OPERATIONS RESEARCH
Vol. 55, No. 4, July–August 2007, pp. 615–629
ISSN 0030-364X | EISSN 1526-5463 | 07 | 5504 | 0615

DOI 10.1287/opre.1070.0405

OR PRACTICE

Building Efficient Product Portfolios at John Deere and Company

Tallys H. Yunes
Department of Management Science, School of Business Administration, University of Miami, Coral Gables, Florida 33124-8237, tallys@miami.edu

Dominic Napolitano
Deere & Company, Technology Center, Moline, Illinois 61265-8098, napolitanodominic@johndeere.com

Alan Scheller-Wolf, Sridhar Tayur
Tepper School of Business, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213-3890
{awolf@andrew.cmu.edu, stayur@andrew.cmu.edu}

John Deere & Company (Deere), one of the world's leading producers of machinery, manufactures products composed of various features, within which a customer may select one of a number of possible options. On any given Deere product line, there may be tens of thousands of combinations of options (configurations) that are feasible. Maintaining such a large number of configurations inflates overhead costs; consequently, Deere wishes to reduce the number of configurations from their product lines without upsetting customers or sacrificing profits. In this paper, we provide a detailed explanation of the marketing and operational methodology used, and tools built, to evaluate the potential for streamlining two product lines at Deere. We illustrate our work with computational results from Deere, highlighting important customer behavior characteristics that impact product line diversity. For the two very different studied product lines, a potential increase in profit from 8% to 18% has been identified, possible through reducing the number of configurations by 20% to 50% from present levels, while maintaining the current high customer service levels. Based on our analysis and the insights it generated, Deere recently implemented a new product line strategy. We briefly detail this strategy, which has thus far increased profits by tens of millions of dollars.

Subject classifications: industries: machinery; marketing: retail/product line optimization; production: applications.
Area of review: OR Practice.
History: Received April 2004; revisions received September 2005, June 2006; accepted June 2006.

1. Introduction

Deere & Company (Deere) manufactures equipment for construction, commercial, and consumer applications, as well as engines and power train components. As a major player in many equipment markets, Deere maintains multiple product lines; within each line, there may be several thousand, to several million, different product variants. Variants are built by selecting for each feature available on a machine (e.g., engine type, transmission, and axle) one of a number of possible options (e.g., 200, 250, or 300 horsepower (HP) for engines). Not all options are compatible; a feasible combination of options is called a *configuration*.

Deere speculates that maintaining too many configurations reduces profits, by elevating what Deere calls *complexity cost*. This cost, over and above the inventory carrying costs of each configuration, captures factors such as reduced manufacturing efficiency, frequent line changeovers, and the general overhead of maintaining documentation and support for a configuration. This definition is similar to that given in Thonemann and Brandeau (2000, p. 1), where complexity cost is "the cost of indirect functions at a company and its suppliers that are caused by component variety; complexity cost includes, for instance, the cost of designing, testing, and documenting a component variant." In this paper, we describe the marketing and operational methodology and tools we developed to reduce Deere's complexity costs by concentrating product line configurations while maintaining high customer service, thus elevating overall profits. We illustrate our work with applications to two lines at Deere; details of the products have been disguised, but the lines differ in significant ways (e.g., costs, profits, and sales), making them a diverse test bed for our optimization algorithm.

A primary component in our algorithm is our *customer migration model*, quantifying the behavior of Deere's customers: A customer may want a specific configuration, but if his or her first choice is unavailable, he or she may *migrate* to an alternative configuration that does not differ too greatly from the first choice. Using actual sales, along with customer segmentations and part-worths utilities provided by Deere, we probabilistically model every

customer as individually identifying a set of acceptable configurations, sorted in decreasing order of preference: their *migration list*. When the top configuration on his/her list is not available, a customer will buy the next available configuration. When no configuration is available, the customer defects to a competitor.

Using each customer's migration list, as well as costs and profits for all feasible configurations (found via constraint programming; see Marriott and Stuckey 1998), we build a mixed-integer program (MIP) to maximize Deere's profits within a product line. Application of our algorithm provides: (i) a general method to determine which configurations are least profitable and thus may be candidates for elimination; (ii) recommendations for how Deere can significantly focus specific product lines; and (iii) identification of the high-level drivers of product line efficiency. Based on our results, Deere has instituted an incentive program to steer customers toward a core subset of their product lines, which has resulted in increased profit in line with our predictions—tens of millions of dollars annually (see §7 for details).

In §2, we begin with a review of the literature. In §3, we give a high-level overview of our optimization algorithm, including a description of how we generate and cost out all feasible configurations for a product line. In §4, we detail how we generate customer utilities and migration lists. The development and solution of our MIP is related in §5. We present results of the experiments on the Deere product lines in §6, describe Deere's actual implementation in §7, and conclude by summarizing our work and findings in §8.

2. Literature Review

Reducing product line complexity is a common goal among companies. For example, Raleigh (2003) describes how Unilever uses its product logic framework to simplify its global home and personal care product portfolio. Similarly, Stalk and Webber (1993), Henkoff (1995), and Schiller (1996) describe pruning of product lines in several other industries. From a marketing perspective, there are often fears that such a reduction may detract from brand image or market share (Kahn 1995, Chong et al. 1998, Kekre and Srinivasan 1990). Randall et al. (1998) discuss the dangers of extending a product line. These latter concerns are in keeping with findings that reducing the breadth of lines and focusing resources on popular products ("favorites") may actually increase sales (see Quelch and Kenny 1994, Broniarczyk et al. 1998, and Fisher et al. 1995). Our model is consistent with both of these streams of thought: If a customer finds a product that meets their needs (i.e., a "favorite"), he or she will make a purchase; if such a product is no longer part of the product line, they will not (and market share will go down).

The motivation behind reducing product lines is often cost containment; there is a long history of tying product line complexity to increased costs. This has been done analytically by Hayes and Wheelright (1984), Abegglen and Stalk (1985), and Kekre (1987), and empirically by Foster and Gupta (1990), Anderson (1995), and Fisher and Ittner (1999). Deere has spent considerable time prior to this project establishing the linkage between variety and cost, ultimately resulting in a proprietary function linking the two, called the *complexity cost*.

Due to the ubiquitous and cross-functional nature of product line optimization, interdisciplinary research has long been advocated; Yano and Dobson (1998b) and Ramdas (2003) provide an extensive literature survey, which we briefly summarize. McBride and Zufryden (1988), Kohli and Sukumar (1990), and Nair et al. (1995) use math programs to solve the product portfolio problem, but do not include costs related to the breadth of the line, as this is fixed. Kohli and Sukumar (1990) apply their method to data from a real (but small) problem for telephone hand sets. Green and Krieger (1985) and Dobson and Kalish (1988, 1993) present math-programming formulations and solution heuristics to the product selection and pricing problem, which do include product line breadth costs. More generally, De Groote (1994), Raman and Chhajed (1995), and Yano and Dobson (1998a) use iterative solution procedures to determine attributes, prices, and production processes, each having a specific fixed cost. Morgan et al. (2001) solve a similar problem using a mathematical program. None of these algorithms are suitable for problems anywhere near the size of Deere's because their models are not likely to scale well. Likewise, none of these works include applications to any actual industrial data. Recently, there have been alternative formulations: Chen et al. (1998) and Chen and Hausman (2000) tested on very small, nonindustrial data.

Thus, while the problem of determining optimal product lines has a long and important place within the marketing and operations literatures, actual industrial scale applications of analytical techniques to optimize product lines based on detailed company data have never been performed. We provide these for the first time.

3. Solution Overview

Our optimization procedure is comprised of three main steps: (1) build and cost out feasible configurations, (2) create migration lists, and (3) optimize using a MIP. We outline these steps here and discuss them in greater detail in the subsequent sections.

Before selecting which configurations to include in a product line, we need to know which configurations can feasibly be built, along with their corresponding unit costs and profits. Generating this data is no trivial task, as Deere's machines can have as many as 23 features (engines, transmissions, attachments) and multiple options within each feature (tires may have 25 options). In principle, this could create millions of possible configurations. In practice, however, many combinations of options are not physically possible; for example, certain transmissions cannot be used with certain axles. Consequently, every product line has a unique set of configuration rules that determine how the

line's options can be combined. These rules typically take the form "if A is true then B must be true," where A and B may be complicated sequences of conjunctions, negations, and disjunctions. The two product lines we considered at Deere, referred to as the Gold Line and Silver Line, have 39 and 68 combination rules, respectively. Each line also has pricing rules (35 for the Gold Line and 212 for the Silver Line), which determine the total cost of a machine given a choice of options. Once the price and total cost for a specific configuration are known, its profit contribution can be computed.

