Managing Broader Product Lines through Delayed Differentiation Using Vanilla Boxes

Jayashankar M. Swaminathan • Sridhar R. Tayur
Walter A. Haas School of Business, University of California, Berkeley, California 94720
GSIA, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

In an attempt to reduce cost while maintaining good customer service, some of the leading manufacturers in the computer industry are delaying product differentiation (by exploiting component commonality) while managing broader product lines. In an environment where demands are stochastic, it seems a good strategy to store inventory in the form of semi-finished products (vanilla boxes) that can serve more than one final product. However, finding the optimal configurations and inventory levels of the vanilla boxes could be a challenging task. In this paper, we model the above problem as a two-stage integer program with recourse. By utilizing structural decomposition of the problem and (sub)gradient derivative methods, we provide an effective solution procedure. A special case, a variant, and several extensions are also discussed. In our computational section, we utilize our model to study several new research issues. We provide insights on the effect of demand variance, correlation, and capacity limitations on the optimal configuration and inventory levels of vanilla boxes and the performance of a vanilla assembly process. In addition, we compare the performance of the vanilla assembly process to make-to-stock and assemble-to-order processes and provide managerial insights on the conditions under which one might be better than the others. Finally, we discuss the characteristics of an IBM product line (which motivated this work) and the effectiveness of a heuristic tailored for that application.

(Stochastic Multi-Product Assembly Systems; Product Variety; Component Commonality; Demand Correlation; Inventory Management; Stochastic Integer Programs)

1. Introduction
In the last decade, there has been an increasing emphasis on producing specialized products for different segments in the market, as a means for providing better service to customers. The Marketing literature (Bagozzi 1986, Cravens and Woodruff 1986) indicates that broader specialized product lines lead to higher market share, whereas the Operations Management literature predicts that cost and complexity may increase when there is greater product variety. Manufacturers have been adopting strategies including (but not limited to) (i) integration of marketing, manufacturing, and distribution activities within the supply chain (Cohen and Lee 1988, Lee and Billington 1993), (ii) accurate response to changes in market needs (Fisher et al. 1994), (iii) flexibility in manufacturing process using concepts such as delayed differentiation (Lee 1996, Lee and Tang 1998), (iv) incorporation of commonality at the product design stage (Ulrich and Pearson 1993, Hayes et al. 1988), (v) adoption of flexible manufacturing technologies (Jaikumar 1986), and (vi) implementation of just-in-time practices such as set-up reduction (Hall 1983), in order to reduce costs and improve customer satisfaction. Kekre and Srinivasan (1990) suggest that such strategies may mitigate any possible adverse effects of product proliferation.

The final assembly stage of a leading product line of IBM is the research site that motivated this work. Different models across the product line show a high degree of component commonality. In addition, there is a
one-to-one relationship between components and features. As a result, products are defined in terms of features that are ordered by customers (Srinivasan and Swaminathan 1997). Since demand for different models is highly stochastic and correlated, the existing mode of operation was to start final assembly only after a firm customer order was received. In the recent past, this process often finished later than the customer requested arrival date, causing a sizeable percentage of orders to be late. Orders were late because customers who once were satisfied with delivery within a fortnight were now requiring products to be shipped within five to seven days. This change in customer requirements was mainly due to competition in the industry and an increase in service expectations.

Management decided to pilot an assembly process based on semi-finished products called vanilla boxes. Figure 1 shows a fictitious product structure consisting of three products ($P_1$, $P_2$, and $P_3$) and four components ($a$, $b$, $c$, and $d$). $P_1$, $P_2$, and $P_3$ correspond to three model types of a product line. The components $a$, $b$, $c$, and $d$ correspond to memory card, processor, hard disk, and floppy drive, respectively. The bills of material for the products are $P_1 = (a, b, c)$, $P_2 = (b, c, d)$, and $P_3 = (a, b, d)$. $V_1$, $V_2$, and $V_3$ are examples of feasible vanilla boxes for the product structure. $V_1$ (containing memory card and processor) can be used in the assembly of $P_1$ and $P_3$ because these products can be assembled from it, by the addition of appropriate components ($c$, a hard disk for $P_1$ and $d$, a floppy drive for $P_3$). When a customer order came in, additional components were often added to a vanilla box and the product was shipped to the customer within the customer requested due date. Thus a vanilla assembly process enabled assembly of customized products within much shorter lead times. To achieve this, though, the manufacturer had to carry additional inventory of vanilla boxes.

The vanilla assembly process was piloted at a final assembly plant which had a nearby satellite plant. The satellite plant could build, test, and ship vanilla boxes to the final assembly plant. Thus, the main issues at the final assembly plant were to determine how many and what type of vanilla boxes to keep, and how to allocate vanilla boxes to final products in order to minimize the expected stock-out costs for lost product demand and holding costs for left-over vanilla boxes. Our basic model addresses the above issues. Subsequently we extend the basic model to incorporate the case where final assembly and vanilla building are done at the same plant (and share the same capacity) in a multi-period setting.

