Multi-product Inventory Planning with Downward Substitution, Stochastic Demand and Setup Costs

Uday S. Rao † Jayashankar M. Swaminathan ‡ Jun Zhang *

CBA, University of Cincinnati, Cincinnati, OH-45221-0130 †

Kenan-Flagler Business School, University of North Carolina, Chapel Hill, NC-27599 ‡

*A.B. Freeman School of Business, Tulane University, New Orleans, LA 70118

March 2001, Revised January, June, July 2002

Abstract

In this paper we consider a single period multi-product inventory problem with stochastic demand, setup cost for production, and one-way product substitution in the downward direction. We model the problem as a two stage integer stochastic program with recourse where the first stage variables determine which products to produce and how much, and the second stage variables determine how the products are allocated to satisfy the realized demand. We exploit structural properties of the model and utilize a combination of optimization techniques including network flow, dynamic programming, and simulation based optimization to develop effective heuristics.

Through a computational study, we evaluate the performance of our heuristics by comparison with the corresponding optimal solution obtained from a large scale MILP. The computational study indicates that our solution methodology can be very effective (98.8% on average) and can handle industry-size problems efficiently. We also provide several new qualitative insights on issues such as the effect of demand variance and cost parameters on the optimal number of products setup, the amount produced or inventoried, and the benefits of allowing substitution.

Keywords: Multi-product inventory, Stochastic demand, Setup cost, Substitution, Product variety, Computational insights.

1 Introduction

In this paper, we consider a single period, multi-product, stochastic inventory problem with substitution and setup costs. We allow a one-way downward substitution structure in that demand for product j may be satisfied only by using stock of those products i with $i \leq j$. This downward substitution structure occurs in several practical settings such as semiconductor chips (see Hsu and Bassok 1999) where a faster processor can be substituted for a slower processor, memory chips (Leachman 1987) and in the steel industry (Wagner and Whitin 1958). Our motivation for studying this problem came from alternative modes of customization tried at IBM. Swaminathan and Tayur (1998) considered one of the alternatives which involved storing semi-finished inventory called *vanilla boxes* and then customizing the product after receiving the order. Another approach involved storing inventory of *cadillac boxes* which contain all (or most of) the features and then removing features (or giving them free) based on the actual demand realized. This latter approach corresponds to a multi-product inventory problem with downward substitution and setup costs under stochastic demand. The setup costs represent product-specific tooling and equipment costs for production or testing and managerial effort for handling the increased product variety.

In this paper, we present a model, properties, and an effective solution methodology that exploits the problem structure and utilizes a combination of optimization techniques including network flows, dynamic programming and infinitesimal perturbation analysis. We also obtain qualitative and managerially relevant insights through a computational study. We focus on a single period problem corresponding to short (fast) cycle products. However, our approach may be applied to a multi-period problem in which the product set selection decision is made once at the beginning and can not be revised later. In our model, there are N products and each product has a continuous stochastic nonnegative demand with finite mean. Costs include setups, production, overage, stockout, and substitution. There are three sets of decisions:

D1: Which products to produce,

D2: How much to produce, and

D3: How to allocate products to satisfy realized demand.

Different versions of the substitution problem have been studied in the past (see literature review in §2). However, to the best of our knowledge, none of the earlier work has considered multiple (more than two) products, stochastic demand, setup costs and product substitution in an integrated model. As shown in Herer and Rashit (1997), even for a two product problem with setup costs, it is not easy to characterize the optimal regions of initial inventory values for which no products, both products, or only one product is produced. Hence, we focus on developing fast and effective heuristic solution methodologies for the multi-product problem.

We model this problem as a two stage integer stochastic program with recourse. The first stage decisions, corresponding to *D1* and *D2*, are made before demand is realized, while the second stage allocation decisions, *D3*, are made after demand materializes. For the first part of the problem, *D1*, we develop two heuristics, based on a stochastic demand version of the Wagner-Whitin algorithm, to determine the set of items to produce. We provide an optimal solution procedure for the second part, *D2*, through simulation based optimization using infinitesimal perturbation analysis (IPA). We formulate the third part, *D3*, as a network flow problem and present a single-pass greedy algorithm to solve certain instances of this problem. For small problem instances with discrete demand scenarios, we obtain an optimal solution for the entire problem using a large scale MILP formulation.

Through detailed computational testing, we demonstrate that our algorithms find near optimal solutions (on average, about 1.2% from optimal across a wide range of over 2000 problem instances). Further, we present qualitative managerial insights on various issues including (i) the effect of demand variance and cost parameters on the optimal number of products setup and the amount produced / inventoried; (ii) the benefits of allowing substitution; (iii) the complicating effects of having starting inventory in the system. Among other results we find that (1) the major portion of cost benefits due to flexibility of substitution result from considering substitution while selecting the set of items to produce and their production quantities – these benefits are more striking when the substitution costs are lower and demand uncertainty is higher; (2) although substitution provides greater flexibility and cost benefits, the total inventory in a system with substitution may be larger than the corresponding inventory in a system without substitution; (3) with setup costs and substitution permitted, as demand variability increases, the total number of setups almost always remains the same or decreases and the total amount produced could increase or decrease. The reduction in the amount produced / inventoried contradicts the traditional intuition that greater demand variability would *increase* inventory.

The contributions of this paper are: (a) Incorporation of setup costs and more general substitution costs into related multi-product stochastic inventory models studied earlier; (b) Development of new and efficient solution heuristics which combine dynamic programming and simulation based optimization while exploiting the network flow structure of the allocation problem; (c) Detailed computational validation of our solution methods which indicate that these are very effective in terms of accuracy (98.8% on average) and time, and are capable of solving large scale problems; (d) Computational results on several new qualitative managerial insights.

The rest of the paper is organized as follows: In §2, we review the relevant literature. In §3, we present the model formulation and some structural properties. We discuss solution methodology in §4 and present computational results in §5. We discuss model extensions in §6 and conclude in §7.

2 Relevant Literature

Deterministic versions of the substitution problem has been studied by Chand et al. (1994) and Tripathy et al. (1999). In this section, we restrict attention to the stochastic version. The multi-product inventory problem with one-way substitution and zero setup costs was originally considered by Ignall and Veinott (1969) and more recently by Bassok et al. (1999), where they demonstrate that myopic base stock policies are optimal. Hsu and Bassok (1999) consider a single period problem with one input resulting in random yield of multiple, downward substitutable products. They show how the network structure of the problem can be used to devise an efficient algorithm. Our model is different in that the unit substitution cost need not be identical as in Bassok et al. (1999), and, although we do not consider random yield, we allow setting up multiple products with associated setup costs as compared to a single product with no setup in Hsu and Bassok (1999). Further, our paper focuses on developing fast and efficient algorithms which we use in computational testing.

Two-product problems with zero setup cost and two-way probabilistic substitution have been well studied. McGillivray and Silver (1978) consider a case where products have identical costs and there is a fixed probability that a customer demand for a stocked out product can be substituted by another available product. For the case where the substitution probability is one, Pasternack and Drezner (1991) compare the optimal stocking levels to the corresponding inventory levels without substitution. For problems without setup costs, convex analysis and dynamic programming may be used to demonstrate optimality of base stock policies. Incorporating setup costs results in several difficulties, since the cost function is no longer convex. As a result, the multi-product inventory problem with setup costs, substitution and random demand considered in this paper has not been adequately studied before. Herer and Rashit (1997) have shown that even for the two-product, one period special case, the optimal ordering policy is complex and cannot be characterized using extensions of the single product (s,S) policy. This suggests that the optimal solution to the multi-product problem is also likely to be difficult to characterize, which necessitates use of effective heuristics.

There are several other research papers which are related to our problem − *Transshipment Problem*: Robinson (1990) and Herer, Tzur, and Yucesan (2000); *Assortment problem*: Pentico (1974) and Pentico (1988); *Component commonality problem*: Gerchak et al. (1988) and Henig and Gerchak (1989); *Continuous review inventory system with substitution*: Lee (1987); *Plant level flexibility problem*: Jordan and Graves (1995); *Vanilla box problem*: Swaminathan and Tayur (1998); *Random yield and substitution*: Bitran and Dasu (1992), Nahmias and Moinzadeh (1997); *Product pricing, customer preferences and substitution*: Agrawal and Smith (2000) and Sen and Zhang (1999); *Hotel yield management problems*: Queyranne and Fill (2000).

3 Problem Formulation and Analysis

In this section we present our model and assumptions in greater detail and discuss properties of the optimal solution.

3.1 Assumptions and Stochastic Programming Formulation

The sequence of events in our model is as follows: First, the decision maker selects which products to produce and how much. We assume that these products become available before demand is realized. Next, demand is observed, a substitution decision is made and existing inventory of different products is used to satisfy each product's demand. Any unsatisfied demand is lost. Then, production costs, inventory holding costs, shortage penalty costs and substitution costs are incurred. Finally, all left-over inventory is salvaged at a discounted price. We use the following notation:

Decision Variables:

$X = (x_1, x_2, ..., x_N)$: initial inventory position at the beginning of the period;

$Y = (y_1, y_2, ..., y_N)$: inventory position after production;

$z = (z_1, z_2, ..., z_N)$: $z_j = 1$ if product j is produced, 0 otherwise;

w_{ij} : amount of substitution of product i to j, where $i \leq j$;

u_j^+ : left-over inventory for product j (at end of period);

u_j^- : shortage for product j;

Data / Problem Parameters:

$\xi = (\xi_1, \xi_2, ..., \xi_N)$: random vector representing product demand;

$F(\cdot)$: cumulative probability distribution function of ξ.

$K = (K_1, K_2, ..., K_N)$: setup cost;

$C = (c_1, c_2, ..., c_N)$: unit variable production cost;

$H = (h_1, h_2, ..., h_N)$: unit overage cost (= holding cost $-$ salvage value);

$P = (p_1, p_2, ..., p_N)$: unit shortage penalty cost;

s_{ij} : unit substitution cost of using product i to satisfy product j demand, for $1 \leq i \leq j \leq N$, with $s_{ii} = 0$ and $s_{ij} = \infty$ for $i > j$.

One of the distinguishing features of our model is the presence of a setup cost, K, for producing products. Clearly, setup costs characterize several real environments as discussed in the introduction. We assume that total substitution costs are linear (i.e., proportional to the amount of substitution from i to j). One could view the per unit substitution costs as (i) the cost of removing components (or removing features via software switches) from the product before substituting it or (ii) the lost goodwill or reduced revenue when i is substituted for j. This is consistent with assumptions made in previous literature on substitution models. As in Hsu and Bassok (1999), the overage cost, H, can be negative when the unit salvage value exceeds the per period holding cost. We assume that demand is unaffected by substitutions, because customers often have little or no knowledge of substitutions by the manufacturer (and may only be happy to get a better product).

As is common in the inventory literature, we assume that $c_j + h_j > 0$, $p_j > 0$, for $j = 1, ..., N$. If this assumption is violated, then it is trivially optimal to produce either nothing (when $p_j \leq 0$) or an infinite amount of product j (when $c_j + h_j \leq 0$). In addition, we make the following assumptions on cost parameters:

A1. *Substitution Feasibility*: (i) $h_i \leq h_j + s_{ij}$: cost to hold product i is no more than cost to convert it to $j > i$ and hold j. (ii) $p_j \leq p_i + s_{ij}$: it is cheaper to incur a shortage of j than to satisfy product j demand by converting i to j and incurring a shortage of i.

A2. *Substitution Cost-Effectiveness*: $s_{ij} \leq h_i + p_j$, for *some* $i < j$. This assumption ensures that substitution is profitable for at least one pair of products (here i and j), otherwise, the problem reduces to a model without any substitution.

In addition, in our computational testing we restrict attention to cases where the triangle inequality, $0 \leq s_{ik} \leq s_{ij} + s_{jk}$, $1 \leq i \leq j \leq k \leq N$, holds. In practical situations we would expect the cost of a direct substitution from i to k to be less than the cost of two or more substitutions that generate the same result.

Given a vector of initial inventory, X, and a sufficiently large constant, M, we formulate the problem as a two stage stochastic program with recourse.

$$U^*(X) \equiv \min_{y_i, z_i} \left\{ U(X,Y,Z) = \sum_{i=1}^{N} c_i(y_i - x_i) + \sum_{i=1}^{N} K_i z_i + \int_{\xi} L(Y,\xi) dF(\xi) \right\} \tag{1}$$

subject to

$$0 \leq y_i - x_i \leq M z_i; \qquad z_i = 0, 1; \qquad \text{for all } i, \tag{2}$$

where (*the recourse problem*),

$$L(Y,\xi) = \min_{w_{ij}, u_i^+, u_i^- \geq 0} \sum_{i=1}^{N} \left(h_i u_i^+ + p_i u_i^- + \sum_{j=i}^{N} s_{ij} w_{ij} \right) \tag{3}$$

subject to

$$\textit{(I)} \ \sum_{i=1}^{j} w_{ij} + u_j^- = \xi_j, \qquad \text{and} \qquad \textit{(II)} \ \sum_{k=j}^{N} w_{jk} + u_j^+ = y_j, \qquad \text{for } j = 1, ..., N. \tag{4}$$

In the above formulation, $y_i - x_i$ determines the production quantity for item i and constraint (2) models the setup cost. In (3), $L(Y,\xi)$ represents the inventory overage, shortage and substitution cost given inventory position Y (after production) and demand realization ξ. Constraints (4) control the substitution amounts, w_{ij}, based on demand realization and the inventory level; here w_{ii} represents the amount of item i used to satisfy demand for i. Consistent with past research (Ignall and Veinott 1969 and Bassok et al. 1999), at the second stage, we assume that all the demand is known immediately. In reality this demand may occur in a dynamic fashion (customers coming one at a time, as at a car-rental agency), however, such a dynamic substitution model becomes extremely complex. Note that, it is easy to show that an expected profit maximization formulation (Parlar and Goyal 1984) of this problem can be converted to an equivalent expected cost minimization.

3.2 The Recourse Problem

The recourse problem, $L(Y,\xi)$, can be modeled as a network flow problem. The network consists of $2N+2$ nodes: Nodes j^I and j^{II} respectively represent the two constraints in (4) corresponding to product j for $j = 1, \ldots, N$; nodes $+$ and $-$ represent an inventory sink and a shortage source. There is demand outflow ξ_j at j^I and inventory inflow y_j at node j^{II}. Arcs $j^{II} \rightarrow +$ and $- \rightarrow j^I$ correspond respectively to flow of excess inventory and shortage.

For i, $j \le N$, arc $i^{II} \to j^I$ corresponds to a flow of w_{ij} units of product i, which represents substitution of i to satisfy demand for product j. Figure 1 illustrates this network for a four product problem. Due to this network structure, the recourse problem can be solved efficiently. Before elaborating on solution algorithms, we present a few characteristics of this problem.

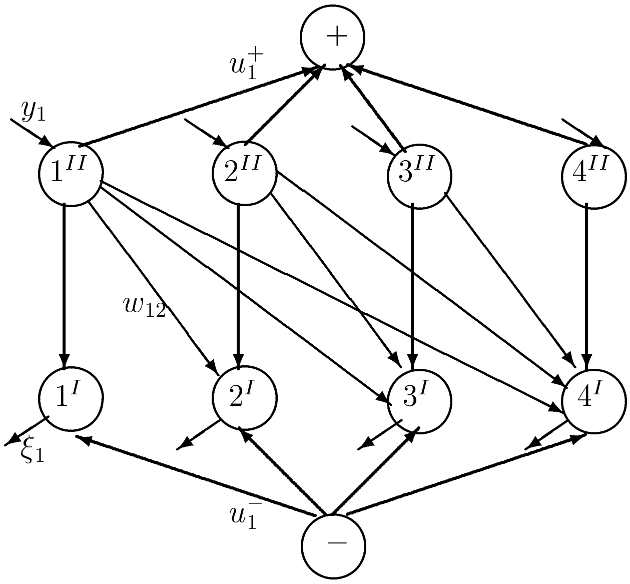

Figure 1: A typical network for $N = 4$

Proposition 1 *If s_{ij} can be written as $\alpha_i + \beta_j$, for all pairs i, j with $j > i$, then the recourse problem may be reformulated as an equivalent problem with zero substitution costs ($s'_{ij} = 0$) for all i, $j \ge i$ and overage and penalty costs, respectively, equal to $h'_i = h_i - \alpha_i$ and $p'_i = p_i - \beta_i$.*

Proof: Substituting $s_{ij} = \alpha_i + \beta_j$, the objective of the recourse problem becomes:

$$\begin{aligned}
& \textstyle\sum_{i=1}^N \sum_{j=i}^N (\alpha_i + \beta_j) w_{ij} + \sum_{i=1}^N h_i u_i^+ + \sum_{j=1}^N p_j u_j^- \\
= \ & \textstyle\sum_{i=1}^N \alpha_i (\sum_{j=i}^N w_{ij}) + \sum_{j=1}^N \beta_j (\sum_{i=1}^j w_{ij}) + \sum_{i=1}^N h_i u_i^+ + \sum_{j=1}^N p_j u_j^- \\
= \ & \textstyle\sum_{i=1}^N \alpha_i (-u_i^+ + y_i) + \sum_{j=1}^N \beta_j (\xi_j - u_j^-) + \sum_{i=1}^N h_i u_i^+ + \sum_{j=1}^N p_j u_j^- \\
= \ & \textstyle\sum_{i=1}^N \alpha_i y_i + \sum_{j=1}^N \beta_j \xi_j + \sum_{i=1}^N (h_i - \alpha_i) u_i^+ + \sum_{j=1}^N (p_j - \beta_j) u_j^- . \qquad (5)
\end{aligned}$$

where the second equality follows from constraints (4). Since the first two terms in expression (5) are constants in the recourse problem, substituting the definitions of h'_i and p'_j into

(5) yields the required result. □

Note that even though the substitution costs may be zero, selection of the set of items to produce will affect $L(Y, \xi)$, through the node costs h'_i and p'_i. By Proposition 2 below, an efficient "greedy" algorithm solves certain instances of the recourse problem.

Proposition 2 *Under the conditions in Proposition 1, the recourse problem is solved to optimality by the single-pass, greedy algorithm in Figure 2 when the cost parameters satisfy* $h'_1 \leq h'_2 \leq ... \leq h'_n$*, and* $p'_1 \geq p'_2 \geq ... \geq p'_n$*.*

Proof: See Appendix. □

Step 1 {Satisfy the demand for each product using its own inventory}

For $i = 1$ to N

$w_{ii} = \min\{\xi_i, y_i\}$; $u_i^+ = \max\{-w_{ii} + y_i, 0\}$; $u_i^- = \max\{\xi_i - w_{ii}, 0\}$;

if $(u_i^+ > 0)$: $\theta_i = \delta_i = -h'_i$; if $(u_i^- > 0)$: $\theta_i = \delta_i = p'_i$;

$F = \{\tau_1 = \{1\}, \tau_2 = \{2\}, ..., \tau_N = \{N\}\}$, where F is a forest of trees, τ_j.

Step 2 {Determine substitutions}

For $j = 1$ to N

While $(u_j^- > 0)$

for $i = j - 1$ down to 1

if $u_i^+ = 0$ or $-h'_i > p'_j$ go to next i; else

if $u_i^+ < u_j^-$ { $w_{ij} = u_i^+$; $u_i^+ = 0.0$; $u_j^- = u_j^- - w_{ij}$;

$\tau_j = \tau_j \cup \tau_i, F = F \backslash \tau_i$; for all k in τ_j, $\theta_k = \delta_k = p'_j$ }.

if $u_i^+ \geq u_j^-$ { $w_{ij} = u_j^-$; $u_j^- = 0.0$; $u_i^+ = u_i^+ - w_{ij}$;

$\tau_i = \tau_i \cup \tau_j$; $F = F \backslash \tau_j$; for all k in τ_i, $\theta_k = \delta_k = -h'_i$ }.

Figure 2: Single-pass greedy algorithm for recourse problem when $s_{ij} = \alpha_i + \beta_j$

Corollary 3 *When* $s_{ij} = T(j - i)$ *or* $s_{ij} = T(c_i - c_j)$*, where* T *is a constant, the recourse problem can be solved to optimality using a single-pass greedy algorithm.*

Given the above results, a reasonably large class of general substitution problems are equivalent in structure to problem instances with zero substitution costs. Since the unit substitution cost, s_{ij}, includes costs of adding or removing components or features and the lost-goodwill cost, often, $s_{ij} \approx f(c_j) - g(c_i) + c^{\mathcal{G}}$, where $f(\cdot)$ and $g(\cdot)$ are functions and $c^{\mathcal{G}}$ represents loss of goodwill costs. Thus s_{ij} can be of the form $\beta_j + \alpha_i$. The case of general substitution costs may be solved using fast min-cost network flow algorithms or a standard Linear Programming code such as CPLEX.

The speed of the algorithm for solving the recourse problem is critical because this problem must be solved repeatedly for different demand scenarios, ξ, and different inventory positions, Y. This requirement also motivated Bassok et al. (1999) and Hsu and Bassok (1999) to develop greedy heuristics for their problems. The algorithm in Figure 2 is different from the one in Hsu and Bassok (1999) in that our algorithm requires only a single pass, whereas theirs is a two pass algorithm. To empirically assess the reduction in run-time from using a single-pass approach vs. the one in Hsu and Bassok 1999), we implemented both approaches. We generated 2304 problem instances with five products (§5), each of which were run for several values of demand scenarios. Based on this test set, we observed that run-time of two-pass ÷ run-time of single-pass was, on average 1.58.

3.3 Properties of the Cost Function

The following results will be used by our solution procedures (described in Section 4). Proposition 4 is used in the gradient search for the optimal Y, given Z; Proposition 5 validates the IPA gradient estimate; and Proposition 6 reduces the search space for the optimal set of items to produce.

Lemma 4 *$L(Y, \xi)$ is convex and piecewise linear in Y for each ξ.*

Proof: The second stage recourse problem is a linear program for each ξ with Y as the right hand sides of the constraints. Hence, $L(Y, \xi)$ is convex and piecewise linear in Y. □

Proposition 5 *$G(Y) \equiv CY + E_\xi[L(Y, \xi)] = CY + \int_\xi L(Y, \xi) dF(\xi)$ is convex and continuously differentiable over Y values for which $G(Y)$ is finite.*

Proof: See Appendix. □

Proposition 6 *Let $S = argmin_Y G(Y)$, and let $[1], [2], \ldots, [N]$ denote the products listed in increasing order of setup costs (i.e., $K_{[i]} \leq K_{[j]}$ for $i < j$). Let $K(i) = \sum_{j=[1]}^{[i]} K_j$. Then, if initial inventory position, X, satisfies $G(X) < G(S) + K(i)$, for some i, the optimal number of products to produce is less than i. In particular, if $K_{min} = \min_j K_j$ and $G(X) \leq G(S) + K_{min}$, it is optimal to produce nothing.*

Proof: By contradiction: From (1), $U(Y, X, Z) = G(Y) - CX + \sum_{j=1}^{N} K_j z_j$. If we produce i or more products, then total costs $\min_Y U(Y, X, Z) \geq G(S) - CX + K(i)$; if we produce nothing, total costs will be $G(X) - CX$. By the condition in the proposition, $G(X) - CX \leq G(S) - CX + K(i) \leq \min_Y U(Y, X, Z)$. Consequently, not producing incurs lower cost than producing i or more products. □

4 Solution Procedure

In this section, we first present a large scale MILP formulation to find the optimal solution when demand can be represented by discrete scenarios. Subsequently, we present our heuristic solution procedure.

4.1 A Large Scale MILP Formulation

One way of optimally solving an approximate model is to generate (using standard Monte Carlo methods) only m representative scenarios of demand, ξ^ℓ, $\ell = 1, \ldots, m$, with scenario l having probability π^ℓ. This leads to the following formulation:

$$\min_{y_i, z_i, w_{ij}^\ell, u_i^{\ell+}, u_i^{\ell-} \geq 0} \left\{ \sum_{i=1}^{N} c_i(y_i - x_i) + \sum_{i=1}^{N} K_i z_i + \sum_{\ell=1}^{m} \pi^\ell \sum_{i=1}^{N} \left(h_i u_i^{\ell+} + p_i u_i^{\ell-} + \sum_{j=i}^{N} s_{ij} w_{ij}^\ell \right) \right\}$$

$$0 \leq y_i - x_i \leq M z_i; \qquad z_i = 0, 1; \qquad \text{for all } i,$$

$$\sum_{i=1}^{j} w_{ij}^\ell + u_j^{\ell-} = \xi_j^\ell, \text{ and } \sum_{k=j}^{N} w_{jk}^\ell + u_j^{\ell+} = y_j, \quad \text{for } j = 1, ..., N, \quad \text{for all } \ell.$$

This formulation consists of a large number of decision variables (directly proportional to the number of demand scenarios considered), as a result its optimization could be time intensive. We conducted a preliminary analysis to test the robustness of the final solution with respect to the number of demand scenarios. Based on our analysis, we decided to use 500 scenarios (demand vectors) in all our computational experiments. In §5, we report on optimal costs and run times obtained from solving the above MILP using CPLEX. Since the same 500 demand scenarios were used in the simulation based optimization to determine production quantities, we are able to compare the MILP optimal to the expected cost estimates from our heuristics (described below).

4.2 Heuristic Algorithms

Our solution procedure decomposes the problem into three parts- *D1*: Determining which products to produce; *D2*: Determining the optimal production quantities for the products given the decisions in *D1*; and *D3*: Allocating products to satisfy realized demand by solving the recourse problem (§3.2). We solve *D3* using the greedy algorithm from Figure 2. For simplicity, we restrict attention only to instances for which *D3* is optimally solved by a greedy approach, consequently, all our computational results are restricted to these instances. We do so because the main focus of this paper is on decisions *D1* and *D2*, the recourse problem *D3* must be solved repeatedly for different demand scenarios and hence we needed a quick solution approach.

Optimal Production Quantities, Given Items to Produce: In this section we focus on determining the optimal production quantities, given the set of items to produce:

Problem 1 *Given initial inventory,* $X = (x_1, ..., x_N)$*, and a subset* $\mathcal{S} \subset \{1, \ldots, N\}$*, find the optimal produce-up-to level* $Y \geq X$*, with* $y_j = x_j$ *for* $j \notin \mathcal{S}$*, to minimize the total expected costs* $G(Y) = CY + E[L(Y, \xi)]$.

By Proposition 5, finding the optimal Y is a convex minimization problem, which may be solved by gradient based search (Sherali, Bazaraa, and Shetty 1992). $G(Y)$ and its gradient is accurately and efficiently estimated using IPA (Glasserman 1991). In fact, the following IPA gradient estimate can be shown to be valid for our problem.

Proposition 7 *The IPA estimator for the (*right*) derivative of* $G(Y)$ *w.r.t.* y_j *is* c_j *plus the average value of the dual price of constraint 4(II) corresponding to product* j.

Remark 8 *When the greedy algorithm in Figure 2 is applicable, the dual price used in the IPA estimator in Proposition 7, is equal to* θ_j*, which is finite and can be stated as follows:*

$$\theta_j = \begin{cases} -h'_i & \text{if node } j \text{ belongs to tree } \tau_i \text{ and } u_i^- = 0, \\ p'_i & \text{if node } j \text{ belongs to tree } \tau_i \text{ and } u_i^- > 0. \end{cases}$$

Note: Trees $\{\tau_i\}$ *have been defined by the algorithm in Figure 2.*

<u>**Items to Produce:**</u> To determine which products to produce, one could enumerate over combinations of products. Although this procedure guarantees an optimal solution, it can be computationally intensive (as is the MILP approach from §4.1). Hence, we develop two heuristics to determine which products to produce. In both heuristics, once we determine which products to produce, we solve Problem 1 using IPA to estimate optimal production quantities, given Z.

A Heuristic Based on Wagner-Whitin Algorithm (SWW): We call this new heuristic SWW because it is related to the Stochastic Wagner-Whitin problem (with uncertain demand and shortages permitted). The traditional Wagner-Whitin algorithm solves the single product, multi-period dynamic lot-sizing problem with deterministic demand and no shortages. The deterministic version of our problem is equivalent to dynamic lot-sizing with backlogging: The N products may be viewed as N time periods; using product i's inventory to satisfy product j's demand corresponds to holding the product in period i to satisfy demand in period j. Based on this observation, our algorithm will first determine the set of products to produce by solving a shortest path problem.

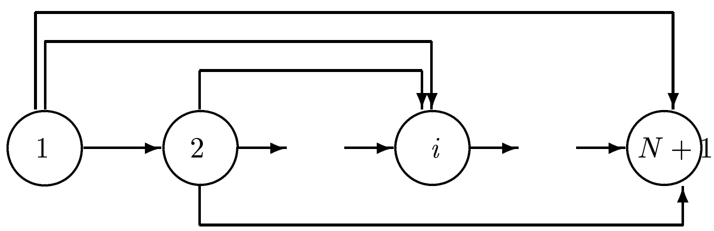

Figure 3: Shortest path network for SWW

The shortest path network for SWW, shown in Figure 3, consists of $(N+1)$ nodes representing the N products and a dummy product. Arc $i \to j$ corresponds to a (potential) setup of product i with production of i used to satisfy demand for products i through $j-1$. The length of this arc is the minimum expected cost,

$$\min_{y,z} \left\{ K_i z_i + c_i(y_i - x_i) + E\left[\min_{w,u} \sum_{l=i}^{j-1} \left(p_l u_l^- + h_l u_l^+ + \sum_{k=l}^{j-1} s_{lk} w_{lk} \right) \right] \right\},$$

$$s.t. \qquad (I) \sum_{k=i}^{l} w_{kl} + u_l^- = \xi_l, \text{ and } (II) \sum_{k=l}^{j-1} w_{lk} + u_l^+ = y_l, \text{ for } l = i, \ldots, j-1,$$
$$y_l = x_l \text{ for } l \neq i, \qquad 0 \leq y_i - x_i \leq M z_i.$$

The arcs in the shortest path from node 1 to node $N+1$ define the items to produce. Product i is produced iff, for some j, arc $i \to j$ is in the shortest path and the cost of this arc was based on $z_i = 1$. Note that, finding the shortest path from 1 to $N+1$ does not necessarily imply that we always produce product 1.

Since finding the shortest path in an acyclic network is easy, the bulk of the computational effort in SWW is in determining the length of each arc in the network. This is done by solving the following problem using an IPA approach similar to that for Problem 1.

Problem 2 *If the initial inventory position of products $i, ..., j-1$ is $(x_i, \ldots, x_{j-1})$, find the optimal produce-up-to level y_i for product i so as to minimize the expected cost incurred by products i through $j-1$ (assuming no inventory of products $k < i$ is available).*

To compute all the arc lengths, we need to solve $N(N+1)/2$ instances of Problem 2. Note that SWW is a heuristic because, while evaluating the length of arc $i \to j$, we ignore the fact that, as a recourse action, any excess inventory of upstream products may be used to satisfy the demand for products i through N and that any excess inventory of products i through $j-1$ may be used to satisfy demand for downstream products with labels greater than $j-1$.

A Heuristic Using Mean Demand Information (DWW): DWW determines the set of products to produce by solving a shortest path problem on the network in Figure 3 with arc lengths determined by assuming that each product's demand is deterministic and equal to its mean value. Computation of these arc lengths needs some care to incorporate issues that do not exist in the traditional Wagner-Whitin. For instance, while computing the arc cost for $i \to j$, there may be a node k, $i < k < j-1$, such that it is never optimal to convert i to products with label greater than k. In this case, the production quantity for i is just the demand for products i through k and products with label greater than k incur shortage penalty if their initial inventory is less than the mean demand. Thus, this computation of arc costs for DWW is akin to a Wagner-Whitin problem with perishability (so that beyond period k, it is uneconomical to hold inventory from period i).

We also considered a **critical fractile based heuristic (FWW)** in which the set of items to produce was determined as in DWW with demand specified by mean + z $\times$ (standard deviation), where z denotes the critical fractile corresponding to the critical ratio of $(p_i - c_i)/(p_i + h_i)$. We compared the performance of FWW with DWW and SWW on 1728 problem instances with five products. The average errors for the DWW, SWW and FWW heuristics were respectively 1.42%, 1.47% and 1.94%. Hence, in our computational experimentation (§5), we only present results for the SWW and DWW heuristics.

5 Computational Study

In this section, we first compare the performance of the heuristics to the optimal solution procedure both in terms of accuracy and running time. Next, we provide insights on the benefits of allowing product substitution when setup costs are incurred for item production. Finally we explore changes in inventory strategy and product setup under different operational conditions.

5.1 Experimental Setup

The tests are conducted on problem instances with N = 5, 10, 15, 20, 25 products. For each value of number of products, N, we varied the following parameters:

Unit Cost of Products, c_i: We chose the three sets of unit costs, with $c_i = 1.0 + \eta(N - i)$ for $\eta = 0.1$, 0.2 and 0.5, with higher η corresponding to greater breadth of the product line (in terms of features present).

Setup Costs, K_i: We chose four values for the setup costs: $10c_i$, $25c_i$, $50c_i$ and $100c_i$. These setup cost values were considered representative because the optimal set of items produced varied from only one item (for high K_i) to all N items (for low K_i).

Substitution Costs, s_{ij}: We set $s_{ij} = T(c_i - c_j)$, for $i < j$. (Experiments with $s_{ij} = T(j - i)$ yielded qualitatively similar results.) We chose three values for T: $T = 0.0$, $T = 0.5$, $T = 1.0$, corresponding to different substitution cost levels.

Overage and Penalty Costs, h_i & p_i: We chose values for overage costs corresponding to -65% and -70% of c_i. (Problem instances with non-negative overage costs of 0% and 15%, corresponding to low salvage values, were also tested with similar results.) For each value of h_i, the penalty costs were chosen so that the expected service level or "critical ratio", $(p_i - c_i)/(p_i + h_i)$, took on values 0.5, 0.7, 0.8, 0.85, 0.9, 0.95; only 0.8 and 0.85 were used for normal and uniform demand.

Demand: We modeled demand for each product using a finite normal distribution with mean 100, and coefficient of variation, denoted by cv, of 0.1 or 0.3. Experiments were also conducted for correlated-normal, gamma and uniform demand. For the gamma distribution, we considered cvs of 0.1, 0.3, 0.6, and 1.0.

For each value of N, the above set of parameters give rise to 288 problem instances for each of the normal and uniform demand cases and to 1728 problem instances for gamma demand. For simplicity, all our computational testing (except §5.4) was done with a starting inventory of zero for all products. Product demands were generated by sampling 500 values from the demand distribution. The initial step size for the gradient based search (IPA) was set to 1.0; if the objective function did not improve in 15 consecutive iterations, the step size was reduced by 25%. The search was terminated when the step size (which governs changes in solution value) became less than 0.01 or when the Euclidean norm of the gradient was less than 0.02.

In the following sections we will highlight key insights from our computational study. Note that the insights are specific to the problem parameters used in our study and there may be instances where they need not apply. However, with that caveat, we do find several interesting insights that were consistent across the set of parameters that we tested.

5.2 Performance of Heuristics

We define the relative error of a heuristic to be = (heuristic cost − optimal cost)/ (optimal cost). The run time of the heuristics is measured in CPU seconds on a Sun Sparcstation 10. Our computational results are summarized in tables 1 through 5 and in figures 4 and 5. In Table 1, we present the average, std.dev., and maximum relative error of the two heuristics compared to the optimal for the normal demand case. Across all the problem instances (normal, gamma, and uniform distribution for the five product case), the average error for SWW and DWW was, respectively, 1.20% and 1.23%. Thus both heuristics perform reasonably well on the tested instances. Table 2 reports the speed of the heuristics. We also ran the heuristic for larger problems with more products, without determining the optimal solution (which was computationally intensive). A comparison between costs of SWW and DWW for $N = 5, 10, 15, 20, 25$ is shown in Figure 4.

Performance in accuracy and speed: From Tables 1 and 2, we see that, *for normal demand, both heuristics are effective* with an average error of less than 1.1%. Further, *the savings in running time are large compared to the MILP solution approach*, particularly for the larger problem with ten products. The optimal solution was not computed for $N \geq 15$, because, in preliminary testing, the MILP could not determine a single feasible solution better than the heuristics even after seven hours.

Table 1: Performance of the two heuristics for normal demand (288 instances for each N)

	Error for $N = 5$ Products			Error for $N = 10$ Products		
	Average	Std.Dev.	Maximum	Average	Std.Dev.	Maximum
SWW	0.55%	0.67%	3.76%	1.03%	0.16%	4.27%
DWW	0.46%	0.53%	2.14%	0.94%	0.13%	3.63%
Randomized	19.92%	25.40%	141.10%	18.32%	4.77%	103.11%

We were somewhat surprised that DWW, a simple mean-demand based selection of the set of items to produce, could perform so well on many test cases. In order to test if the product setup selection significantly affected total expected costs, we compared our heuristics with alternative random product selection procedures which in expectation produced the same

Table 2: Average running time, in CPU seconds, for normal demand (288 instances for each N), NA = Not Applicable

N	5 Products	10 Products	15 Products	20 Products	25 Products
MILP	123.97	3351.49	NA	NA	NA
SWW	0.28	2.60	8.77	16.42	33.21
DWW	0.18	1.33	3.12	3.97	12.16

number of items as SWW (or DWW). Results in the Randomized row of Table 1 correspond to selecting the items to produce as follows: Let $\gamma = (\#$ of items produced by SWW$)/N$. For each product i, let γ_i be a generated uniform random variate between 0 and 1. Then Randomized produces i if $\gamma_i < \gamma$. As seen in Table 1, this Randomized performs several orders of magnitude worse than DWW and SWW. Therefore, we confirm that *both our heuristics effectively solve a non-trivial product setup selection problem. Further, we note that the choice of which products to produce may be made using expected value information as long as the stock levels are chosen taking the distribution of demand into consideration.*

Correlated demand: Table 3 presents our results for the five product instances with demand correlation coefficients of -0.2, 0.0 and $+0.2$, with $h_i = 0.0$ or $0.15c_i$. From the table we see that for the test problem instances, the *performance of the heuristic is not very sensitive to demand correlation.*

Table 3: Effects of demand correlation on heuristic performance for normal demand (144 instances for each Correlation)

	Negative Correlation		Zero Correlation		Positive Correlation	
	Avg. Error	Max. Error	Avg. Error	Max. Error	Avg. Error	Max. Error
SWW	1.91%	10.32%	1.95%	9.70%	1.92%	9.09%
DWW	0.83%	3.44%	1.05%	4.02%	1.19%	4.49%

Number of products, N: To investigate if the performance of DWW or SWW depended on the number of products, N, we generated 288 problem instances for each value of $N = 15$, 20 and 25 and computed the objective values U^*_{DWW} and U^*_{SWW}. Figure 4 plots the average

% cost difference, $100(U^*_{DWW} - U^*_{SWW})/U^*_{SWW}$. We note that *as the number of products increases, the average performance of SWW improves slightly relative to DWW.* While the magnitude of the % cost difference varies significantly with problem instance, the increasing trend in Figure 4 often held true. For instance, for $N = 15$ the % cost difference increased with N in 51% of instances, this increased to 66% for $N = 20$. The figure also demonstrates that *the performance of the heuristics shows a similar behavior when demand is uniform* with the same mean and variance parameters as the normal.

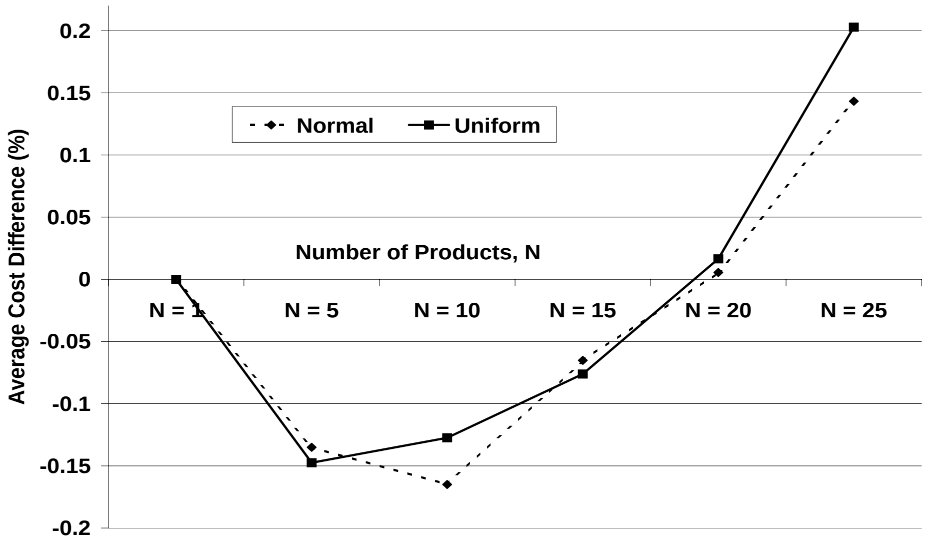

Figure 4: Cost comparison of SWW and DWW solutions (288 instances for each N)

Effects of demand variation: Table 4 shows the performance of the two heuristics under different coefficients of variation of demand (cv). We find that *the performance of both heuristics deteriorates at higher cv.* This may be explained as follows. In DWW, we only use the mean demand information, as a result, when cv increases the mean information becomes less valuable and the performance deteriorates. In SWW, the error comes from the leftover inventory of upstream products when we compute the length of the arc connecting node i and node j. This arc length is set equal to the optimal cost of producing product i to satisfy the demand for products i through $j - 1$, under the assumption that leftover inventory of upstream products can not be used to satisfy demand for products $k \geq j$.

On account of this assumption, SWW's arc lengths overestimate costs. With increase in demand variance the overestimated costs makes arc lengths more unreliable, as a result, the performance of SWW worsens.

Table 4: Effects of demand variation on heuristics' performance – average error, std.dev.– for normal demand (144 instances for each variance and N)

	$N = 5$ Products		$N = 10$ Products	
	Low Variance ($cv = 0.1$)	High Variance ($cv = 0.3$)	Low Variance ($cv = 0.1$)	High Variance ($cv = 0.3$)
SWW	0.23%, 0.20%	0.88%, 0.80%	0.48%, 0.12%	1.57%, 0.20%
DWW	0.23%, 0.19%	0.70%, 0.63%	0.48%, 0.07%	1.40%, 0.20%

To test the performance of heuristic for higher cv values (0.1, 0.3, 0.6, 1.0), we ran experiments for the five product problem instances using gamma distributed demand and data parameters identical to the zero correlation normal case. This corresponded to a total of 1728 problem instances, from which all our results pertaining to the case of gamma demand are based. We noted that, *the heuristics' remain effective for the more variable gamma demand* with (Avg. Error, Max. Error) respectively (1.42%, 13.81%) and (1.47%, 9.50%) for SWW and DWW. Averaging over the 432 instances for each $cv = 0.1, 0.3, 0.6, 1.0$, the Avg. Error of SWW was 0.7%, 1.8%, 2.0% and 1.3%; the Avg. Error of DWW was 0.4%, 1.2%, 2.0%, 2.4%. These errors suggest that *for gamma demand, SWW may be more competitive than DWW at higher cv* (≥ 1).

Effects of setup cost, K_i: From Figure 5, we note that *for instances with five products and normal demand, the relative error for SWW is smaller than the relative error of DWW when the setup cost, K_i, is large* ($K = 50c$ and $K = 100c$). As mentioned in the previous paragraphs, the shortest path used in SWW overestimates inventory and substitution costs. Consequently, the effect of this overestimation (which leads one to produce the wrong set of products) is amplified when the setup costs are low and many products are produced. Hence SWW's performance deteriorates with decrease in setup cost. Overall, a similar behavior was observed for the case of ten products and for different demand distributions.

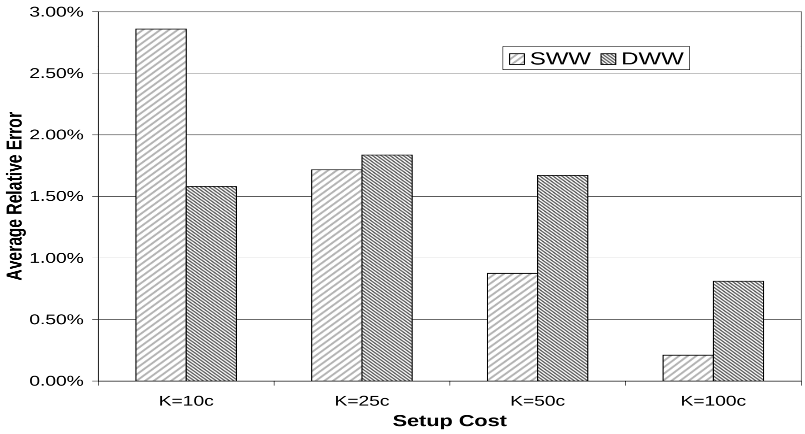

Figure 5: Effects of setup cost on heuristics' performance (5 products, normal demand, 72 instances for each K)

Other Effects: As seen in Table 5, the average relative error increases with the critical ratio, $(p_i - c_i)/(p_i + h_i)$, for gamma demand. Similarly, we noted that the relative errors increase as the salvage value decreases (even if the critical ratio remains unchanged). We also observed that the heuristics' performance does not change very significantly with changes in substitution costs.

Table 5: Performance of heuristics under different critical ratio for gamma demand (288 instances for each Critical Ratio)

Critical Ratio	0.5	0.7	0.8	0.85	0.9	0.95
SWW	0.39%	0.94%	1.41%	1.65%	1.91%	2.19%
DWW	1.28%	1.32%	1.51%	1.56%	1.59%	1.58%

5.3 Substitution Effects

In this section, we study the effects of considering substitution when setup costs are incurred for item production. First, we provide results which illustrate the cost benefits of substitution. Next we discuss the differences in inventory strategy when substitution is allowed. Finally, we investigate the effect of increase in variance on the number of products setup and their inventory levels.

Cost Benefits of Substitution: We compare total costs with substitution to the corresponding optimal costs when (i) substitution is not allowed (called *no substitution allowed*) and (ii) when substitution is allowed but is not taken into account when making production decisions (called *no substitution considered*). The cost of the *no substitution allowed* case is computed by independently finding the optimal (s_i, S_i) parameters for each of the products, which determine the respective production quantities (produce-up-to S_i, if item i's inventory is s_i or less). Given these production quantities, we evaluate its cost using Monte-Carlo simulation. For the case of *no substitution considered*, we compute the production quantities as above but allow substitution as a recourse action.

The relative difference from the MILP optimal cost is calculated as (cost of *no substitution allowed* − optimal cost)/(optimal cost); a similar relative difference measure is used for the case with *no substitution considered.* These performance metrics represent the potential percentage reduction in costs resulting from exploiting the substitution option. From the results in Table 6, we see that *considering the substitution option (in determining which items to setup and their production quantities) can result in substantial savings* (upto 46%). Further, when substition is not considered in selecting the products to produce and their production quantities, allowing substitution during recourse (*no substitution considered*) is, on average, only slightly (1 or 2%) better than not converting at all (*no substitution allowed*).

Table 6: Overall cost saving (576 problem instances)

	Optimal vs. *no substitution allowed*		Optimal vs. *no substitution considered*	
	Avg. Saving	Max. Saving	Avg. Saving	Max. Saving
5 Products	12.53%	46.60%	11.14%	44.22%
10 Products	15.54%	51.17%	13.75%	48.13%

Based on experiments with different values of demand cv, setup cost K, and substitution cost c_{ij}, we find that *the savings are greater when the substitution cost is small and the demand variance or setup cost is high.* This is intuitive because, when the demand variance or setup cost is high, there is greater potential for cost-effective substitution. Table 7 illustrates results for different combinations of substitution and setup costs.

Table 7: Average cost saving under different setup costs and substitution costs (288 instances with 5 products)

	Optimal vs. *no substitution allowed*				Optimal vs. *no substitution considered*			
	$K = 10$	$K = 25$	$K = 50$	$K = 100$	$K = 10$	$K = 25$	$K = 50$	$K = 100$
$T = 0$	3.26%	7.16%	16.18%	35.81%	1.50%	5.54%	14.69%	34.48%
$T = 0.5$	2.68%	5.38%	13.89%	27.96%	1.10%	3.92%	12.56%	26.71%
$T = 1.0$	2.36%	3.38%	9.62%	23.94%	0.94%	2.10%	8.48%	22.95%

Changes in Inventory Strategy: Popular intuition suggests that the substitution option provides greater flexibility and should hence result in lower total inventory levels when compared to the case with no substitution allowed. However, *with non-zero setup costs, optimal inventory levels may actually increase when substitution is permitted.* This observation was noted in other environments by Gerchak and Mossman (1992) and Henig and Gerchak (1989).

An intuitive explanation for the above phenomenon is that when substitution is not allowed, the presence of setup costs may make it optimal to not produce a product. On the other hand, when substitution is allowed, the benefits from producing a product in larger volumes and converting it into other products may be large, which could lead to larger total inventory. Although inventory is not reduced the costs can still be significantly reduced when substitution is allowed. Further, as expected, under downward substitution, the inventory of the first product does not decrease and the inventory of the last product does not increase as compared to the optimal solution when substitution is not allowed.

Demand Variance and Number of Products Setup: *As demand variance increases, we observe that, when substitution is allowed, the number of setups almost always remains the same or decreases.* For instance, the number of setups was nonincreasing in the demand variance over all the 1728 test cases of five-product problem instances with gamma demand and cv = 0.1, 0.3, 0.6 and 1.0. Comparing the number of setups for each cv with setups for the next higher cv, we noted that, of the 432 instances at each cv = 0.1, 0.3, and 0.6, the number of setups remained the same in 383, 330, and 320 instances while the number of setups decreased with cv in 49, 102, and 112 instances. Similar results were obtained for normal and uniform demand; however, the number of setups did increase with cv in a few cases.

This empirical observation is different from the usual risk pooling because in our case what decreases is not necessarily the total inventory level, but the number of products setup. The substitution option allows a certain form of risk pooling in that a subset of products can be produced to satisfy the demand for the whole set. On the other hand, there is a per unit substitution cost which goes against production of fewer products. When variance in demand is higher, the benefits of risk pooling are likely to be greater as compared to the cost incurred due to substitution and as a result, fewer products are produced.

5.4 Effects of Initial Inventory

We conducted some preliminary experiments with different initial inventory levels and found that *there is a complex relation between initial inventories and optimal production quantities.* It seems to be difficult to come up with general results on the effect of initial inventories in our problem environment, other than simple observations such as: If the initial inventory of an item i is greater than a threshold value, s_i, and the initial stocks of other items are bounded below, then item i will not be produced. We also noted that certain existing results on inventory problems with substitution do not always apply. For instance, Bassok et al. (1999) have shown that, when no setup cost is incurred for production, the order up to level for any item i is non-increasing in the initial inventory level of item $j \neq i$. With setup cost, the order up to level may increase with initial inventory.

6 Model Extensions / Variants

In this section we discusse two extensions: partial substitution and two-way substitution.

Partial Substitution: Here we show how the model developed in §3.1 can be extended to incorporate the case where the customer fraction that accepts substitutions is known and fixed at a value that may be less than 1.0. We consider two cases of partial substitution that have been observed in practice. In the first case, a deterministic portion f_j, $0 \leq f_j \leq 1$, of customers demanding product j will accept a converted product. To model this case, we add the following constraints for all products whose demand cannot be met by its own on-hand inventory (*i.e.* , $\xi_j > y_j$):

$$\sum_{i=j}^{N} w_{ij} \leq f_j(\xi_j - y_j). \tag{6}$$

In the second case, only a portion, f_{ij}, of the customers whose demand can not be satisfied by product j will buy a substitute product i, $i \leq j$. This corresponds to adding the constraints $w_{ij} \leq f_{ij}(\xi_j - y_j)$. Under both cases, with the additional constraints, the second stage recourse problem becomes a capacitated network flow problem. Hence fast network algorithms still apply to this extension of our model.

Two-way Substitution: Another model extension involves two-way substitution where product j may be used (i) to satisfy demand for product $k > j$ (downward substitution) or (ii) to satisfy demand for product $i < j$ (upward substitution or customization by adding features). The large scale MILP formulation from §4.1 can be easily modified to permit two-way substitution. Further, the SWW and DWW heuristics may also be tailored to the two-way substitution structure as follows: Let the length of arc i to k in SWW denote the expected total cost resulting from producing the optimal quantity of item j, $i \leq j < k$, to satisfy the demand for products i through $k-1$. Thus, this arc length is the optimal objective value of the following problem (solved using IPA):

$$\min_{i \leq j < k} \min_{y,z} \left\{ K_j z_j + c_j(y_j - x_j) + E\left[\min_{w,u} \sum_{l=i}^{k-1} \left(p_l u_l^- + h_l u_l^+ + \sum_{m=i}^{k-1} s_{lm} w_{lm} \right) \right] \right\},$$

subject to the constraints

$$\sum_{m=i}^{k} w_{lm} - \sum_{m=i}^{k} w_{ml} + u_l^- - u_l^+ = \xi_l - y_j, \quad \text{for } i \leq l < k,$$

$$y_l = x_l \quad \text{for } l \neq j, \qquad 0 \leq y_j - x_j \leq M z_j.$$

The optimal choice of j may be obtained by implicitly enumerating over all j between i and $k-1$. The corresponding optimal production level y_j is found by an IPA gradient based search. Given y_j, the recourse problem may be solved using CPLEX. The arcs in the shortest path from 1 to $N+1$ in the network would specify the products to setup. The actual production quantity for these items may subsequently be determined using IPA to compute the optimal production level vector (as in SWW).

7 Summary

In this paper, we have modeled the single period multi-product inventory system with downward substitution and setup costs as a two stage, mixed integer, stochastic program with recourse. Fast solution methodologies, that utilize the inherent structure of the problem and combine alternative optimization techniques, are developed. Over a wide range of parameter settings, our solutions are shown to be very effective (98.8%) on average. Through a computational study, we explored the total cost (and inventory) benefits of substitution and the effects of setup or substitution costs and demand variance.

Future work would consider (1) multi-period problems, (2) quick algorithms for the second stage recourse problem with general (two-way) substitution and their application within algorithms to select the set of products to produce, and (3) a study of efficient quasi Monte-Carlo techniques for generating demand vector scenarios. We feel that developing effective solutions to the multi-period problem with general substitution is likely to be challenging.

Acknowledgements: We thank Yigal Gerchak and two anonymous referees for several comments that helped improve the content and presentation of this paper. Research of the second author was supported in part by the NSF CAREER award # 9984252.

References

Agrawal, N. and S. A. Smith (2000). Management of multi-item retail inventory systems with demand substitution. *Opns. Res. 48*, 50–64.

Bassok, Y., R. Anupindi, and R. Akella (1999). Single period multiproduct inventory models with substitution. *Opns. Res. 47*, 632–642.

Bitran, G. and S. Dasu (1992). Ordering policies in an environment of stochastic yields and substitutable demands. *Opns. Res. 40*, 999–1017.

Chand, S., E. Ward, and Z. Weng (1994). A parts selection model with one-way substitution. *Eur. Jnl. of Opnl. Res. 73*, 65–69.

Gerchak, Y., M. J. Magazine, and A. B. Gamble (1988). Component commonality with service level requirements. *Mgmt. Sci. 34*, 753–761.

Gerchak, Y. and D. Mossman (1992). On the effect of randomness on inventories and costs. *Opns. Res. 40*, 804–807.

Glasserman, P. (1991). *Gradient Estimation via Perturbation Analysis*. Kluwer Academic Publishers.

Henig, M. and Y. Gerchak (1989). Component commonality in assemble-to-order systems: Models and properties. *Naval Res. Log. 36*, 61–68.

Herer, Y. and A. Rashit (1997). Lateral stock transshipments in a two-location inventory system with fixed and joint replenishment costs. *Naval Res. Log. 46*, 525–547.

Herer, Y., M. Tzur, and E. Yucesan (2000). The multi-location transshipment problem. Working Paper, INSEAD.

Hsu, A. and Y. Bassok (1999). Random yield and random demand in a production system with downward substitution. *Opns. Res. 47*, 277–290.

Ignall, E. and A. Veinott (1969). Optimality of myopic inventory policies for several substitute products. *Mgmt. Sci. 15*, 284–304.

Jordan, W. C. and S. C. Graves (1995). Principles on the benefits of manufacturing process flexibility. *Mgmt. Sci. 41*(4), 577–94.

Lee, H. (1987). A multi-echelon inventory model for repairable items with emergency lateral transshipments. *Mgmt. Sci. 33*, 1302–1316.

McGillivray, A. and E. Silver (1978). Some concepts for inventory control under substitutable demands. *INFOR 16*, 47–63.

Nahmias, S. and K. Moinzadeh (1997). Lot sizing with randomly graded yields. *Opns. Res. 45*, 974–986.

Parlar, M. and S. Goyal (1984). Optimal ordering decisions for two substitutable products with stochastic demands. *OPSEARCH 10*, 96–100.

Pasternack, B. A. and Z. Drezner (1991). Optimal inventory policies for substitutable commodities with stochastic demand. *Naval Res. Log. 38*, 221–240.

Pentico, D. (1974). The assortment problem with probabilistic demands. *Mgmt. Sci. 21*, 268–290.

Pentico, D. (1988). The discrete two-dimensional assortment problem. *Opns. Res. 36*, 324–322.

Queyranne, M. and T. Fill (2000). Hotel yield management: stochastic integer programming model with a monge property. Working Paper, Faculty of Commerce, University of British Columbia.

Robinson, L. (1990). Optimal and approximate policies in multiperiod multilocation inventory models with transshipments. *Opns. Res. 38*, 278–295.

Rockfellar, R. (1970). *Convex Analysis.* Princeton, N.J.: Princeton U. Press.

Sen, A. and A. Zhang (1999). The newsboy problem with multiple demand classes. *IIE Trans. 31*, 431–444.

Sherali, H. D., M. S. Bazaraa, and C. M. Shetty (1992). *Nonlinear programming: theory and algorithms.* John Wiley & Sons.

Swaminathan, J. M. and S. Tayur (1998). Managing broader product lines through delayed differentiation using vanilla boxes. *Mgmt. Sci. 44*, S161–S172.

Tripathy, A., H. Sural, and Y. Gerchak (1999). Multidimensional assortment problem with an application. *Networks 33*, 239–245.

Wagner, H. and T. Whitin (1958). Dynamic version of the economic lot size model. *Mgmt. Sci. 5*, 89–96.

Appendix: Proof of Propositions 2 & 5

Proof of Proposition 2 (outline): The dual of the recourse problem with zero substitution cost is:

$$\begin{aligned} \min_{\theta_i,\delta_i} \quad & \textstyle\sum_{i=1}^{N} -y_i\theta_i + \sum_{j=1}^{N} \xi_j\delta_j \\ s.t. \quad & -\theta_i \le h_i', \quad \text{for all } i; & (7) \\ & \delta_j \le p_j', \quad \text{for all } j; & (8) \\ & -\theta_i + \delta_j \le 0, \quad \text{for all } i \le j. & (9) \end{aligned}$$

The optimality of the greedy algorithm results from the following easily proven facts:

- Fact 1: During the algorithm, primal feasibility is maintained.
- Fact 2: During the algorithm, complementary slackness is maintained.
- Fact 3: When the algorithm ends, the network is divided into a forest, F, of trees. Each tree is composed of consecutive nodes in the network. In the final solution, each tree as a whole has either excess inventory or excess demand. (By continuity of demand, the probability of exactly zero excess inventory and zero excess demand is negligibly small, so we do not consider this case.) All the nodes k in tree τ_i have the same dual price, $\theta_k = \delta_k$. If tree τ_i has excess inventory (demand), the dual price is $-h_i'$ (p_i').
- Fact 4: When the algorithm ends, dual feasibility (optimality) is reached.

Facts 1, 2 and 4 taken together prove the optimality of the greedy algorithm (Proposition 2). Fact 3 is used to prove Fact 4. Proof details are available from the authors. □.

Proof of Propostion 5: We only consider Y values at which $G(Y)$ is finite. Clearly, $G(Y)$ is nonnegative and convex (since expectation preserves convexity and CY is linear). From Theorem 25.2 and Corollary 25.5.1 of Rockfellar (1970), $G(Y)$ will be continuously differentiable everywhere if the partial derivatives

$$\frac{\partial G(Y)}{\partial y_i} = \lim_{\epsilon\to 0}\int_{R_+^N}[c_i + \frac{L(Y+\epsilon e_i,\xi) - L(Y,\xi)}{\epsilon}]dF(\xi) \tag{10}$$

exist and are finite, where R_+^N denotes the nonnegative orthant of N-space and e_i denote an N-vector of 0's except for a 1 in the ith component.

Note that $|\frac{L(Y+\epsilon e_i,\xi)-L(Y,\xi)}{\epsilon}| \leq h_i$, since the value of $L(Y+\epsilon e_i,\xi)$ is no more than the cost of using only Y to satisfy demand (which is $L(Y,\xi)$) plus the cost of holding the extra ϵ units of product i (which is ϵh_i). Therefore the integrand of (10) is bounded above, which implies $\frac{\partial G(Y)}{\partial y_i}$ is also bounded above. This allows us to bring the limit inside of the integral in (10).

Since $L(Y,\xi)$ is convex, then by Theorem 25.3 of Rockfellar (1970), its partials must exist everywhere except at a countable number of points. Since the demand distribution is continuous, these points will have measure zero. Thus (10) becomes $\frac{\partial G(Y)}{\partial y_i} = c_i + \int_{R_+^N - S} \frac{\partial L(Y,\xi)}{\partial y_i} dF(\xi)$, where S is the set of measure zero at which the partials of $L(\cdot)$ do not exist. Since the partial derivatives exist everywhere outside of this set S, and are bounded from above, $G(Y)$ is continuously differentiable everywhere. $\square$