thirty, 2-paricle neighborhoods. The CPU time increases as the number of neighborhoods increases, since a topology with more neighborhoods requires that more local bests have to be calculated and stored in each iteration.

6.4 A comparison of Particle Swarm Optimization with Genetic Algorithms

This section will benchmark the performance of PSO against current state of the art algorithms. Previous studies have shown that Genetic Algorithms outperform Dynamic Programming (Balakrishnan and Jacob, 1996), as well as Beam Search (Alexouda and Paparrizos, 2001; Balakrishnan *et al.*, 2004), and together with Simulated Annealing provided the best performance among the 9 most important algorithms that have been applied to the problem (Belloni *et al.*, 2008). Since Simulated Annealing is a single-best approach, the performance of PSO will be compared with that of GAs. The performance of PSO and GAs will be compared without the incorporation of retaliatory responses from competition, an issue that will be elaborated in the subsequent section. Based on the results of the preceding analysis, the approach that will be applied is the PSO with inertia and constriction under the SPV mapping, using a topology of 6 neighborhoods each consisting of 10 particles. The other PSO parameters are the same as in Table 6.3, except for the number of iterations which is not fixed. Instead, a convergence criterion is employed, according to which the algorithm will terminate when the best solution is not improved for 20 succeeding generations. As for the GA configuration, the findings of previous studies which benchmark GAs against other approaches will be used.

6.4.1 Genetic Algorithm implementation

Genetic Algorithms will be implemented with the use of an integer representation scheme (instead of a binary) as in Balakrishnan *et al.* (2004), since all genetic operators (crossover, mutation) have to be applied to entire attributes (genes), (instead of a portion of binary bits that form an attribute) in order for feasible solutions to be produced. The size of the GA population is set to 100 (Balakrishnan and Jacob, 1996), and a random initialization is performed. In accordance with the findings of the sensitivity analysis performed by Steiner and Hruschka (2002), the crossover probability is set to 0.9, and the mutation rate to 0.04. A uniform crossover process is adopted, where half of the attributes exchange values between the two parents. An elitist strategy is employed as the reproduction process, where the 40 best chromosomes are selected to survive into the succeeding generation, as in

Alexouda and Paparrizos (2001). The moving average rule is adopted as the convergence criterion, where the algorithm stops iterating when the increase of the mean fitness value of the best 3 chromosomes is less than 0.2%, in comparison to the last 5 generations.

6.4.2 Performance results

Given that conjoint methods are being applied to more and more complex settings, it is important to demonstrate that PSO scales to more realistic larger problems than the previous solved, in which GAs have already been tested. For this reason, a fractional factorial experiment was designed, consisting of 5 factors each varying at two levels (Table 6.8).

Table 6.8: Factors and levels used in the experiment

Factor	Levels	
Number of attributes	5	9
Number of attribute levels	4	8
Number of products in the line	6	9
Number of competing firms	5	8
Number of customers	200	700

In a problem instance with 9 attributes and 8 levels per attribute the possible combinations for a single product are 134,217,728, while for a line of 9 products the number of possible solutions is over 10^{30}. Finding and verifying the global optimal solution through complete enumeration of the search space in problems of such sizes would require more than a month of computational time. Hence, the best solutions provided by PSO and GA will not be compared with the optimum. Instead, a relative comparison of the two algorithms' performance will be made regarding a number of variables of interest. As before eight

profiles were created and 20 replicates were generated for each, which results in a total of 160 different data sets. In order for the two algorithms to be compared with regard to the best solution found, the *ratio of the best solution found by PSO to that found by GA* will be calculate, as well as the percentage of problem instances in which a) PSO finds a better solution than GA, and b) GA finds a better solution than PSO. Table 6.9 presents the mean values across the 160 data sets.

Table 6.9: Performance results regarding the best solution found by the PSO and GA

Average PSO/GA	1.0126
PSO better than GA	23.12%
GA better than PSO	20.62%

PSO found a better solution than GA in 37 out of the 160 runs (23.12%), GA found a better solution than PSO in 33 runs (20.62%), while in 90 runs (56.25%) the two algorithms gave the same best fitness. On average, PSO performs 1.26% better than the GA. The average *iteration where best solution was found* is also estimated, as well as the *time* required for the algorithms to converge, and the *average fitness of the solutions in the final population*, and the *fitness of the worst solution in the final population*. The last two variables are calculated as a percentage of the best solution found by the two algorithms in each problem, and will enable the evaluation the quality of the entire set of solutions that each algorithm provides. Furthermore, the diversity of the alternative solutions is assessed through the estimation of the *number of unique solutions in the final population*, the *percentage of unique solutions whose fitness is at least 95% of the best solution's fitness* and the *standard deviation of their fitness*. Two solutions are considered different if they differ in at least one product, and two products are considered different if they differ in the level of at least one attribute. The mean values of the above variables of interest are illustrated in Table 6.10.

Table 6.10: Mean values of the variables of interest for PSO and GA

Variable	PSO	GA
Iteration best solution was found	775.62	32.37
Computational Time*	86.96	14.51
Percentage of unique solutions in the final population	85%	48%
Percentage of unique solutions with fitness at least 95% of the best	38.3%	9%
Average fitness of the solutions in the final population	94.97%	96.72%
Worst solution's fitness in the final population	85.68%	89.44%
Standard deviation of solutions' fitness in the final population	0.016	0.009

* CPU Time is measured in seconds on a 2.4 core 2 duo PC with 4GB of RAM

Whereas PSO completes each iteration 4 times faster than GA, it requires 6 times more computational time to converge. Fifty one out of the sixty (85%) final solutions proposed by PSO are unique, while only 48 out of the 100 are unique in the GA's final population. The average fitness of the final population is higher in GA (94.72%) than in PSO (92.97%) due to the higher number of identical solutions in the GA's final population. The wider range of good solutions that PSO provides is also indicated from the *percentage of unique solutions whose fitness is at least 95% of the best solution's fitness*, which is 38.3% (23 out of the 60 solutions) for PSO and only 9% for GA (9/100). The low performance of GA in the specific variable is an outcome of the reproduction and crossover processes, which essentially recycle the same genes (attribute levels) among the different solutions, resulting in a final population that contains many copies of 3-5 good chromosomes. This explains the 7 times higher standard deviation of fitness values in the final population of PSO compared to that of GA. The low diversity of the GA's final population can be mitigated by an increase in the mutation probability, since mutation can produce chromosomes that correspond to new undiscovered regions of the search space. Higher mutation rates were tested, but whereas this increased the GA's population diversity, it significantly impacted the fitness of the best

solution (a 3%-5% reduction), since higher mutation increases the randomness of the search. Larger GA's population sizes were also explored, but the gain was too small in relation to the extra time required for the algorithm to converge. Another effect of the reproduction process is the early elimination of the bad solutions. As a consequence, GA performs better (89.44%) in the *worst solution's fitness in the final population* compared to PSO (86.68%). The results indicate that PSO provides the decision maker with a wide range (51 out of 60) of unique high quality solutions (94.97% average fitness), among which he can choose the best 20 (which exhibit fitness over 95% of the best solution) for further evaluation. GA on the other hand, while running 6 times faster than PSO, it converges to a final population of low diversity with multiple copies of less than 10 good solutions. Such a small set of unique chromosomes can be too restrictive in many situations, since the solutions may proved to be almost identical, representing product lines that differ in only a single attribute level of one product. On the contrary, PSO searches a much larger part of the entire solution space, in an acceptable amount of time for a marketing application (less than one and a half minute). The neighborhood topology enables PSO to converge to several local optima (usually the same as the number of neighborhoods) and provide a broad range of good solutions around them.

7 Modeling competitive reactions

In this chapter the retaliatory actions from competitors are modeled in a game theoretic concept. Each firm is treated as a Nash player that optimizes its product line using PSO until a market equilibrium is reached. The approach will be demonstrated through a scenario based on the data from the milk survey.

7.1 Introduction

As illustrated in chapter 3, a number of optimization algorithms have been applied to the problem, guaranteeing even the global optimality for the single product design problem. However, all studies treat the competition as being static, where the companies that participate to the market do not react to the new entrant's move, staying with the product line configurations that were already offering. It is obvious that this does not constitute a proper representation of real world conditions. A model that is based in such an assumption can only provide an instant picture of the market that is valid only for a short period after the introduction of the new product line. However, companies which are about to spend a lot of money in developing and launching their new products are interested for the market share and profits obtained in the longer term. Only when competitors have made their moves and the market has stabilized the new entrant can recover its investment and obtain some profit.