We use Eclipse, a constraint-programming (CP) language (Tsang 1993, Mariott and Stuckey 1998) to build, cost out, and find the profit for all of the valid configurations, leveraging CP's inherent expressive power and intelligent search mechanisms. After all necessary constraints are added to the CP model, the solver outputs all feasible configurations as well as their prices. We provide examples of configuration and pricing rules in Appendix A.

Our next step is to model customer behavior; we require a construct to represent how customers evaluate different configuration choices that will also lend itself well to our MIP formulation. To accomplish this task, we devised the concept of a *customer migration list*, which can be thought of as a personal ranking of configurations each customer carries around in his or her mind. In §4, we discuss the construction of customer migration lists based on marketing data provided by Deere. Once we have generated the all of the possible configurations, or the solution space of our problem, and calculated our customer migration lists, or the customer data, we combine these in a MIP with the objective of maximizing profit. We describe the formulation of this program, including how it uses the migration list data, in §5.

4. The Customer Migration Model

When Deere approached us, they did not have a specific customer behavior model in mind, but they did know which product features were important and what types of trade-offs customers were likely to make. We were tasked with giving structure to this knowledge; we were to provide a tool to harness Deere's understanding, incorporate customer purchase data (over 15,000 data points), and be well suited for use by our optimization engine. We were not tasked with evaluating customer behavior, for example, by estimating utility values; rather, we were to provide a framework for Deere to use, now and in the future, based on their understanding of customer behavior.

To accomplish this task, we first sought a high-level view of how a typical Deere customer behaves. We posited the following model: If provided with the entire selection of Deere products, the customer would rank them in a mental list in order of preference, based on the different options each configuration possessed. If money was no object, the customer would purchase the configuration at the top of their list, if available. If it was too expensive or not available, the customer would move down their list until they found an available and affordable machine. At some point, if too many of their top choices were either unavailable or exceeded their budget, a customer would give up and leave the Deere dealer without making a purchase.

This idea formed the basis for our construct capturing customer behavior: the *customer migration list*. To operationalize this model, guided by Deere's marketing research, we generated a set of randomized migration lists for each product line—one list for each purchase from Deere's sales history. The lists are generated prior to solving our MIP to increase solution speed and allow study of the lists in their own right. This also facilitates experimental replication and sensitivity analysis (see §6).

What remains is to explain how we model customers' formation of migration lists, i.e., which configurations they will consider (their *potential configurations*), how they evaluate these potential configurations and decide what is "too much to pay," and at what point give up and leave. In §4.1, we discuss how (and why) we probabilistically place customers into different segments, how we then specify key parameter values (contingent on segment), and determine the customer's potential configurations—those configurations that could possibly satisfy what the customer wants. These are then ranked, according to part-worths utilities, as described in §4.2. The final migration list is formed according to these rankings, with probabilistic cut-offs based on price and utility. This is described, along with a complete summary of our generation algorithm, in §4.3. We discuss extensions of our migration list algorithm in §4.4.

Throughout, when discussing the generation of the migration list of an individual customer, C_0 will denote the configuration this customer purchased, with price P_0, and summed part-worths utilities equal to U_0.

4.1. Segmenting Customers and Determining Parameter Values

Our first step was to identify customer segments that accurately capture market behavior. Different segments correspond, for example, to private users and commercial users in different industries. While this is not mandatory for our algorithm, Deere felt that there were a number of distinct, but overlapping customer segments for their products—three for the Gold Line and four for the Silver Line. Because different segments make purchases from the same, single-product line, they must be considered simultaneously. Below we define the parameters that place customers into segments and the parameters that determine potential configurations for customers within each segment.

The most direct way to place a customer into a segment is to match the options on his or her purchased configuration to a segment likely to purchase it. If options on a configuration are attractive to different segments, then the configuration will have to be assigned in some probabilistic manner. Moreover, for those configurations that are not

Table 1. Segments for the customer segmentation example.

Segment	cond	q	s
1	Engine = 450 HP	0.9	0.2
2	Axle = 2WD and transmission = SST	0.75	0.55
3	Stereo = six-disc CD	0.2	0.25

(probabilistically) assigned, there will have to be a procedure to place these into segments as well. These ideas are formalized in the two segmentation parameters below, which are defined for each segment. We then present an illustrative example.

(cond, q): Based on whether certain options denoted by the logical condition cond (e.g., "engine is 400 HP and axle is 4WD") are present in C_0, the customer is placed in the segment with probability q (the proportion of purchases with these options from that segment). Multiple segments for which C_0 satisfies cond are considered sequentially, until the customer is assigned. Should a customer fail to be placed, he or she is assigned according to s, below:

s: This is the percentage of unit sales that come from the segment. All customers not assigned according to the rule above are randomly assigned based on these probabilities. The sum of s values over all segments is thus equal to one.

Customer Segmentation Example. Assume that there are three customer segments with the parameters in Table 1.

Assume that configuration C has a 450 HP engine, 4WD transmission, and a six-disc CD stereo. It will be placed in segment 1 with 90% probability. If this does not happen, segment 2 will be bypassed, as C is not 2WD. Because C has a six-disc stereo, it will then be tried in segment 3, and assigned there with a 20% probability. If C is not being assigned after these attempts, it will be assigned to one of the three segments with probabilities equal to 20%, 55%, and 25%, respectively.

After placing the customer into a segment, we assign values to five key parameters—proxies for the critical elements of customer behavior.

c: This is the *commonality factor*, the minimum number of options a configuration must have in common with C_0 for the customer to be willing to purchase it. If $c = 0$, all machines may be considered; as c grows, fewer deviations from C_0 are allowed, until (when c is maximal) no configuration except C_0 will be considered.

$\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$: This is a list (possibly empty) of fixed features. Any feature that appears in this set must not have its option changed from that on C_0.

r: This models customers' reservation prices. A customer who purchased a machine with price P_0 has his or her reservation price drawn uniformly from the interval $[P_0, (1+r)P_0]$. Customers are willing to buy machines costing at most their reservation price.

u: This models customers' reservation utilities. Analogous to r, a customer who purchased a machine with summed part-worths utilities equal to U_0 has his or her reservation utility drawn uniformly from the interval $[(1-u)U_0, U_0]$. Customers are willing to buy machines with summed part-worths utilities equal to no less than their reservation utility.

β: This is the first-choice probability. With probability β, C_0 will be placed first on the customer's migration list, independent of the utilities generated by the model. With probability $1-\beta$, the position of C_0 on the list will be determined by its utility. If $\beta = 0$, we have a pure choice model, and if $\beta = 1$, customers always prefer their initial purchase.

Because these behaviors vary across segments, they are defined for each segment. Moreover, to capture heterogeneity within segments, many of the parameters are either probabilities themselves or are selected probabilistically. Taken together, these five parameters determine a customer's potential configurations; they describe how closely the elements of the migration list will hew to the purchased configuration, C_0, that spawned it. The first two, c and $\mathcal{F}$, determine the size of the search space a customer is willing to consider in the neighborhood of C_0. The next two, u and r, determine how flexible the customer is willing to be within this search space, with respect to price and summed part-worths utilities of configuration. The final parameter, β, controls whether or not C_0 will be forced to the top of the migration list. Values for each of these parameters, by segment, were determined by Deere.

After a customer is placed into a segment and the above parameters are established, we use constraint programming to determine the customer's set of potential configurations—those configurations having at least c features in common with C_0, including all of those on list $\mathcal{F}$, and price no more than the customer's reservation price. The next step is to determine how the customer will evaluate these configurations, i.e., how he or she will rank them on his or her migration list.

4.2. Using Option Utilities to Rank Configurations

To model customers' ranking of configurations, we utilize part-worths utilities; in a standard application of part-worth utilities, the (static) value of all of the options on a configuration would be summed to give the total value of the configuration. We randomize this procedure to allow for greater heterogeneity between customers within the same segment. This randomization is crucial to our algorithm; without it, customers in the same segment who purchased the same configuration would have identical, or nearly identical, migration lists. They could differ only in where they end, due to reservation price, and whether C_0 was first, if $\beta \neq \{0, 1\}$.