The need to capture the stochastic and correlated nature of product demand, the commonality of parts in the product line, and assembly speed-ups due to the usage of vanilla boxes added significant complexity to the modeling task. After considering several alternatives we decided to develop a discrete time model with finite assembly capacity in which the manufacturer experiences stochastic demand for multiple products. We utilize a two-stage stochastic programming framework. The first stage decision corresponds to choosing the configuration of vanilla boxes (what features they should contain) and their inventory levels. The demand process is modeled using a set of demand scenarios each of which is assigned a probability of occurrence. Once the demand occurs, the second stage decision corresponds to how the vanilla boxes are allocated to the different products within a limited assembly capacity. The objective is to minimize the sum of stock-out costs of unsatisfied demand and holding costs of left-over vanilla boxes.
1.1. Literature Review and Summary of Paper

Our problem has several similarities to the component commonality problem (Collier 1982, Sauer 1985, Gerchak and Henig 1986, Baker et al. 1986, Gerchak et al. 1988). The component commonality problem has been considered either with stock-out costs or service level constraints. Multi-period problems with service constraints can be extremely difficult to solve as indicated in Srinivasan et al. (1992). Tayur (1994) uses a (sub)gradient based method to solve a multi-period problem with stock-out costs. In our problem, we optimize the vanilla configuration and inventory while taking into account capacity constraints on assembly, a factor which is not considered in component commonality problems. Another related problem is the multi-product substitution problem (Chand et al. 1994, Bassok et al. 1995, Bassok and Hsu 1995). In our problem, substitution is possible when there is a choice in the vanilla box to be used for a product. From a methodological point of view, we utilize stochastic programming with recourse with first stage integer variables. These are difficult models to solve computationally (Wollmer 1980). Additional complicating factors are discussed in §2.1.

In this paper, we model and analyze an assembly process based on vanilla boxes which enables customization of products within short lead times. This is a new approach in final assembly and is being piloted at many manufacturing organizations including IBM, US Filter, and 3Com. Our model finds the optimal configuration of vanilla boxes as well as their inventory levels. The model simultaneously addresses issues related to product line characteristics (such as commonality), demand characteristics (such as variance and correlation of product demands), and assembly time characteristics (such as reduction in assembly time when semi-finished products are used for customization). As per the knowledge of the authors this is the first model that addresses all the above issues in an integrated manner.

The richness of the model enables us to provide several new insights in our computational study. These include the effect of correlation in product demands and changes in assembly capacity on (1) the performance of a vanilla assembly process and (2) the optimal inventory and (3) configuration of vanilla boxes. In addition, we compare the performance of the vanilla assembly process against assemble-to-order and make-to-stock environments, providing managerial insights on the effectiveness of each under different demand conditions and capacity restrictions. Finally, in our computational section, we discuss the characteristics of the IBM product line that motivated this work and provide details about a sequential heuristic tailored to handle industry size problems. Our results indicate that our heuristic can solve industry size problems in a reasonable time while providing cost effective solutions.

The rest of the paper is organized as follows. In §2 we discuss the basic model and explain the algorithm. Further, we extend our basic model to include both final assembly and vanilla building in the same plant within a multi-period setting. In §3 we present computational results and provide insights into the vanilla assembly process. In §4 we discuss a special case, a variant, and other extensions of the basic model.

2. Model

In this section, we first describe our basic model in which there is a satellite facility which produces vanilla boxes for the final assembly plant. Subsequently, we expand our model to multi-period settings where capacity at the assembly plant is utilized to make both final products and vanilla boxes. Several other extensions of the basic model are discussed in §4.3.

2.1. Basic Model

The basic model considers a single period. C is the capacity available to assemble products from vanilla boxes or from basic components. All vanilla boxes are produced before the beginning of the period. Demands are realized at the beginning of the period before decisions need to be made regarding the assembly of the final products. Every product \(i(1, \ldots, N)\) may either be assembled directly from its components, or from any vanilla box whose component set is a subset of those required by \(i\), thus avoiding redundant components. Demands are random but follow one of \(L\) given scenarios, each with a given likelihood. Holding costs are incurred for unused vanilla boxes at a box type specific rate, and shortage costs for unsatisfied demand, at a product specific rate. Without loss of generality, we assume that the bill of material in terms of the components is binary.