Recent approaches that optimize product line designs using conjoint data neglect to explicitly consider potential retaliating moves from the competitors. Actually, only two

approaches have incorporated competitive reactions using game theory in the conjoint analysis context. Choi and DeSarbo (1993) apply a specialized branch and bound optimization algorithm, while modeling the competitive responses in a Nash equilibrium framework. Green and Krieger (1997), also employ the Nash equilibrium concept with the use of divide and conquer heuristic. While being the first studies to apply game theory to the product design, both approaches solve only the single product design, using traditional single-best optimization approaches. A possible reason for which a dynamic competition approach has not yet been considered in the Optimal Product Line Design problem is that conjoint attributes are typically discrete. In a discrete space the existence of a Nash equilibrium cannot be guaranteed through an analytical closed-form solution. Even if a Nash equilibrium is found, its uniqueness cannot be proved.

7.2 A Nash equilibrium approach

The above problems will be faced in the present thesis through the employment of an empirical sequential iterative process for computing a Nash equilibrium, if it actually exists. The market will be considered to be dynamic, where a new entrant introduces a line of products, and the incumbent firms respond by optimizing their products' configuration according to the objective function (equation 35). This is a game where each player (firm) acts with perfect information, by observing the competitors' earlier moves. The process continues until a Nash equilibrium (if there exists at least one) is reached. A Nash equilibrium is a situation where, given the objective function, none of the players that participate to the game can make any further gains (market share increase) by moving (altering attribute levels) unilaterally. Since an analytical closed-form solution cannot be calculated, the problem will be solved using an iterative *tatonnement* process. Tatonnement process is a term used in game theory to describe the process by which markets find their way to equilibrium. The tatonnement process can be either simultaneous, where all players form their strategies at the same time, or sequentially, where each player moves in turn, in a sequential predetermined order. In real markets, companies usually observe competitors' moves and then alter their strategies, rather than acting all together in a simultaneous manner. Hence, a sequential tatonnement process is employed, like in Choi and DeSarbo

(1993), and Green and Krieger (1997), since this constitutes a better representation of real world circumstances.

7.2.1 An illustrative real world case

The proposed approach will be demonstrated using actual conjoint data from the real market survey concerning milk buyers. According to the scenario, ALPHA plans to become one of the largest players in the Greek retail market of milk, by redesigning its product line. As mentioned in chapter 4, four companies are the main players in the market. For illustrative purposes, it is assumed that each competitor is currently offering 3 different products. Under such a scenario, a game of five players (including the new entrant) is formulated, where each alters each product strategy (optimizes its product line) in a sequential predetermined order, until a Nash equilibrium is reached. Table 7.1 presents the current market scenario, with the initial product configurations of the four incumbent firms, together with the corresponding total market share for each firm. Each product is described with four integers, each representing the selected level of the corresponding attribute. For example, the product 4312 describes a 2-litre, Goat milk, with 1.5% fat in a plastic package.

Table 7.1: The existing situation of the market before the entrance of the new firm

Firms	Products			Market Share (%)
	1	*2*	*3*	
BETA	2111	1221	3312	30.86
GAMMA	4122	3211	1322	28.02
DELTA	3122	2212	4311	25.44
EPSILON	1111	4222	2321	15.68

The game begins when the new entrant (ALPHA) introduces a line of three new products, becoming the fifth player of the market. ALPHA designs its line based on consumers' preferences (parth-worhts) and the current competitive products' configuration (Table 13), using the PSO algorithm. The best solution that PSO provides is the 3112 2221 1321. The new market shares are now ALPHA:25.44%, BETA:20.15%, GAMMA:22.03%, DELTA:19.64%, EPSILON:12.74%. The entrance of ALPHA to the market results in a 35% reduction in BETA's share and a 20% reduction in the share of the rest three firms. ALPHA competes mostly with BETA, since two of their product are almost identical, differing only in the level of a single attribute. As a consequence, it is expected that the first player that will react to the launch of ALPHA's product line is probably BETA, which will try to redesign its line and win back the lost market share. In such a situation, BETA will optimize its product line based on the new market scenario that includes ALPHA, and the consumer's part-worths, which for the purpose of the study are considered stable. In total, there are 24 different sequences of competitive moves when ALPHA initiates the game. The order of movement for the incumbent firms must be determined exogenously, usually through managerial judgment. For illustrative purposes it is assumed that the players will respond in a lexicographic order. In such a case, a Nash equilibrium is found after 8 moves, with GAMMA being the last to react. A ninth move will not be made, since neither DELTA nor any of the rest players can find a new solution that will result in an increase in their market share. The product configurations after the market has reached the Nash equilibrium along with the final market shares are shown in Table 7.2.

Table 7.2: The new situation of the market after the Nash equilibrium has been reached

Firms	Products			Market Share (%)
	1	*2*	*3*	
ALPHA	1212	2221	1321	22.31
BETA	2111	4211	2322	22.74
GAMMA	4121	3211	4321	23.42
DELTA	3122	2212	3111	19.38
EPSILON	1221	4222	2321	12.15

The configuration of more than half of the products that form the market has changed compared to the initial market scenario shown in Table 7.1. Significant differences are also displayed in the firms' market shares. BETA exhibited the greatest loss due to the entrance of ALPHA, and GAMMA was the firm that managed to improve its position after the stabilization of the market. This indicates that when the market is treated as dynamic, the outcome of the optimization process is substantially different from the static case.

Now it is assumed that the market initially comprises 5 players, with ALPHA participating with its initial product line 3112 2221 1321. The sequence in which firms move is changed from ALPHA-BETA-GAMMA-DELTA-EPSILON to GAMMA-EPSILON-DELTA-BETA-ALPHA. The market reaches a Nash equilibrium after 9 moves this time, with BETA being the last acting player. The final market shares in this case are: ALPHA:20.16%, BETA:24.09%, GAMMA:21.50%, DELTA:20.91%, EPSILON:13.34%. As it is observed, the order in which the firms make their moves affects the final outcome, since this time BETA and DELTA gained more from the competitive game than the other three players. An interesting finding is that the "first mover advantage" comes under question in the second scenario, since GAMMA that initiated the sequence exhibited a small decline in its share compared to the first scenario. On the other hand, the player that makes the last move (GAMMA in the first, BETA in the second scenario) seems to be in a favored position.

However, both issues constitute empirical findings that cannot be easily generalized. Probably the first mover advantage would hold, if the first player could foresee the future moves of the competition. In the proposed approach where a player cannot predict the future competitors' actions but has, on the contrary, perfect information concerning the other players' previous moves, the last moving firm is in a position to optimize its product line without encountering any further competitive reactions. Since however, the firm that will make the last move cannot be determined in advance, the value of such a finding is rather limited.

An issue that must be further examined is the fact that each firm optimizes its entire product line every time it makes a move. This seems to be a quite strong assumption, since in real markets production as well as advertisement costs would prevent firms to alter more than a couple of products every time they observe a competitive action. Nevertheless, modeling competitive reactions in the concept of Nash equilibrium provides useful insights, such as identifying those attribute levels that can stay intact in the long term.