Within each feature (for example, engine), we initially assign values determined by Deere, by segment, to all available options (for example, 200 HP, 400 HP). This also defines the feature's mean *relative importance*—the difference between the largest and smallest utility value for any of its options—which captures the feature's baseline importance to customers in the segment. For each customer, we

then randomly perturb these mean relative importance values using deviation parameters, again estimated by Deere. This captures the fact that individual customers from the same segment may vary considerably in how they value different features. We could have defined subsegments to capture this phenomenon, but given the data available (heterogeneity is known to be present but is hard to quantify), we preferred the current method.

To perturb relative importance values, our algorithm randomly selects a fixed number of features and rescales their relative importances using a factor uniformly drawn from the feature's mean relative importance plus or minus its deviation parameter. We then rescale the relative importances of those features not selected to keep the sum of all the relative importance parameters constant. Next, the option utilities within each feature are rescaled by the ratio of the new relative importance value to the old. Only then do we sum the individual part-worths utilities, yielding the total value for each configuration. This operation is done uniquely for all customers within the segment. We formalize this procedure below and then provide an example.

First, we randomly select n features, where n is a fixed parameter for all customers, in all segments. Next, we randomly generate a relative importance for each of the n features using a uniform distribution over the feature's mean relative importance, plus or minus its deviation. Then, we determine the relative importance of the remaining features by scaling them such that the total sum of relative importances has the same value as it had before the perturbation step. Finally, we rescale all option utilities based on the new relative importances of their features.

Utility Calculation Example. We consider a simplified machine with three features—engine, transmission, and axle. For a given customer segment, Table 2 first presents coded options available in each feature, with their corresponding initial utilities. The relative importance of the three features (mean, deviation) are $(21, 5)$, $(20, 3)$, and $(15, 4)$, respectively.

The sum of the mean relative importances is 56. Assume that $n = 1$ and transmission was selected to have its relative importance changed to 22 (Steps 1 and 2). Thus, the new relative importances of engine and axle have to sum to 34 $(=56 - 22)$; the relative importance of engine becomes 19.83 $(=21/(21 + 15) \times 34)$, and the relative importance of axle becomes 14.17 $(=15/(21 + 15) \times 34)$ (Step 3). We then rescale the option utilities by new RI of feature/old RI of feature (Step 4). The lower part of Table 2 shows the results.

Table 2. Initial and perturbed utilities for the utility calculation example.

Engine		Transmission		Axle	
Option	Utility	Option	Utility	Option	Utility
Initial utilities					
200 HP	0	MPT	0	AWD	0
250 HP	7	HWT	10	NWD	15
300 HP	14	LBT	20		
350 HP	21				
Perturbed utilities					
200 HP	0	MRT	0	AWD	0
250 HP	6.61	HWT	11	NWD	14.17
300 HP	13.22	LBT	22		
350 HP	19.83				

4.3. Formation of the Final Migration List

After we have obtained all of a customer's potential configurations (as described at the end of §4.1) and determined their part-worths utilities (as outlined in §4.2), we form the customer's final migration list. We list all of the potential configurations in decreasing order of their summed part-worths utilities, stopping when we reach a configuration with summed utility less than the customer's reservation utility. More formally, for every customer in the sales history, we generate a migration list as follows (recall that C_0 is the configuration bought by the customer):

1. Assign the customer to a segment as described in §4.1; this determines c, $\mathcal{F}$, and β;
2. Determine reservation price and reservation utility for the customer, as defined in §4.1;
3. Determine the customer's potential configuration list—all configurations that have at least c features in common with C_0, including all those on list $\mathcal{F}$, with price no more than the reservation price. Call this list L;
4. Determine the option utility values for the customer as in §4.2;
5. Sort L in decreasing order of summed part-worths utilities, truncating the list when a configuration has utility value less than the reservation utility;
6. Use the parameter β to determine whether to move C_0 to the top of L;
7. Optionally, truncate L, ensuring that C_0 remains in L after truncation (in accordance with wishes of Deere);
8. Output the resulting list L.

4.4. Summary and Extensions

Our algorithm generates a set of lists; each list models an individual customer's choices, while in aggregate the lists capture behavior over segments. Customer heterogeneity is captured in two ways: based on their initial purchases, different customers consider different potential configurations. Within these potential configurations, due to our randomization of features' relative importances, customers may value configurations differently.

We can use lists generated from different parameter settings to explore the sensitivity of product lines to different beliefs about customer behavior (as in §6.5). If lists are generated via a different method, such as expert surveys, these could be used in our MIP, enabling us to compare the solutions to tune customer behavior parameters. In addition, we can optimize different random replications of the same

customer behavior parameters to estimate the variation of the optimal product portfolio for the same parameter set (as in §6.3).

Our list generation algorithm could be extended to include machines from Deere's competitors, possibly with a utility for brand equity, modeling a richer competitive environment. Likewise, we could allow customers to make multiple purchases, include "lost" customers who were unable to make purchases due to factors at the dealers, and/or customers who did not find their first choice, but did buy a different machine. In the first case, we could generate multiple identical (or correlated) lists for selected customers, based on historical purchase data. To deal with "lost" customers, we could generate an additional set of customers and their initial choices, possibly using the s parameters. Finally, we could assign an explicit probability that any customer's purchase was not actually his or her first choice (although the β parameter captures this to some extent). None of these actions was used for the project reported.

One extension not readily made is to include negotiations with customers, as all prices are taken as exogenous. Endogenous pricing raises significant and complex modeling issues outside of the scope of this project, but if data on actual sales prices were made available, profits could be rescaled if discounts were significant. In general, if Deere were to provide us with more detailed data, our algorithm should improve. Additional information which would be most helpful would focus on customer migration behavior, for example: How long should customer migration lists be? Should these lengths change according to segment? How elastic are reservation prices and utilities (and utilities in general)? How do incentives affect migration behavior and customer impressions? Through their implementation (see §7), Deere is gathering some of this data now.

5. The Optimization Model

In this section, we describe the MIP model that selects the configurations to build in order to maximize total profit. Although it is in theory potentially of exponential complexity (it enumerates the search space), our MIP can efficiently solve Deere's base problem (5,000–15,000 customers and 3,500–24,000 configurations) using commercially available solvers, and also allows the incorporation of additional managerial constraints. Examples of these include limits on the number of configurations built, a minimum service level, and specifying configurations that must be included (or dropped). The performance of our algorithm with some of these constraints is shown in §6.6.

The input for the optimization problem consists of the following data:

- T: set of all relevant configurations, $T = \{1, 2, \ldots, t\}$.
- C: set of all customers.
- L_i: ordered set of configurations in the migration list of customer i $\forall i \in C$.
- N_j: set of customers whose migration lists contain j: $N_j = \{i \in C \mid j \in L_i\}$.
- p_j: profit (price − cost) of configuration $j \in T$.

Our model uses two sets of binary decision variables:

- $y_j = 1$ if configuration j is produced, 0 otherwise ($j \in T$).
- $x_{ij} = 1$ if customer i buys configuration j, 0 otherwise ($i \in C,\ j \in L_i$). Determining the y_j values likewise determines the x_{ij} values, given a set of migration lists.

Our model uses the following constraints:

$$x_{ij} \leqslant y_j \quad \forall j \in T,\ i \in N_j, \tag{1}$$

$$\sum_{k \text{ after } j \text{ in } L_i} x_{ik} + y_j \leqslant 1 \quad \forall j \in T,\ i \in N_j, \tag{2}$$

$$y_j \in \{0, 1\} \quad \forall j \in T, \tag{3}$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in C,\ j \in L_i. \tag{4}$$

Constraints (1) prescribe that only available configurations can be purchased, and Constraints (2) that configurations that appear earlier in a migration list of a customer have higher priority. Note that Constraints (2) also ensure that every customer buys at most one configuration.

Deere wishes to maximize the following objective, which is similar to many in the literature (for example, it is a more detailed version of that in Dobson and Kalish 1988, 1993):

max shareholder value added (SVA)

$$= \sum_{j \in T} \left((p_j - K) \sum_{i \in N_j} x_{ij} \right) - a - \textit{overhead}. \tag{5}$$

Overhead is the sum of fixed costs, and the constant K captures the inventory and capital equipment costs incurred for each unit sold. Reducing the number of configurations in the product line will *not* reduce K; only selling fewer units will. Thus, K differs from complexity cost, a, which *does* depend on the number of configurations in the product line via (6)–(10) below.

The term a in (5) is our piecewise-linear approximation of Deere's complexity cost function. Use of this cost form dates from before the initiation of our project; its determination and/or validation was outside of the scope of our work. We discuss the use of an alternative, and potentially more accurate, complexity cost function in Equations (11)–(13).