Our proposed planning approach starts with the enumeration of all possible vanilla box configurations using...
Managing Broader Product Lines Using Vanilla Boxes

SWAMINATHAN AND TAYUR

at most $K$ different types or a suitable subset thereof. With $n$ the number of components there are $2^n - n - 1$ different vanilla boxes since we exclude vanilla boxes with zero or one component. So, if $K$ optimal vanilla boxes have to be chosen then we need to consider $(C_k^{2^n - n - 1})$ configurations. This number may be lower in practice because there may be vanilla choices that do not support even a single product. Still, our algorithm is exponential and is not suitable for solving large problems exactly. We feel that the insights obtained in this paper can help in tailoring an approximation (or heuristic) strategy for very large problems. In §3.3 we describe a greedy sequential heuristic developed for the IBM problem. For a given choice of the vanilla box configuration, the remaining problem consists of determining optimal inventory levels for the vanilla boxes used, and for each scenario, an optimal assembly plan. These can be determined via the following 2-stage stochastic programs. Let

- $t_{i0}$: per unit assembly time for product $i$ from components;
- $t_{ik}$: per unit assembly time for product $i$ from vanilla box $k$ ($t_{ik} = \infty$ if product $i$ cannot be made from vanilla box $k$);
- $\pi_i$: per unit per period stock-out cost for product $i$;
- $h_k$: per unit per period holding cost for a vanilla box $k$;
- $q = (q_1, \ldots, q_k)$: vector of starting inventory levels of vanilla boxes $k$ ($1, \ldots, K$);
- $\xi_l$: a realization ($\xi_l, \ldots, \xi_{N_l}$) of $L$ product demands in scenario $l$ where $\xi_1, \ldots, \xi_N$ have a joint distribution $F$; and
- $s_{kl}$: quantity of product $i$ made using vanilla box $k$ in scenario $l$ ($k = 0$ implies that product $i$ is assembled directly from components).

The two-stage stochastic program corresponding to a vanilla box configuration $U$ can be formulated as:

$$P(I(U)): \min_{q} E_0 Q(q, U, \xi_l),$$

where

$$Q(q, U, \xi_l) = \min_{s_l} \sum_{i=1}^{N} \left( \pi_i \left( \xi_{il} - \sum_{k=0}^{K} s_{kl} \right) \right) + \sum_{k=1}^{K} \left( h_k \left( q_k - \sum_{i=1}^{N} s_{kl} \right) \right)$$

s.t.

$$\sum_{i=1}^{N} \sum_{k=0}^{K} t_{ik} s_{kl} = C \quad \forall l,$$

$$\sum_{i=1}^{N} s_{kl} = q_k \quad \forall k \geq 1, \forall l,$$

$$\sum_{k=0}^{K} s_{kl} = \xi_{il} \quad \forall i, l,$$

$$s_{kl}, q_l \in \mathbb{R}_+.$$

Let $\psi(q^0)$ denote the optimal value of the above linear program when fixing $q = q^0$. It is well known that the value of the linear program is jointly convex in the right-hand side values of its constraints. Thus, $\psi(q)$ is jointly convex. There are two common approaches for solving the above problem—a large scale linear programming approach where there are $O(NKL)$ variables and a (sub)gradient based approach in which the problem is decomposed into $L$ separate linear programs, each with only $O(NK)$ variables, once $q$ is fixed. A detailed comparison of these two methods and other related properties of our formulation are discussed in Swaminathan and Tayur (1995). Based on those results we have decided to utilize the (sub)gradient based approach.

Recall that a (sub)gradient based method starts with an arbitrary initial vector $q^0$, and at the $k$th iteration updates $q^k$ using

$$q^{k+1} = q^k + \text{step}^{k+1} \nabla \psi(q^k).$$

Here $\nabla \psi(q^k)$ represents an arbitrary (sub)gradient of $\psi(q^k)$ and step$^{k+1}$ is the step size. The function $\psi(q)$ is differentiable in $q = q^0$ when a unique solution exists in each of the $L$ linear programs. Whether this is the case or not, the (sub)gradient can always be obtained from the dual variables associated with constraints (2). The above method converges to the optimal solution as long as the step sizes step$^{k+1}$ are chosen appropriately, e.g., when step$^k \rightarrow 0$ and $\sum_{i=1}^{N} \text{step}^i \rightarrow \infty$; see Bazaara and Shetty (1979).

We conclude this section with an observation which may reduce the number of vanilla configurations that need to be considered. Define a vanilla box as maximal if the addition of any component to it reduces the number of products that can be assembled using the vanilla box. If the holding costs of all vanilla boxes are identical, then it is straightforward to see that an optimal configuration exists with maximal vanilla boxes only.
2.2. Extended Model
The basic model above can be expanded to settings where the assembly capacity is used to produce vanilla boxes as well as final products, and even to multi-period settings under the following two assumptions: (1) Demands in consecutive periods are independent; (2) A base stock policy is adopted for managing the inventory so that each period starts with the same base stock policy is adopted for managing the inventory. We assign overtime costs for bringing the inventory level to the target level if it is lower at the end of a period.