8 The System

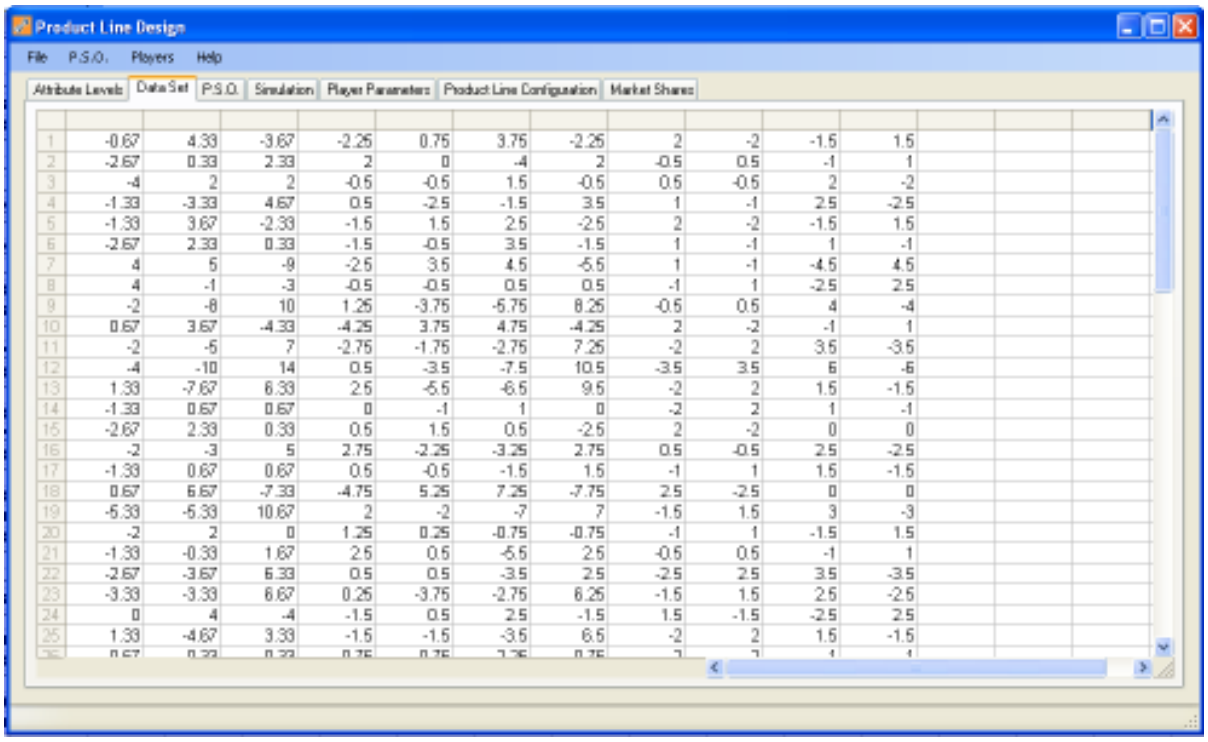
The models described in the previous chapters were all developed with the use of the R2008a edition of the MATLAB programming language. The user interface of the system was built with the use of the Microsoft Visual Basic 2008 express edition programming language. In this section, the system that incorporates the proposed models is presented. The case study example that was described in the previous chapter will be used for the illustration of the system.

The customers' partworths constitute the main input of the system. The user is provided with 3 choices for entering the customers' partworths:

1. Importing from a database.
2. Importing from a Spreadsheet.
3. Entering the data manually into a form.

The partworths are entered through the "Load Partworths" selection from the "File" menu. After the procedure has been completed the loaded partworths can be accessed through the "Data Set" tab (Figure 8.1)

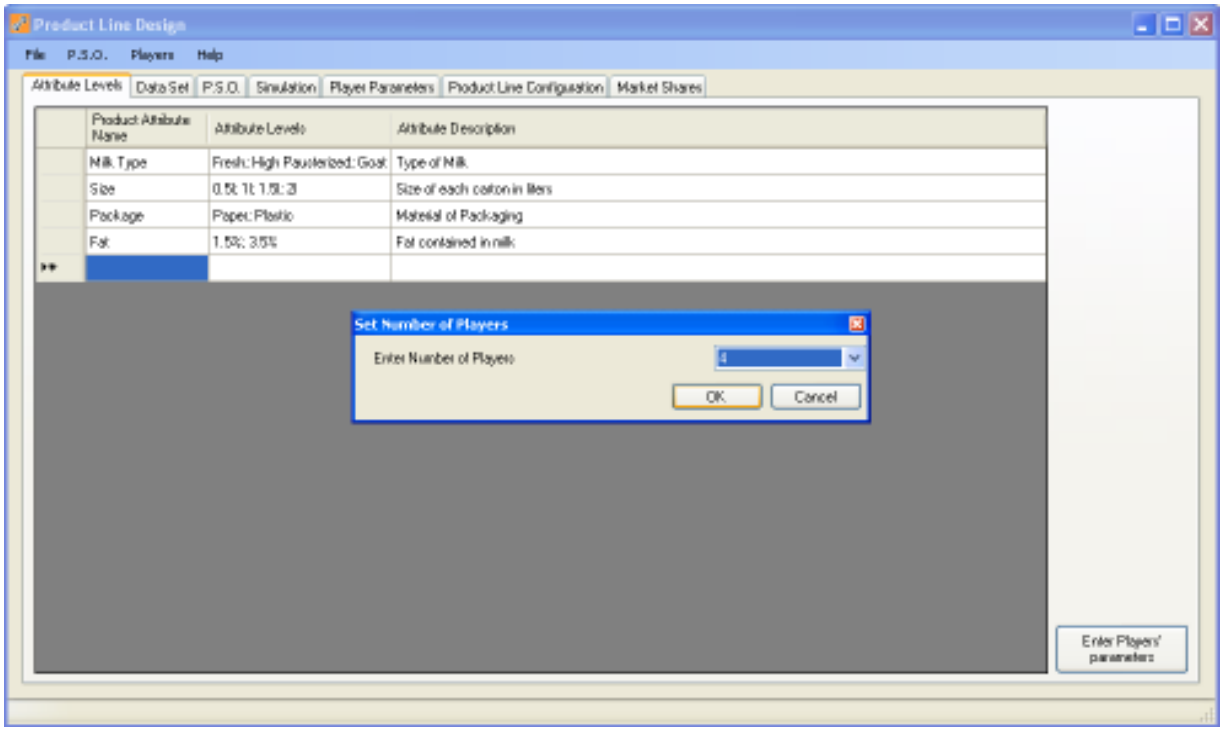
Figure 8.1: The partworths matrix



1	-0.67	4.33	-3.67	-2.25	0.75	3.75	-2.25	2	-2	-1.5	1.5
2	-2.67	0.33	2.33	2	0	-4	2	-0.5	0.5	-1	1
3	-4	2	2	-0.5	-0.5	1.5	-0.5	0.5	-0.5	2	-2
4	-1.33	-3.33	4.67	0.5	-2.5	-1.5	3.5	1	-1	2.5	-2.5
5	-1.33	3.67	-2.33	-1.5	1.5	2.5	-2.5	2	-2	-1.5	1.5
6	-2.67	2.33	0.33	-1.5	-0.5	3.5	-1.5	1	-1	1	-1
7	4	5	-9	-2.5	3.5	4.5	-5.5	1	-1	-4.5	4.5
8	4	-1	-3	-0.5	-0.5	0.5	0.5	-1	1	-2.5	2.5
9	-2	-8	10	1.25	-3.75	-5.75	8.25	-0.5	0.5	4	-4
10	0.67	3.67	-4.33	-4.25	3.75	4.75	-4.25	2	-2	-1	1
11	-2	-5	7	-2.75	-1.75	-2.75	7.25	-2	2	3.5	-3.5
12	-4	-10	14	0.5	-3.5	-7.5	10.5	-3.5	3.5	6	-6
13	1.33	-7.67	6.33	2.5	-5.5	-6.5	9.5	-2	2	1.5	-1.5
14	-1.33	0.67	0.67	0	-1	1	0	-2	2	1	-1
15	-2.67	2.33	0.33	0.5	1.5	0.5	-2.5	2	-2	0	0
16	-2	-3	5	2.75	-2.25	-3.25	2.75	0.5	-0.5	2.5	-2.5
17	-1.33	0.67	0.67	0.5	-0.5	-1.5	1.5	-1	1	1.5	-1.5
18	0.67	6.67	-7.33	-4.75	5.25	7.25	-7.75	2.5	-2.5	0	0
19	-5.33	-5.33	10.67	2	-2	-7	7	-1.5	1.5	3	-3
20	-2	2	0	1.25	0.25	-0.75	-0.75	-1	1	-1.5	1.5
21	-1.33	-0.33	1.67	2.5	0.5	-5.5	2.5	-0.5	0.5	-1	1
22	-2.67	-3.67	6.33	0.5	0.5	-3.5	2.5	-2.5	2.5	3.5	-3.5
23	-3.33	-3.33	6.67	0.25	-3.75	-2.75	6.25	-1.5	1.5	2.5	-2.5
24	0	4	-4	-1.5	0.5	2.5	-1.5	1.5	-1.5	-2.5	2.5
25	1.33	-4.67	3.33	-1.5	-1.5	-3.5	6.5	-2	2	1.5	-1.5

In order for the system to process these partworths, the attributes that represent the product together with the corresponding levels must be entered to the system. The user must give the name of each attribute and the corresponding levels separated by semicolons. The attribute levels must be entered in the same order as they appear in the file containing the partworths. A short description for each attribute may also be given. The process is accomplished through the "Enter product configuration" selection from the "File" menu which opens the "Attribute Levels" tab (Figure 8.2).