To define a, we need an extra set of variables and constraints. Let B be the set of grid points we choose to approximate the complexity function. For every $k \in B$, let n_k and o_k be, respectively, the number of configurations and Deere's complexity cost corresponding to the kth grid point. Finally, let λ_k be real-valued variables that determine a convex combination of two consecutive grid points. Then, using SOS2, a device available in most commercial MIP solvers that ensures at most two of its arguments assume positive values (and if two arguments are positive they are consecutive), we have

$$\sum_{j \in T} y_j = \sum_{k \in B} n_k \lambda_k, \tag{6}$$

$$a = \sum_{k \in B} o_k \lambda_k, \tag{7}$$

$$\sum_{k\in B} \lambda_k = 1, \tag{8}$$

$$\mathrm{SOS2}(\lambda_1, \ldots, \lambda_{|B|}), \tag{9}$$

$$\lambda_k \in [0, 1] \quad \forall k \in B. \tag{10}$$

An alternative formulation of a complexity cost could use a step function over B. Let z_i ($i \in \{1, \ldots, |B|\}$) be a binary variable indicating whether the number of configurations is at least n_i. Setting $o_{\text{zero}} = 0$, we modify the previous objective function by substituting $\sum_{i=1}^{|B|-1}(o_i - o_{i-1})z_i$ for a in (5). To complete this new formulation, we drop (6)–(10) and include (11)–(13):

$$\sum_{j\in T} y_j \leqslant \sum_{i=1}^{|B|-1} (n_{i+1} - n_i)z_i + n_1, \tag{11}$$

$$z_{i+1} \leqslant z_i \quad \forall i \in \{1, \ldots, |B| - 2\}, \tag{12}$$

$$z_i \in \{0, 1\} \quad \forall i \in \{1, \ldots, |B| - 1\}. \tag{13}$$

We illustrate the performance of our algorithm under this objective function in §6.6.

We conclude this section by noting that while our initial formulation presented satisfactory computational performance for Deere's problems, solving larger problems may require adding additional constraints (cuts) to the MIP given by (1)–(4). A discussion of some potential constraints is given in Yunes (2006).

6. Computational Results

In this section, we report computational results from two product lines at Deere, henceforth referred to as the Gold Line and the Silver Line. The fundamental characteristics of these lines can be gleaned from Tables 3–5: the Silver Line is a higher-cost, higher-margin product line with fewer sales and a larger number of features and feasible configurations. Not surprisingly, the Silver Line's customer base is also more heterogeneous than the Gold Line's—the Silver Line has more segments, and the differences between how segments value features is greater. In addition, *within* segments many Silver Line features are of approximately the same importance to customers. This leads to larger and more variable migration lists, seen in Table 4, which significantly affects optimization.

We first report in §6.1 instance sizes and representative generation times for the feasible configurations and migration lists. Sections 6.2 to 6.6 detail solutions of particular instances of the problems introduced in §6.1. Section 6.7 summarizes our findings. All execution times are expressed as CPU time of a Pentium 4, 2.3 GHz, with 2 GB of RAM. Execution parameters have been disguised to maintain proprietary information.

Table 3. Features, feasible configurations, and generation times.

	Features	Feasible configurations	Typical generation time (seconds)
Gold	9	3,696	2
Silver	23	24,144	60

Table 4. Customers, typical migration lists, and generation times.

	Customers	Typical mean/std. dev. of list size	Typical generation time (hours)
Gold	15,844	8.18/5.81	0.8
Silver	5,278	606.18/191.86	28.5

6.1. Generating Feasible Configurations and Migration Lists

The first step in the optimization is to determine the number of configurations in a product line, as described in §3. As seen in Table 3, in all cases configuration generation was quite swift.

The next step is to generate the migration lists, as outlined in §4. The commonality factor, c, plays a significant role in this process by limiting potential configurations: a smaller c allows a customer to vary more features, or be more flexible with respect to what he or she will buy, reducing the weight the algorithm places on C_0. We allow customers on the Gold Line and the Silver Line to vary two and three features, respectively, from their C_0 (we experimented with this; see §6.5). In addition to making customers closer to their purchases, thus preserving heterogeneity, these restrictions had the practical effect of keeping list-generation times manageable, as seen in Table 4. Nevertheless, list generation is time consuming, but because the lists may be generated off-line and stored, this does not adversely impact the performance of the algorithm. Moreover, if new customers needed to be added to an extant set of lists, their list could simply be appended to the older set, as long as the problem parameters were consistent.

6.2. Single-Instance Optimization

Initially, for both the Gold Line and the Silver Line, customer migration lists were truncated to length at most four (we experimented with this later; see §6.5). This limitation made customers more selective and reduced solution times (typically to a few hours). The majority of this time was spent making incremental improvements or verifying optimality. In all instances, the algorithm found an optimal or near-optimal solution (true gap within 0.5%) in under an hour.

As shown in Table 5, reducing product lines can lead to a significant increase in optimal expected profit, listed in millions of dollars, as compared to the initial profit, of Deere's original product portfolio. Table 5 also reports the number of configurations sold by Deere in the initial portfolio (see the column "Initial configs."). This does *not*

Table 5. Optimal solutions for the Gold Line and the Silver Line.

	Initial profit	Optimal profit	Profit ratio	Service level	Initial configs.	Optimal configs.	Time (hours)
Gold	40.180	47.627	1.1853	96.30	698	282	1.62
Silver	292.720	316.600	1.0816	99.96	816	539	2.18

include those configurations produced, but not sold. Our solutions reduce the number of configurations even below these levels, but our service level, defined as (1 − number of customers lost/total number of customers), remains high: 96% on the Gold Line and virtually 100% on the Silver Line. Moreover, if the migration lists of customers who defect from the system are extended to their untruncated length without reoptimization, the actual service levels (and profits) would increase. Note that the solutions in Table 5 are to single instances, and as such show the potential for improvement. Whether or not a single portfolio can be constructed to perform this well across different instances is a different question. This will be explored in §6.4.

An important measure of customer satisfaction is the percentage of customers who found their "favorite" product, i.e., who purchased machines at the top of their migration list. Table 6 shows that in both instances, at least 76% of the customers had either their first or second choice as part of the product line, and well over 90% had one of their top three. Customers are more focused around their top choices for the Gold Line than for the Silver Line; this is a manifestation of the greater customer heterogeneity in the Silver Line. We will see it repeated in later experiments.

6.3. Replicate Experiments

Recognizing that the solution to a single instance would typically not be sufficient to construct an effective product portfolio, we solve replicate instances with identical parameter settings to identify common characteristics of optimal portfolios. Each instance uses the same parameter settings, but due to our randomization results in a different set of migration lists, or a different set of "customers." As with all sampling-based optimization, generating additional replications would provide additional data on effective portfolios; nevertheless, after generating 10 instances for the Gold Line and the Silver Line, some fundamental properties become apparent. Using these properties, in §6.4 we will explore different ways of building good product portfolios.

Table 6. Percentage of sales per position in the migration list.

	Customers buying (%)			
	1st	2nd	3rd	4th
Gold	43.64	40.32	9.54	2.78
Silver	31.14	44.69	15.57	8.56

Table 7 reports the solution of 10 instances for the Gold Line. These 10 instances' optimal configurations vary by less than 10%, between 274 and 295, and the ratios of optimal profit to Deere's solution are even closer, differing by less than 0.01. Likewise, running times were in general similar, with one outlier. (This was instance 2, which also has the lowest profit and largest number of configurations, likely due to lists that were unusually disjoint.) Overall, this table paints a relatively consistent high-level picture of the optimal product line, particularly with respect to breadth and profit. Looking at the 10 solutions in detail though, yields a slightly different picture.

In Figure 1, we display a histogram of the number of appearances of individual configurations in the 10 optimal solutions, where for reasons of scale we show only the 612 configurations that appear at least once among these 10 instances of the problem. From the figure, we can identify the "core" of the optimal product line, the 103 configurations that appear in every optimal solution. On the other end of the spectrum, there are 245 configurations that appear only once or twice—these are configurations that fill out a portfolio, and likely are substitutes for each other. In between are the configurations that appear more frequently, but are not absolutely critical. We will see in §6.4 that this relatively large core (over 35% of any instance's optimal portfolio) implies that there are many different ways of constructing a good portfolio for the Gold Line—as long as Deere gets the core right, they will do quite well. The Silver Line is more challenging.