Under these two assumptions, each period’s planning problem is entirely independent of that of the other periods. Here is some additional notation: (1) \( x_{il} \): production of vanilla box \( k \) in scenario \( l \); (2) \( r_j \): additional cost incurred (overtime charges) per unit of vanilla type \( k \) assembled at the end of the period; (3) \( c_{jk} \): amount of assembly capacity utilized per unit of vanilla type \( k \) assembled during the period. The period's planning problem can be expanded to settings under the following two assumptions: (1) De-...
Figure 2  Bill of Material for X1

\[
\begin{pmatrix}
  a & b & c & d & e \\
  P1 & 1 & 0 & 0 & 1 & 1 \\
  P2 & 0 & 1 & 0 & 1 & 1 \\
  P3 & 1 & 1 & 0 & 1 & 1 \\
  P4 & 1 & 1 & 1 & 1 & 0 \\
  P5 & 1 & 0 & 1 & 1 & 1
\end{pmatrix}
\]

- Speed up in Assembly: We assume in our computations that it takes 1 unit of time to assemble a component. As a result, the time to assemble any product from raw components is equal to the number of components in the product. Therefore, using a vanilla box reduces the final assembly time.
- Product Demands: A sample of demand vectors for products is generated from the multivariate normal distribution using a standard procedure. The different vectors represent different scenarios, each with equal probability.

Tables 1 and 2 represent the average cost incurred by the vanilla process ($K = 1, 2$), make-to-stock (MTS), and assemble-to-order (ATO) for the 12 instances of problem X1. Table 3 shows the changes in the optimal vanilla configuration and inventory levels as capacity is varied.

- **Variance and Correlation of Product Demands.**
  (1) We find that an increase in variance increases the cost incurred (Tables 1 and 2). (2) The vanilla process incurs lower cost under negative as compared to positive correlation (Tables 1 and 2). When $\rho = -0.25$, while the demands for individual products are variable the total demand across all products is fairly constant. As a result, the demand for common components across the product line is stable. The demand for vanilla boxes made out of these common components can be predicted accurately. So, one would expect a vanilla process to perform better under negative correlation.

### Table 1  Effect of Changes in Capacity and Variance of Demand on the Cost Incurred under Negative Correlation ($\rho = -0.25$)

<table>
<thead>
<tr>
<th>Capacity</th>
<th>$K = 1$</th>
<th>$K = 2$</th>
<th>ATO</th>
<th>MTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low–35%</td>
<td>11532.80</td>
<td>1844.47</td>
<td>29722.71</td>
<td>1278.61</td>
</tr>
<tr>
<td>Med–65%</td>
<td>1793.79</td>
<td>297.10</td>
<td>16967.93</td>
<td>575.36</td>
</tr>
<tr>
<td>High–95%</td>
<td>202.27</td>
<td>10.98</td>
<td>6227.49</td>
<td>154.22</td>
</tr>
<tr>
<td>Low–35%</td>
<td>7193.37</td>
<td>332.05</td>
<td>23878.92</td>
<td>276.39</td>
</tr>
<tr>
<td>Med–65%</td>
<td>92.05</td>
<td>0.00</td>
<td>12256.44</td>
<td>0.85</td>
</tr>
<tr>
<td>High–95%</td>
<td>0.00</td>
<td>0.00</td>
<td>1033.46</td>
<td>0.00</td>
</tr>
</tbody>
</table>

### Table 2  Effect of Changes in Capacity and Variance of Demand on the Cost Incurred under Positive Correlation ($\rho = 0.5$)

<table>
<thead>
<tr>
<th>Capacity</th>
<th>$K = 1$</th>
<th>$K = 2$</th>
<th>ATO</th>
<th>MTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low–35%</td>
<td>23709.84</td>
<td>9314.11</td>
<td>35259.06</td>
<td>1839.39</td>
</tr>
<tr>
<td>Med–65%</td>
<td>12497.41</td>
<td>2506.60</td>
<td>39588.44</td>
<td>1151.24</td>
</tr>
<tr>
<td>High–95%</td>
<td>202.27</td>
<td>10.98</td>
<td>6227.49</td>
<td>154.22</td>
</tr>
<tr>
<td>Low–35%</td>
<td>14166.75</td>
<td>3655.16</td>
<td>37137.78</td>
<td>655.44</td>
</tr>
<tr>
<td>Med–65%</td>
<td>4391.04</td>
<td>48.92</td>
<td>24996.42</td>
<td>290.30</td>
</tr>
<tr>
<td>High–95%</td>
<td>0.00</td>
<td>0.00</td>
<td>1033.46</td>
<td>0.00</td>
</tr>
</tbody>
</table>

### Table 3  Influence of Capacity on Optimal Vanilla Configurations and Inventory ($K = 2$)

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Vanilla</th>
<th>Inventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>15%</td>
<td>$((a , d), , (a , b , c , d))$</td>
<td>(535.19, 299.21)</td>
</tr>
<tr>
<td>35%</td>
<td>$((a , d), , (a , b , c , d))$</td>
<td>(700.64, 299.21)</td>
</tr>
<tr>
<td>65%</td>
<td>$((a , d , e), , (b , d))$</td>
<td>(585.66, 521.73)</td>
</tr>
<tr>
<td>95%</td>
<td>$((a , d , e), , (b , d))$</td>
<td>(501.88, 448.70)</td>
</tr>
</tbody>
</table>
never, at very low capacities (15% to 20%), additional experiments revealed that this behavior is not always observed.

- **Capacity Restrictions.** (1) An increase in capacity reduces the total cost incurred (Tables 1 and 2). Since capacity appears on the right-hand side of the recourse step of the stochastic program, we can show that the total cost function is convex and decreasing with respect to capacity. (2) We observe the following stages as capacity is increased (Table 3): First, the inventories of the vanilla boxes increase, but the configuration does not change. Next, the configuration changes, with the new vanilla boxes typically having smaller numbers of components. In the final stage, the inventory levels are reduced but the configuration does not change. Our intuition is as follows. Under very tight capacity restrictions vanilla boxes, even if available in plenty, cannot be utilized because there is not enough capacity to assemble products even from vanilla boxes, and as a result, the optimal inventory is low. Under a larger capacity level vanilla boxes are more useful, and as a result, there is an increase in optimal inventory. Finally, when capacity is high, many vanilla boxes may not be required since products can be assembled from components, and as a result, the optimal inventory levels decrease.

- **Number of Vanilla Types (K).** (1) An increase in the number of vanilla boxes leads to a decrease in the cost incurred (Tables 1 and 2).

### 3.2. Comparison with Make-to-Stock and Assemble-to-Order

Make-to-stock (MTS) and assemble-to-order (ATO) processes are extreme cases where the configuration of vanilla boxes corresponds with the set of final products and with \( K = 0 \), respectively. In this section, we compare the performance of the vanilla assembly process to ATO and MTS processes. We utilized the same 40 problems and an additional 10 problems with nine products and six components each, under strong negative correlation in demand, to understand conditions under which a vanilla process may be superior.

Table 4 displays a comparison between the vanilla assembly process and MTS under negative correlation, for the 40 problems used in the pilot study. The first (second, third) numbers within parentheses denote the number of problems in which MTS does better than (equal to, worse than) the vanilla assembly process. Table 5 compares the cost incurred by the vanilla assembly process and MTS for the problems with nine products.

- **Demand Correlation.** (1) We find that the vanilla process (with \( K = 2 \)) performs significantly better than MTS under negative correlation and medium and large capacity (Table 4). (2) MTS performs better than the vanilla assembly process (\( K = 1, 2 \)) in all cases with
positive demand correlation and as a result, we have not provided a table similar to Table 4.\(^1\) (3) The performance of the vanilla assembly process and MTS is better than or equal to ATO in all considered cases (Tables 1 and 2).

- **Capacity Restrictions.** (1) Under medium and high capacity, the vanilla assembly process with \(K = 2\) performs better than MTS under negative correlation (Tables 1, 2, and 4). (2) The cost of MTS or the vanilla process, relative to ATO, improves as capacity increases (Tables 1 and 2).

- **Variance.** The costs of ATO, MTS, and the vanilla assembly process all increase with \(cv\) but those of MTS and the vanilla process increase faster than that of ATO; due to the increased uncertainty, costlier mistakes are made in the up-front production of vanilla boxes or final products (Tables 1 and 2).

- **Number of Vanilla Boxes.** (1) A larger number of vanilla boxes clearly improves the performance of the vanilla assembly process (Tables 1, 2, and 4). Swaminathan and Tayur (1995) note that the expected cost appears convexly decreasing in \(K\). In addition they note that when \(K = N\), an optimal vanilla process could be better and different from MTS which stocks the \(N\) final products. For example, if the products are \((a, b, c), (b, c, d), \) and \((a, c, d)\), then the best three vanilla boxes could be \((b, c), (a, c), \) and \((c, d)\).

Based on the above results, it became evident that a vanilla assembly process could perform better than MTS under negative correlation and medium to high capacity. In order to further confirm the above we generated 10 problems (P1–P10) with nine products and six components each and tested them under conditions similar to those in the IBM problem that motivated this work.

The negative correlation in demand as evidenced at IBM and other firms is of the following type. Out of \(N\) products in the product line only a subset of them are demanded in a given period. To model this, we first generated independent demands with \(cv = 0.125\). Subsequently, we modified the demands so that in each scenario demand occurred for only three products. For example, demand for products 1, 4, 7 would occur in four scenarios and demand for products 2, 5, 8 would occur in four other scenarios. The demands in scenarios were adjusted so that the mean demand for each product computed over all the scenarios did not change. Table 5 shows the performance of MTS and the vanilla process with \(K = 2\) for the ten problems P1–P10 under medium (60%) and high (90%) capacity with the above demand process.

We find that the vanilla process performs better than MTS in 17 out of 20 cases and in all the 10 cases under high capacity (Table 5). In addition, our results indicate that it is possible to stock just two vanilla boxes and significantly outperform MTS (with nine products) under the conditions of our experiment (Table 5).

### 3.3. Application to Industry Size Problems

In this section, we provide a brief description of the problem that motivated this work, describe a sequential heuristic, and provide results based on industry size problems. Our research site was an assembly plant of an IBM product line. There were 50 main products in the product line and each one of them was assembled from 10 major components (features). In reality each product was assembled from hundreds of components. However, approximately ten major components (sub-assemblies) including memory, hard-file, power-unit, processor, disk drive, floppy drive, graphic cards, and communication cards constituted 80–90% of the cost of a product. Management was interested in efficiently managing the assembly of these components. Typical steps in final assembly involved getting components together (kitting), putting them in the right place (assembly), testing, loading software (preloading), and packaging the final product. The existing mode of operations was an assemble-to-order (ATO) process.

A make-to-stock environment was not being used because these products were customized from a set of components and it was a daunting task to keep track of all the possible choices. The 50 products considered in our study constituted more than 90% of the demand for the product line. In addition, demands for products were random and correlated. For example, sometimes there was demand for only 20% of the above 50 products and at other times the remaining 80% were in great demand. In such an environment, an assemble-to-order

---

\(^1\) Note that a vanilla assembly process with \(K = N\) will perform at least as well as MTS since MTS is a special case of that vanilla assembly system.
system provided flexibility to change production based on realized demand. However, this came at the expense of orders often being late or lost due to quoted lead times being excessive.

Management decided to experiment with delayed differentiation using vanilla boxes at the final assembly stage. The assembly plant had a satellite plant where vanilla boxes could be made. Management restricted the number of types of vanilla boxes to at most three in order to make it easier to pilot the process. In addition, they wanted to gradually introduce the different types, one at a time, and stop when they were either satisfied with the performance or if they found that the vanilla assembly process was too difficult to implement.

Since the problem size was too large for our algorithm to handle, we developed a greedy heuristic which at the nth stage selects the vanilla box that provides the maximum cost reduction when used with the n − 1 previously selected vanilla boxes. Swaminathan and Tayur (1995) report that the above heuristic provides optimal solutions for 87% of a sample set of 80 problems with K ≤ 5.

We created five problems: P11, P12 (50 products and 9 components) and P13, P14, and P15 (50 products and 10 components) which do not directly correspond to any IBM problem, although they are of the same size. However, the bill of materials for the problems were generated based on the bill of materials for two products lines provided to us by IBM. In each of these problems we considered 100 scenarios to capture the demand process. The problems were run with a utilization rate of 70% and cv = 0.125. In any given scenario, demands occur for only 10 products. We generated the demand using a similar technique as described in section 3.2. The number of vanilla configurations that were evaluated was in the range of 1200 to 1500.2

Table 6 displays the costs of the vanilla, ATO and MTS processes as well as the time required (in hours) to compute these numbers. We find that a greedy sequential heuristic provides effective solutions and can solve the industry size problem in reasonable time. In addition, we also find that the vanilla process significantly outperforms the current assemble-to-order process in all examples and the MTS in all but P15. It is to be noted that the vanilla boxes found by our heuristic may not be optimal and as a result, the performance of the optimal vanilla assembly could be even better.

4. A Special Case, a Variant, and Other Extensions of the Basic Model

In this section we discuss a special case and a variant and show that under certain conditions a greedy solution is optimal for the second stage recourse problem. We also consider various extensions of the basic model.

4.1. Product Line Specific Vanilla Boxes

It is often easier to select vanilla boxes when each vanilla box uniquely caters to a specific segment in the product line. In this section, we consider a special case where each product can be made either from a unique vanilla box or from raw components. Let l(k) denote the set of products assembled from the kth vanilla box. k[i] is the index of the vanilla box to be used for product i and k[i] = 0 implies that the product is assembled from raw components. p_i is the probability of occurrence of the lth demand scenario. The problem can be formulated as follows:

\[ P'(q, U): \min_{s_1, \ldots, s_N} \sum_{l=1}^{L} p_l \left( \sum_{i=1}^{N} (\pi_i (\xi - s_{\hat{a}[i]}) \right) \]

\[ + \sum_{k=1}^{K} \left( b_k (q_k - \sum_{i \in l(k)} s_{\hat{a}[i]}) \right) \]

s.t. for \( l = 1, \ldots, L \)

\[ \sum_{i=1}^{N} t_{\hat{a}[i]} s_{\hat{a}[i]} \leq C, \quad (1) \]

\[ \sum_{i \in l(k)} s_{\hat{a}[i]} = q_k, \quad (2) \]

\[ s_{\hat{a}[i]} = \xi, \quad \forall i, \quad (3) \]

\[ s_{\hat{a}[i]}, q_k \in R_+. \quad (4) \]

The above problem is a continuous knapsack problem with simple and generalized upper bounds in each scenario and can be solved using a greedy rule. The greedy

