RC23433 (W0411-142) November 15, 2004
Mathematics

IBM Research Report

Push-Pull Production Planning

Feng Cheng, Markus Ettl, Grace Lin, Yingdong Lu
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

David D. Yao
IEOR Department
302 Mudd Building
Columbia University
New York, NY 10027

Push-Pull Production Planning

Feng Cheng, Markus Ettl, Grace Lin, Yingdong Lu
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
fcheng,msettl,gracelin,yingdong@us.ibm.com

David D. Yao*
IEOR Department, 302 Mudd Building
Columbia University, New York, NY 10027
yao@columbia.edu

September 2003
Revised: October 2004

Abstract

We study a manufacturing system with a two-stage production referred to as the Fabrication and Fulfillment processes. The first stage (Fabrication) is a Build-to-Plan (Push) process in which the parts are replenished, tested, and assembled into subassemblies according to the product level build plan. The subassembly inventory is kept in stock ready for the final assembly of the end products. The second stage (Fulfillment) is an Assembly-to-Order (Pull) process, which means that final assembly starts after the customer order is received and no finished goods inventory is kept for end products.

One important issue in the planning process is to address the trade-off between the capacity utilization and inventory cost reduction while striving to meet the quarter-end peak demand. We present a nonlinear optimization model to minimize the total inventory cost subject to the service level constraints and the machine capacity constraints. This results in a convex program with linear constraints. Efficient solution algorithms are developed using piecewise linear approximation, and heuristics are developed based on the Dynamic Programming formulation of the problem. Several variations of the model are formulated to incorporate additional features in practice. Numerical analyses are presented to show the performance improvement generated by the optimal solutions over the “as is” production-smoothing strategy.

*Research undertaken while an academic visitor at IBM Research Division, T.J. Watson Research Center.

1 Introduction

A push system is a traditional supply chain where production and distribution are based on forecasts. However, it is difficult to predict customer demand, and therefore difficult to match supply and demand. Thus, a push system is very susceptible to the bullwhip effect in the supply chain. On the other hand, in a pure pull system, production is driven by customer demand rather than forecast, which leads to zero inventory, reduced impact from the bullwhip effect, and improved service levels. Unfortunately, it is very difficult to implement a pull supply chain strategy if there are certain constraints in the supply chain, such as supply and production capacity limitations, long lead times, etc. Furthermore, a pull strategy is less likely able to take advantage of economies of scale offered by batch production or fully loaded trucks, since production and distribution are in response to specific customer demand.

Electronic commerce and the Internet are fundamentally changing the nature of supply chains. Consumers are becoming more and more comfortable using the Internet as a purchasing source. Typically, Internet customers demand immediate fulfillment of customized products. The challenges imposed by the Internet economy have led companies to look for a new supply chain strategy that takes advantage of the best of both push and pull systems. A hybrid push-pull supply chain system has been suggested to be the right model that combines mass production (push, or make-to-stock, like detroit flow lines) with small-batch production (pull, or make-ot-order, like flexible manufacturing) to achieve "mass customization" for production in the Internet age.

IBM Server Group (SG) manufactures high-end server computers. To deal with increasingly complex configuration requirements and tremendous pressures for responsive order fulfillment, a two-stage Push/Pull production strategy, also known as the Fabrication and Fulfillment strategy, has been implemented at IBM SG. The first stage (Fabrication) is a Build-to-Plan (Push) process in which the parts are replenished, tested, and assembled into subassemblies according to product level build plans. The second stage (Fulfillment) is an Assembly-to-Order (Pull) process, which means that final assembly starts after the customer order is received and no finished goods inventory is kept for end products. By having most of manufacturing tasks completed in the first stage of production, hence allowing a shorter cycle time in the second stage, SG can delay the commitment of resources until customer orders are finalized with detailed product configurations and required ship dates.

It has been observed that customer order volumes for high-end server computers exhibit a cyclic pattern typically skewed to the end of a quarter (and more so to the end of a year), creating an undesirable demand/capacity mismatch. Obviously, it would not be cost effective to simply build a production capacity to meet the peak demand. The two-stage production strategy enables the management of such demand and capacity mismatch by building up the component/subassembly inventory in the first stage during the low-demand periods, in preparation for the end-of-quarter peak demand.

There exist two types of capacity constraints in the entire production process, namely the machine capacity and the labor capacity. Typically, the labor capacity is shared to perform different manufacturing tasks, and can be adjusted according to demand by using overtime work force at a premium cost. Meanwhile, the share of machine capacity is more restricted, only among those tasks that involve components of a certain type, and the capacity is relatively fixed. Hence, the decision making on the build quantities over the planning horizon is largely driven by capacity constraints and the associated costs at the first stage of production process. A production-smoothing strategy is often adopted by planners in practice to meet the end-of-quarter peak demand and provide optimal capacity utilization.

A simple, however ad-hoc production planning policy has been practiced in SG to determine the monthly build plan given the quarterly volume forecasts for end products, that is to use a pre-specified quarter-to-month spread ratio to calculate the monthly build quantities from the quarterly volume forecasts. The spread ratio is usually determined based on factors including capacity utilization, plant and supplier requirements' stability, and the ability to ramp up for end-of-quarter peak demands. Such practice simplifies the decision making by dealing with the capacity issue at an aggregated level. However, the issue of minimizing inventory cost is not addressed directly with this heuristic approach. In the server computer business, inventory-driven costs, which include financing, inventory write-downs and inventory write-offs (obsolescence) have tremendous cost impact on performance. A challenging task is to generate a build plan that minimizes the inventory cost while meeting customer demand with a given service level in a capacity constrained environment.

Simchi-Levi describes the idea of the hybrid push/pull supply chain systems with a number of interesting examples ranging from DELL in the PC manufacturing, Amazon.com in the book industry, to Peapod in online grocery business. (note: more literature needed)

With regard to research relating to the capacitated production-inventory models with stochastic demand, Federgruen and Zipkin (1986a,b) establish the optimality of the modified base-stock policy in a discrete-time single item, single location finite capacity production system under an average cost criterion and a discounted cost criterion, respectively. Aviv and Federgruen (1997) allow the values of parameters such as demand distributions, capacity levels, and costs, to vary in a periodic pattern. They prove that a periodic modified base-stock policy is optimal, and show that the problem may be solved using a value-iteration method. Kapuscinski and Tayur (1996) also consider a periodic model; they compute inventory levels using infinitesimal perturbation analysis, a simulation-based method of estimating derivatives discussed by Glasserman and Tayur (1995). However, all these studies, as well as the majority of a vast body of related research, have limited themselves on single-item and/or single-echelon systems.

A special case of the two-stage production systems has been previously studied in the context of postponement strategy. For example, Aviv and Federgruen (2001) provide an analysis on capacitated multi-item inventory systems where the items are produced in two stages. They investigate the benefits of various delayed product differentiation (postponement) strategies and other related issues in a two-stage production system. The typical setting of two-stage production systems discussed in the postponement literature usually involves a common intermediate product in the first stage and a simple localization operation in the second stage. However, the two-stage production process in the IBM SG manufacturing environment is far more complex. There are multiple subassemblies/products with complex bills of material to be built in both stages of production. Build plans for each stage are subject to capacity constraints that are defined based on the types of parts involved in the operation.

In fact, the two-stage production strategy practiced at IBM SG can be viewed as a Configure-To-Order (CTO) system, a special case of the Assembly-To-Order (ATO) systems, which have been becoming increasingly popular in manufacturing industry and also have generated a great deal of research interests. A detailed survey of the state-of-the-art research on ATO systems has appeared in Song and Zipkin 2001. Several authors focus on performance evaluation and optimization techniques for the base-stock policy, assuming that demand in each period has a multivariate normal distribution. Hausman et al. (1998) examine the problem of maximizing the the probability filling all demand in a period within a time window subject to a linear budget constraint, and develop heuristic methods. The model of Agrawal and Cohen (2001) minimizes the total expected component inventory costs, subject to constraints on the order fill rates using an allocation policy characterized by partial FCFS and fair-share allocation. Cheng et al. (2002) study the problem of minimizing average component inventory holding cost subject to product-family dependent fill rate constraints for a CTO system and provide an exact algorithm and as well as a greedy heuristic for computing the optimal inventory policy. Nevertheless, the optimization models for ATO systems generally assume unconstrained production capacity.

In our paper, we will address the production-inventory optimization problem in the above-mentioned Push/Pull production environment. We consider a multiple-period problem with service level constraints defined at the product level for each period. In addition, there are production capacity constraints defined at the component level by component type for each period. The rest of the paper is organized as follows. We formulate the optimization problem with the decision variables representing the product build quantities at the machine-type model (MTM) level in the following section. Two alternative formulations of the problem are introduced in Section 3 with build quantities defined at the component level instead of the MTM level. Computation methods for solving the problem are described in Section 4, where a linear approximation approach and a temporal decomposition (TD) algorithm are developed to provide efficient numerical solutions to the nonlinear optimization problem. Numerical results are presented in

Section 5. Several extensions of the models are presented in Section 6. The paper is concluded with a summary and remarks for future research.

2 An MTM-Based Model

We formulate an nonlinear optimization problem to model the manufacturing planning problem under study. Instead of using a quarter-to-month spread ratio to derive the build plan as with the heuristic approach, we define the weekly build quantities of MTM's as the decision variables to capture potentially greater benefits from the use of analytic approach.

2.1 Formulation

We use the superscript K to index the MTM types, and the subscript i to index components (items or subassemblies). Let $\mathcal{K}$ denote the set of all MTM types, and $\mathcal{I} = \{1, ..., m\}$ denote the set of all components. We further partition $\mathcal{I}$ into disjoint subsets, $\mathcal{I} = \cup_{j=1}^{n} A_j$, with each A_j representing a collection of similar component, e.g. CPU, memory, and so forth.

Each MTM type is characterized by a subset of components, i.e., $K \subseteq \mathcal{I}$, along with the usage counts: u_i^K, $i \in K$, i.e., the number of units required from each component i.

Let $t = 1, ..., T$ index the time periods. For instance, when each period is a week, $T = 12$ represents a quarter.

Assume the following are given data:

- $C_{j,t}^M$, $j = 1, ..., n$, the machine capacity limit, in terms of the maximum number of components $i \in A_j$ that can be processed in period t. Note that this capacity is shared among all components in A_j, and only those components. We do not impose a labor capacity limit in this formulation. However, an extension of the model with labor constraints included will be discussed in Section 6.

- h_t^K, $K \in \mathcal{K}$, inventory cost for each unit of type K MTM left over at the end of period t (after supplying demand). For instance, $h_t^K = \sum_{i \in K} c_i u_i^K$, with c_i being the cost (for raw materials, processing and labor) associated with each unit of component i. (Note that since the final assembly is only built to order, the "MTM left over at the end of period t" really refers to the ensemble of its components.)

In addition, assume for each MTM type K, we know its demand for each period, D_t^K. Suppose

$$D_t^K = \mu_t^K + \sigma_t^K Z_t,$$

with Z_t denoting the standard normal variate; assume Z_t's are i.i.d. over time. Denote

$$D^K(1,t) := \sum_{s=1}^{t} D_s^K;$$

and similarly denote

$$\mu^K(1,t) := \sum_{s=1}^{t} \mu_s^K,$$

and

$$\sigma^K(1,t) := [\sum_{s=1}^{t}(\sigma_s^K)^2]^{1/2}.$$

The decision variables are: x_t^K, the build quantity for each MTM type K in each period t. Note, again, that the MTM's will *not* be actually "built" as they are assembled to orders only. The MTM build quantities are surrogates for component build quantities. Once x_t^K's are decided, so will the build quantity for each component: for component i, this is equal to $\sum_{K\ni i} u_i^K x_t^K$. Denote:

$$x^K(1,t) := \sum_{s=1}^{t} x_s^K.$$

The objective is to minimize the total inventory cost:

$$\min \sum_{t=1}^{T}\{ \sum_{K\in\mathcal{K}} h_t^K \mathsf{E}[x^K(1,t) - D^K(1,t)]^+. \tag{1}$$

The constraints are:

$$\sum_{i\in A_j}\sum_{K\ni i} u_i^K x_t^K \le C_j^M, \qquad j = 1, ..., n; \tag{2}$$

$$x^K(1,t) \ge \mu^K(1,t) + \sigma^K(1,t)\cdot \epsilon_{\alpha^K}, \qquad K \in \mathcal{K},\ t = 1, ..., T. \tag{3}$$

The last constraint above is equivalent to

$$\mathsf{P}[x^K(1,t) \ge D^K(1,t)] \ge \alpha^K,$$

which enforces the service levels, represented by α^K, for product K, with ϵ_{α^K} denoting the point corresponding to the α^K percentile of the standard normal distribution; e.g., $\epsilon_{\alpha^K} = 1.64$ when $\alpha^K = 95\%$.

Note in the objective function in (1), the inventory part assumes full backlog of demand. The expectation term in (1) can be derived as follows:

$$\mathsf{E}[x^K(1,t) - D^K(1,t)]^+$$

$$
\begin{aligned}
&= \mathsf{E}[x^K(1,t) - \mu^K(1,t) - \sigma^K(1,t)\cdot Z]^+ \\
&= \sigma^K(1,t)\mathsf{E}\Big[\frac{x^K(1,t) - \mu^K(1,t)}{\sigma^K(1,t)} - Z\Big]^+ \\
&= \sigma^K(1,t) H\Big(\frac{x^K(1,t) - \mu^K(1,t)}{\sigma^K(1,t)}\Big), \qquad (4)
\end{aligned}
$$

where

$$
H(x) := \mathsf{E}[x - Z]^+ = \int_{-\infty}^{x} (x - z)\phi(z)dz = x\Phi(x) + \phi(x), \qquad (5)
$$

with Φ and ϕ being the distribution and the density functions of Z.

Since $H(x)$ is both increasing and convex in x, the optimization problem has an increasing and convex objective function, and a set of linear constraints.

Proposition 1 *(Feasibility condition) An optimal solution exists to the problem defined by (1)-(3) if the following conditions are satisfied.*

$$
\sum_{i\in A_j}\sum_{K\ni i} u_i^K(\mu^K(1,t) + \sigma^K(1,t)\cdot \epsilon_{\alpha^K}) \le C_j^M * t, \qquad j = 1,...,n, t = 1,...,T. \qquad (6)
$$

Discussions

1. What's referred to as MTM here is really at the level of "boxes" or final products. That is, each MTM has a *fixed* configuration (bill of material, or BOM) in terms of components and their usage counts.

2. Decisions (build quantities) are made at the level of MTM (as defined above), by type and by period. Since each MTM has a fixed BOM, the decisions are readily translated into build quantities of components and subassemblies.

3. Inventory cost is charged to the *end* inventory of each period.

4. The definition of $C_{j,t}^M$ relates readily to more primitive data as follows. For instance in the case of constant capacity, suppose there are m machines for testing the subset of components in A_j, and suppose each machine handles a batch of B units at a time, and the testing time is w weeks on average. Then, $C_j^M = mB/w$ per week.

3 A Component-based Model

The MTM-based model presented above implicitly requires the components as defined by the BOM for an MTM be built in a "square" set, meaning that all the components required to assembly a particular

MTM needs to be built or replenished at the same time. This is simply because the variables defined in the MTM-based model present the MTM-level build quantities. To convert the MTM-level build plan to a component-level build plan, one would simply apply the usage rates as defined in the BOM to get the build quantities of the components. With decision variables defined at the component-level, the "square" set restriction can be removed to allow additional flexibility when optimizing the build plan with the capacity constraints.

3.1 Component-based Model Formulation

To model the planning of build quantities at the component level, we introduce a set of variables to represent the build quantity of each component. Let w_t^i be the build quantity for component i in period t, $i \in \mathcal{I}$ and $1 \leq t \leq T$. We also define the accumulative build quantity for component i from period 1 to period t as

$$w^i(1,t) = \sum_{s=1}^{t} w_s^i.$$

The objective is to minimize the total component inventory cost.

$$\min \quad \sum_{t=1}^{T} \{ \sum_{i \in \mathcal{I}} h_t^i \mathsf{E}[w^i(1,t) - \sum_{K \in \mathcal{K}} \sum_{K \ni i} u_i^K D^K(1,t)]^+. \tag{7}$$

The constraints are:

$$\sum_{i \in A_j} w_t^i \leq C_j^M, \qquad j = 1, ..., n; \tag{8}$$

$$x^K(1,t) \geq \mu^K(1,t) + \sigma^K(1,t) \cdot z_{\alpha^K}, \qquad K \in \mathcal{K},\ t = 1, ..., T; \tag{9}$$

$$w^i(1,t) \geq \sum_{K \in \mathcal{K}} \sum_{K \ni i} u_i^K x^K(1,t), \qquad i \in \mathcal{I},\ t = 1, ..., T. \tag{10}$$

The last constraint above ensures that the MTM-level build plan is always feasible given the tested parts available at any given time. In fact we can combine (9) and (10) to get the service level constraints defined at the component level as the following

$$w^i(1,t) \geq \sum_{K \in \mathcal{K}} \sum_{K \ni i} u_i^K \left(\mu^K(1,t) + \sigma^K(1,t) \cdot z_{\alpha^K} \right),\ t = 1, ..., T,$$

and eliminate the $x^K(1,t)$ variables from the formulation. Also note that h_t^i is now the inventory holding cost for component i in period t.

Proposition 2 *The minimimal cost of the component model is no greater than the minimimal cost of the corresponding MTM model.*

Remarks

- The benefits of the component-based model over the MTM-based model are achieved by relaxing the requirement for building "square sets" at the component level. The difference between the MTM-based model and the component-based model reflects the cost reduction due to this factor. Since the capacity constraints are defined at the component level, it is clear that better capacity utilization can be achieved through optimizing the component level build plan. Especially when the capacity is tight for some parts in certain periods, one has the flexibility to adjust the build schedules for the parts to avoid capacity constraints in those periods. On the other hand, the MTM-based model would be more restrictive in this case since parts are to be built in "square sets", therefore, less flexible in making adjustments of the build plan.

- The another potential factor that could lead to additional cost savings is the risk-pooling effect on the demand variability when we switch from the MTM-based model to a component-based model. This is based on the fact that parts commonality exists among different MTM's. Since the demand variability is reduced through demand aggregation at the component-level, the safety stock required to maintain the same level of service should be less than what would be required if the safety stock was kept at the MTM level.

- However, the cost savings due to the demand variability reduction is not reflected in the current component-based model because the demand variability reduction is not captured in the model. To take the advantage of the risk-pooling effect, we would need to model the demand variability reduction that could be achieved by forecasting the component-level demand directly. Furthermore, a component-based (or build block-based) planning methodology should be adopted such that the common components are indeed shared by the MTMs that require them rather than individually allocated to each of those MTMs.

4 Two Solution Approaches

Both the MTM-based model and the component-based model have nonlinear objective functions, more specially convex functions, with linear constraints. Today's high-power nonlinear solvers are probably capable of solving this type of problems. However, there are certain structures of the problems that can be explored and utilized to facilitate the computation involved in the optimization. We have developed two computation approaches that demonstrated improved computation performance through numerical results. One approach is to approximate the nonlinear objective function through linearization so that a near-optimal solution can be obtained by using a linear program solver. The second approach is to use a backward-recursion algorithm to solve the problem through a temporal decomposition.

4.1 Piece-wise Linearization

First notice that the penalty part of the objective function in (1) can be replaced by $\pi_t u_t^K$, with

$$\sum_{K\in\mathcal{K}} x_t^K - C_t^L = v_t - w_t, \tag{11}$$

and $v_t, w_t \geq 0$ being additional slack variables. Furthermore, since

$$x_t^K = x^K(1,t) - x^K(1,t-1),$$

we can let

$$y_t^K := x^K(1,t), \qquad t = 1, ..., T;\ K \in \mathcal{K}$$

be the new variables (and denote $y_0^K := 0$ for all K). This way, the optimization problem becomes:

$$\min \sum_{t=1}^{T} \left\{ \pi_t v_t + \sum_{K\in\mathcal{K}} h_t^K \sigma^K(1,t) H\Big(\frac{y_t^K - \mu^K(1,t)}{\sigma^K(1,t)}\Big) \right\}, \tag{12}$$

subject to the following constraints:

$$\sum_{i\in A_j} \sum_{K\ni i} u_i^K [y_t^K - y_{t-1}^K] \leq C_j^M, \qquad j = 1, ..., n; \tag{13}$$

$$\sum_{K\in\mathcal{K}} [y_t^K - y_{t-1}^K] - C_t^L = v_t - w_t, \qquad t = 1, ..., T; \tag{14}$$

$$y_t^K \geq \mu^K(1,t) + \sigma^K(1,t) \cdot \epsilon_{\alpha^K}, \qquad K \in \mathcal{K},\ t = 1, ..., T; \tag{15}$$

$$y_T^K \geq y_{T-1}^K \geq \cdots \geq y_1^K \geq 0, \qquad K \in \mathcal{K} \tag{16}$$

$$v_t \geq 0, \quad w_t \geq 0, \qquad K \in \mathcal{K},\ t = 1, ..., T. \tag{17}$$

Clearly, the above is a separable convex programming problem.

From (5), it is readily verified that

$$H(x) \approx x, \quad \text{for} \quad x \geq 3,$$

and

$$H(x) \approx 0, \quad \text{for} \quad x \leq -3.$$

Hence, the only non-linearity of the function is within the interval $[-3, 3]$. (And, over this interval it is convex).

Hence, we should be able to approximate the H function by piecewise linear functions with a relatively small number of pieces. Suppose we preselect the following points on the x-axis:

$$-3 := z_0 < z_1 < \cdots < z_p < z_{p+1} := 3. \tag{18}$$

For any x, suppose $x \in [z_\ell, z_{\ell+1}]$, for some $\ell \le p$. We can write

$$x = z_\ell + \lambda(z_{\ell+1} - z_\ell),$$

for some $\lambda \in [0, 1]$. We can then approximate $H(x)$ as follows:

$$H(x) \approx H(z_\ell) + \lambda[H(z_{\ell+1}) - H(z_\ell)],$$

which is nothing more than linear interpolation. In general, we can write any $x \in (-3, +3)$ as

$$x = \sum_{\ell=0}^{p} \lambda_{\ell+1}(z_{\ell+1} - z_\ell),$$

and

$$H(x) \approx \sum_{\ell=0}^{p} \lambda_{\ell+1}[H(z_{\ell+1}) - H(z_\ell)].$$

These, along with $H(x) = x$ for $x \ge 3$ and $H(x) = 0$ for $x \le -3$, approximate the H function by $p + 2$ linear pieces.

We now apply this linearization procedure to the H function in (12). Thanks to the constraint in (15), we can start at

$$z_0 = \min_{K \in \mathcal{K}} \{\epsilon_{\alpha^K}\}, \tag{19}$$

instead of $z_0 = -3$ as in (18). (The other preselected z_ℓ's are the same as in (18).)

Write

$$y_t^K = \mu^K(1,t) + \sigma^K(1,t) \sum_{\ell=0}^{p} \lambda_{\ell+1}^{K,t}(z_{\ell+1} - z_\ell). \tag{20}$$

Then, we have

$$H\Big(\frac{y_t^K - \mu^K(1,t)}{\sigma^K(1,t)}\Big) = H\Big(\sum_{\ell=0}^{p} \lambda_{\ell+1}^{K,t}(z_{\ell+1} - z_\ell)\Big) = \sum_{\ell=0}^{p} \lambda_{\ell+1}^{K,t}[H(z_{\ell+1}) - H(z_\ell)].$$

Consequently, the original optimization problem becomes:

$$\min_{\lambda} \sum_{t=1}^{T} \quad \left\{ \pi_t v_t + \sum_{K \in \mathcal{K}} h_t^K \sum_{\ell=0}^{p} \lambda_{\ell+1}^{K,t}[H(z_{\ell+1}) - H(z_\ell)] \right\}; \tag{21}$$

with the following constraints:

$$\sum_{i\in A_j}\sum_{K\ni i} u_i^K \sum_{\ell=0}^{p}(\lambda_{\ell+1}^{K,t}-\lambda_{\ell+1}^{K,t-1})(z_{\ell+1}-z_\ell) \le C_j^M, \qquad j=1,...,n; \tag{22}$$

$$\sum_{\ell=0}^{p}\lambda_{\ell+1}^{K,t}(z_{\ell+1}-z_\ell) \ge \epsilon_{\alpha^K}, \qquad K\in\mathcal{K},\ t=1,...,T; \tag{23}$$

$$\sum_{K\in\mathcal{K}}\sum_{\ell=0}^{p}(\lambda_{\ell+1}^{K,t}-\lambda_{\ell+1}^{K,t-1})(z_{\ell+1}-z_\ell) - C_t^L = v_t - w_t, \qquad t=1,...,T; \tag{24}$$

$$\sum_{\ell=0}^{p}\lambda_{\ell+1}^{K,t}(z_{\ell+1}-z_\ell) \ge \sum_{\ell=0}^{p}\lambda_{\ell+1}^{K,t-1}(z_{\ell+1}-z_\ell), \qquad K\in\mathcal{K},\ t=1,...,T; \tag{25}$$

$$0 \le \lambda_\ell^{K,t} \le 1, \qquad \forall K,t,\ell. \tag{26}$$

The above is a linear programming with $\lambda_i^{K,t}$ as decision variables. Once the optimal solution to $\lambda_i^{K,t}$ is obtained, the optimal solution to y_t^K follows from (20).

4.2 Temporal Decomposition

Rewrite the component-based model (7-10) as follows.

$$\min \sum_{t=1}^{T}\{\sum_{i\in\mathcal{I}} h_t^i \mathsf{E}[w^i(1,t)-d^i(1,t)]^+\}, \tag{27}$$

Subject to:

$$\sum_{i\in A_j} w_t^i \quad\le\quad C_j^M, \qquad j=1,...,n; \tag{28}$$

$$w^i(1,t) \quad\ge\quad B^i(1,t), \qquad t=1,...,T. \tag{29}$$

where

$$d^i(1,t) = \sum_{K\in\mathcal{K}}\sum_{K\ni i} u_i^K D^K(1,t),$$

and

$$B^i(1,t) = \sum_{K\in\mathcal{K}}\sum_{K\ni i} u_i^K\left(\mu^K(1,t)+\sigma^K(1,t)\cdot z_{\alpha^K}\right),\ t=1,...,T.$$

Notice that the new problem is separable in j. Therefore, we can decompose the problem by j and solve a number of smaller problems each with simple constraints. We can rewrite the constraint (29) as

$$\sum_{i\in A_j} w^i(1,t) \ge \bar{C}_j(1,t),$$

where $\bar{C}_j(1,t)$ is given by

$$\bar{C}_j(1,t) = \sum_{i\in A_j} w^i(1,t+1) - C_j^M.$$

In fact this new formulation is also separable in t. Therefore, we can further simplify the sub-problem for a given j, as defined in (27-29), by decomposing it for each $t = T, ..., 1$, i.e.,

$$\min \sum_{i \in A_j} h_t^i \mathsf{E}[w^i(1,t) - d^i(1,t)]^+, \tag{30}$$

Subject to:

$$\sum_{i \in A_j} w^i(1,t) \geq \bar{C}_j(1,t), \tag{31}$$

$$\bar{B}^i(1,t) \leq w^i(1,t) \leq w^i(1,t+1), \qquad i \in A_j. \tag{32}$$

where

$$\bar{B}^i(1,t) = B^i(1,t) + w^i(1,t+1) - \bar{B}^i(1,t+1)$$

with $\bar{B}^i(1,T+1) := 0$. Note that $w^i(1,t+1)$ is a constant given by the solution for period $t+1$ and $w^i(1,T+1) := 0$. The structure revealed in this simplified formulation allows us to decompose the original problem by j and by t into a sequence of smaller problems, thus reducing the complexity of the problem and hence the total computation time required.

The objective function of the sub-problem given j and t is given by

$$g(j,t|\mathbf{w}(j,t)) = \sum_{i \in A_j} h_t^i \mathsf{E}[w^i(1,t) - d^i(1,t)]^+,$$

where $\mathbf{w}(j,t)$ denotes the vector $w^i(1,t) : i \in A_j$. The value function is then defined as

$$v(j,t) = \min \sum_{k=t}^{T} g(j,k|\mathbf{w}(j,k)).$$

The new formulation of the problem with the temporal decomposition can be presented as follows.

$$v(j,t) = \min_{\mathbf{w}(j,t) \in \mathcal{U}_t} \left(v(j,t+1|\mathbf{w}(j,t) + g(j,t+1|\mathbf{w}(j,t+1)) \right),$$

where $t = 1, ..., T, j = 1, ..., n$ and $v(j,T) = 0, \forall j$. $\mathcal{U}_t$ is defined by (31) and (32).

Given the structure of the sub-problem for given t and j, it can be observed that the optimal solution $\mathbf{w}(j,t)$ has the following properties

Proposition 3 *The optimal solution $\mathbf{w}(j,t)$ is always at the boundaries defined by (31) and (32).*

Proof. Since the objective function of the sub-problem g is convex and non-decreasing in $\mathbf{w}(j,t)$, the optimal value must be attained at the boundaries.

There are three cases that an optimal solution can be attained.

- If constraint (32) is binding from below, then the optimal solution is directly given by the lower bounds defined by (32);
- If constraint (32) is binding from above and constraint (31) is not binding, then the optimal solution is directly given by the upper bounds defined by (32);
- If constraint (31) is binding, the optimal solution lies on the line defined by (31) and between the two intersects of the line with the bounds defined by (32).

The resulting problem with $i \in A_j$ for a given j can be solved using a backward recursion algorithm outlined below. Note that each stage of the recursion involves solving a nonlinear optimization problem with a convex objective and linear constraints. The solution to this problem (30-32) can be easily obtained by performing a greedy search.

- Step 0: Initialize $w_i(1,t) = \lceil B^i(1,t) \rceil,\ t = 1, ..., T$ ($\lceil x \rceil =$ the least integer greater than or equal to x), and let the step size be Δw.
- Step 1: If for $t = T$, $\sum_{i \in A_j} w^i(1,t) > C_j^M(1,t)$, then the problem is infeasible. Exit. Otherwise continue to Step 2.
- Step 2: For $t = T-1, ..., 2$,
  If for some $t < T$, $\sum_{i \in A_j} w^i(1,t) > C_j^M(1,t)$, find $i^* = \min\{h_t^i \Phi((w^i(1,t) - \mu^i(1,t))/\sigma^i(1,t))\}$.
  Let $w^{i^*}(1,t) := w^{i^*}(1,t) - \Delta w$ and $w^{i^*}(1,t-1) := w^{i^*}(1,t-1) + \Delta w$. Repeat this step until $\sum_{i \in A_j} w^i(1,t) \leq C_j^M(1,t)$.

One added benefit of the backward recursion algorithm is that we can easily restrict the solutions to be integers by letting $\Delta w = 1$. Adding the integer constraints in the original formulation would increase the computational complexity substantially and most likely would render the problem intractable when a standard NLP solver is used.

5 Numerical Results

We have conducted two types of numerical analyses. The first one is aimed to provide computational performance comparisons between different computation methods/algorithms. The second type of analysis is designed to show the estimated cost reduction potentially achieved by different optimization models described in this paper.

5.1 Computational Performance Analysis

In this part of the numerical results, we compare the computational performance of three different algorithms: the exact solution implemented using IPOPT, a nonlinear solver developed by A. Waechter, et al., the piecewise linear approximation (PWLA) solution, and the temporal decomposition (TD) solution for the component-based model. The IPOPT and PWLA solutions are implemented for both the MTM-based model and the component-based model. The results obtained using IPOPT are used as a benchmark to compare with those by the PWLA solution and the TD solution.

To test the performance of the three computation methods, a data set that resembles a typical real-world problem is compiled based on real data. The data set includes 23 MTM partnumbers, 40 component partnumbers of 4 different component types, and weekly demand data for 12 periods. The results for the MTM-based model are shown in Table 1. The m value is the number of segments used in the PWLA solution, `obj` is the objective function value computed, `error` is the error relative to the exact optimal solution by IPOPT, and T is the computation time (seconds) used (on a Pentium 4 Intellistation). The PWLA results are reasonably accurate compared to the IPOPT results and faster in terms of the computation time required.

PWLA						IPOPT
m	3	5	10	15	20	-
Obj	119570	119336	119366	119130	119128	119078
error	0.41%	0.22%	0.24%	0.04%	0.04%	-
T	0.90	1.67	2.95	4.44	6.24	11.69

Table 1: Performance Comparison of IPOPT and PWLA solutions for the MTM model

Similar results are also obtained for the component-based model. Comparisons are shown in Table 2. Particularly, the results of the TD solution are also included here. Note that the data set used for the component model is the same as the one used for the MTM model except that the means and variances of the component-level demands need to be derived explicitly to compute the objective values as defined in the component-based model. Furthermore the capacity data are the same for the results in Tables 1 and 2.

	PWLA			TD			IPOPT
$m/\Delta w$	3	5	10	0.01	0.1	1	-
Obj	51978	51972	51970	51969	51969	51972	51969
error	0.014%	0.006%	0.0014%	0.001%	0.001%	0.006%	-

Table 2: Performance Comparison of IPOPT, PWLA and TD solutions for the component model)

Note that in Table 2 the numbers reported in the row labeled $m/\Delta w$ are the number of pieces for the PWLA results and the step size used for the greedy search in the TD solution respectively. The TD algorithm actually performs very well in terms of the accuracy of the solution. The computation time of the TD algorithm is less than 0.1 second when Δw=0.1, much faster than either IPOPT or PWLA. Furthermore, a step size of one is in fact more practical for real-world applications.

5.2 MTM model vs. Component model

To demonstrate the advantage of the Component-based model over the MTM-based model, we compiled Table 3 based on the results presented in the previous subsection. To provide a comparable cost analysis, we compute the cost of each MTM as the sum of the costs of all components used to assembly that MTM. Furthermore, for the MTM model, we compute the build quantities of the components using the usage rates based on the optimal build quantities obtained for MTM's, then use these component build quantities to calculate the objective function as defined by the component model. In particular, to calculate the expectation in (7), the demand variability should stay the same as in (1), namely the MTM demand variability, NOT the component demand variability.

When comparing the results of the two models, we notice that the results obtained from the component model may not be feasible for the MTM model for a given set of capacity constraints. This is because the MTM model assumes that all the parts are built in "square sets", thus requiring more available capacity in general and becoming less flexible when making build plans for individual parts. Table 3 provides a performance comparison between the MTM model and the component model. The expected performance of the "As Is" scenario is also included for reference. There are six cases in the table representing two different sets of capacity constraints (C1 and C2; on average C2 is about 60% of C1) and three different service levels (S1=50%, S2=80% and S3=95%).

Capacity	Service	As Is	MTM	COMP
C1	S1	5.6476	2.4729	2.3604
C1	S2	6.8945	4.1571	3.6239
C1	S3	8.1478	6.3649	5.1224
C2	S1	5.6476	2.9254	2.3756
C2	S2	6.8945	5.0326	3.6553
C2	S3	8.1478	7.7770	5.1777

Table 3: Performance comparison between MTM model and component model

The column labeled as "As Is" represents the "as is" practice using a quarter-to-month spread ratio of (30%,40%,30%) to determine the monthly build quantities. "MTM" is the optimal objective value obtained from the MTM model, and the column labeled as "COMP I" shows the optimal objective values obtained by the component model. As we can see, the overall inventory cost can be reduced by about 20% or more if the solution of the MTM model is implemented. Furthermore, optimizing the build quantities at the component level will generate as much as 10-20% more savings comparing to the MTM model.

To have a better understanding of how additional savings can be achieved by the component model, let us take a close look of the results in two cases. The first one is when the capacity constraints are not too tight (ex., C1 & S1). In this case, parts can be built as late as possible even with the "square set" requirement which can be accommodated within the capacity limits. Therefore, the inventory costs of both models can be minimized almost to the extent that just meets the service level constraints.

The second case is when the capacity constraints become tighter (ex., C2 & S3). Because of the demands in the later periods exceed the capacity, parts need to be built in earlier periods when the demands are low. Particularly for the MTM model, the "square set" requirements impose additional restrictions that force the parts to be built even earlier, whereas in the component model production for some of parts can be delayed as long as the capacity is available for building those parts (not the "whole set") in later periods.

6 Extensions

There are a number of extensions to the models described in the earlier part of the paper. Additional considerations that reflect the real-world scenarios are incorporated in these extensions.

6.1 A Modified Formulation

In reality, some of the test capacities are not specific to a certain type of components (called commodity type). For example, MCM parts and MEMORY parts may go through similar tests requiring the same capacity. Here we define the capacity constraints in terms of the type of tests to be performed (called driver type). Denote the subset of $\mathcal{K}$ with components requiring the test capacity of driver type j by V_j. Note that V_j may not be disjoint since some components are tested with more than one driver type. We further define that C_j^V be the test capacity of driver type j, $j = 1, ..., n_v$ (n_v is the number of different driver types), and v_{ij} be the usage rate by component i when driver type j is used. The new constraints are:

$$\sum_{i \in V_j} v_{ij} w_t^i \leq C_j^V, \qquad j = 1, ..., n_v; \tag{33}$$

$$\tag{34}$$

Furthermore, some tests may also have dependency requirements. For example, test 1 must be completed for a component before test 2 can be done for the same component. To deal with such dependency requirements, we introduce a new variable representing the number of the components that have completed the first test each time that such a dependency occurs. Let $\hat{\mathcal{I}}$ be the set of components going through two tests with a dependency requirement. For each $i \in \hat{\mathcal{I}}$ and $t = 1, ..., T$, we define a new variable $w^{i'}(1,t)$ and a new constraint

$$w^{i'}(1,t) \geq w^i(1, t + \tau_i),$$

where τ_i is the lead time (rounded to weeks) of the first test. The cases where the dependency is defined for a component in more than two tests can be treated in a similar fashion.

6.2 Labor Capacity Constraints

Let us introduce the parameters related to labor capacity.

- C_t^L, the labor capacity limit, in terms of the maximum number of MTM's, over all types, that can be built in period t, $t \leq T$.
- π_t, penalty cost for over-time labor, when the capacity limit C_t^L is exceeded.

The objective is to minimize the total inventory cost and penalty cost for over-time labor:

$$\min \sum_{t=1}^{T} \{ \sum_{K \in \mathcal{K}} h_t^K \mathsf{E}[x^K(1,t) - D^K(1,t)]^+ + \pi_t [\sum_{K \in \mathcal{K}} x_t^K - C_t^L]^+ \}. \tag{35}$$

The constraints are:

$$\sum_{i \in A_j} \sum_{K \ni i} u_i^K x_t^K \leq C_j^M, \qquad j = 1, ..., n; \tag{36}$$

$$x^K(1,t) \geq \mu^K(1,t) + \sigma^K(1,t) \cdot \epsilon_{\alpha^K}, \qquad K \in \mathcal{K},\ t = 1, ..., T. \tag{37}$$

7 Concluding Remarks

In this paper, we formulated a nonlinear optimization problem with the objective to minimize the inventory cost subject to both capacity constraints and service level constraints. Several different models are formulated with decision variables defined as either the build quantities of MTMs or the build quantities of components. We developed algorithms for solving the problems efficiently. The numerical analysis based on real data are presented to demonstrate the performance improvements achieved by the optimization models over the “as is” scenarios.

References

[1] AGRAWAL, M. AND COHEN, M., "Optimal Material Control and Performance Evaluation in an Assembly Environment with Component Commonality". Naval Research Logistics 48 (2001), 409-429.

[2] AVIV, Y. AND FEDERGRUEN, A. (1997), "Stochastic Inventory Models with Limited Production Capacity and Periodically Varying Parameters," Probability in the Engineering and Informational Sciences, 11, 107-135.

[3] AVIV, Y. AND FEDERGRUEN, A. (2001), "Capacitated Multi-item Inventory Systems with Random and Seasonally Fluctuating Demands: Implications for Postponement Strategies," Management Sciences, 47(4), 512-531.

[4] CHENG, F., ETTL, M., LIN, G.Y., AND YAO, D.D., "Inventory-Service Optimization in Configure-to-Order Systems", Manufacturing and Service Operations Management 4 (2002), 114-132.

[5] FEDERGRUEN, A. AND ZIPKIN, P. (1986a), "An Inventory Model with Limited Production Capacity and Uncertain Demands I. The Average-Cost Criterion.," Mathematics of Operations Research, 11(2), 193-207.

[6] FEDERGRUEN, A. AND ZIPKIN, P. (1986b), "An Inventory Model with Limited Production Capacity and Uncertain Demands II. The Discounted-Cost Criterion.," Mathematics of Operations Research, 11(2), 208-215.

[7] GLASSERMAN, P. AND TAYUR, S. (1995), "Sensitivity Analysis for Base-Stock Levels in Multi-Echelon Production-Inventory Systems," Management Science, 42(5), 263-281.

[8] HAUSMAN, W.H., LEE, H.L. AND ZHANG, A.X., "Order Response Time Reliability in a Multi-Item Inventory System", European J. of Operational Research, 109 (1998), 646-659.

[9] KAPUSCINSKI, R. AND TAYUR, S. (1998), "A Capacitated Production-Inventory Model with Periodic Demand," Operations Research, 46(6), pp.899-911.

[10] SONG, J.S. AND ZIPKIN, P. (2004), Supply Chain Operations: Assemble-to-Order Systems, Chapter 11 of Supply Chain Management, T. De Kok and S. Graves, eds., in Handbooks in Operations Research and Management Science, Vol. XXX, North-Holland.

[11] ZIPKIN, P. (2000), "Foundations of Inventory Management", McGraw-Hill.