We show similar results for the Silver Line in Table 8 and Figure 2. Looking first at Table 8, the Silver Line solutions are again largely consistent; there is one outlier with respect to the optimal number of configurations—instance 1 has 539, well below the others, this time likely due to less disjoint migration lists—but the ratios of optimal profit are nearly identical, varying no more than 0.003. Running times did show some variation, but were reasonable in all cases.

Despite this consistency, the frequency histogram generated by the Silver Line experiments is qualitatively different from that of the Gold Line. As seen in Figure 2, again showing only the 2,208 configurations that appear at least once, there is a much smaller "core" to the product line—94 configurations, or roughly one sixth of an optimal product line, appear in at least nine of the optimal solutions. In contrast to this, 1,042 configurations, almost twice a product line, appear only once. There are two factors driving this dispersion: the first is the heterogeneous

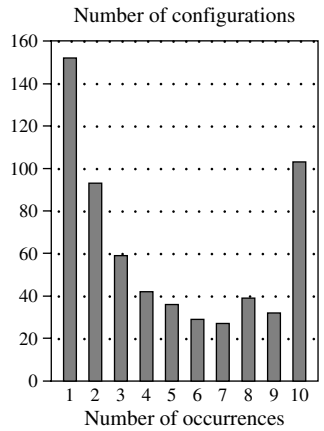
Table 7. Optimal solutions for 10 instances of the Gold Line.

Instance	Profit ratio	Service level % trunc. 4/nontrunc.	Opt. configs.	Time (hours)	Choice % 1st	2nd	3rd	4th
1	1.1853	96.30/97.08	282	1.62	43.64	40.32	9.54	2.78
2	1.1790	96.01/97.07	295	7.45	41.85	39.16	12.19	2.81
3	1.1866	96.52/97.13	283	0.63	43.45	41.23	9.11	2.73
4	1.1848	96.50/97.16	283	0.74	42.90	41.17	10.10	2.32
5	1.1855	96.55/97.26	291	1.05	45.32	37.57	10.91	2.76
6	1.1848	96.28/97.08	287	0.83	45.13	34.39	12.96	3.79
7	1.1847	96.23/97.00	279	0.94	42.79	40.63	9.81	2.99
8	1.1855	96.50/97.30	288	0.71	45.36	36.35	12.08	2.71
9	1.1871	96.01/96.74	280	0.84	43.81	37.74	11.05	3.41
10	1.1850	96.30/97.22	288	1.28	46.82	34.47	12.38	2.63

Silver Line customer base, as described in §6. The second factor is the (virtually) 100% service level in all 10 Silver Line instances; due to the larger margins for these machines, even customers whose preferences make them outliers are being served. (By comparison, service levels are typically 96%–97% on the Gold Line.) Satisfying these outliers is expensive, but worthwhile, on the Silver Line due to higher profit margins. This combination of heterogeneity and profitability leads our algorithm to make less dramatic reductions in the number of configurations on the Silver Line as compared to the Gold Line, and thus our algorithm increases profits on the Silver Line by 8% as compared to 18% on the Gold Line. *Thus, high margins combine with customer heterogeneity to broaden product lines.*

The combination of a heterogeneous customer base and very high service level also increases the percentage of customers served with lower choices in their migration lists: the model forces more of them to accept substitute configurations. Whether such substitution is likely within a customer base, and what strategies may make such substitution more likely, is a question Deere (or any similar company) must study very closely. A salient question thus becomes whether our algorithm can ascertain whether such substitution is likely. We will see this in the next section.

Figure 1. Histogram of Gold Line configuration frequency in optimal solutions from Table 7.



6.4. Constructing Good Product Portfolios

Our ultimate goal is to find a small set of configurations that are highly profitable, i.e., that appeal to the majority of the customer base. We call this a *good portfolio*. Based on the results of the previous section, in particular, the relatively large core of the Gold Line as compared to the Silver Line, we would expect constructing a good Gold Line portfolio to be easier than constructing a good Silver Line portfolio. To explore this, we tested two methods of constructing portfolios. In the first, we simply tested four of the optimal portfolios for the Gold Line and the Silver Line on 10 additional randomly generated instances. The performance of this method is reported in the first two rows of Table 9.

We see from the table that this provides good performance for the Gold Line, but the Silver Line is not broad enough; its service level has declined dramatically. Note that the performance for the Silver Line rebounds when we use nontruncated lists—(virtually) all of the customers do want machines in the portfolios, but these machines are not among their top four choices. This is a manifestation of the fact that Silver Line customers place similar weights on different features, so our perturbation of features' importances can dramatically reshuffle lists. This implies that Deere must be careful when trying to reduce the Silver Line—having features that customers value roughly equivalently means that customers may be willing to substitute, but customer heterogeneity implies that there may be a significant amount of substitution, and this may come at a price. We will see how Deere acted on this insight in §7.

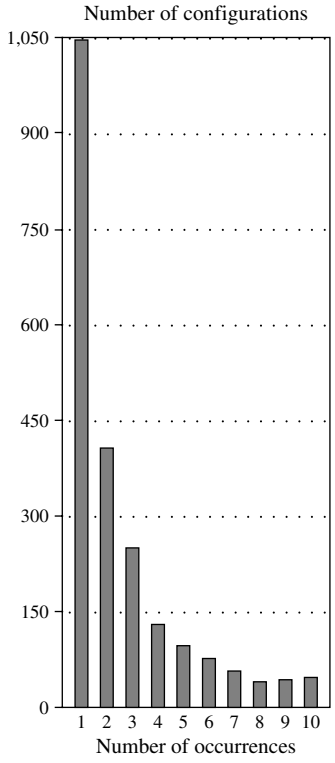
Given that a single optimal Silver Line portfolio may be too narrow, we tried the more conservative solution of taking every configuration that appeared in one of the optimal solutions, or the union of the elements on the histograms of Figures 1 and 2. As seen in the last two rows of Table 9, this is still quite good for the Gold Line, although less than using a single instance, and better (but still not satisfactory) for the Silver Line, at least for the truncated lists. This reinforces the message that constructing a good portfolio for the Gold Line is relatively easy, due to this line's large core, but for the Silver Line it is more difficult, due to customer heterogeneity and high optimal service levels.

Table 8. Optimal solutions for 10 instances of the Silver Line.

Instance	Profit ratio	Service level (%) trunc. 4/nontrunc.	Opt. configs.	Time (hours)	Choice (%) 1st	2nd	3rd	4th
1	1.0816	99.96/99.96	539	2.18	31.14	44.69	15.57	8.56
2	1.0800	100/100	580	1.39	29.30	43.72	17.60	9.38
3	1.0817	100/100	570	1.04	29.12	44.00	18.39	8.49
4	1.0827	100/100	563	0.79	29.30	44.67	17.08	8.94
5	1.0803	100/100	582	0.82	29.46	46.51	15.86	8.18
6	1.0796	100/100	583	5.14	29.23	44.80	16.63	9.34
7	1.0800	100/100	575	1.01	30.44	43.80	16.52	9.24
8	1.0807	100/100	578	5.31	29.30	46.94	15.44	8.32
9	1.0820	100/100	582	0.90	26.86	45.05	17.96	10.13
10	1.0809	100/100	569	1.40	29.15	42.28	19.21	9.36

From Table 9, as well as the results in §6.3, it appears that for the Gold Line a good portfolio can be constructed with approximately 300 configurations. The number of configurations in a good Silver Line is less clear. To try to establish this, we used weighted sampling from Table 2 to construct four different Silver Line portfolios, and then varied each portfolio's size. We "offered" these portfolios to a single instance of the Silver Line problem, plotting the profit of each of portfolio as a function of its size. Figure 3 shows the results of this experiment; a good target portfolio size for the Silver Line appears to be around 1,400 configurations, and 1,200, twice a single optimal portfolio, appears to be a minimum.

Figure 2. Histogram of Silver Line configuration frequency in optimal solutions from Table 8.



6.5. Sensitivity Analysis

In this section, we conduct experiments varying parameters affecting: (i) customers' potential configurations or their search space (§6.5.1), and (ii) how customers evaluate these potential configurations (§6.5.2). In the former case, we vary c, the number of features a customer may vary around C_0; the truncation point of the migration lists (default was four); and u and r, which affect reservation prices and utilities. With respect to how customers evaluate configurations, we experimented with changing n, the number of features having their relative importance scaled; the magnitude of these variations; and finally β, which captures how true customers are to their purchases. All experiments were conducted on instances of the Gold Line from Table 7.