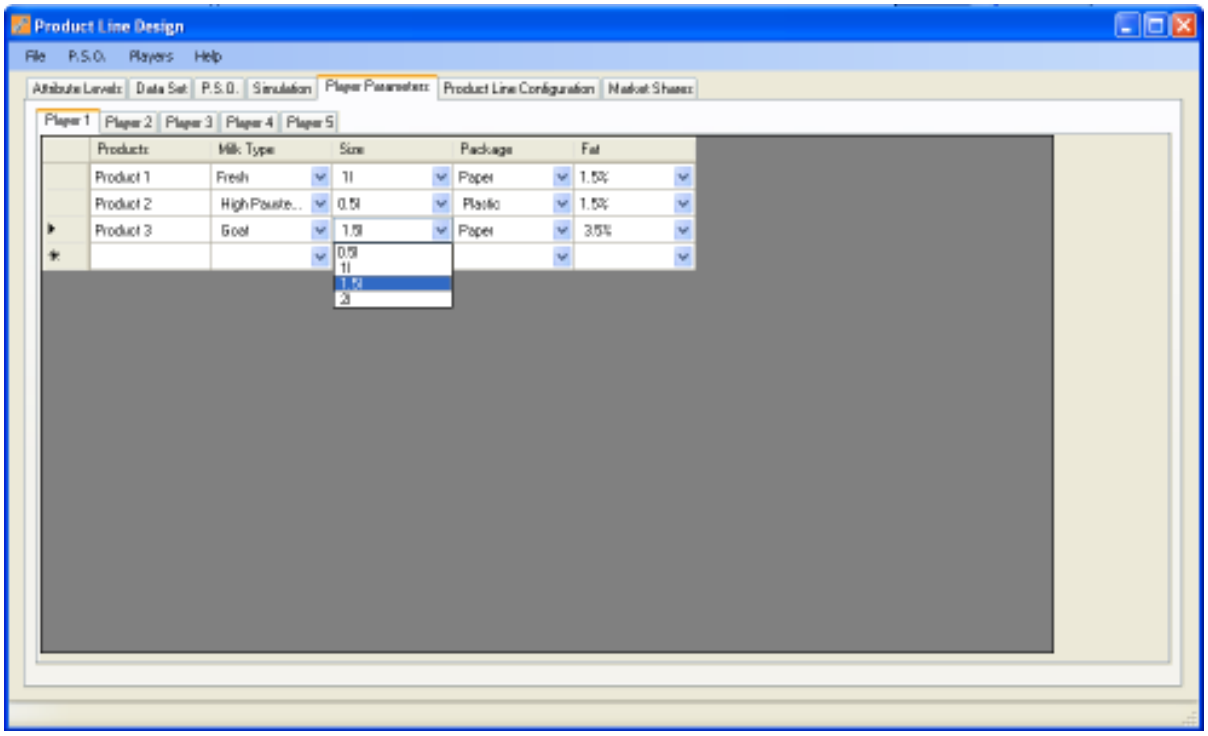
Figure 8.2: The attributes and levels that form the products



After completing the above process, the number of firms (players) that form the market must be set. This is implemented through pressing the “Enter Players’ parameters” button in the “Attribute Levels” tab. This opens a textbox where the number of players (excluding the new entrant) that will participate to the game must be entered (Figure 8.2).

The product configuration of the incumbent firms of the market (players) along with the corresponding market shares must be entered next. These data constitute the second and last input of the system. This is implemented with through the “Player Parameters” tab that opens after the number of players has been set (Figure 8.3).

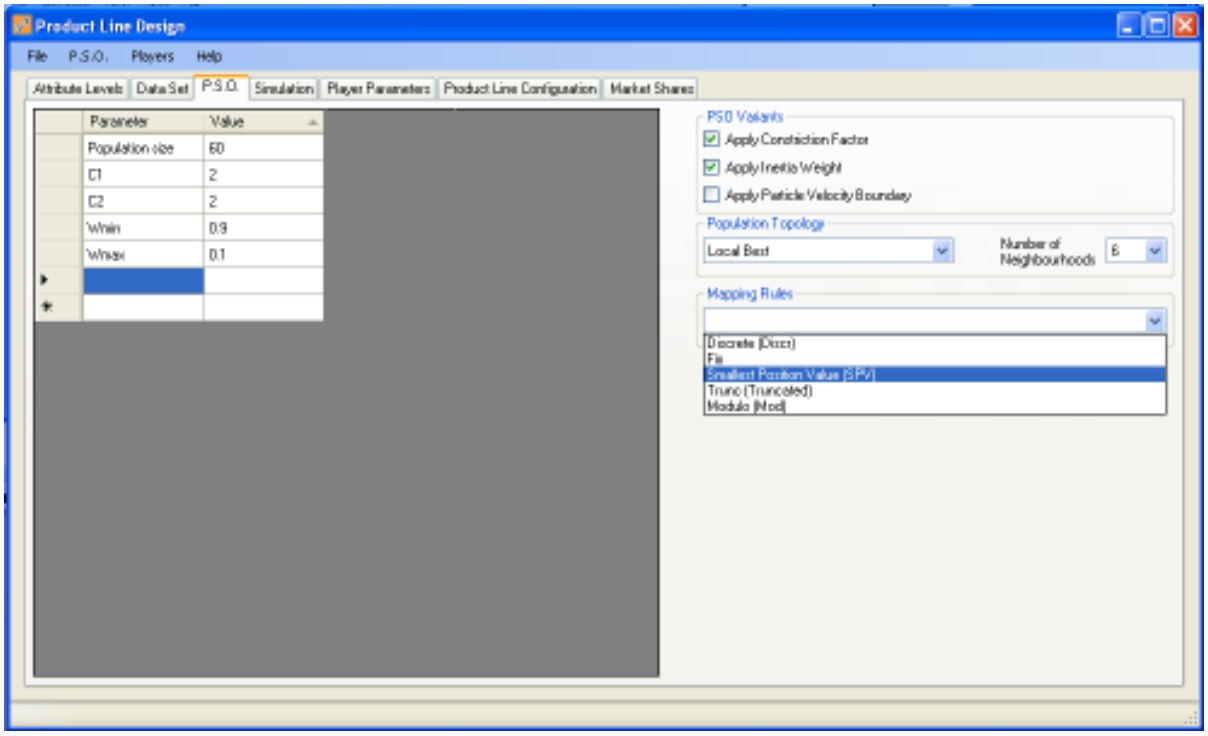
Figure 8.3: The product configurations of the incumbent firms



Next, the current market shares of the current competitors must be entered through the "Enter market shares" selection from the "Players" menu. This opens the 'Market Shares" tab, where the name of each firm along with its market share must be entered in a new line. For a new entrant only its name is entered.

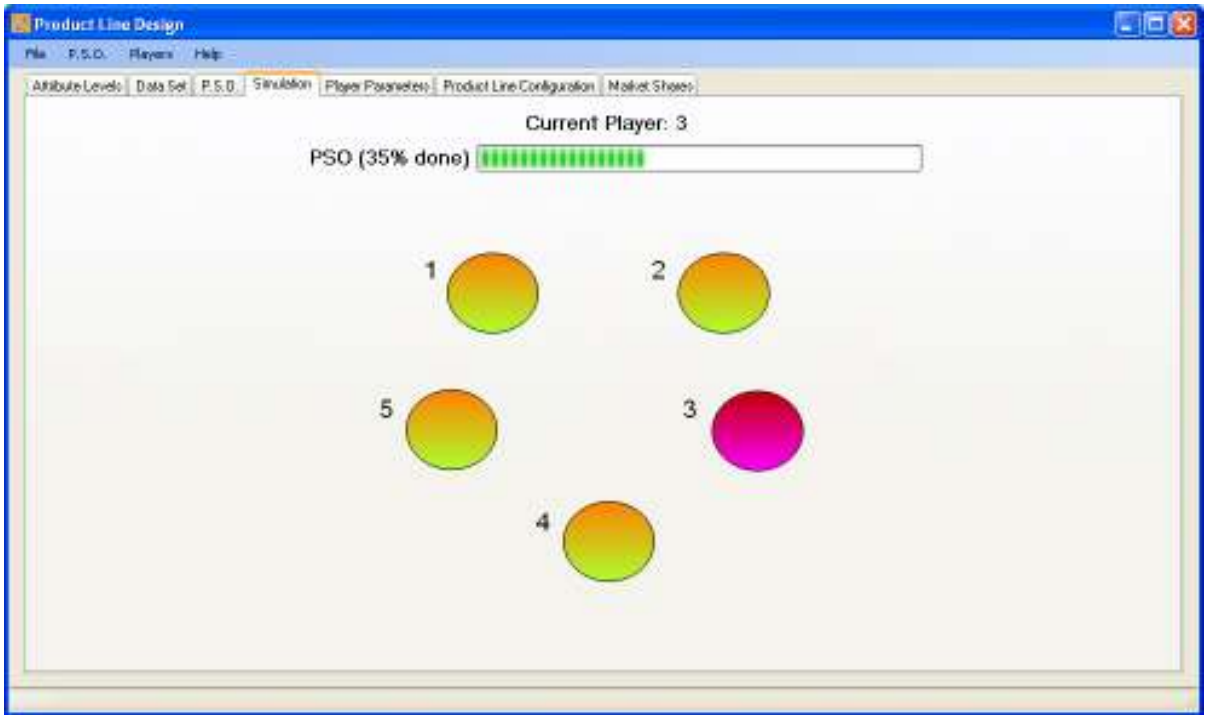
The last step is the determination of the values of the PSO parameters that will be used in the optimization process. This is implemented through the "Define parameters" selection from the "P.S.O." menu (Figure 8.4).