---

2 Note that some vanilla boxes need not be considered as they do not support any product. In one example, only about 500 out of 1023 vanilla box possibilities had to be considered. Since K = 3, the number of configurations evaluated are about 1500. The exact number of vanilla boxes to be evaluated depends on the BOM.
Table 6  Performance of Sequential Heuristic on Industry Size Problems

<table>
<thead>
<tr>
<th></th>
<th>Vanilla Process</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time Taken</td>
<td>K = 1</td>
<td>K = 2</td>
<td>K = 3</td>
<td>Assemble-to-Order</td>
</tr>
<tr>
<td>P11</td>
<td>8.51</td>
<td>109525</td>
<td>59114.33</td>
<td>26958.32</td>
<td>180485.5</td>
</tr>
<tr>
<td>P12</td>
<td>4.68</td>
<td>76214.60</td>
<td>23962.81</td>
<td>2041.42</td>
<td>144485.53</td>
</tr>
<tr>
<td>P13</td>
<td>6.06</td>
<td>54279.21</td>
<td>6591.45</td>
<td>114.19</td>
<td>137682.34</td>
</tr>
<tr>
<td>P14</td>
<td>5.15</td>
<td>88486.10</td>
<td>28215.86</td>
<td>3562.71</td>
<td>171986.05</td>
</tr>
<tr>
<td>P15</td>
<td>6.13</td>
<td>191343.25</td>
<td>134830.52</td>
<td>98155.55</td>
<td>274825.20</td>
</tr>
</tbody>
</table>

4.2. Cost of Substitution

In this section we consider a variant of the basic model where capacity and speed-up restrictions are removed and costs are assigned for choosing a particular vanilla box for a product (based on its distance from the product). The problem can be formulated as (VP1):

\[
\text{VP1}(\mathbf{q}, \mathbf{U}): \min \sum_{i=1}^{L} \sum_{j=1}^{N} p_i \left( \sum_{k=1}^{K} \left( \pi_i \left( \xi_{il} - \sum_{k=0}^{K} s_{ikl} \right) \right) \right) + \sum_{k=1}^{K} \left( h_k \left( q_k - \sum_{i=1}^{N} s_{ikl} \right) \right) + \sum_{i=1}^{N} c_{ik} s_{ikl} \\
\text{s.t. for } l = 1, \ldots, L \\
\sum_{i=1}^{N} s_{ikl} \leq q_k \quad \forall k, \quad (1) \\
\sum_{k=0}^{K} s_{ikl} \leq \xi_{il} \quad \forall i, \quad (2) \\
s_{ikl}, q_i \in \mathbb{R}_+, \quad (4)
\]

where \( c_{ik} \) is the cost of using the vanilla box \( k \) for product \( i \) and \( s_{ikl} \) is quantity of product \( i \) from vanilla box \( k \) in demand scenario \( l \). \( c_{ik} \) can be set to infinity if the vanilla box cannot be used for that product.

4.3. Other Extensions

In this section we discuss other extensions of the basic model and describe how the methodology in this paper could be extended to handle them.

1. Redundancy of Components. In the basic model we allow products to be assembled from vanilla boxes which have a subset of the components required for the product. However, it might be a good strategy under certain conditions to provide the customer with redundant components instead of losing the demand. Our model can be extended to address this redundancy by introducing a cost \( c_{ik} \) that is incurred when vanilla box \( k \) is used in product \( i \). This could reflect the cost of additional components that are present in the vanilla box not required in product \( i \). The time to assemble product \( i \) from vanilla box \( k \), \( t_{ik} \), should reflect the speed-up obtained only based on the required components for product \( i \) that are present in vanilla box \( k \). The objective changes to the following:

\[
Q(\mathbf{q}, \mathbf{U}, \xi) = \min \sum_{i=1}^{N} \left( \pi_i \left( \xi_i - \sum_{k=0}^{K} s_{ik} \right) \right) + \sum_{k=1}^{K} h_k \left( q_k - \sum_{i=1}^{N} s_{ik} \right) + \sum_{i=1}^{N} c_{ik} s_{ik}.
\]

2. Substitution among Products. In the basic model we assumed that there is no scope for substitution among products. We can model substitution by introducing a cost for interchanging products. The solution procedure will still be valid. However, now the model would have to be redefined by introducing (a) new variables \( r_{ij} \) which represent the amount of product \( i \)'s demand that is satisfied by product \( j \), and (b) costs \( c_{ij} \) which represent the per-unit cost of such a substitution. The objective function becomes:

\[
Q(\mathbf{q}, \mathbf{U}, \xi) = \min \sum_{i=1}^{N} \left( \pi_i \left( \xi_i - \sum_{j=1}^{N} r_{ij} \right) \right) + \sum_{k=1}^{K} \left( h_k \left( q_k - \sum_{i=1}^{N} s_{ik} \right) \right) + \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} r_{ij}.
\]
and the following inventory constraints complete the model:

$$\sum_{j=1}^{N} r_{ij} = \sum_{k=0}^{K} s_{ik} \quad \forall i.$$ 

The variable $c_{ij} = 0$ for $i = j$ because there is no cost for self-substitution. $c_{ij}$ captures all the different kinds of substitution that are possible. For example, some of the special cases are (i) $c_{ij} = \infty$ for $i \geq j$ represents full downward substitution, (ii) $c_{ij} = 0 \forall i, j$ represents a single product problem, (iii) $c_{ij} = \infty \forall i \neq j$ reduces to the basic model.

3. Raw Component Inventory. We neglected raw component inventory (i.e., inventory of components not in the vanilla box) in our basic model. We can model the optimization of raw component inventory in the following manner. In $P1 (U)$, we introduce additional variables $b_{ijk}$, $h_i'$, and $r_j$, $b_{ijk}$ represents the amount of component $j$ required when product $i$ is assembled from vanilla box $k$. So, $b_{ijk} = 0$ if $a_{ij} = 0$ or $(x_{ik} = 1$ and $n_{kj} = 1)$. $h_i'$ is the per-unit holding cost and $r_j$ is the amount of inventory for component $j$. We need to modify the objective function and add the following inventory constraints:

$$Q(q, U, \xi) = \min_{s} \sum_{i=1}^{N} \left( \pi_i \left( \xi_i - \sum_{k=0}^{K} s_{ik} \right) \right)$$

$$+ \sum_{k=1}^{K} \left( h_k \left( q_k - \sum_{i=1}^{N} s_{ik} \right) \right)$$

$$+ \sum_{i=1}^{N} \left( r_j \left( 1 - \sum_{i=1}^{N} (a_{ij} s_{ij} + \sum_{k=1}^{K} b_{ijk} x_{ik} s_{ik}) \right) \right)$$

$$\sum_{i=1}^{N} (a_{ij} s_{ij} + \sum_{k=1}^{K} b_{ijk} s_{ik}) \leq r_j \quad \forall j.$$ 

The introduction of the $r_j$’s and the optimization over them is likely to reduce the speed of our solution procedure.

4. Substitution among Components. In this model we consider cases where there may be more than one grade of a particular component that could be used in a product. For example, a 40 MHz chip could be replaced by a 60 MHz chip. In such cases, a vanilla box could possibly be used for “lower” grade products if it contains higher grade components. We can model such a scenario by assigning a cost for using a “higher” grade vanilla box $k$ for product $i$ given by $c_{ik}$. The set of products that can be made from the vanilla box $PC(U_i)$ is all the products that contain a subset of components present in $U_i$. The speed-ups due to the vanilla boxes are computed as before. The objective function is changed to:

$$Q(q, U, \xi) = \min_{s} \sum_{i=1}^{N} \left( \pi_i \left( \xi_i - \sum_{k=0}^{K} s_{ik} \right) \right)$$

$$+ \sum_{k=1}^{K} \left( h_k \left( q_k - \sum_{i=1}^{N} s_{ik} \right) \right) + \sum_{k=1}^{K} \sum_{i=1}^{N} c_{ik} s_{ik}.$$ 

This model differs from the model described in item 3 above because the latter considers substitution only at the product level.

5. Kitting Problem. The kitting problem relates to choosing the optimum set of tools required by a service person facing uncertainty regarding what specific set of tools are needed on site; see Mamer and Shogan (1987). Our methodology can be extended to handle the kitting problem by considering field problems as products and tools as components. We are interested in finding the set of components to keep in a single vanilla box (with unit inventory) such that the expected cost of operations is minimized. The product demands are either zero or one and in any given scenario one or more product demands could be positive. The interesting aspect is that the inventory of the vanilla box does not get used up (same tools can be used for solving more than one problem on site). So, in that sense, the problem becomes simpler. We can solve the kitting problem by searching over sets of components and calculating the cost incurred by each of them using a scenario based approach.

6. A Practical Issue. We have neglected some design and manufacturing infeasibilities corresponding to a vanilla box. For example, a vanilla box containing a monitor and a hard drive is not very useful without a processor. We can incorporate these infeasibilities by keeping a list of such vanilla boxes in a taboo list and not considering them in the logical choices.\(^3\)

---

\(^3\) The authors are grateful to Professor Awi Federgruen, the associate editor, and two anonymous referees for their valuable comments and suggestions which have helped in improving this paper greatly. Many thanks to Alan Scheller-Wolf for his editorial help. Further, they want...
to thank one of the referees for help in developing a fair comparison measure between MTS and the vanilla assembly process. The authors thank Dr. Chae An (IBM) for introducing this problem to the first author during his stay at T. J. Watson Research Center, for providing them with the data, and for his continued support. The authors wish to thank Professor George Shanthikumar for providing information about the kitting problem and Professor Suresh Chand for his detailed comments on an earlier version of this paper.

References