6.5.1. Parameters Affecting Search Space. We first studied increasing c from two (the default for the Gold Line) up through seven. Surprisingly, the effects of these changes were negligible; simply allowing customers to change more features from C_0 left migration lists, and hence optimal portfolios, largely unaffected. We hypothesize that this is because changing c did *not* affect how most customers evaluated configurations; the same set of configurations tended to be preferred, even though customers had more potential choices, because the initial C_0 was likely a high-utility configuration (so customers tended to stick to configurations "close" to C_0). This need not have been the case; if a large proportion of customers purchased configurations with features that do not have high estimated utilities, increasing c would have likely had a more dramatic effect on the migration lists, and hence the optimal portfolios.

Does this insensitivity carry over to changing parameters affecting reservation prices and utilities, r and u, because these also only affect the configurations a customer will consider? No. Even though relaxing these constraints (making r and u larger) does not change how customers evaluate configurations, this does lead to product lines that have higher profits and contain fewer variants, by enabling our algorithm to concentrate customers around higher end models. (The correlation between utility and margin for the Gold Line is 0.9038.) We see this in detail in Table 10,

Table 9. Portfolio characteristics.

Method	Line	Profit ratio	Service level % trunc. 4/nontrunc.	Opt. configs.	Choice % 1st	2nd	3rd	4th
Single instance	Gold	1.1499	93.82/95.43	289	44.16	35.54	9.11	3.00
Single instance	Silver	0.9345	87.96/99.84	571	28.27	40.92	12.45	6.31
Union	Gold	1.1284	98.36/98.66	612	60.22	33.5	4.08	0.56
Union	Silver	0.9922	100/100	2,208	53.93	40.21	4.88	1.09

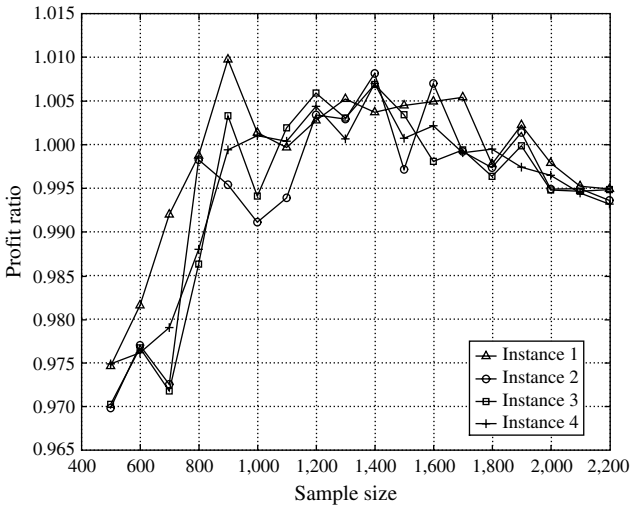
where we vary p and u for Gold Line instances 1, 2, 3, and 10 from Table 7.

Note that there is one less obvious effect of this change—the proportion of customers purchasing their first choice decreases dramatically as reservation price (or utility) relaxes—customers are being "steered" toward higher end products. Increasing u also makes the problem correspondingly more difficult to solve—for the four instances with $2u$, the algorithm only comes to within the optimality gap shown in parentheses in 24 hours. We saw very similar effects when increasing the truncation point for migration lists: solution times and profits increased as service levels remained virtually unchanged, as optimal portfolios served the same number of customers using fewer configurations.

Summarizing the effects of changing c, r, u and truncation points, we conclude that simply giving customers more choices will not necessarily increase profits, but if increasing their choices makes customers more flexible (or if such flexibility can be induced; see §7), then lines may be condensed and profits increased.

6.5.2. Parameters Affecting Evaluation. Now we alter how customers evaluate configurations, examining the effects of changing n (number of features selected to have their importance rescaled), feature importance variation, and β.

Figure 3. Profit vs. portfolio size for four Silver Line instances.



Similar to c, changing n had no significant effect on our solutions. This is not overly surprising—recall from §4.2 that the features not selected also have their importances scaled to keep the summed mean feature values constant. Thus, all utilities are in fact being scaled for $n > 0$.

We next investigated changing the variance of feature importance, experimenting with Gold Line instances 1, 2, 3, and 10 from Table 7. We found the Gold line to be very stable; Table 11 reports the results of optimizing these instances after increasing these variations by 20 times. At this point, profits decrease by just over 10% (essentially to Silver Line levels) as service levels remain constant, but require more configurations to be achieved. Increasing feature variability elevates customer heterogeneity, making the Gold Line customers behave more like the Silver Line's. Note that the Gold Line's customers still have more in common than their Silver Line counterparts: they agree on their first choices much more, as over 80% now receive their first choice. This "agreement" does not lead to higher profits as compared to the Silver Line, as Silver Line customers pay higher margins on average.

Finally, we turn to the effects of changing β, and how likely it is that C_0 is forced to the top of a migration list. In Figure 4, we see how the profit ratio, service level, and first choice probability change with β for Gold Line instance 1 from Table 7. As β changes from zero to one (customers' preference for their purchased machine grows), profits decrease by approximately 5%, service levels drop slightly, and first choice percentage drops dramatically. (Although not shown, changes in the the breadth of the product line show no discernable pattern.) This effect of β on profit is somewhat surprising, as changing β does not change the contents of any customer's migration list; only the order in which the items appear: Any solution to an instance with $\beta = 0$ will satisfy exactly the same set of customers as in the $\beta = 1$ instance. So what is driving this change? The answer lies, again, in the correlation between utility and margin for the Gold Line.

When $\beta = 0$, customers are sorting their choices solely by utility, so their first choices tend to be machines with large margins. The optimization offers these, giving customers their first choices while also reaping greater profits. Conversely, when $\beta = 1$, customers will choose C_0 whenever it is available. Thus, to steer customers to higher margin machines, the optimization does not offer as many first choices. Thus, estimating β is critical; it affects the structure of the optimal product line and also has potentially

Table 10. Sensitivity analysis with respect to reservation price and utility for the Gold Line.

Change	Profit ratio	Service level % trunc. 4/nontrunc.	Opt configs.	Time (hours)	Choice % 1st	2nd	3rd	4th
$r/2$	1.1295	96.79/97.22	296	0.51	61.03	24.38	8.71	2.67
$r/2$	1.1285	96.93/97.34	311	0.79	56.14	30.12	8.19	2.48
$r/2$	1.1279	96.52/97.15	276	1.14	64.37	21.62	7.88	2.64
$r/2$	1.1278	96.46/96.92	288	0.71	60.02	26.72	7.09	2.63
$2r$	1.2897	95.83/96.94	276	0.73	34.85	43.81	13.92	3.26
$2r$	1.2944	96.12/96.98	263	0.48	35.50	45.34	12.41	2.88
$2r$	1.2872	96.25/97.30	263	0.60	38.18	42.34	13.02	2.70
$2r$	1.2876	96.27/97.34	268	3.98	39.37	41.63	11.94	3.33
$u/2$	1.1670	96.50/97.21	298	0.44	55.54	30.08	8.66	2.23
$u/2$	1.1664	96.30/97.15	295	0.80	50.44	35.43	8.39	2.02
$u/2$	1.1661	96.35/97.10	286	1.32	54.82	31.27	8.12	2.16
$u/2$	1.1665	96.45/97.02	293	1.05	52.71	33.61	7.95	2.19
$2u$	1.2252	97.55/98.64	280	>24 (0.14%)	27.03	46.01	19.32	5.19
$2u$	1.2270	97.07/98.45	260	>24 (0.12%)	27.28	46.06	18.82	4.90
$2u$	1.2247	97.39/98.62	277	>24 (0.18%)	26.62	48.06	17.51	5.21
$2u$	1.2244	97.15/98.50	268	>24 (0.24%)	26.29	45.76	19.55	5.55

significant ramifications regarding the number of customers who find their "favorite" machine.

6.6. Algorithmic Modifications and Additional Constraints

We now briefly examine the effect on algorithm performance of (i) modifying the objective function to a step-wise form (as described in §5), and (ii) including additional managerial constraints in the MIP. Specifically, the constraints we add establish:

1. A lower bound $\mathscr{L}$ on configurations (to preserve product line breadth); $\sum_{j\in T} y_j \geqslant \mathscr{L}$;

2. An upper bound on configurations, $\mathscr{U}$ (to contain complexity costs); $\sum_{j\in T} y_j \leqslant \mathscr{U}$; or

3. A lower bound on service level, γ; $\sum_{i\in C}\sum_{j\in L_i} x_{ij} \geqslant \gamma$.

These are only a few of the possible modifications of the MIP; many others can be considered, for example, ensuring that certain "flagship" or "core" configurations are present in the solution.