Figure 8.4: Determination of the PSO parameters



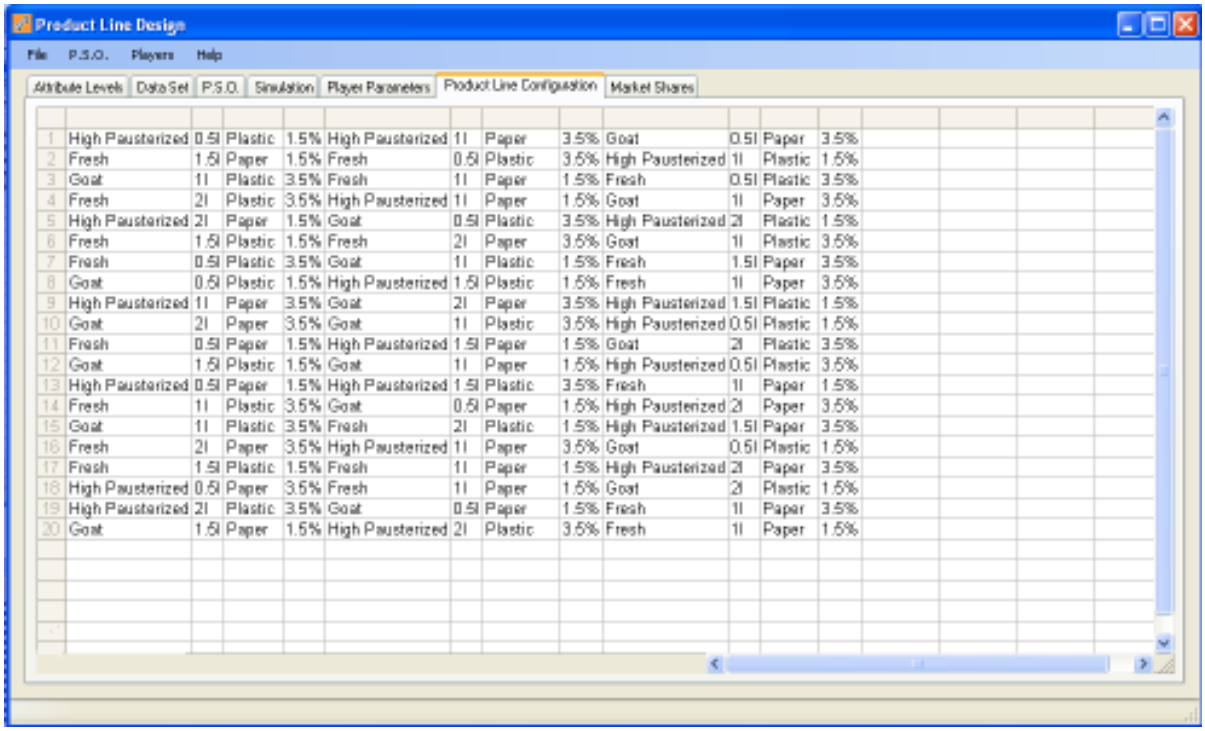
Now the simulation of the game is ready to begin through the selection of the "Run" choice from the "File" menu. Initially, the market simulation model is calibrated with the use of the customer partworths and the competitors' current market shares. When this process is completed the "Simulation" tab opens. Here each competitor (player) is represented through an orange-green disk. The player that is currently optimizing its product line is denoted by a purple color. A progress bar shows the percentage of completion of the optimization process of the active player (Figure 8.5).

Figure 8.5: Simulation of the market



When the simulation is completed the "Product Line Configuration" tab opens. Here the 20 best solutions (product line configurations) for the new entrant are illustrated (Figure 8.6).

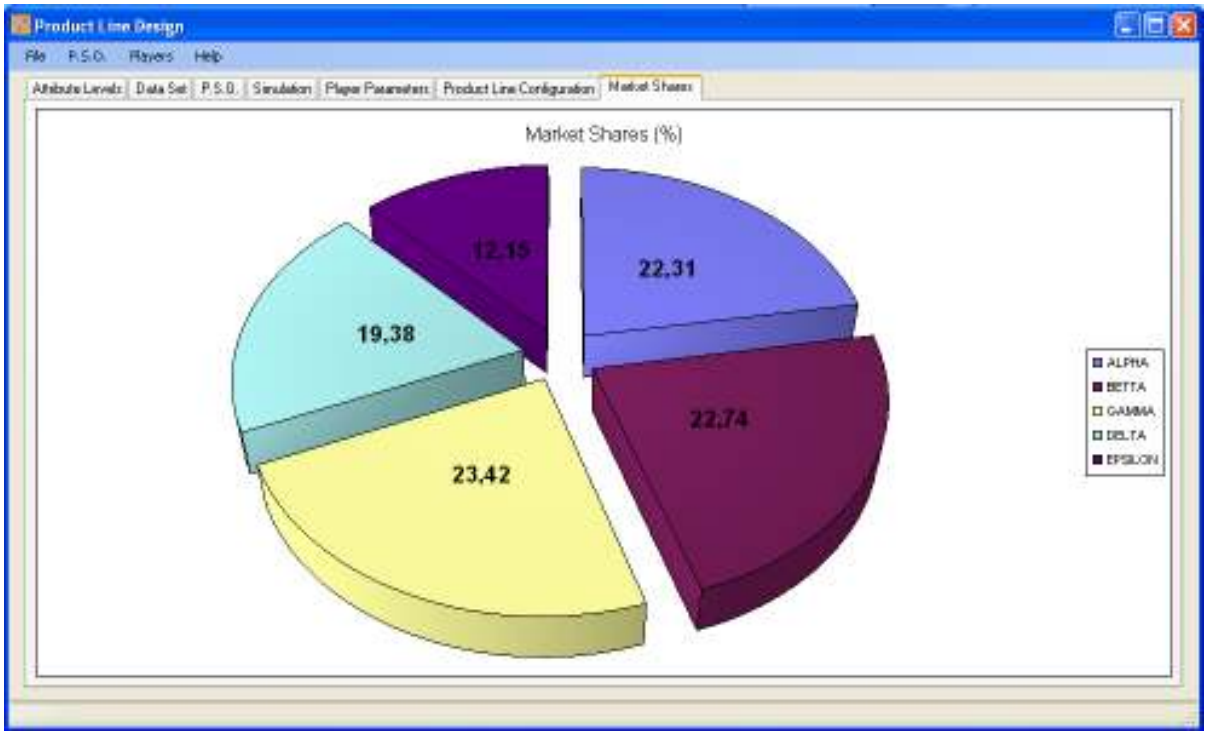
Figure 8.6: The 20 best product line configurations for the new firm.



1	High Pausterized	0.5l	Plastic	1.5%	High Pausterized	1l	Paper	3.5%	Goat	0.5l	Paper	3.5%
2	Fresh	1.5l	Paper	1.5%	Fresh	0.5l	Plastic	3.5%	High Pausterized	1l	Plastic	1.5%
3	Goat	1l	Plastic	3.5%	Fresh	1l	Paper	1.5%	Fresh	0.5l	Plastic	3.5%
4	Fresh	2l	Plastic	3.5%	High Pausterized	1l	Paper	1.5%	Goat	1l	Paper	3.5%
5	High Pausterized	2l	Paper	1.5%	Goat	0.5l	Plastic	3.5%	High Pausterized	2l	Plastic	1.5%
6	Fresh	1.5l	Plastic	1.5%	Fresh	2l	Paper	3.5%	Goat	1l	Plastic	3.5%
7	Fresh	0.5l	Plastic	3.5%	Goat	1l	Plastic	1.5%	Fresh	1.5l	Paper	3.5%
8	Goat	0.5l	Plastic	1.5%	High Pausterized	1.5l	Plastic	1.5%	Fresh	1l	Paper	3.5%
9	High Pausterized	1l	Paper	3.5%	Goat	2l	Paper	3.5%	High Pausterized	1.5l	Plastic	1.5%
10	Goat	2l	Paper	3.5%	Goat	1l	Plastic	3.5%	High Pausterized	0.5l	Plastic	1.5%
11	Fresh	0.5l	Paper	1.5%	High Pausterized	1.5l	Paper	1.5%	Goat	2l	Plastic	3.5%
12	Goat	1.5l	Plastic	1.5%	Goat	1l	Paper	1.5%	High Pausterized	0.5l	Plastic	3.5%
13	High Pausterized	0.5l	Paper	1.5%	High Pausterized	1.5l	Plastic	3.5%	Fresh	1l	Paper	1.5%
14	Fresh	1l	Plastic	3.5%	Goat	0.5l	Paper	1.5%	High Pausterized	2l	Paper	3.5%
15	Goat	1l	Plastic	3.5%	Fresh	2l	Plastic	1.5%	High Pausterized	1.5l	Paper	3.5%
16	Fresh	2l	Paper	3.5%	High Pausterized	1l	Paper	3.5%	Goat	0.5l	Plastic	1.5%
17	Fresh	1.5l	Plastic	1.5%	Fresh	1l	Paper	1.5%	High Pausterized	2l	Paper	3.5%
18	High Pausterized	0.5l	Paper	3.5%	Fresh	1l	Paper	1.5%	Goat	2l	Plastic	1.5%
19	High Pausterized	2l	Plastic	3.5%	Goat	0.5l	Paper	1.5%	Fresh	1l	Paper	3.5%
20	Goat	1.5l	Paper	1.5%	High Pausterized	2l	Plastic	3.5%	Fresh	1l	Paper	1.5%

The user can select one of the 20 solutions and open the “Market Shares” tab, to see the final market share for each competitor under the selected scenario (Figure 8.7).

Figure 8.7: The final firms' market shares under a specific solution



Now the user has all the required information for assessing the different solutions and selecting the most beneficial.

9 Concluding remarks

The Optimal Product Line Design is a NP-hard combinatorial optimization problem, where several approaches have been applied over the past thirty years. The three main critical properties of the problem are the simulation of customer choice behavior, the optimization algorithm employed, and the modeling of competitors' reactions. Most approaches have employed deterministic choice rules for simulating the customer choice process, in order to reduce the problem's complexity, which however, suffer from serious limitations. Market simulation models have never been employed in any approach or marketing system, since they significantly raise the algorithm's complexity. However, the large increase in computers' speed, as well as the advances in optimization algorithms, can now compensate the extra complexity that market simulation models add to the problem.

As far as optimization algorithms are concerned, survey results have shown that methods that work with full product profiles (Genetic Algorithms, Simulated Annealing) perform better than methods that work with partial product profiles (Dynamic Programming, Beam Search). This holds because the latter methods investigate in each iteration, only the most promising solutions and disregard the others, thus it is possible that they disregard (near) optimal solutions in a very early stage. Until now, global optimality has been guaranteed in tractable time, only for the Single Product Design problem, with the use of Lagrangian Relaxation with Branch and Bound. Among the methods that have been applied to the Optimal Product Line Design problem, Genetic Algorithms and Simulating Annealing have shown the best performance. Genetic Algorithms have an extra benefit compared to Simulating Annealing, as they work with a set of candidate solutions rather than a single one. In this way they provide the decision maker with a wide range of different product lines, which constitutes an important issue in real world marketing problems. Genetic Algorithms provide the manager with the capability to select among a set of high quality product lines the one that best satisfies the company's objectives. Genetic Algorithms constitute also the most advanced optimization method that has been incorporated into a marketing system that deals with the problem.

However the GA-based marketing systems have been implemented in such a way that provide the decision maker with only a single best solution, thus they fail to capitalize on the method's main advantage. Furthermore, the issue of dynamic competition has received very little attention in the current literature. Retaliatory actions have only been considered for the single product design problem, using basic traditional optimization algorithms. There is still no approach that incorporates competitors' responses to the optimization of a Product Line Design. In addition, dynamic competition options have not yet been embedded into any marketing system that deals with the problem.

In the present thesis, an integrated approach for dealing with the Optimal Product Line Design problem was developed. A user friendly marketing system was also presented, which incorporates the innovative methods developed for the three properties of the problem.

The first property, simulating market behavior, constitutes one of the most critical success factors of new product design and development. The effectiveness of a market simulation depends on the forecasting accuracy of the shares estimation algorithm, as well as the proper modeling of human choice behavior. The latter was clearly illustrated through the differences in the tested models' performance between the simulated and the real data set. Whereas the calibration of the choice models is an adequate procedure for the achievement of high predictive accuracy in synthetic data sets, the incorporation of a corrective method into the choice rule is necessary when dealing with real world data. Traditional market simulation approaches either fail to achieve low prediction errors, or do not correctly represent customer purchasing behavior. An integrated market simulation model that performs well on both issues was developed. The study showed that the calibration of choice models using the *range*, *kurtosis*, and *skewness* of the customer's product utilities distribution, maximizes their predicting validity. In particular, the calibration of the Pessemier model using as exponent the linear combination of the three coefficients, gives better results than the ALPHA rule, which is the current state of the art approach in commercial applications. The model displays the differential impact that an attribute may have on particular alternatives, as well as the substitution and the attraction effects among different products, through the incorporation of the corrective method into the choice rule. The value of the corrective method was illustrated on the real world scenario, where it

enabled the model to effectively deal with the similar items in the choice set, and improve its performance.

Next, a novel approach for attribute level optimization was presented, where product lines are constructed directly from customer part-worths using the Particle Swarm Optimization Algorithm. Since this is the first reported application of the PSO algorithm to the Optimal Product Line Design problem, various enhancements of the basic PSO algorithm were evaluated, and the best values for the algorithm's tuning parameters were explored through an experimental design. In a comparison of five different mappings that relate the continuous space in which particles fly with the discrete domain required by the problem, the Smallest Position Value rule gave the best results. Furthermore, it was showed that the PSO topology that mostly fits to the problem is the one with 6 neighborhoods each comprising 10 particles, and that the incorporation of both an inertia weight and a constriction factor improved the algorithm's performance. Compared to the Genetic Algorithm approach, PSO performed better as it displayed a 1.26% mean improvement in the quality of the best solution found, but required on average 6 times more computational time to converge. However, the main contribution of the PSO approach is the generation of a wide range of different near optimal solutions. 85% of the final solutions provided are unique, and the fitness of half of them is within the 5% of the best value. Given such a set of different yet high quality product lines, the firm can select the one that satisfies a number of secondary objectives which are not included in the share of choices calculation such as strategic fit, production costs, technological feasibility etc.

Finally, a dynamic approach for the Product Line Design problem was developed, where each firm optimizes its strategy using the Particle Swarm Optimization algorithm, until a Nash equilibrium is reached. The incorporation of game theoretic concepts resulted in totally different solutions than that obtained in the static case. Hence, product lines that might look attractive in the short run, may proved to be suboptimal in the longer term. The modeling of retaliatory actions from competitors through the Nash equilibrium framework provided useful insights. In a situation where no analytical closed-form solution can be provided, the player that initiates the game, and the sequence in which firms act constitute determinant factors of the final outcome as well as the speed of convergence. Furthermore, since the proposed approach treats all firms as Nash players, where none of them can

foresee the others' next moves, the first mover advantage does not seem to hold. Instead, late-movers may be in a better position, since they are able to act with perfect information, by observing the competitors' previous moves. While the Nash equilibrium constitutes a theoretical concept that employs highly simplifying assumptions, its incorporation to the product optimization framework reveals valuable information for a firm such as the product attributes that resist to the retaliatory moves from competitors, or the incumbent firms that will benefit most in the long term.

Whereas the presented methodology is the first to integrate the three critical properties of the optimal product line design problem, it does not come without limitations. The market simulation model is still at an early stage of its development. Much of the support for the approach is provided on the basis of simulations, which can only reflect the model that generated these simulated data. Hence, the model's performance needs to be further evaluated in a wider range of real world cases and situations. Furthermore, due to the discrete nature of most conjoint attributes the existence of a Nash equilibrium cannot be guaranteed. Even if one exists, it has to be computed empirically through an iterative sequential process, and still its uniqueness can hardly be proved. The existence of multiple equilibria is not a desirable property for a firm, since there is no information concerning the probability with which each of them may occur. There is still much work to be done in this direction.

Nevertheless, the developed system constitutes a useful tool for marketing managers involved in new product design decisions. A firm can evaluate the potential success of a set of new product concepts before they enter the production stage, using data obtained from a market survey. The effective modeling of customer choice behavior and competitors' retaliatory moves assists managers in minimizing the uncertainty and risks associated with new product launches. The high predictive accuracy of the system's underlying methodology supports the design of optimal products that will significantly contribute to the firm's profitability, without cannibalizing its existing product line. The presented system is the first that provides the user with a wide variety of good near-optimal solutions. The significance of this innovative property can be illustrated by a managerially important issue raised by Balakrishnan et al. (2004): The fact that the optimization of the selected objective (market share, profit) is done at the product line level, may result in large

variances in share among the products that form the line. This will probably cause dissatisfaction among the product managers, especially to those assigned a low-share product. The existence of different high quality product line configurations might mitigate this undesirable organizational conflict through the selection of the product line which, while providing a close to the optimal overall market share, it gives the minimum possible variation in the expected share among the different items of the line.

This may also constitute an interesting area for future research: The use of an algorithm that optimizes different objectives at the same time, for example market share maximization, and minimum variation in the expected share among the different items of the line. This would require the application of Multi-Objective optimization approaches, instead of the single-objective algorithms that has been employed so far to the optimal product line design problem. A multi-objective approach along with a Pareto Optimal analysis will enable the firm to concurrently optimize two or more conflicting goals (e.g. production cost minimization, and market share maximization), while setting a number of constraints (e.g. market share not less than 20%). Another promising future research area is the use of dynamic models not only for representing the competitive reactions of the firms, but also for modeling the way that customer preferences change over time. So far, customer preferences are measured once and are considered stable for the rest of the simulation. However, it is accepted that consumer preferences in the real world are changing over time. In such an approach, the market would stabilize to different equilibria over time, depending on the change in customer part-worths.

10 REFERENCES

Addelman, S. (1962). Orthogonal Main Effect Plans for Asymmetrical Factorial Experiments. *Technometrics*, 4, 21-46.

Alexouda, G. (2004). An evolutionary algorithm approach to the Share of Choices problem in the product line design. *Computers and operational research*, 31, 2215-2229.

Alexouda, G. (2005). A user-friendly marketing decision support system for the product line design problem using evolutionary algorithms. *Decision support systems*, 38, 495-509.

Alexouda, G., and Paparrizos K. (2001). A Genetic Algorithm approach to the product line design problem using the Seller's Return criterion: an exhaustive comparative computational study. *European journal of operational research*, 134(1), 165-178.

Allenby G, Fennell G, Huber J, Eagle T, Gilbride T, Horsky D, Kim J, Lenk P, Johnson R, Ofek E, Orme B, Otter T and Walker J (2005). Adjusting Choice Models to Better Predict Market Behavior. *Marketing Letters* **16**(3-4): 197-208.

Baier D and Gaul W (2001). Market simulation using a probabilistic ideal vector model for conjoint data. In: Gustafsson A and Hermann A (eds). *Conjoint Measurement, methods and applications* (2nd ed.). Springer: 97-120.

Balakrishnan, P, and Jacob, V. (1996). Genetic algorithms for product design. *Management Science*, 42(8), 1105-1117.

Balakrishnan, P., Gupta, R., and Jacob, V. (2004). Development of hybrid genetic algorithms for product line designs. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(1), 468-483.

Baltas G and Doyle P (2001). Random utility models in marketing research: a survey. *Journal of Business Research* **51**: 115-125.

Banks, A., Vincent, J. and Anyakoha, C. (2007). A review of particle swarm optimization. Part I: background and development, *Natural Computing*, 6 (4), 467-484.

Banks, A., Vincent, J. and Anyakoha, C. (2008). A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications, *Natural Computing*, 7, 109-124.

Belloni, A., Freund, R., Selove, M. and Simester, D. (2005). Optimizing Product Line Designs: Efficient Methods and Comparisons. *Working Paper*, MIT Sloan School of Management.

Belloni, A., Freund, R., Selove, M. and Simester, D. (2008). Optimizing Product Line Designs: Efficient Methods and Comparisons. *Management Science*, 54(9), 1544-1552.

Ben-Akiva M and Lerman SR (1985). *Discrete Choice Analysis*. MIT Press: Cambridge.

Bradley, R.A., and Terry, M.E. (1952). Rank analysis of incomplete block designs I: The method of paired comparisons. *Biometrika*, 39, 324–345.

Brice, R. (1997). Conjoint Analysis A Review of Conjoint Paradigms and Discussion of the Outstanding Design Issues. *Marketing and Research Today*, November, 260-66.

Camm, J.D., Cochran, J.J., Curry, D.J. and Kannan, S. (2006). Conjoint Optimization: An Exact Branch-and-Bound Algorithm for the Share-of-Choice Problem. *Management Science*, 52(3), 435-447.

Chen, K.D., and Hausman, W.H. (2000). Technical Note: Mathematical properties of the optimal product line selection problem using choice-based conjoint analysis. *Management Science*, 46(2), 327-332.

Choi, S.C. and DeSarbo, W.S. (1993). Game Theoretic Derivations of Competitive Strategies in Conjoint Analysis. *Marketing Letters*, 4(4), 337-348.

Clerc, M. and Kennedy, J. (2002). The particle swarm: explosion, stability and convergence in a multi-dimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6, 58-73.

Currim IS (1982). Predictive testing of consumer choice models not subject to independence of irrelevant alternatives. *Journal of Marketing Research* **19**(2): 208-222.

Dobson, G., and Kalish, S. (1993). Heuristics for pricing and positioning a product line using conjoint and cost data. *Management Science*, 39(2), 160-175.

Downs, B.T. and Camm, J.D. (1996). An exact algorithm for the maximal covering problem. *Naval Research Logistics*, 43, 435-461.

Eberhart, R.C., Simpson, P., and Dobbins, R. (1996). Computational intelligence PC tools. *AP Professional*, 212–226.

Elrod T (2001). Recommendations for Validations of Choice Models. *Proceedings of the 2001 Sawtooth Software Conference*. Sequim, WA: 225-243.

Engelbrecht, A.P. (2007). Computational Intelligence: An Introduction, John Wiley and Sons, England.

Goldberg, D.E. (1989). Genetic Algorithm in Search, Optimization and Machine Learning, Addison Wesley.

Green, P.E., Carroll, J.D., and Goldberg, S.M. (1981). A general approach to product design optimization via conjoint analysis. *Journal of Marketing*, 45, 17-37.

Green, P.E., and Krieger, A. (1985). Models and Heuristics for Product Line Selection. *Marketing Science*, 4, 1-19.

Green, P.E., and Krieger, A.M. (1987). A consumer-based approach to designing product line extensions. *Journal of product innovation management*, 4, 21-32.

Green, P.E., and Krieger, A.M. (1988). Choice Rules and Sensitivity Analysis in Conjoint Simulators. *Journal of the Academy of Marketing Science*, 16(3), 114-127.

Green, P.E., Krieger, A.M., and Zelnio, R.N. (1989). A componential segmentation model with optimal design features. *Decision Sciences*, 20(2), 221-238.

Green, P.E., and Krieger, A.M. (1992). An application of a product positioning model to pharmaceutical products. *Marketing Science*, 11, 117-132.

Green, P.E., and Krieger, A. (1997). Using conjoint analysis to view competitive interaction through the customer's eyes. In: Day, G.S., and Reibstein, D.J. (eds), Wharton school on dynamic strategy, Wiley and Sons, New York, 343-368.

Green, P. E. and V. R. Rao (1971), "Conjoint Measurement for Quantifying Judgmental Data," Journal of Marketing Research, 8 (August), 355-363

Hagerty, M.R. (1985). Improving the Predictive Power of Conjoint Analysis: The Use of Factor Analysis and Cluster Analysis. *Journal of Marketing Research* **25**(2): 175-186.

Holland, J.H. (1975). Adaptation in Natural and Artificial systems. Ann Arbor. MI: The University of Michigan Press.

Huber J, Orme B and Miller R (2001). Dealing with product similarity in conjoint simulation. In: Gustafsson A, Herrmann A and Huber F (eds). *Conjoint Measurement-Methods and Applications*. Springer.

Huber J and Puto C (1983). Market Boundaries and Product Choice: Illustrating Attraction and Substitution Effects. *Journal of Consumer Research* **10**.

Huber J, Wittink DR, Fiedler JA and Miller RL (1993). The Effectiveness of Alternative Preference Elicitation Procedures in Predicting Choice. *Journal of Marketing Research* **30**: 105-114.

Huber J and Zwerina K (1996). The importance of utility balance in efficient choice designs. *Journal of Marketing Research* **23**: 307-317.

Kaul, A., and Rao, V.R. (1995). Research for product positioning and design decisions: an integrative view. *International Journal of Research in Marketing,* 12(4), 293-320.

Kennedy, J., and Eberhart, R.C. (1995). Particle Swarm Optimization. *Proceedings of the IEEE International Conference on Neural Networks*. Piscataway, NJ, USA, 1942-1948.

Kennedy, J., and Eberhart, R.C. (1997). A discrete binary version of the particle swarm algorithm. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*. Orlando, FL, USA, 4104-4108.

Kennedy, J., and Eberhard, R.C. (2001). Swarm Intelligence. Morgan Kaufmann Publishers.

Kohli, R., and Krishnamusti, R. (1987). A heuristic approach to product design. *Management Science*, 33(12), 1523-1533.

Kohli, R., and Krishnamusti, R. (1989). Optimal product design using conjoint analysis: Computational complexity and algorithms. *European Journal of Operational Research*, 40(2) 186-195.

Kohli, R., and Sukumar, R. (1990). Heuristics for product line design using conjoint analysis. *Management Science*, 36(12), 1464-1478.

Kotler P and Armstrong G (2009). *Principles of Marketing* (13th ed.). New Jersey: Pearson, Prentice Hall.

Kotler P and Keller KL (2009). *Marketing Management* (13th ed.). New Jersey: Pearson, Prentice Hall.

Krieger, A.M., and Green, P.E. (2002). A decision support model for selecting product/service benefit positionings. *European journal of operational research*, 142, 187-202.

Krieger A.M., Green P.E., and Wind Y (2004). *Adventures in conjoint analysis: A practitioner's guide to trade-off modelling and applications*. Warton School: Pennsylvania.

Land, A.H., and Doig, A.G. (1960). An automatic method for solving discrete programming problems. *Econometrica*, 28, 497-520.

Laskari, E.C., Parsopoulos, K.E., and Vrahatis, M.N. (2002). Particle Swarm Optimization for Integer Programming. *Proceedings of the IEEE International Congress on Evolutionary Computation*. Honolulu, HI, 1582-1587.

Lesourne J (1977). *A theory of the Individual for Economic Analysis*. North- Holland.

Liao, C.J., Tseng, C.T., Luarn, P. (2007). A discrete version of particle swarm optimization for flowshop scheduling problems. *Computers & Operations Research*, 34, 3099- 3111.

Luce, R.D. (1959). *Individual choice behavior: a theoretical analysis*. New York: Wiley.

Manrai, A.K. (1995). Mathematical models of brand choice behaviour. *European Journal of Operational Research*, 82, 1-17.

Matsatsinis N.F. and Samaras AP (2000). Brand choice model selection based on consumers' multicriteria preferences and experts' knowledge. *Computers & Operations Research* **27**: 689-707.

Matsatsinis, N.F. and Siskos, Y. (1999). MARKEX: An Intelligent Decision Support System for Product Development Decisions. *European Journal of Operational Research*, 113(2):336-354.

McBride, R.D., and Zufryden, F. (1988). An integer programming approach to the optimal line selection problem. Marketing Science, 7, 126-140.

McFadden, D. (1974). Conditional Logit Analysis of Qualitative Choice Behaviour, in Zaremka P., editor. *Frontiers in Econometrics*, Academic Press, N.Y.

Michalek, J., Feinberg, F., Ebbes, P., Adiguzel, F. and Papalambros, P. Coordinated Positioning and Design of Product Lines for Markets with Heterogeneous Preferences. *Management Science*, unpublished.

Michalewicz, Z. (1994). Genetic Algorithms + Data Structures = Evolution Programs, Springer Verlag

Nair, S.K., Thakur, L.S., and Wen K. (1995). Near optimal solutions for product line design and selection: Beam Search heuristics. *Management Science*, 41(5), 767-785.

Orme B (2006). *Getting Started with Conjoint Analysis: Strategies for Product Design and Pricing*. Research Madison, WI: Research Publishers.

Orme B and Huber J (2000). Improving the Value of Conjoint Simulations. *Marketing Research*, Winter.

Orme B and Johnson R (2006). External effects adjustments in Conjoint Analysis. *Research Paper Series*. Sawtooth Software: Sequim, WA.

Paffrath R (1997). Practical Ways to Minimize the IIA-bias in Simulation Models. *Proceedings of the Sawtooth Software Conference*: Sequim, WA.

Papadimitriou, C.H., & Steiglitz, K. (1983). *Combinatorial Optimization - Algorithms and Complexity*. Upper Saddle River, NJ: Prentice-Hall.

Pessemier EA, Burger P and Teach R (1971). Using laboratory brand preference scales to predict consumer brand purchases. *Management Science* **17**: 371-385.

Poli, R., Kennedy, J., and Blackwell, T. (2007). Particle swarm optimization. An overview, *Swarm Intelligence*, 1, 33-57.

Radcliffe, N.J. (1991). Forma analysis and random respectful recombination. In *Proceedings of the 4th International Conference on Genetic Algorithms*.

Salman, A., Ahmad, I., and Al-Madani, S. (2002). Particle Swarm Optimization for task assignment problem. *Microprocessors and Microsystems*, 26, 363–371.

Sawtooth Software (2003). Advanced Simulation Module (ASM) for product optimization. *Sawtooth Software technical paper series.*

Shi, L., Olafsson, S., and Chen, Q. (2001). An optimization framework for product design. *Management Science*, 47(12), 1681-1692.

Shi Y and Eberhart R (1998). A modified particle swarm optimizer. *Proceedings of 1998 IEEE World Congress on Computational Intelligence*, 69-73.

Shocker, A.D., and Srinivasan, V. (1974). A consumer-based methodology for the identification of new product ideas. *Management Science*, 20, 927-937.

Siskos Y and Yannacopoulos D (1985). UTASTAR: An ordinal regression method for building additive value functions. *Investigacao Operational* **5**(1): 39-53.

Steiner, W., and Hruschka, H. (2002). A probabilistic one-step approach to the optimal product line design problem using conjoint and cost data. *Review of Marketing Science working papers*, 1(4), 1-36.

Steiner, W., and Hruschka, H. (2003). Generic Algorithms for product design: how well do they really work? *International journal of market research*, 45(2), 229-240.

Tasgetiren, M.F., Sevkli, M., Liang, Y.C., and Gencyilmaz, G. (2004). Particle Swarm Optimization Algorithm for Single Machine Total Weighted Tardiness Problem. *Proceedings of the IEEE International Congress on Evolutionary Computation.* 1412-1419.

Tsafarakis S, Grigoroudis E and Matsatsinis NF (2008). Targeting the undecided customer. *Proceedings of the 37th EMAC Conference*: Brighton, UK.

Tversky A (1972). Elimination by aspects: A theory of choice. *Psychological Review* **79**.

Tversky A and Simonson I (1993). Context-dependent preferences. *Management Science* **39**(October): 1170-1189.

Vriens M, Wedel M and Wilms T (1996). Metric Conjoint Segmentation Methods: A Monte Carlo Comparison. *Journal of Marketing Research* **33**: 73-85.

Wedel M and Steenkamp EM (1989). Fuzzy Clusterwise Regression Approach to Benefit Segmentation. *International Journal of Research in Marketing* **6**: 241-258.

Wittink DR and Cattin P (1981). Alternative Estimation Methods for Conjoint Analysis: A Monte Carlo Study. *Journal of Marketing Research* **28**: 101-106.

Wittink DR, Vriens M and Burhenne W (1994). Commercial Use of Conjoint Analysis in Europe: Results and Critical Reflections. *International Journal of Research in Marketing* **11**: 41-52.

Zufryden, F. (1977). A conjoint measurement-based approach for optimal new product design and market segmentation. In *Analytical approaches to product and marketing planning*, Shocker A.D. (ed.), Marketing Science Institute, Cambridge, MA.

Zufryden, F. (1979). ZIPMAP: a zero-one integer programming model for market segmentation and product positioning. *Journal of the Operational Research Society*, 30, 63-70.