Adding constraints on the number of configurations ($\geqslant$400 or $\leqslant$200) tends to speed the algorithm, while requiring a higher service level or using the stepwise complexity function tends to increase solution times, but in both cases the computational times remained reasonable. Thus, our algorithm, at least in these preliminary tests, is robust with respect to running time for typical problem variants.

Managerially, we examine how increasing the Gold Line service level affects solutions. In Table 12, we report average profit and portfolio size for different service levels, over instances 1, 2, 3, and 10 from Table 7. As expected, increasing the service level broadens the product line appreciably, but it has only a slight effect on profits: the extra cost of the additional configurations is almost compensated for by the additional revenue. Thus, for the Gold Line, Deere need not be overly aggressive in reducing the line—keeping a broader line and serving more customers is still very profitable.

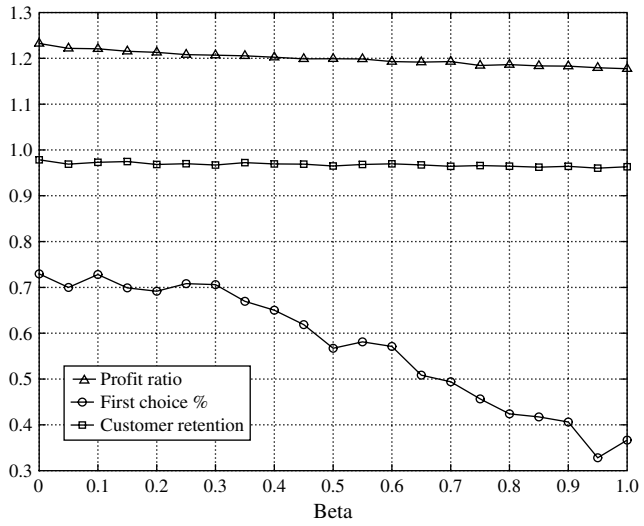
6.7. Summary of Computational Results

Our computational results illustrate a number of characteristics of Deere's problem, and portfolio optimization in general. With respect to Deere, the Gold Line and the Silver Line exhibit significant differences under our optimization: the Gold Line more readily accepts optimization because Gold Line customers more readily accept optimization—they have more in common than their Silver Line counterparts. As such, a well-defined "core" of an optimal Gold

Table 11. Optimal solutions of four Gold Line instances when relative importances vary widely.

Instance	Profit ratio	Service level % trunc. 4/non trunc.	Opt. configs.	Time (hours)	Choice % 1st	2nd	3rd	4th
1	1.0779	96.28/97.98	314	9.36	80.59	5.69	3.17	6.82
2	1.0781	96.51/98.14	332	3.74	80.42	6.27	3.53	6.29
3	1.0771	96.12/97.95	308	3.27	80.54	5.86	3.31	6.41
10	1.0773	96.33/98.16	326	3.27	80.69	5.66	3.70	6.29

Figure 4. Profit, service level, and first choice vs. β for a single Gold Line.



product line is apparent in our replicate experiments. Any reduced line that includes this core should lead to a significant increase in profits, through offering fewer configurations and possibly "steering" customers to higher-margin machines.

The Silver Line is more tricky—there is a much smaller core visible through replication. We can trace this to a specific type of heterogeneity among Silver Line customers—many Silver Line configurations have roughly equal mean parts-worth utility, so different customers may have quite different tastes, even though they are shopping for "comparable" machines (having similar total utility and price). When this fact is combined with the high Silver Line margins, which imply that serving all customers is optimal, broader Silver Lines result. This means that Deere should be more conservative in their efforts to reshape the Silver product line.

Our sensitivity analysis focused on those parameters that affect customer behavior. We isolate two different types of parameters affecting customer behavior: those that affect the configurations a customer will consider, and those that affect how these configurations are evaluated. In general, widening the search space without changing customers' evaluations does not change the problem. In contrast, changing how customers think by relaxing their reservation price or utilities, making them more or less variable in their evaluation of feature values, or tying them more or less closely to their purchased configuration, does change the problem considerably. In a nutshell, if customers can be made more flexible, lines can more easily be consolidated, and profits can be improved. This is the message we brought to Deere, and this is the message that shaped their implementation.

7. Implementation at Deere

There was a cross-functional team at Deere responsible for implementation, comprised of representatives from Deere's corporate analysis group, the order-fulfillment groups of the appropriate business units, Deere's product marketing group, sales leadership, and their dealer council. Given the insights from our analysis, they decided that rather than remove configurations, they would offer discounts on specific options to "steer" customers toward a smaller set of configurations. This reduced set may be thought of as the "core" of the product line; we will call this set C. Clearly, offering greater incentives will increase migration, but also increase costs; thus, there is a trade-off that regulates how much effort Deere should exert to induce those customers outside of C to migrate into C. The results of our analysis helped Deere determine specific migration targets for each of the two product lines (both in excess of 50%) for the percentage of customers who migrate, as well as the level of incentives to offer.

Based on the demand before and after the incentives were implemented, Deere has already achieved their targets by providing discounts of 10%–15% on selected options (accounting for 0.5% to 2% of total model cost) without having to publicly announce a product line compression, or "force" customers away from their first choices. Deere also found, in line with our results, that more migration is possible, with smaller incentives, in the Gold Line than in the Silver Line.

This implementation raises a question: Without actually discontinuing products, can Deere reap profits from more concentrated product lines, especially in a primarily make-to-stock environment? The short answer is yes: so far, Deere has estimated savings in the tens of millions of dollars for the two studied project lines. To calculate these savings, Deere forecasted what their profits would have been without the modifications suggested by our analysis. Then, Deere subtracted this amount from the actual profits post implementation. The portion of this difference attributable our project, rather than other initiatives, constitutes the estimated savings.

Where are these increased profits coming from? Previously, Deere built more configurations, some of which were not even sold once. Now Deere builds fewer variants in advance (but still provides customers with any feasible configuration, custom built, that they want). Moreover, a smaller set of configurations are eventually sold, reducing the complexity of servicing these configurations throughout their lifetime.

Table 12. Impact of service level constraints for the Gold Line.

Service level (%)	Profit ratio	Opt. configs.
97	1.1835	312
98	1.1808	359
99	1.1743	433

8. Summary and Conclusions

While the literature contains much work on optimizing product lines, there have been no industrial applications, nor even any algorithm presented, capable of solving problems of the size and level of complexity found at Deere. We optimally solve such instances—with tens of thousands of customers and possible configurations—in several hours. One factor critical to this efficiency is our use of customer migration lists, capturing customer behavior in a form amenable to optimization.

Leveraging the efficiency of our algorithm, we solve different variants of Deere's basic model, as well as multiple randomized replications of the same instance. This helps us to identify the configurations that form the core of any good solution, which is useful in building optimal portfolios. It also led to several insights into the product line optimization problem, specifically regarding the effects of different modeling choices for customer flexibility and heterogeneity. Moreover, by quantifying the benefits of customer flexibility, we have helped Deere design incentive program which has proved to be very effective at steering customers to a smaller set of configurations, elevating profits by tens of millions of dollars. In addition, Deere's incentive program has helped them avoid the appearance of denying customers their first choice by discontinuing models.

Our work is being applied more widely at Deere; our tool is now part of the standard decision-support process they use when product enhancements are undertaken. In addition, we are currently in the process of applying our methodology at another leading company in the heavy equipment sector. In general, our algorithm can likely be applied by any company with high-margin products having many product variants. This will require data similar to that gathered by Deere, and might also include additional data such as scenario-based estimates of future demand. Such data should be available, or obtainable, from the company's marketing department or directly from their customer base. Obtaining this data, and leveraging it as we do, holds the promise of greater efficiencies, increased profits, and greater customer satisfaction as resources are focused on products that better reflect their desires.

Appendix A. Examples of Configuration and Pricing Rules

We provide examples of configuration and pricing rules below, and then illustrate how the first of these rules would be incorporated as a constraint into our CP formulation.

Sample configuration rules:

1. Feature MidPTO unavailable on machines with both a 2WD axle and SST transmission;

2. For engine 32_130DLV and transmission PRT, axle cannot be 2WD.

Sample pricing rules:

1. If engine type is 28_129DLV and the machine does not have a MidPTO, SST transmission costs \$1,000 for a 2WD axle and \$1,200 for a 4WD axle;

2. If engine type is 32_130DLV, MidPTO costs \$250 in two circumstances: either the axle is 4WD, or the axle is 2WD and the transmission type is not SST.

Configuration Rule Example. Let the line under consideration have n different features. A vector x of n variables $x_1, \ldots, x_n$ represents an option choice for each of the n features; if feature i has m_i different options, the domain of x_i is defined as $\{1, \ldots, m_i\}$. We write all combination and pricing rules as constraints on the x variables. For instance, if transmission is the second feature and SST is its third option for transmissions, axle is the third feature and 2WD is its first option, and MidPTO is the fifth feature and it is present when $x_5 = 1$, then configuration rule 1 above is

$$x_2 = 3 \wedge x_3 = 1 \implies x_5 \neq 1.$$

Appendix B. Properties of Optimal Solutions to the Optimization Model

Proposition 1. *When looking for optimal solutions, imposing integrality on all the y variables alone implies integrality of all the x variables, and vice versa. Moreover, at least one of the integrality conditions* (3) *or* (4) *is necessary to avoid the possibility of fractional optimal solutions.*

Proof. Let $y_j \in \{0, 1\}$ for all $j \in T$. Let $Y_0 = \{j \in L_i \mid y_j = 0\}$ and $Y_1 = \{j \in L_i \mid y_j = 1\}$, the sets of configurations on the list which are not, and are, produced, respectively. Constraints (1) say that $x_{ij} = 0$ for all $j \in Y_0$. For every customer $i \in C$, let $j_i \in Y_1$ be the produced configuration that appears earliest in L_i. If configuration k comes before j_i in L_i, (1) makes $x_{ik} = 0$. If configuration k comes after j_i in L_i, (2) makes $x_{ik} = 0$. Finally, x_{ij_i} will be equal to one due to the objective function (here we assume that all configurations have positive profit). If no such j_i exists for a given customer i, then all x variables for customer i are set to zero because of (1).

Now let $x_{ij} \in \{0, 1\}$ for all $i \in C$ and $j \in L_i$. Let $Y_2 = \{j \in T \mid x_{ij} = 1 \text{ for some } i \in C\}$ be those configurations that have to be produced. Constraints (1) will force $y_j = 1$ for all $j \in Y_2$. Also, (2) will make $y_j = 0$ whenever there exists a customer $i \in C$ such that $x_{ik} = 1$ and j precedes k in L_i. Finally, all remaining y variables that have not been fixed will be set to zero because we are maximizing, and a is a nondecreasing function of the sum of all y variables.

For the second part, let us consider an example in which constraints (3) and (4) are relaxed, i.e., $\{0, 1\}$ is replaced by $[0, 1]$. Let $C = \{1, 2, 3\}$, $T = \{1, 2, 3\}$, $L_1 = \{1, 2\}$, $L_2 = \{2, 3\}$, and $L_3 = \{3, 1\}$. In addition, let $p_1 - K = 101$, $p_2 - K = 103$, $p_3 - K = 102$, $overhead = 0$, and let the complexity cost of producing 1, 2, or 3 configurations

be 200, 250, and 300, respectively. If we decide to build only one configuration, the best choice is to make $y_2 = 1$, which yields a profit of $2 \times 103 - 200 = 6$. If we decide to build two configurations, the best choice is to make $y_2 = y_3 = 1$, which yields a profit of $2 \times 103 + 102 - 250 = 58$. Finally, if we make $y_1 = y_2 = y_3 = 1$, our profit is equal to $101 + 103 + 102 - 300 = 6$. But, if we make $y_1 = y_2 = y_3 = 0.5$, we will have the maximum profit of $101 + 103 + 102 - 225 = 81$. This gives us an integrality gap of $(81 - 58)/58 \approx 39.7\%$. □

Acknowledgments

This project was supported by Deere & Company and the National Science Foundation. The authors thank the associate editor and two anonymous referees, as well as Tom Surma and Loren Troyer of Deere & Company, who all helped to improve both the content and presentation of the latest version of the paper.

References

Abegglen, J., G. Stalk. 1985. *KAISHA, The Japanese Corporation*. Basic Books, New York.

Anderson, S. 1995. Measuring the impact of product mix heterogeneity on manufacturing overhead cost. *Accounting Rev.* **70**(3) 363–387.

Broniarczyk, S., W. Hoyer, L. McAlister. 1998. Consumers' perceptions of the assortment offered in a grocery category: The impact of item reduction. *J. Marketing Res.* **35** 166–176.

Chen, F., J. Eliashberg, P. Zipkin. 1998. Customer preferences, supply-chain costs, and product-line design. T.-H. Ho, C. S. Tang, eds. *Product Variety Management: Research Advances*. Kluwer Academic Publishers, Norwell, MA, 123–144.

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

Chong, J., T. Ho, C. Tang. 1998. Product structure, brand width, and brand share. T.-H. Ho, C. S. Tang, eds. *Product Variety Management: Research Advances*. Kluwer Academic Publishers, Norwell, MA, 39–64.

De Groote, X. 1994. Flexibility and marketing/manufacturing coordination. *Internat. J. Production Econom.* **36** 153–167.

Dobson, G., S. Kalish. 1988. Positioning and pricing a product line. *Marketing Sci.* **7**(2) 107–125.

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

Fisher, M., C. D. Ittner. 1999. Impact of product variety on automobile assembly operations. *Management Sci.* **46**(6) 771–786.

Fisher, M., A. Jain, J. MacDuffie. 1995. Strategies for product variety: Lessons from the auto industry. E. Bowman, B. Kogut, eds. *Redesigning the Firm*. Oxford University Press, Oxford, UK, 116–154.

Foster, G., M. Gupta. 1990. Manufacturing overhead cost driver analysis. *J. Accounting Econom.* **12** 309–337.

Green, P., A. Krieger. 1985. Models and heuristics for product line selection. *Marketing Sci.* **4**(1) 1–19.

Hayes, R., S. Wheelright. 1984. *Restoring Our Competitive Edge*. John Wiley & Sons, New York.

Henkoff, R. 1995. New management secrets from Japan. *Fortune* (November) 135–146.

Kahn, B. 1995. Consumer variety-seeking among goods and services. *J. Retailing Consumer Services* **2**(3) 139–148.

Kekre, S. 1987. Performance of a manufacturing cell with increased product mix. *IIE Trans.* **19**(3) 329–339.

Kekre, S., K. Srinivasan. 1990. Broader product line: A necessity to achieve success? *Management Sci.* **36**(10) 1216–1231.

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

Marriott, K., P. Stuckey. 1998. *Programming with Constraints: An Introduction*. MIT Press, Cambridge, MA.

McBride, R., F. Zufryden. 1988. An integer programming approach to the optimal product line selection problem. *Marketing Sci.* **7**(2) 126–140.

Morgan, L., R. Daniels, P. Kouvelis. 2001. Marketing/manufacturing trade-offs in product line management. *Management Sci.* **33** 949–962.

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

Quelch, J., D. Kenny. 1994. Extend profits: Not product lines. *Harvard Bus. Rev.* **72**(5) 153–160.

Raleigh, A. 2003. Product logic. Seminar: Managing Complexity for Competitive Advantage. New England Council for Operations Excellence, Washington, D.C.

Raman, N., D. Chhajed. 1995. Simultaneous determination of product attributes and prices, and production processes in product-line design. *J. Oper. Management* **12** 187–204.

Ramdas, K. 2003. Managing product variety: An integrative review and research directions. *Production Oper. Management* **12**(1) 79–101.

Randall, T., K. Ulrich, D. Reibstein. 1998. Brand equity and vertical product line extent. *Marketing Sci.* **17**(4) 356–379.

Schiller, Z. 1996. Make it simple. *Business Week* (September) 96–104.

Stalk, G., Jr., A. Webber. 1993. Japan's dark side of time. *Harvard Bus. Rev.* **71**(4) 93–102.

Thonemann, U., M. Brandeau. 2000. Optimal commonality in component design. *Oper. Res.* **48**(1) 1–19.

Tsang, E. 1993. *Foundations of Constraint Satisfaction*. Academic Press, New York.

Yano, C., G. Dobson. 1998a. Designing product lines for diverse customer markets. H. L. Lee, S.-M. Ng, eds. *Global Supply Chain and Technology Management*. POMS Series in Technology and Operations Management, Vol. 1, 152–158.

Yano, C., G. Dobson. 1998b. Profit optimizing product line design, selection and pricing with manufacturing cost considerations. H. L. Lee, S.-M. Ng, eds. *Global Supply Chain and Technology Management*. POMS Series in Technology and Operations Management, Vol. 1, 103–122.

Yunes, T. H. 2006. On the integration of optimization techniques. Ph.D. dissertation, Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA.