SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NEW YORK 14853

TECHNICAL REPORT NO. 806

June 1988

ANALYSIS OF MULTISTAGE PRODUCTION SYSTEMS

by

John A. Muckstadt
Robin O. Roundy

This research was supported by National Science Foundation grants DMC–85–09131, ECS–8404641 and DMC–8451984, and also by AT&T Information Systems and DuPont Corporation.

1. INTRODUCTION

The control of inventories of physical objects has been the theme of literally thousands of technical and trade journal articles during this century. Since Harris (1915) described the basic trade–offs that need to be made when making these control decisions, numerous models and algorithms have been devised that indicate how basic control parameters should be established. In his work, Harris, to our knowledge, provided the first derivation of the economic lot size formula, or economic order quantity (EOQ) formula. Later Raymond (1931) wrote the first full book on the topic of inventory control in which he explained, without the aid of mathematical theory, how extensions of the now classic EOQ problem could be implemented.

It was not until the 1950s when more serious mathematical analyses of various inventory problems were undertaken. Whitin's (1953) stochastic version of the EOQ model and works by Arrow, Harris, and Marshak (1951), Dvoretzky, Kiefer, and Wolfowitz (1952, 1953) and Arrow, Karlin, and Scarf (1958) provided the mathematical basis for subsequent extensions of the single location model. Also see Chapter 2 of this book. The first multi–stage models were developed at about the same time. The important works of Clark and Scarf (1960, 1962) and the papers found in the book edited by Scarf, Gilford, and Shelly (1963) generated considerable interest from both academics and practitioners. The often quoted text by Hadley and Whitin (1963) and the survey paper by Veinott (1966) provide excellent summaries of many of these early modeling efforts. While much of this work concentrated on stochastic modeling, there was and currently is a strong interest in developing further extensions of Harris' basic EOQ model.

These extensions include single item, single location models that consider backordering, finite production rates, and quantity discounts of several types, and multi–item models that represent joint replenishment problems and constraints of many types. Most of these single location models are discussed in some detail in Peterson and Silver (1979). Recently, efforts in single stage modeling have focused on different issues such

as the trade–offs between setup cost reduction and inventory costs (Porteus (1985, 1986, 1987)) and the effects of process quality on inspection and lot–sizing policies (Porteus (1986), Lee and Rosenblatt (1987), and Muckstadt (1988)).

Other important advances have come from the study of various types of multi–stage systems. These include the extension of the EOQ model to consider serial, assembly, and distribution systems as well as more general system structures. We will examine these other systems in detail in subsequent sections. However, our coverage will by necessity be limited. There are many important contributions that we will not discuss including, for example, the extensive research on systems with non–constant demands and the research on setup cost reduction. We also will ignore heuristics for multi–stage lot–sizing without provable performance bounds, such as the ones developed by Blackburn and Millen (1982), Williams (1981), Crowston and Wagner (1973), Crowston, Wagner, and Henshaw (1972), Crowston, Wagner, and Williams (1973), Graves (1981), Jensen and Khan (1972), McLaren (1976), and many others, some of which are listed in the references. The same applies to the joint replenishment problem; we review only heuristics with provable performance bounds. We refer the reader to Silver and Peterson (1985) for a discussion of other heuristics for the joint replenishment problem.

Our purpose in this chapter is to review optimization based methods for solving lot sizing problems for a range of serial, assembly, distribution, and other more general system structures with constant demand rates. Furthermore, we will study only a class of algorithms that has a provable bound on their performance. To be precise, our goal is to present models and algorithms that can be used to determine what we call consistent and realistic reorder intervals for each stage in each of the types of systems we have mentioned. We have chosen to develop our presentation in terms of reorder intervals, that is, the time between placement of orders, rather than in terms of lot size. Although the classic EOQ

model is formulated in terms of lot sizes, we believe that there are several important reasons for formulating the model in terms of reorder intervals.

First, in many environments we have examined, we found it easier to think in terms of planning the frequency of production rather than calculating the lot size quantities, because the resources required to manufacture and distribute components, subassemblies, and finished products are generally related to the number of production runs per unit of time rather than just to the lot sizes. For example, the frequency of production dictates the number of setups, planned production orders, requests for tooling and fixtures, and requirements for moving batches of work–in–process inventories. It is easier to think in terms of producing items, say either weekly, monthly, quarterly, or yearly rather than to plan production to optimal lot sizes that indicate production should occur every 2.796 weeks for one item, 3.921 weeks for a second and $\sqrt{\pi}$ weeks for a third.

Second, the mathematical representation of the problem is simplified when the model's decision variables are reorder intervals rather than lot sizes. When the decision variables are production lot sizes, constraints must often be stated to ensure that an adequate supply of each component is available to produce the lot. These constraints, which must be stated in terms of installation stock, are not always easy to work with. As a result, the algorithm for finding the optimal lot sizes can be quite complex. Choosing the decision variables to be the reorder intervals makes the model easy to state and solve.

Third, when the demand pattern for each item and the lead time for inventory replenishment are known exactly, then once the reorder intervals are established it is obviously easy to compute the corresponding lot sizes. However, in most real situations demand forecasts change somewhat from time to time. Consequently, when there are small variations in the demand process, it is generally easier to keep reorder intervals constant and to adjust the lot sizes than to change the frequency of production intervals. Thus from a practical viewpoint, this choice simplifies the subsequent scheduling task and greatly reduces the so–called nervousness found in some MRP systems.

As mentioned, each of the models we will examine ensures that consistent production decisions are made for each stage. By this we mean that the reorder intervals must be determined so that a feasible and cost effective flow of material will occur from stage to stage. Furthermore, the reorder intervals between successive stages should be planned so that schedules can be generated that can be implemented realistically on the shop floor. That is, a solution is not useful when it indicates that a component's reorder interval is $\sqrt{3}$ times longer than that of the assembly into which it is placed. Consequently, we believe that restricting attention to a certain class of solutions is desirable.

The policies considered throughout this chapter are based on ones we have observed in practice. We began our research that led to the material we will present by studying the production control system used in stamping plants for a major U.S. automotive manufacturer. In that environment, we observed that demand for products (doors, body panels, rear decks, etc.) remains relatively constant over a lengthy period of time. The schedules determined by the production planner required each operation or component to be produced exactly once, twice, or four times during each 4–week period. The amount produced in each run was planned to be approximately the same. By having the time between production runs remain constant for an operation or component, the planner could assign a specific operation to time slots on each machine so that, for example, every fourth week the same operation takes place at the same time on the same machine. By restricting the production schedules to the type we have described, the planner could create schedules relatively quickly. His objective was to choose the week, or weeks, in which to perform an operation so that the machine utilization across the 4–week horizon was balanced. The selection of the day and shift on which to run an operation was also guided by the desire to balance the work across shifts, and across days of the week.

The planner also had to coordinate production among stages. To do so, he selected reorder intervals so that production could not occur at an operation unless it also occurred at all of the immediate successor operations. This production tactic was chosen to prevent build up of work–in–process inventory. Thus if production at one operation occurred every second week, then the immediate successor operations had production every week or every second week. Policies of this type are called nested.

Scheduling operations on several hundred machines was obviously a difficult task. It was clear to us that the planner's effectiveness rested heavily on the fact that each operation was scheduled to occur every week, every other week, or every fourth week. If this 1, 2, 4 type of policy had not been followed, the scheduling process would have been considerably more time and resource consuming.

Subsequent to our automative manufacturing study, we examined production planning systems used by manufacturers of reprographic machines, computers and pneumatic tools. In each case we observed that these firms made similar assumptions to those described previously when planning production. In particular, the major portion of the products being manufactured had relatively constant demand and the firms used nested policies. Furthermore, these firms were also using policies similar to the 1, 2, 4 type. We found, for example, in one case that production lots were equal to demand expected during 3 months, 6 months, or 1 year.

The assumptions and policies considered in this chapter are guided by these observations. The models we will present restrict attention for the most part to policies that are nested and stationary. By stationary, we mean that the time between production runs at each stage or operation is constant. Although policies of this form are not necessarily optimal (Muckstadt and Singer (1978); Williams (1982)), they are of significant practical importance. The nestedness assumption is made primarily for ease of presentation. The models can easily be modified to consider nonnested solutions at particular stages or

operations. More is said about this generalization in Roundy (1985a,1986) and Maxwell and Muckstadt (1985). In section 9 we explicitly consider the effects of using nonnested policies.

Furthermore, we also consider only solutions for which each stage or operation can produce a multiple of two (i.e., 1, 2, 4, 8, ... , 2^k, ... , where k is a nonnegative integer) times per reorder interval for any predecessor stage. We also assume that a *base planning period* exists, such as a shift, day, week, or month, and that all reorder intervals must be an integer multiple of this period. As we have discussed, restricting attention to such solutions greatly simplifies the task of scheduling the actual production for each workcenter. When limiting the choice of a solution to this restricted class of policies, the optimal objective function value does not differ substantially from that obtained when the policy space is unconstrained. In sections 2 and 3 we prove that, when restricting the solutions to this class, the average cost can never be more than 6% higher than can be achieved following any other nested policy. Other simple policies, such as 3^k or 5^k, do not have as tight a worst case bound.

The remainder of this chapter is organized as follows. In the next section we introduce some notation and present the powers–of–two extension of Harris' single item, single stage EOQ problem. We then examine several different multi–stage generalizations of the powers–of–two model. In section 3 we study a system having a serial structure. That section contains some mathematical background that applies to all subsequent models as well as detailed proofs showing why the average annual cost of following the powers–of–two policy cannot exceed the cost for any other policy by more than 6%. In section 4 we examine a system with an assembly system structure and in section 5 we discuss a model and an algorithm for a distribution system. Section 6 contains a model of a general system structure. All the other models described in this chapter are special cases of this model. In section 7 we again examine a general system structure, but we also consider the effect of constraints on available time for setup and production in each workcenter found in a manufacturing and distribution system.

2. POWERS–OF–TWO SOLUTION FOR THE SINGLE ITEM, SINGLE STAGE DETERMINISTIC LOT SIZING PROBLEM

The purpose of this section is to introduce some basic concepts and nomenclature and to restrict Harris' EOQ problem to a powers–of–two lot–sizing problem. We use the single item, single stage deterministic lot size model to develop the powers–of–two lot–sizing approach because it is so simple and permits us to introduce concepts without cumbersome notation. We will show how optimal powers–of–two solutions can be found and will compare the solution to the one obtained when solving Harris' classic EOQ problem. Many of the ideas put forth here are basic to the more complicated ones that will be subsequently developed in the following sections.

2.1 Harris' EOQ Model

We begin by reviewing the classic EOQ problem. Recall that this problem is one of determining the constant reorder quantity that minimizes the average annual cost of purchasing and carrying inventory. The assumptions underlying this problem are as follows:

1) The demand rate λ is constant and continuous. λ is measured in units per year.

2) No backordering is allowed.

3) Production is instantaneous, or equivalently, the production rate is infinite.

4) A fixed cost K is incurred whenever an order is placed.

5) An inventory holding charge is incurred proportional to the on hand inventory. For each unit held in stock for one year the holding cost is h dollars.

6) All other costs are assumed to be independent of the ordering decision.

7) The lead time is zero.

Based on these assumptions the classic EOQ lot sizing model is given by

$$z = \min_{Q \geq 0} \frac{\lambda K}{Q} + \frac{1}{2} hQ \,, \tag{1}$$

where Q is the constant reorder quantity. This problem is known to be a useful mathematical approximation for determining lot sizes. However, rather than analyzing it directly, we choose to reformulate it in terms of reorder intervals rather than reorder quantities, where a reorder interval is the time between successive production runs. Obviously there is a relationship between the reorder intervals and the reorder quantities. If T is the reorder interval, then clearly $T = Q/\lambda$. Hence Problem (1) can be restated in terms of T as follows:

$$z = \min_{T \geq 0} \frac{K}{T} + \frac{1}{2} \lambda h T . \tag{2}$$

We call this problem the economic reorder interval problem. Letting $g = \frac{1}{2} \lambda h$, its solution is

$$T^* = \sqrt{\frac{K}{g}} \tag{3}$$

and

$$z^* = 2\sqrt{Kg} . \tag{4}$$

Thus the optimal solution is to place an order every T^* time units. Since the lead time is zero, the optimal policy is obviously to place and receive an order only when the on hand inventory level is zero.

It is well known that the average annual cost is relatively insensitive to the choice of T. For example, if $T = 2T^*$, then the corresponding average annual cost exceeds z^* by only 25%. The robustness of the cost to the value of T is an important factor affecting the usefulness of the powers–of–two model.

2.2 Powers–of–Two Restriction of the Economic Reorder Interval Problem

Since the value of T^* can be any positive real number, the solution to Problem (2) is often impractical to implement. For example, T^* might equal $\sqrt{2}$ weeks. Typically there are practical reasons why orders can be placed only in certain intervals, such as a day, a week, etc. We assume a minimum reorder interval exists, which we called a *base planning period* in section 1. We denote this base planning period by T_L. It can be a shift, day, week, or other appropriate time period.

We assume that the reorder interval T must be a power of two (i.e., 1, 2, 4, 8, ... , 2^k , ... , where k is a nonnegative integer) times the base planning period. That is,

$$T = 2^k \cdot T_L, \quad k \in \{0,1,...\} .$$

As we will now see, when restricting ourselves to a policy of this form we can never have a solution whose average annual cost exceeds z^* by more than about 6%.

The powers–of–two reorder interval problem is

$$z_c = \min_{T \geq 0} \quad \frac{K}{T} + gT \tag{5}$$

$$T = 2^k T_L, \quad k \in \{0,1,...\} .$$

To find its solution we employ a first differencing argument. Let

$$f(T) = \frac{K}{T} + gT .$$

Note that if T^* is given by (3) and $T = \alpha T^*$ then $f(T) = \frac{1}{2}(\alpha + \frac{1}{\alpha})f(T^*)$.

Since $f(T)$ is a convex function, we find the solution to Problem (5) by finding the smallest nonnegative integer value k for which

$$f(2^{k+1} \cdot T_L) \geq f(2^k \cdot T_L) .$$

This condition reduces to finding the smallest nonnegative integer k for which

$$2^k \geq \frac{1}{T_L}\sqrt{\frac{K}{2g}} = \frac{1}{\sqrt{2}T_L} T^* . \tag{6}$$

In what follows, we assume T_L is chosen such that $T^* \geq T_L$. Since k is chosen to satisfy (6), it is clear that

$$\frac{1}{\sqrt{2}} T^* \leq 2^k \cdot T_L < \sqrt{2}T^* . \tag{7}$$

Therefore, the optimal power–of–two solution must be close to T^*; it must be at least .707 times T^* and no more than 1.41 times T^*. Furthermore, observe that

$$f\left(\frac{T^*}{\sqrt{2}}\right) = f(\sqrt{2}T^*) = (\sqrt{2} + \frac{1}{\sqrt{2}})\sqrt{Kg}$$

$$= (\sqrt{2} + \frac{1}{\sqrt{2}}) \frac{z^*}{2}$$

$$\cong 1.06z^* . \tag{8}$$

Since $f(T)$ is strictly convex,

$$f(2^k \cdot T_L) \leq f(\sqrt{2}T^*) \cong 1.06z^*$$

or

$$\frac{z_c}{z^*} \leq 1.06 .$$

Thus the powers–of–two solution has an objective function value that must be very close to z^*. The average cost of a powers–of–two solution is better yet. If we assume that $2^k T$ is uniformly distributed over the interval $(T^*/\sqrt{2}, \sqrt{2}T^*)$ then the expected value of z_c/z^* is $\sqrt{.5}\,(3/4 + \ell n\, 2) \cong 1.0205$.

In summary, the powers–of–two restriction of the economic reorder interval problem produces solutions that are similar in both cost and the values of the reorder intervals to those found when solving Problem (1). As we will observe, these results hold in even the most general situations we will discuss. This makes the powers–of–two solutions particularly useful, as we shall observe.

3. SERIAL SYSTEMS

Perhaps the simplest extension to the single stage economic reorder interval problem is the serial system reorder interval problem. Rather than having a single production stage, we now assume that there are n such stages, as displayed in Figure 1.

FIGURE 1

Graph of a Serial System

This graph indicates that each unit that is produced of a single item must go through n distinct stages, beginning with stage n and ending with stage 1. The problem is to

determine the reorder intervals for each stage.

The assumptions on which our analysis is performed are relatively obvious extensions of the ones made in the previous sections and account for the presence of n rather than one production stage. We will next introduce some notation and terminology. Some of the notation may seem cumbersome and perhaps unnecessarily complicated to describe this simple problem. We have chosen to introduce it here because it is more easily interpreted in this case and will make the material in subsequent sections more easily understood.

First, we let G represent the directed graph representing the serial production system with N(G) the node set and A(G) the arc set corresponding to G. The graph G is analogous to a bill of material network. The elements of N(G) represent the production stages and the elements in A(G) indicate the precedence constraints implying the order in which operations must be performed. In this serial system $N(G) = \{1, \ldots, n\}$ and $A(G) = \{(n, n-1), (n-1, n-2), \ldots, (2,1)\}$. For each $i \in N(G)$, λ_i represents the total demand rate for the units produced at stage i. Note that λ_i need not be the same for all stages which would allow us, for example, to consider situations in which several units at stage j are required to produce one unit at stage $j-1$. For simplicity, however, in this section we assume $\lambda_i = \lambda$ for all $i \in N(G)$.

Next, let T_i represent the reorder interval for stage i. As in the single stage model, T_i is expressed as a powers–of–two multiple of the base planning period, T_L.

The costs considered in the model are fixed setup costs K_i and holding costs h_i, for all $i \in N(G)$. The holding costs are incremental echelon holding costs, i.e., $h_i = h_i' - h_{i+1}'$ where h_i' is the conventional holding cost at stage i. These echelon holding costs are charged for stage i proportional to the inventory on hand in stage i and all its successor stages. That is, the holding costs for stage i are charged on the inventory on hand in stages 1 through i. Lastly, we let $g_i = h_i\lambda/2$, the average yearly echelon holding cost for stage i when $T_i = 1$. Before developing the model, we discuss why echelon inventory holding costs are used rather than the usual holding costs. We also discuss the form of the optimal policy.

3.1 Echelon Inventory and Nested Policies

Let us first examine a two stage system to gain an understanding of the importance of using echelon inventory as a basis for charging holding costs and also to establish the form of the policies that will be used.

We restrict our attention to policies that are both stationary and nested, as discussed in section 1. Although stationary and nested policies need not be optimal for all types of production systems, they are optimal for the serial system we are now examining. To see why a nested policy is optimal consider a two stage system.

Suppose production occurs at time t at stage 2 while no production occurs at stage 1. Suppose that $t' > t$ is the first time following t that production occurs at stage 1. Hence the inventory produced at time t at stage 2 must be held until at least time t' before it is used at stage 1. Consider an alternative production plan in which the production at stage 2 that occurred at t is postponed and is initiated at t' instead. All other production times remain unchanged. Since the number of setups in the two plans is the same and the holding costs are lower in the second one, it is obvious that it is preferable to produce at stage 2 only when production occurs at stage 1.

A formal proof of this assertion is easily constructed for an n stage system by following an argument similar to the one we have stated. We do not present such a proof but state

THEOREM 1. For an n stage serial system it is optimal to follow a nested policy.

For a proof of this result see Schwarz (1973).

We note, however, that it is possible to have production at stage $i-1$ without having production at stage i. Thus $T_i \geq T_{i-1}$.

Returning to our two stage example, suppose that $T_1 = 1/2\ T_2$. Then the on–hand inventory graphs for stages 1 and 2 are given in Figure 2. Note the

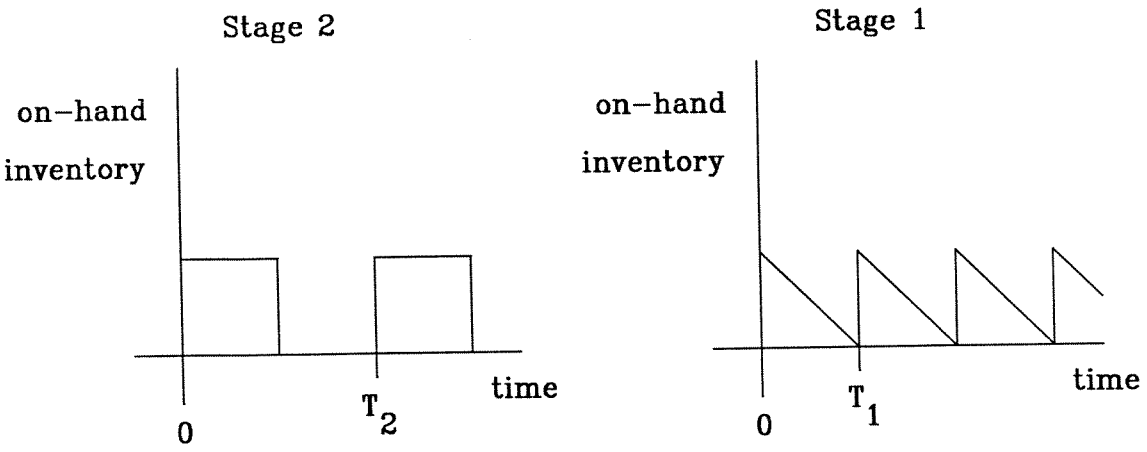


FIGURE 2

Graphs of On–Hand Inventory

shapes of these graphs. The graph for stage 1 is the usual saw–toothed curve that is found in the single stage system. However, the graph for stage 2 is not of this form. Furthermore, the average on–hand inventory is not $\frac{\lambda T_2}{2}$, but is, in this case, $\frac{\lambda T_2}{4}$. In fact the average on–hand inventory at stage 2 is a function of both T_1 and T_2. On the other hand, the average stock on hand at stage 1 is always $\frac{\lambda T_1}{2}$, independent of the choice of T_2. The fact that the average on–hand inventory at stage 2 depends on both T_1 and T_2 makes it undesirable to calculate holding costs using the on–hand inventories.

Now suppose we construct the graphs for echelon stock. The echelon stock at stage 1 is the same as the graph of on–hand inventory. The graph of stage 2 echelon stock is quite different. The echelon stock for stage 2 consists of the stock on hand

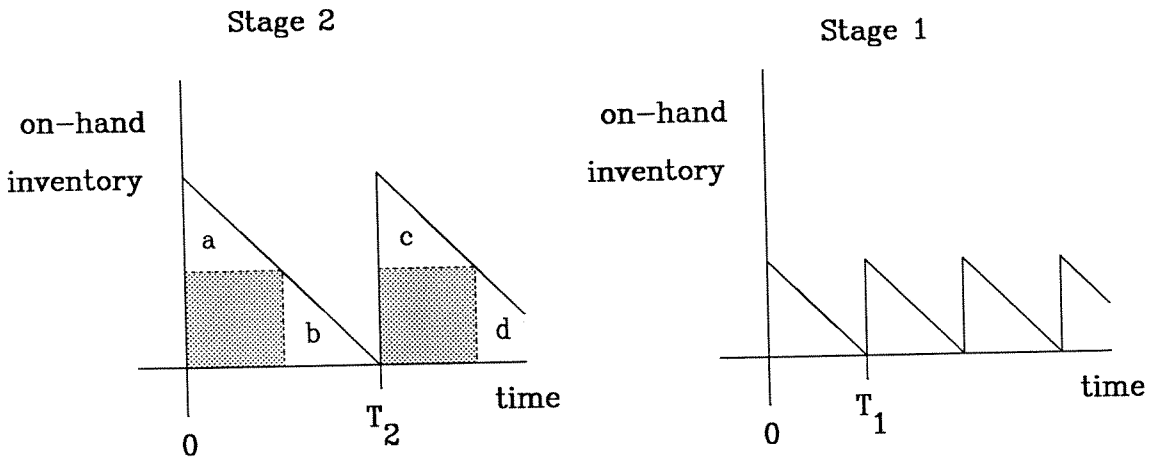


FIGURE 3

Graphs of Echelon Stock

at stage 2, the shaded area in Figure 3, plus the amount on hand at stage 1, represented by triangles a, b, c, and d in the graph. Note that the graph of echelon stock in both stages is saw–toothed in shape.

The average echelon inventory level at stage 2 depends only on T_2. A similar statement applies to stage 1. Thus if inventory costs are charged proportional to echelon inventory levels, then the average costs are $h_2 \cdot \frac{\lambda T_2}{2}$ and $h_1 \cdot \frac{\lambda T_1}{2}$ for stages 2 and 1, respectively. This corrects the serious deficiency observed earlier when attempting to charge holding costs based on average on–hand inventory. But what do these costs measure? How do they compare with the holding costs based on on–hand inventory?

Returning to Figure 2, we see that holding costs based on on–hand stock are equal to $(h_1 + h_2) \cdot \frac{\lambda T_1}{2}$ for stage 1 and $h_2 \cdot \frac{\lambda T_2}{4}$ for stage 2. The echelon holding cost for stage 1 is $h_1 \cdot \frac{\lambda T_1}{2}$ and for stage 2 is $h_2 \cdot \frac{\lambda T_2}{2} = h_2 \lambda T_1 = h_2 \cdot \frac{\lambda T_1}{2} + h_2 \cdot \frac{\lambda T_1}{2} = h_2 \cdot \frac{\lambda T_2}{4} + h_2 \cdot \frac{\lambda T_1}{2}$. Hence the total carrying costs are the same whether they are

calculated using echelon or on–hand inventories. For obvious reasons we choose to use echelon stock as the basis for calculating holding costs.

Clearly, $\lambda T_i/2$ is the average echelon stock for stage i in any serial system. It is also true that $\lambda T_i/2$ is the average echelon stock for node $i \in N(G)$ for the more general graphs examined later in this chapter.

3.2 A Reorder Interval Model and Its Relaxation

Using the definitions, assumptions, form of the policies considered, and the echelon inventory method for calculating holding costs, we see that the reorder interval model is

$$z_9 = \text{minimize} \quad \sum_{i \in N(G)} [K_i/T_i + g_i T_i]$$

$$T_i = 2^{\ell} T_L, \quad \ell \in \{0, 1, \dots\} \tag{9}$$

$$T_i \geq T_{i-1} \geq 0 .$$

This problem is a nonlinear, integer programming problem. The integer decision variable is ℓ. However, due to its special structure we can easily solve it even when G is an arbitrary acyclic graph. As was the case for the single stage system, Problem (9) turns out to have a very close relationship to its following relaxation:

$$z_{10} = \text{minimize} \quad \sum_{i \in N(G)} [K_i/T_i + g_i T_i]$$

$$T_i \geq T_{i-1} \geq 0 . \tag{10}$$

We now characterize the solution to Problem (10) and show its relationship to Problem (9).

3.3 Characterizations of the Optimal Solution to Problems (9) and (10)

We begin by establishing the correspondence between the solutions to Problem (9) and the ordered partitions of the graph G. These relationships are important for the more general graphs we shall discuss subsequently. Consequently, we use definitions that are more general than needed here.

Define a *subgraph* G' of G to consist of a node set $N(G') \subset N(G)$ together with an arc set $A(G') \subset A(G)$ and where $(i,j) \in A(G')$ if and only if $i,j \in N(G')$. An ordered collection of subgraphs $(G_1, \ldots, G_N)$ of G is said to be *ordered by precedence* if for any $1 \leq \ell < k \leq N$ there does not exist a node $j \in N(G_\ell)$ and a node $i \in N(G_k)$ such that $(j,i) \in A(G)$. Consequently, there is no node in $N(G_\ell)$ that precedes any node in $N(G_k)$, $k > \ell$. We say that a collection of subgraphs $(G_1, G_2, \ldots, G_N)$ forms an *ordered partition* of G if

i) the node subsets form a partition of $N(G)$

and ii) the collection of subgraphs is ordered by precedence.

For a serial system, an ordered partition of G has the form $N(G_1) = \{1, \ldots, n_1\}$, $N(G_2) = \{n_1 + 1, \ldots, n_2\}, \ldots, N(G_N) = \{n_{N-1}, \ldots, n\}$.

Next we define a directed cut of the subgraph G' to be an ordered (binary) partition (G'^-, G'^+) of G'.

Suppose we have a subgraph G'. Consider the following relaxed problem

minimize $\sum_{i \in N(G')} [K_i/T_i + g_i T_i]$

subject to

$$T_i \geq T_{i-1} \geq 0 \text{ , for all } (i, i-1) \in A(G') \text{ .} \tag{11}$$

Suppose further that all reorder intervals T_i, $i \in N(G')$, must be equal; that is, $T_i = T$, $i \in N(G')$. The optimal value of T is

$$T = \left[\frac{\sum_{i \in N(G')} K_i}{\sum_{i \in N(G')} g_i} \right]^{1/2} .$$

If we let $K(G') = \sum_{i \in N(G')} K_i$ and $g(G') = \sum_{i \in N(G')} g_i$, then

$$T = [K(G')/g(G')]^{1/2} .$$

Suppose next that we have a feasible solution to Problem (10), and suppose there are N distinct reorder intervals associated with this solution. We denote these N reorder intervals by $T^*(1) < T^*(2) < \ldots < T^*(N)$. Then for each reorder interval there must correspond a subset of the production stages sharing that value. Let $G_1, \ldots, G_N$ represent the subgraphs that correspond to these N distinct intervals. Since $\bigcup_{i=1}^{N} N(G_i) = N(G)$ and the subgraphs are disjoint, these subgraphs form an ordered partition of G. In an optimal solution to Problem (10) we must also have $T^*(k) = [K(G_k)/g(G_k)]^{1/2}$, for $k = 1, \ldots, N$.

Conversely, let $(G_1, \ldots, G_N)$ be an ordered partition of the graph G. Also, let $T^*(k) = [K(G_k)/g(G_k)]^{1/2}$, $k = 1, \ldots, N$, and assume $T_i^* = T^*(k)$ for all $i \in N(G_k)$. Then this solution is feasible to Problem (10) if $T^*(1) \leq T^*(2) \leq \ldots \leq T^*(N)$, or equivalently

$$\frac{K(G_1)}{g(G_1)} \leq \frac{K(G_2)}{g(G_2)} \leq \ldots \leq \frac{K(G_N)}{g(G_N)}.$$

The relationship between the optimal solutions to Problem (10) and the ordered partitions of G is given by

THEOREM 2. Suppose we have an arbitrary collection of N reorder intervals, $T^*(1), \ldots, T^*(N)$. The following conditions are necessary and sufficient for these reorder intervals to provide an optimal solution to Problem (10).

i) There exists an ordered partition $(G_1, \ldots, G_N)$ of G such that

$$T^*(k) = [K(G_k)/g(G_k)]^{1/2}$$

ii) $T^*(1) \leq T^*(2) \leq \ldots \leq T^*(N)$

iii) For each $k = 1,\ldots,N$, there does not exist a directed cut (G_k^-, G_k^+) of G_k for which

$$\frac{K(G_k^-)}{g(G_k^-)} < \frac{K(G_k^+)}{g(G_k^+)}.$$

The proof of this Theorem can be found in Jackson, Maxwell, and Muckstadt (1988).

Any ordered partition of G that satisfies the three above conditions is an *optimal partition.* These conditions also provide a means for identifying optimal solutions for the more general graphs discussed in the succeeding sections of this chapter. The node sets $N(G_1), N(G_2), \ldots, N(G_N)$ are called *clusters.* Note that the solution to (10) can easily be constructed once the clusters are known. In the following section we give an algorithm that solves (10) for a serial system by finding the clusters. In section 6 we show how a standard network algorithm can be used to find the clusters for an arbitrary acyclic graph G.

Now suppose we are given an optimal partition $G_1, \ldots, G_N$ of the serial system graph G. Let $T^*(k)$ be the corresponding optimal solution. Now let us see how we can find the optimal powers–of–two solution to Problem (9). Let $T_i = T(k)$, $i \in N(G_k)$, $k = 1, \ldots, N$, where we find $T(k)$ by solving

$$\begin{aligned} &\text{minimize} && \sum_{i \in N(G_k)} [K_i/T(k) + g_i T(k)] \\ &\text{subject to} && T(k) = 2^{\ell} T_L \,, \quad \ell \in \{0, 1, \ldots\} \,. \end{aligned} \tag{12}$$

$T(k)$ is found using a first differencing approach as discussed for the single stage case. Thus $T(k) = 2^{\ell} T_L$, where ℓ is the smallest nonnegative integer for which

$$2^{\ell} \geq \frac{1}{T_L} \left[\frac{K(G_k)}{2\ g(G_k)} \right]^{1/2} = \frac{1}{\sqrt{2} T_L} T^*(k) \,. \tag{13}$$

Observe that $T^*(k-1) < T^*(k)$ implies that $T(k-1) \leq T(k)$, assuming that ℓ is chosen using (13) for all clusters k. Therefore an optimal solution to (13) satisfies the precedence constraint in Problem (9). Hence this solution is feasible for (9). Furthermore, by applying the fact that there are no directed cuts for G_k satisfying the conditions of part iii) of

Theorem 2, we can show that this solution is *optimal* for Problem (9). The details proving that this ordered partition is optimal can be found in Maxwell and Muckstadt (1985).

3.4 An Algorithm for Finding the Optimal Ordered Partition for a Serial System

The following algorithm can be used to find the optimal ordered partition for a serial system. It consists of three main steps. In the first, we find the clusters, or equivalenly, an optimal partition of G; in the second, we solve problem (10); and in the third, we solve Problem (9). Given an arbitrary node set C, we define $T^*(C) = [(\sum_{i\in C} K_i)/(\sum_{i\in C} g_i)]^{1/2}$.

ALGORITHM 1. SERIAL SYSTEMS.

STEP 1. Find an Optimal Partition of G.

a. Set $C^i \leftarrow \{i\}$ and $\sigma(i) \leftarrow i-1$ for all $1 \le i \le n$, and $S \leftarrow \{1,2,\ldots,n\}$. Set $j \leftarrow 2$.
Note: $\sigma(i)$ is the node that precedes i in the sequence S.

b. If $T^*(C^j) \ge T^*(C^{\sigma(j)})$, go to Step 1d; otherwise, collapse $C^{\sigma(j)}$ into C^j by setting $C^j \leftarrow C^{\sigma(j)} \cup C^j$, $\sigma(j) \leftarrow \sigma(\sigma(j))$, and $S \leftarrow S\backslash\{\sigma(j)\}$.

c. If $\sigma(j) > 0$, go to Step 1b.

d. Set $j \leftarrow j+1$. If $j \le n$, go to Step 1b.

e. Re–index the clusters $\{C^i: i \in S\}$ so that $S = \{1,2,\ldots,N\}$ and if $j \in C^i$, $k \in C^\ell$, and $j < k$ then $i < \ell$.
Comment: $\{C^k: k \in S\}$ are the clusters. The optimal partition is $\{G_k: k \in S\}$ where G_k is the subgraph of G induced by C^k. Thus $C^k = N(G_k)$.

STEP 2. Find the Solution to Problem (10).

For each cluster C^k, $k \in S$ set

$$T^*(k) = T^*(C^k) = [(\sum_{i \in C^k} K_i)/(\sum_{i \in C^k} g_i)]^{1/2}.$$

For each $i \in C^k$ set $T_i^* = T^*(k)$.

STEP 3. Find the Solution to Problem (12).

For each $i \in C^k$ set $T_i = 2^{\ell} T_L$ where $2^{\ell} \geq T^*(k)/\sqrt{2}T_L > 2^{\ell-1}$.

It is sometimes desirable to impose a uniform lower and/or upper bound which applies to all reorder intervals. Suppose we add to Problem (9) the constraint

$$2^{\underline{\ell}} T_L \leq T_i \leq 2^{\bar{\ell}} T_L \quad \forall \ i \in N(G).$$

If $i \in C^k$, an optimal solution to this version of (9) is obtained by selecting $T_i = 2^{\underline{\ell}} T_L$ if $T^*(k) \leq 2^{\underline{\ell}} T_L$, $T_i = 2^{\bar{\ell}} T_L$ if $T^*(k) \geq 2^{\bar{\ell}} T_L$, and selecting T_i as in Step 3 above if $2^{\underline{\ell}} T_L \leq T^*(k) \leq 2^{\bar{\ell}} T_L$. If this is done all claims of optimality and near–optimality made in the sequel still apply.

An alternate procedure for performing Step 3 is found in (Roundy 1986). This procedure has the advantage of producing policies that are within 2% of optimal, rather than the 6% bound that results from Step 3 above. However, this procedure requires that T_L be treated as a variable. For many systems the base planning period is determined by the times at which information is reported and acted upon, and cannot be treated as a variable.

The relaxation (10) of problem (9) was first formulated and solved in a more general setting by Schwarz and Schrage (1975). The system myopic policies they proposed use a rounding scheme that is similar to Step 3, but they do not use the same values of T^*.

3.5 Analysis of Worst–Case Behavior of the Algorithm

To establish the worst case behavior of the algorithm we calculate an upper bound on the optimal objective function value to Problem (9) and compare it to a lower bound. The lower bound is found by solving Problem (10), as we will subsequently demonstrate. We carry out this worst–case analysis for the serial system in detail. A similar analysis can be performed for other graph structures. However, we will not conduct this analysis for these cases but will only assert the conclusions that would result from such an analysis.

Suppose we have an optimal partition of G found using the algorithm described in the previous section. Let $(G_1, \ldots, G_N)$ represent this partition. Then

$$T_i^* = T^*(k) \text{ for all } i \in N(G_k),$$

where

$$T^*(k) = [K(G_k)/g(G_k)]^{1/2}, \text{ and}$$

$$z_{10} = \sum_{k=1}^{N} 2\sqrt{K(G_k) \cdot g(G_k)}\,.$$

Recall that $T(k)$, $k = 1, \ldots, N$, represents the optimal solution to Problem (9), where

$$T(k) = 2^{\ell} \cdot T_L$$

and ℓ is the smallest nonnegative integer satisfying

$$2^{\ell} \geq \frac{1}{\sqrt{2}T_L}\sqrt{\frac{K(G_k)}{g(G_k)}} = \frac{1}{\sqrt{2}T_L}\,T^*(k)\,.$$

Then

$$\frac{T^*(k)}{\sqrt{2}} \leq T(k) < \sqrt{2}T^*(k) .$$

Let

$$f_i(T_i) = \frac{K(G_k)}{T_i} + g(G_k) \cdot T_i, \ i \in N(G_k) .$$

Since $f_i(\cdot)$ is convex and

$$f_i\left[\frac{T^*(k)}{\sqrt{2}}\right] = f_i\left[\sqrt{2}T^*(k)\right] = \left[\sqrt{2} + \frac{1}{\sqrt{2}}\right]\sqrt{K(G_k) \cdot g(G_k)} ,$$

we have

$$f_i(T(k)) \leq \left[\sqrt{2} + \frac{1}{\sqrt{2}}\right]\sqrt{K(G_k) \cdot g(G_k)} .$$

Hence the solution to Problem (9) satisfies

$$z_9 = \sum_{k=1}^{N} \{ \sum_{i \in N(G_k)} [K_i/T(k) + g_i T(k)] \}$$

$$\leq \left[\sqrt{2} + \frac{1}{\sqrt{2}} \right] \sum_{k=1}^{N} \sqrt{K(G_k) \cdot g(G_k)}$$

$$= \frac{\left[\sqrt{2} + \frac{1}{\sqrt{2}} \right]}{2} z_{10}$$

so that

$$\frac{z_9}{z_{10}} \leq 1.06 .$$

This is the same result we obtained in the single stage model and one that holds for all the models discussed in this chapter.

We now show that z_{10} is a lower bound on the long run minimum average cost attainable for a nested policy. This result also holds for more general cases as well.

Let $z(t)$ represent the minimal holding and setup costs incurred over an interval of length t given a nested production plan is followed. We will show that

$$z_{10} \leq \lim_{t \to \infty} z(t)/t .$$

Let $n_i(t)$ be the number of setups for stage i through time t, and let τ_{ik} represent the time at which the kth setup occurs for stage i. We assume $\tau_{i0} = 0$, $i \in N(G)$. Since the production plan is nested,

$$\tau_{ik} = \tau_{i-1,k'}$$

for some $k' \geq k$, $k = 1, \ldots, n_i(t)$.

Now let h_i' represent the holding cost per year for on–hand inventory held at stage i. If τ_i represents the sequence of setup times for stage i over the horizon of length t, we define $I_i(u; \tau_i, n_i(t))$ to be the on–hand inventory at stage i at time u, $u \in [0,t]$, given policy τ_i and $n_i(t)$ setups over the interval of length t. Furthermore, let $E_i(u; \tau_i, n_i(t))$ represent the corresponding echelon inventory at time u, $u \in [0,t]$. Thus the total holding cost during the interval is

$$\sum_{i \in N(G)} h_i' \int_0^t I_i(u; \tau_i, n_i(t))\, dt \,. \tag{14}$$

As we discussed earlier, (14) can be restated as

$$\sum_{i \in N(G)} h_i \int_0^t E_i(u; \tau_i, n_i(t))\, dt. \tag{15}$$

Let us temporarily ignore the fact that we must follow a nested policy. Since there are $n_i(t)$ setups for stage i, it is easy to prove that the time between setups should be equal if holding costs are to be minimized at stage i. The echelon holding costs over the planning period for stage i, assuming equally spaced setups of length $T_i = t/n_i(t)$ time units, is $g_i T_i t$. Since this is a lower bound on echelon holding costs for stage i,

$$h_i \int_0^t E_i(u; \tau_i, n_i(t))\, du \geq g_i T_i t \,.$$

The setup cost for stage i over [0,t] is $n_i(t) \cdot K_i = \frac{t}{T_i} K_i$. Thus

$$z(t) = \underset{n_i(t),\, \tau_{ik}}{\text{minimum}} \sum_{i \in N(G)} \left(n_i(t) \cdot K_i + h_i \int_0^t E_i(u;\, \tau_i,\, n_i(t))du\right),$$

given τ_i is nested

$$\geq \underset{T_i}{\text{minimum}} \left\{ \sum_{i \in N(G)} \left[\frac{K_i}{T_i} + g_i T_i\right] \cdot t:\ T_i \geq T_{i-1} \geq 0 \right\}$$

$$= z_{10} \cdot t\ .$$

Thus $z_{10} \leq \frac{z(t)}{t}$ for all $t > 0$ and therefore $z_{10} \leq \lim_{t \to \infty} z(t)/t$.

3.6 Summary

A number of key ideas introduced in this section will re–appear in several later sections. For this reason we summarize some of the most important elements of the analysis we have just described.

The problem of finding a powers–of–two stationary nested policy with minimal cost was formulated as the nonlinear integer program, Problem (9). The variables in this problem are the reorder intervals. A near–optimal policy was obtained by solving the continuous relaxation of this problem, and rounding off the reorder intervals to get a feasible, stationary, nested powers–of–two policy.

Two facts about the solution to the continuous relaxation, Problem (10), are noteworthy. The first is that the bill of material network G is partitioned into subgraphs. The set of nodes or stages in a given subgraph was called a *cluster*. All stages in a cluster

use the same reorder interval. Local optimality implies that the reorder interval for the stage in cluster C is $[(\sum_{i\in C} K_i)/(\sum_{i\in C} g_i)]^{1/2}$. Thus finding the clusters is equivalent to solving Problem (9).

The second fact is that the solution to Problem (10) was shown to be a lower bound on the average cost of any feasible policy for the original system, including policies that are neither powers–of–two nor stationary. We showed that the cost of the feasible policy we compute is within 6% of the solution of Problem (9). Because nested policies are optimal in serial systems, our heuristic is guaranteed to compute a policy whose cost is at most 6% above the cost of an optimal policy.

Optimization problems similar to Problems (9) and (10) will appear in every section of this chapter. The relationship between them, the policies, and bounds on the cost performance of those policies is similar in every section.

4. ASSEMBLY SYSTEMS

We next consider a multi–stage system in which a single finished product is assembled from a set of parts Each part could be manufactured in several stages and could also be assembled from several other parts. Such systems are called assembly systems and have bill of material networks that are arborescences. Figure 1 is an example of a graph G of this type of system. For ease of discussion, we assume that each node in N(G) represents the fabrication of a part. Hence node i corresponds to the stage in which part i is manufactured.

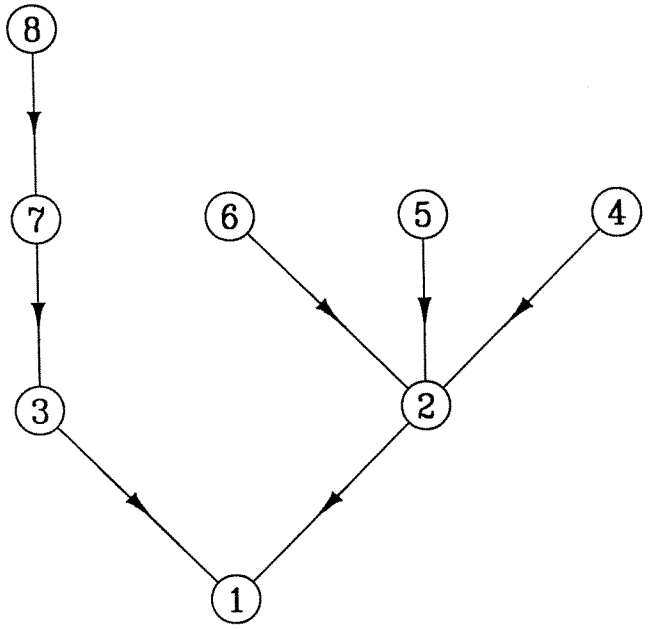


FIGURE 4

An Assembly System

We assume that each part $i > 1$ is consumed at a unique *direct successor* stage. We define part 1 to be the finished product. It has no successor stage; instead, it is consumed by external demand, which, as before, is constant, continuous, and must be met without backlogging.

The other assumptions made in section 2 regarding costs and machine times hold here as well. Furthermore, our goal is to determine reorder intervals for each part so that the long–run average annual setup plus holding costs are minimized.

4.1 Stationary Nested Power–of–Two Policies and Their Costs

As before, we restrict attention to stationary nested powers–of–two policies. In section 3 we argued that nested policies are optimal for serial systems. Each stage of an assembly system has either no external demand and a unique successor, or no successor and

external demand. Therefore, the same argument could be used to show that nested policies are optimal for assembly systems.

Let us introduce some additional notation. Let s_i be the direct successor of component i, p_i be the set of direct predecessors of component i, S_i be the set composed of i and all of its successors, direct or indirect, and let P_i be similarly defined. We assume without loss of generality that if $j \in p_i$, then one unit of j is consumed for each unit of i ordered.

Recall that the *echelon holding cost* for part i is h_i and is the amount by which the holding cost of part i exceeds the sum of the holding costs of its direct predecessors. It is assumed that the echelon holding costs of all of the parts are positive. The *echelon inventory* E_i^t of part i at time t is the sum of the inventories of all j, $j \in S_i$. Using the same argument in section 3, the total rate at which holding costs are being incurred at time t is $\sum_i h_i E_i^t$.

Let the stages of the assembly system be numbered 1 through n. We denote the powers–of–two policy by $\mathcal{T} = (T_i,\ 1 \le i \le n)$. Recall that a *nested* policy is a policy in which an order is placed at the successor of stage i simultaneously with every order at i. Hence in this case, $\mathcal{T}$ is nested if and only if

$$T_i \ge T_{s_i} \quad \text{for all } i\,. \tag{16}$$

It is easy to show that an optimal policy is nested and has the following property:

THE ZERO–ORDERING PROPERTY. Orders are placed at stage i only when the inventory of i is zero.

In the sequel we will restrict attention to stationary nested powers–of–two policies that satisfy the Zero–Ordering Property. This implies that the average cost of $\mathcal{I}$ can be written as

$$c(\mathcal{I}) = \sum_i (K_i/T_i + g_i T_i) \tag{17}$$

where $g_i = h_i\lambda/2$ and λ is the demand rate.

4.2 Structural Results and Computation

The problem of finding an optimal stationary nested powers–of–two policy can therefore be written as

$$\text{minimize} \quad \sum_i (K_i/T_i + g_i T_i)$$

$$\text{such that} \quad T_i \geq T_{s_i} \text{ for all } i, \text{ and} \tag{18}$$

$$T_i = 2^{\ell} T_L, \ \ell = 0,1,2,\ldots, \ T_L > 0 \, . \tag{19}$$

We call this Problem (18) and solve it using the approach of section 3. The first step in solving it is to relax (19) and solve

$$\text{minimize} \quad \sum_i (K_i/T_i + g_i T_i)$$

$$\text{such that} \quad T_i \geq T_{s_i} \text{ for all } i \, . \tag{20}$$

This is done for two reasons. First, it will give us a set of approximate reorder intervals that we will round off to integer powers–of–two. Second, it will give us a lower bound on the average cost of an arbitrary policy for this problem. The lower bound is used in evaluating our heuristic.

As we discussed in section 3, the solution to Problem (20) partitions the network G of Figure 1 into connected subgraphs. The nodes in these connected subgraphs are sets of stages whose costs induce them to place orders simultaneously. As before, we call these node sets *clusters.* We define the *root* of a cluster C to be the unique stage $i \in C$ satisfying $C \subseteq P_i$.

Let C^i be the cluster whose root is stage i, $K(C^i) \equiv \sum_{j \in C^i} K_j$ be the total setup cost for cluster i, and $g(C^i) \equiv \sum_{j \in C^i} g_j$ be the total echelon holding cost for cluster i. For $j \in C^i$ we define $K(j) \equiv \sum_{k \in P_j \cap C^i} K_k$ and $g(j) \equiv \sum_{k \in P_j \cap C^i} g_k$. If $i \neq j \in C^i$ and we were to partition cluster i by letting j be the root of a new cluster, the total setup cost of the new cluster rooted at j would be K(j) and the total holding cost would be g(j). The following lemma is the assembly system analog of Theorem 2 in section 3.

LEMMA 1. $\mathcal{T} = (T_i^*: 1 \leq i \leq n)$ solves Problem (20) if and only if there is a partition of the stages into clusters C^i that satisfies the following conditions:

C^i is the node set of a connected subgraph of graph G with root i ,

$$T_n^* = T^*(i) \text{ for all } n \in C^i \text{ and for all clusters } i , \tag{21}$$

$$T^*(i) = (K(C^i)/g(C^i))^{1/2} \text{ for all clusters } i , \tag{22}$$

$$K(C^i)/g(C^i) \geq K(C^j)/g(C^j) \text{ whenever } s_i \in C^j . \tag{23}$$

$$K(j)/g(j) < K(C^i)/g(C^i) \text{ for all } j \in C^i,\ j \neq i . \tag{24}$$

Equation (24) states that if we try to split cluster i in two by letting j be the root of a new cluster, the constraint $T_j^* \geq T_k^*$, $k = s_j$, will be violated. To see this, suppose that we split cluster i. Note that the total setup cost of the portion of cluster i that does not contain j is $K(C^i) - K(j)$ and the total holding cost is $g(C^i) - g(j)$. Since $K(j)/g(j) < K(C^i)/g(C^i)$, (21) and (22) imply that $T_j^{*2} = K(j)/g(j) < (K(C^i) - K(j))/(g(C^i) - g(j)) = T_k^{*2}$, $k = s_j$.

We are now ready to give an algorithm for computing the clusters. This algorithm is known as the *minimum violators algorithm* and was developed to solve the statistical problem of isotonic regression (Thompson (1962), Best and Chakravarty (1988)). Once the clusters have been identified, Steps 2 and 3 of Algorithm 1 are used to compute solutions to problems (20) and (18).

ALGORITHM 2. THE MINIMUM VIOLATORS ALGORITHM.

STEP 1. Create a singleton cluster for each component in the system. Let the set Q initially consist of roots of all clusters other than the one containing the final product. (Subsequently, the set Q will be the set of roots of clusters C^i for which we do not know that (23) holds.) Let $r = \{1\}$ (Subsequently, r will be the set of roots of the clusters that solve Problem (20).).

STEP 2. If Q is empty, then stop; the clusters C^i, $i \in r$ are optimal. Otherwise, find a cluster $i \in Q$ for which $K(C^i)/g(C^i)$ is minimal. In the event that there is a tie, choose a cluster C^i for which $|S_i|$ is minimal. Let $s_i \in C^j$.

STEP 3. Remove i from Q. If $K(C^i)/g(C^i) \geq K(C^j)/g(C^j)$, then add i to r and go to Step 2. If $K(C^i)/g(C^i) < K(C^j)/g(C^j)$, then collapse cluster i into cluster j ($C^j \leftarrow C^j \cup C^i$) and go to Step 2.

The minimum violators algorithm can be implemented in $O(n \log n)$ time by using an appropriate data structure for $\{K(C^i)/g(C^i): i \in Q\}$. At each iteration of the algorithm we remove the member of this set that lexicographically minimizes $(K(C^i)/g(C^i), |S_i|)$. In many of the iterations we alter the value of one of the elements of the set. Using a heap (or any of several other data structures), these two operations can both be performed in $O(\log n)$ time (Aho et al. (1974)). Since the number of iterations is at most $n-1$, the overall running time of the algorithm is $O(n \log n)$. Another algorithm for computing clusters for this problem is found in Schwarz and Schrage (1975). That algorithm can also be implemented to run in $O(n \log n)$ time.

The proof that the Minimum Violators Algorithm computes the optimal clusters is based on equations (21) – (24). The main ideas are as follows.

It is easily verified that the algorithm maintains the following two properties:

$$\bigcup_{i \in Q \cup r} C^i = N(G).$$

$$K(C^i)/g(C^i) \geq K(C^j)/g(C^j) \text{ for all } i \in Q,\ j \in (r \backslash \{1\}). \tag{25}$$

Furthermore, the value of $\min_{i \in Q} K(C^i)/g(C^i)$ is nondecreasing as the algorithm progresses. Consequently, if $K(C^i)/g(C^i) \geq K(C^j)/g(C^j)$ in Step 3, then $j \in r$; if $K(C^i)/g(C^i) < K(C^j)/g(C^j)$, then $j \in Q \cup \{1\}$. Therefore, once $i \neq 1$ is added to r in Step 3, no changes are made to C^i.

When the algorithm terminates the preceding paragraph implies that (23) holds. The proof that (24) holds is found in Roundy (1984b). Equations (21) and (22) follow from Step 2 of Algorithm 1, which is used to compute the T_n^*'s.

By substituting (21) and (22) into the objective function of (20), we see that the solution to Problem (20) is

$$z^* \equiv \sum_{i \in r} 2\sqrt{K(C^i)g(C^i)} \,. \tag{26}$$

As was done for the serial system in section 3, it can be shown that z^* is a lower bound on the average cost of any feasible solution to the original lot sizing problem. When the reorder intervals T_i^* computed by the Minimum Violators Algorithm are rounded off to an integer powers–of–two times T_L, the cost of the resulting policy computed is known to be within 6% of optimal or, if the roundoff procedure that treats T_L as a variable is used, is within 2% of optimal.

5. DISTRIBUTION SYSTEMS

We now consider another special type of graph G called a distribution network. Figure 5 contains an illustration of this type of network. The nodes correspond to stages, and the arcs indicate the direction in which material flows through the system. Material enters the system at stage 1, and moves down through the different levels of the system until it is consumed by external demand. Each stage $i > 1$ is supplied by a unique *predecessor stage,* so the distribution network G is an arborescence.

External demand can occur at any or all of the stages. The demand rate at stage i is positive if i has no successors in the network. Again the basic assumptions made in section 2, and the costs defined there, apply to this case as well. Additionally, the objective is to find a feasible policy for an infinite time horizon which approximately minimizes the

long-run average order and holding costs. This model and similar models have been addressed by several authors (Schwarz (1973), Graves and Schwarz (1977), Williams (1981,1983)).

5.1 Policies

As before, we restrict ourselves to stationary nested powers–of–two policies. For the systems of sections 3 and 4, nested policies were optimal. However for distribution systems, optimal nested policies can be seriously sub–optimal. Suppose for example that our distribution system consists of three stages. Stage one (the factory) supplies two outlets, stages two and three. The factory is in Albany, outlet one is in New York City, and outlet two is in Singapore. The Singapore outlet has very low demand and a very high order cost.

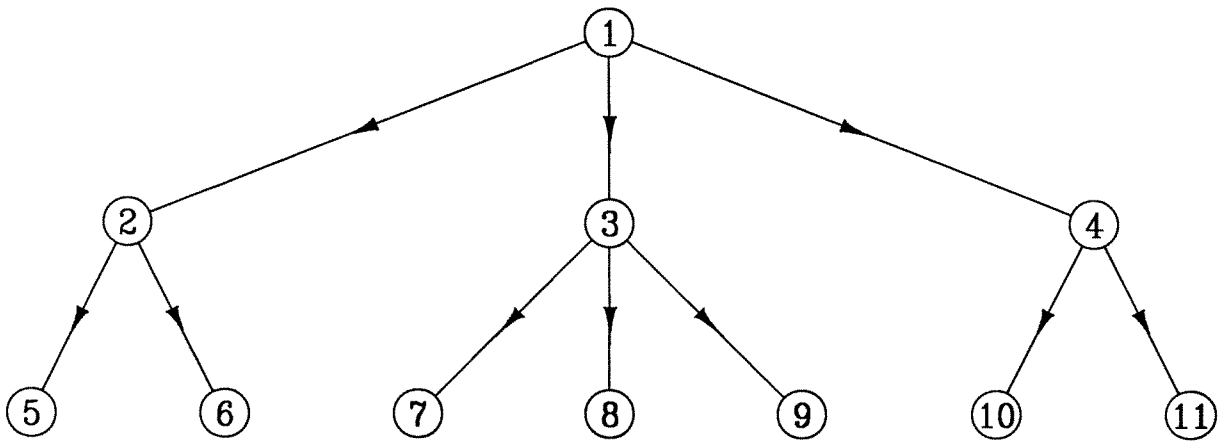


FIGURE 5

A Distribution System

For such a system it might be reasonable to place orders at the factory and at the New York outlet on a weekly basis, and to place orders at Singapore less often, say, once every 32

weeks. However if a nested policy were followed, an order could not be placed at the factory unless a simultaneous order were placed at Singapore. A nested policy would incur either high inventory costs at the factory due to a long reorder interval there, or high order costs at the Singapore outlet due to a low reorder interval there, or both.

As this example illustrates, nested polices tend to become seriously sub–optimal when relatively high order costs are combined with relatively low demand rates. The model of this section is intended for use in situations where all of the stages have fairly high and stable demands. For systems that have stages with low, stable demands and relatively high order costs, the methods of section 9 are more appropriate. For stages that have sporadic demand, or stages for which the inventory is replenished at pre–determined points in time that are beyond our control, our models are not appropriate. These cases will normally be handled by a different model, possibly one that is based on reorder points and/or order quantities.

5.2 Formulation

As we did in section 4, we denote a stationary nested powers–of–two policy by $\mathcal{T} = (T_i: 1 \leq i \leq n)$. Note that stage $i > 1$ in the graph G is supplied by a unique *predecessor stage* p_i. The set of *successor stages* S_i of i includes i and all stages that are directly or indirectly supplied through stage i. Since we consider only nested policies, we require that

$$T_{p_i} \geq T_i \text{ for all } i > 1 , (p_i, i) \in A(G). \tag{27}$$

The *echelon inventory* at stage i at time t is E_i^t, the sum of the inventories at all stages j, $j \in S_i$. Since the *echelon holding cost* h_i at stage i is given by $h_i = h_i' - h_{p_i}'$ where h_i' is the conventional holding cost at stage i, the rate at which holding costs are

accumulating at time t is $\sum_i h_i E_i^t$. The demand for the echelon inventory at stage i is $\lambda_i > 0$, the sum of the external demand rates at the stages j, $j \in S_i$.

As before, we assume that the Zero–Ordering Property of section 4 holds, i.e., that no stage places an order when its inventory level is positive. As in sections 3 and 4, if $\mathcal{I}$ is a stationary nested powers–of–two policy that satisfies the Zero–Ordering Property, then the echelon inventory at stage i follows the traditional sawtooth pattern with a minimum level of zero and a maximum level of $\lambda_i T_i$. Therefore, the average cost incurred by $\mathcal{I}$ is

$$\sum_i (K_i/T_i + g_i T_i) \tag{28}$$

where $g_i = \frac{1}{2}\lambda_i h_i$ as usual. Therefore the problem of finding an optimal stationary nested powers–of–two policy can be written as

$$\text{minimize:} \qquad \sum_i (K_i/T_i + g_i T_i) \tag{29}$$

$$\text{subject to:} \qquad T_{p_i} \geq T_i \quad \text{for all } i > 1, \tag{30}$$

$$T_i = 2^{\ell_i} T_L, \quad \ell_i = 0,1,2,\ldots, \quad T_L > 0\ . \tag{31}$$

We call this Problem (29).

5.3 Solution

As in earlier sections, we relax (31), solve the resulting problem, and then round off the reorder intervals to a powers–of–two times T_L. We will show that the relaxation of Problem (29) can be solved by the algorithm of section 4 via a simple transformation.

The only difference between the relaxation of (29) and Problem (20) of section 4 is the form of the constraint (30). In both cases the constraint takes on the form $T_i \geq T_j$ whenever $(i,j) \in A(G)$. But here the arcs are oriented away from the root of the arborescence, whereas in section 4 they were oriented towards the root.

We transform the relaxation of Problem (29) into an instance of Problem (20) as follows. Let $U_n = 1/T_n$. Then the relaxation can be rewritten as

$$\text{minimize:} \qquad \sum_i (g_i/U_i + K_i U_i) \tag{32}$$

$$\text{subject to:} \qquad U_i \geq U_{p_i} \text{ for all } i > 1 . \tag{33}$$

This problem, which we call Problem (32), is of the form of Problem (20). The roles of the setup cost K_i and the holding cost coefficient g_i have been interchanged. The orientation of the arcs in the distribution network has been reversed to convert it into the bill of material network of an assembly system. Thus, in the language of section 4, p_i is the successor of i. The clusters that solve Problem (32) also solve the relaxation of Problem (29). They can be computed by applying Algorithm 2 of section 4 to Problem (32). Then the reorder intervals that solve Problem (29) are calculated as in Algorithm 1, Steps 2 and 3.

5.4 Performance Analysis

This algorithm computes nested policies, and the ratio of the cost of an optimal nested policy to the cost of an optimal policy can be arbitrarily high. However, the cost of the policy computed by our algorithm can be shown to be within 6% of the cost of an optimal nested policy or, if the base period is treated as a variable, within 2% of the cost of an optimal nested policy.

6. A MODEL OF GENERAL MULTI–STAGE PRODUCTION–DISTRIBUTION SYSTEMS

In this section we present a model for establishing reorder intervals in general multi–stage production–distribution systems. By this we mean that the graph G that describes the system is an arbitrary acyclic directed graph.

Our analysis is based on the assumptions made in section 2. We also continue to assume that only nested and stationary policies are of interest. Since the output rate for all final products is assumed to be constant and continuous, we let λ_i represent the stationary, continuous demand rate for echelon stock for node (operation) i, $i \in N(G)$. As before, h_i represents echelon holding costs, K_i represents the fixed setup costs, and $g_i = h_i\lambda_i/2$.

Based on these definitions and our assumptions, we state the general multi–stage production–distribution system model as

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in N(G)} [K_i/T_i + g_i T_i] \\ \text{subject to} \quad & T_i = 2^{\ell_i} T_L, \ i \in N(G), \\ & T_i \geq T_j, \ (i,j) \in A(G), \\ & \ell_i \text{ integer}, \ i \in N(G). \end{aligned} \tag{34}$$

6.1 Solving Problem (34)

Problem (34) is a large–scale, nonlinear, integer programming problem. In practical situations, the sets $N(G)$ and $A(G)$ could contain many thousands of elements. Consequently, in these cases an optimal solution to Problem (34) cannot be directly obtained using a standard branch and bound algorithm. To circumvent this problem, we

solve it using a two–step procedure. Before presenting the details, we discuss the objectives and approach followed in each step.

In the first step we solve a relaxed version of Problem (34) to establish what groups of operations must have identical reorder intervals. The mathematical formulation of this relaxed problem, which we call Problem (35), is

$$\begin{aligned} &\text{minimize} && \sum_{i \in N(G)} [K_i/T_i + g_i T_i] \\ &\text{subject to} && T_i \geq T_j \geq 0, \ (i,j) \in A(G). \end{aligned} \tag{35}$$

In Maxwell and Muckstadt (1985) it is proven that if the solution to Problem (35) indicates that $T_i = T_j$ for $(i,j) \in A(G)$, then operations i and j share a common reorder interval in the solution to Problem (34). However, the solution to Problem (35) may not indicate all the operations that should have the same reorder interval in the solution to Problem (34). For example, the solution to Problem (35) might state that $T_i = 1.001T_j$ for some $(i,j) \in A(G)$, so that operations i and j would most likely share a common reorder interval in the solution to Problem (34). Thus a second step is needed to find the optimal solution to Problem (34).

The second step of the algorithm uses the solution to Problem (35) to find the optimal solution to Problem (34). This is done using Step 3 of Algorithm 1.

6.1.1 Solving Problem (35)

The Kuhn–Tucker Conditions

The solution to Problem (35) is obtained by recognizing that its solution must satisfy its associated Kuhn–Tucker conditions. Since Problem (35) has a strictly convex objective function and the constraint set is convex, it has a unique optimal solution.

Let us introduce some additional notation. Let $(T_i^*: i \in N(G))$ solve Problem (35), and let

$$P_G(i) = \text{direct predecessor set for node } i \text{ in graph } G,$$
$$S_G(i) = \text{direct successor set for node } i \text{ in graph } G, \text{ and}$$
$$\theta_{ij} = \text{multiplier for constraint } T_i \geq T_j \text{ in Problem (35).}$$

Using these definitions, we can state the Kuhn–Tucker conditions as follows:

$$-K_i/T_i^{*2} + g_i - \sum_{j \in S_G(i)} \theta_{ij} + \sum_{j \in P_G(i)} \theta_{ji} = 0, \quad i \in N(G), \tag{36}$$

$$\left.\begin{aligned} T_i^* &\geq 0, \quad i \in N(G) \\ T_i^* - T_j^* &\geq 0, \quad (i,j) \in A(G) \end{aligned}\right\}, \tag{37}$$

$$\theta_{ij} \geq 0, \quad (i,j) \in A(G), \tag{38}$$

$$\theta_{ij}(T_i^* - T_j^*) = 0, \quad (i,j) \in A(G). \tag{39}$$

Observe that condition (36) implies that in any optimal solution

$$T_i^* = \sqrt{K_i/(g_i - \sum_{j \in S_G(i)} \theta_{ij} + \sum_{j \in P_G(i)} \theta_{ji})}.$$

Thus T_i^* is similar to the solution found for the reorder interval in the classic EOQ problem discussed in section 2; if $\sum_{j \in P_G(i)} \theta_{ji} = \sum_{j \in S_G(i)} \theta_{ij}$, then T_i^* equals the reorder

interval in the EOQ problem. Since multiplier θ_{ij} represents the price for deviating from the economic reorder interval, $\sum_{j \in S_G(i)} \theta_{ij}$ measures the subsidy and $\sum_{j \in P_G(i)} \theta_{ji}$ measures the penalty to i's holding cost induced by i's successor and predecessor operations, respectively.

Also, observe that the multiplier θ_{ij} corresponds to arc $(i,j) \in A(G)$ and hence appears in exactly two constraints of the type (36). Furthermore, the sign of θ_{ij} is different in these two constraints, so that

$$\sum_{i \in N(G)} \left[-\sum_{j \in S_G(i)} \theta_{ij} + \sum_{j \in P_G(i)} \theta_{ji} \right] = 0. \tag{40}$$

Thus, if the T_i^* are chosen to be equal, we find, by summing over the constraints of type (36), that

$$T_i^{*2} = \left[\sum_{i \in N(G)} K_i \right] / \left[\sum_{i \in N(G)} g_i \right], \quad i \in N(G).$$

Let this common value of T_i^{*2} be represented by D(G). Assume the T_i^* are chosen to be equal. If there exists θ_{ij} satisfying (36) and (38), all the Kuhn–Tucker conditions will be satisfied and the optimal solution will have been obtained. Otherwise, all the T_i^* cannot be equal. Finding the optimal solution to Problem (35) in this case requires subdividing G until all the Kuhn–Tucker conditions are satisfied.

6.1.2 Directed and Maximal Cuts

The subdivisions of G are called *directed cuts.* A directed cut is a split of G into two separate acyclic directed graphs, G^+ and G^-, so that

1. $N(G)$ is partitioned into $N(G^+)$ and $N(G^-)$, and
2. For all $(i,j) \in A(G)$ either
   (a) $i,j \in N(G^+)$, and these (i,j) constitute $A(G^+)$,
   (b) $i,j \in N(G^-)$, and these (i,j) constitute $A(G^-)$, or
   (c) $i \in N(G^+)$ and $j \in N(G^-)$.

The directed cut reflects a partition of $N(G)$ into two sets or clusters of nodes or operations, $N(G^+)$ and $N(G^-)$. Each operation in $N(G^+)$ will have a reorder interval of the same length, and each operation in $N(G^-)$ will share a common reorder interval. The notion of a directed cut implies that conditions (38) are satisfied, that is, the common reorder interval for operations in $N(G^+)$ must be at least as large as that for the operations in $N(G^-)$.

Define the value of a directed cut (G^+,G^-) to be

$$v(G^+,G^-) = \sum_{i \in N(G^+)} \{K_i/D(G) - g_i\}.$$

By definition of $D(G)$, $v(G^+,G^-) = \sum_{i \in N(G^-)} \{g_i - K_i/D(G)\}$. Suppose $T^*_{G^+}$ and $T^*_{G^-}$ represent the common reorder intervals for the operations in $N(G^+)$ and $N(G^-)$, respectively, and suppose that $T^*_{G^+} = T^*_{G^-} = \sqrt{D(G)}$. As is clear from the definition of a directed cut, the nodes in G^+ precede those in G^- in the graph G. Because of this property, if $T^*_{G^+}$ is increased above $\sqrt{D(G)}$ or $T^*_{G^-}$ is decreased below $\sqrt{D(G)}$, the

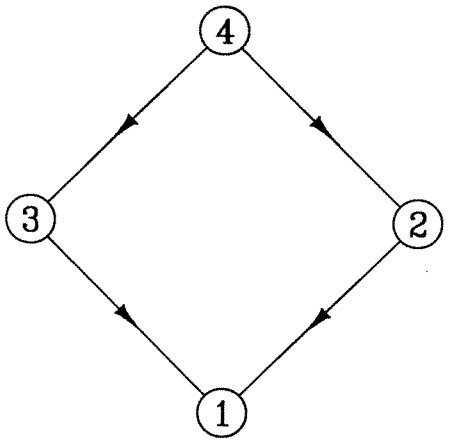


FIGURE 6

Example Graph

resulting solution remains feasible. Furthermore, note that the rate of decrease of the objective function, evaluated at $T^{*2}_{G^+} = D(G)$, when $T^{*}_{G^+}$ increases, is $v(G^+,G^-)$. Similarly, $v(G^+,G^-)$ measures the rate of change or gradient of the objective function as $T^{*}_{G^-}$ decreases. Consequently, if $v(G^+,G^-) > 0$, then the current solution cannot be optimal and the T_i^* cannot all assume the same value in the optimal solution.

We observe that there are many possible directed cuts that can be constructed corresponding to a graph G. Finding the best directed cut corresponds to finding the one having the maximal value, that is, the one yielding the maximal rate of decrease in cost evaluated at the current value of $T_i^* = \sqrt{D(G)}$. If the value of the maximal cut is nonpositive, then the optimal solution is $T_i^* = \sqrt{D(G)}$ for all $i \in N(G)$.

Consider the graph G given in Figure 6. Suppose $K_1 = 4$, $K_2 = 1$, $K_3 = 6$, $K_4 = 1$, and $g_i = 1$ for all $i = 1, \ldots, 4$. Assuming the reorder intervals are equal, $D(G) = (\Sigma K_i)/(\Sigma g_i) = 3$. Observe that there are 4 possible directed cuts associated with this graph,

$N(G^+) = \{4\}$, $N(G^-) = \{3, 2, 1\}$, $v(G^+,G^-) = -2/3$,

$N(G^+) = \{4, 3\}$, $N(G^-) = \{2, 1\}$, $v(G^+,G^-) = 1/3$,

$N(G^+) = \{4, 2\}$, $N(G^-) = \{3, 1\}$, $v(G^+,G^-) = -4/3$, and

$N(G^+) = \{4, 3, 2\}$, $N(G^-) = \{1\}$, $v(G^+,G^-) = -1/3$.

The maximal cut (G^+,G^-) has value $1/3$ with $N(G^+) = \{4, 3\}$ and $N(G^-) = \{2, 1\}$.

6.1.3 Finding the Maximal Cut

The problem of finding the maximal cut for the graph G corresponding to the solution $T_i^* = \sqrt{D(G)}$ for all $i \in N(G)$ is related to the problem of calculating the multipliers that satisfy the Kuhn–Tucker conditions (36). Assume there are n nodes in G numbered 1 through n. Let $a_i = K_i/D(G) - g_i$ for $i = 1, \ldots, n$. Then the multipliers satisfying (36), given $T_i^* = \sqrt{D(G)}$, correspond to flows in a transshipment network having n nodes and having supply available for shipment at nodes for which $a_i < 0$ and requirements for supply at nodes for which $a_i > 0$. Nodes having $a_i = 0$ are transshipment nodes. If the flow satisfying the constraints (36) also satisfies the nonnegativity requirements (38), the graph G does not need to be subdivided; however, if no such solution exists, then at least one $\theta_{ij} < 0$.

To determine the maximal cut, we solve a minimum flow problem on a graph G', which is related to G as follows.

Graph G' has two nodes in addition to those found in G. The first new node, which we call node 0, is a source node; the second new node, labeled node $n + 1$, is a sink node. The other n nodes correspond to those found in G. Furthermore, if $(i,j) \in A(G)$, then $(i,j) \in A(G')$; G' has another 2n arcs in the form (0,i) and $(i,n + 1)$ for $i \in N(G)$. Associated with each arc $(i,j) \in A(G')$ is a lower bound, $\ell(i,j)$, on the flow θ_{ij} over that arc. For all $(i,j) \in A(G)$, let $\ell(i,j) = 0$; for arcs of the form $(0,i) \in A(G')$, let $\ell(i,j) = 0$; for arcs $(i, n + 1) \in A(G')$, let $\ell(i, n + 1) = a_i = K_i/D(F) - g_i$.

The graph G' corresponding to the graph G given in Figure 6 is shown in Figure 7. The numbers on the arcs represent the lower bounds on the flows.

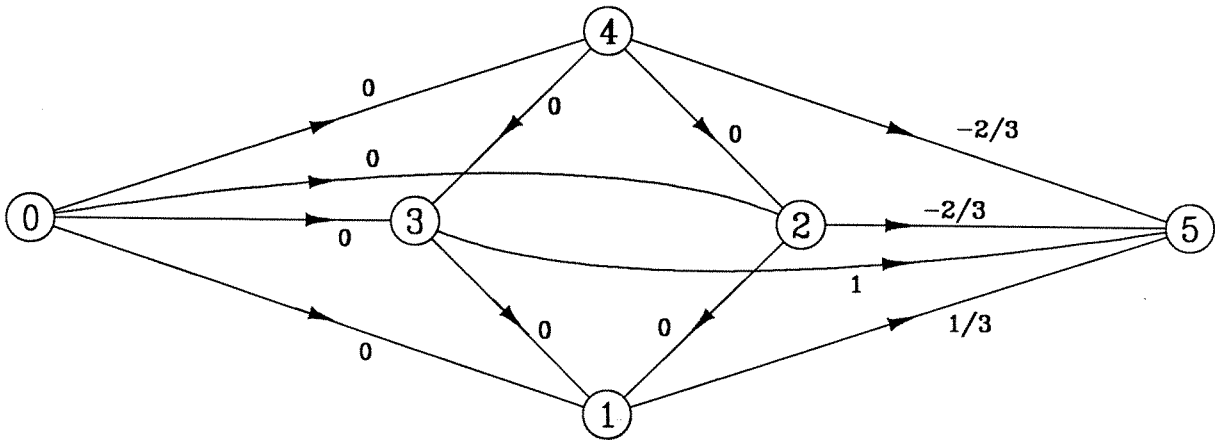


FIGURE 7

Graph of G'

The addition of the source node to the transshipment network ensures that a flow that satisfies conditions related to (36) and (38) exists. In particular, by adding this node, conditions (36) become, for each $i \in N(G)$,

$$\theta_{0i} + \sum_{j \in P_G(i)} \theta_{ji} - \sum_{j \in S_G(i)} \theta_{ij} = a_i, \quad \theta_{0i} = 0.$$

The flow $v = \sum_{i=1}^{n} \theta_{0i}$ measures the total requirement in the original transshipment network that cannot be satisfied without reversing the orientation on at least one arc, that is, by making at least one $\theta_{ij} < 0$. If $v = 0$, then a feasible flow satisfying (36) and (38) exists.

The sink node appended to the original network serves a different purpose. Each arc $(i, n+1)$ has a corresponding lower bound on its flow. If the lower bound is positive, that is, $a_i > 0$, then a node i has a requirement in the original network. If $a_i < 0$, node i has a supply in the original network. In this latter case, the negative lower bound on the arc $(i, n+1)$ indicates that the natural supply orientation of this arc has been reversed. Thus the negative lower bound indicates that a flow can exist from node $n+1$ to i that cannot exceed $-a_i$.

By adding both the source and sink nodes, we can rewrite (36) as

$$\theta_{i,\,n+1} + \sum_{j \in S_G(i)} \theta_{ij} = \theta_{0i} + \sum_{j \in P_G(i)} \theta_{ji},\ \theta_{0i} = 0,\ \theta_{i,n+1} = a_i.$$

The minimum flow problem, called Problem (41), that yields the maximum cut for the graph G is

$$v = \min \sum_{i=1}^{n} \theta_{0i}$$

subject to

$$\theta_{i,\,n+1} + \sum_{j \in S_G(i)} \theta_{ij} = \theta_{0i} + \sum_{j \in P_G(i)} \theta_{ji},\ i = 1, \ldots, n, \tag{41}$$

$$\theta_{ij} \geq \ell(i,j),\ (i,j) \in A'(G').$$

The definition of $D(G)$ implies that $\Sigma_i a_i = 0$. Equation (41) implies that $\Sigma_i \theta_{0i} = \Sigma_i \theta_{i,n+1}$. Therefore $v = 0$ if and only if $\theta_{0i} = 0$ and $\theta_{i,n+1} = a_i$ for all i, i.e., (36) and (38) have a solution.

Problem (41), like the classic maximum flow problem, can be solved efficiently for very large graphs. An algorithm exists to solve it that is at worst of order n^3. The solution identifies the maximum directed cut as follows:

$$0 \in N(G'^{+}),$$

$$n + 1 \in N(G'^{-}),$$

$$v = \sum_{i \in N(G'^{+})} \sum_{j \in N(G'^{-})} \ell(i,j)$$

$$= \sum_{i \in N(G'^{+}) - \{0\}} a_i \geq 0.$$

The maximal cut (G^{+},G^{-}) is $(G'^{+} - \{0\}, G'^{-} - \{n + 1\})$. Note that if $v = 0$, then G need not be subdivided further.

6.1.4 An Algorithm for Solving Problem (35)

The solution to Problem (35) is found using an algorithm that is based on the concept of a directed cut. The algorithm operates as follows. To begin, assume that all operations share a common reorder interval. Problem (41) is then solved to correspond to this feasible solution. If $v = 0$, the Kuhn–Tucker multipliers are nonnegative and the solution is optimal; if not, G must be divided into graphs G^{+} and G^{-}, which are determined by solving Problem (41). The process is then repeated separately on G^{+} and G^{-} until no further splitting of the graphs is needed, that is, until the solutions to the corresponding Problem (41) indicate that nonnegative multipliers exist that satisfy all the Kuhn–Tucker conditions.

Formally, we state the algorithm as follows.

ALGORITHM 3. THE DIVIDE AND CONQUER ALGORITHM

STEP 0. Assume all reorder intervals are identical.

STEP 1. If no directed cut (G^+,G^-) has a positive value, that is, if no directed cut has $\sum_{i\epsilon N(G^+)}(K_i/D(G) - g_i) > 0$, then STOP.

STEP 2. Find the maximal cut, by solving Problem (41), and divide G into two separate subgraphs G^+ and G^-. Apply the Divide and Conquer Algorithm to each of the graphs G^+ and G^-.

In the event that Problem (41) has more than one maximal cut, any one of the maximal cuts may be selected in Step 2.

This algorithm constructs a set of subgraphs which we denote $\{G_k,\ 1 \leq k \leq N\}$. The clusters are $\{N(G_k),\ 1 \leq k \leq N\}$. Steps 2 and 3 of Algorithm 1 are now used to obtain the reorder intervals that solve Problem (35) and Problem (34). A proof that the Divide and Conquer Algorithm does solve Problem (35) to optimality can be found in Maxwell and Muckstadt (1985).

In the event that the reorder intervals T_i, $i \in N(G)$ are constrained to satisfy

$$2^{\underline{\ell}}T_L \leq T_i \leq 2^{\bar{\ell}}T_L\ ,$$

the modified roundoff procedure described in section 3.4 can be used. All claims of optimality and near–optimality made in this section apply to the constrained version of the problem as well. This is also true of the models of sections 4, 5, 7, 8, and 9.

6.3 Computational Complexity of the Algorithm

The solution of Problem (35) is constructed by solving a sequence of minimum flow problems – Problem (41). Each solution to Problem (41) generates either two or no leaf

vertices. However, there can be no more than n leaf vertices in the final binary tree, where n represents the number of nodes in G. Since the number of steps required by the algorithm for finding the optimal solution to Problem (41) is at worst proportional to n^3, the algorithm for solving Problem (35) is an order n^4 or less algorithm.

6.4 An Example Problem

The problem used in this section to illustrate the algorithm was obtained from a supplier of components to the U.S. automotive industry. Figure 8 shows the product structure graph G corresponding to this problem. The graph shows that there are 4 finished products, which correspond to nodes 1, 2, 4, and 6. Each of the 94 nodes in the graph represents an operation. Those nodes represented by triangles indicate the acquisition of raw materials. Squares indicate operations having positive setup times and circles operations requiring no setup time.

Observe that G represents neither a pure assembly nor a pure distribution system. Some raw materials are used in the production of more than one component. Some components are used in more than one assembly or subassembly. For example, the component completed at operation 52 is used in 8 different ways. Furthermore, the subassembly at operation 3 requires 5 different components. Thus this graph G represents a relatively complex production environment.

The reorder intervals for each operation were obtained using Algorithm 3. Figure 9 gives the first partition of G into G^+ and G^-. Underlining of a leaf vertex in the solution tree indicates that the leaf vertex is in a permanent state. Since all leaf vertices are underlined in the tree shown in Figure 10, the optimal reorder intervals for Problem (35) have been found. The reorder interval for operations in G^{--} is 0.074 years, in G^{-+-} is 0.090 years, in G^{-++} is 0.106 years, in G^{+--} is 0.118 years, in G^{+-+-} is 0.140 years, in G^{+-++} is 0.167 years, and in G^{++} is 0.350 years. Thus, for example, operations 76 through 80, the elements of G^{+-++}, should have the same reorder interval

and should be produced about once every 2 months. Note that the magnitude of the values of the reorder intervals correspond to an ordering of the sets $G^{\alpha_1 \ldots \alpha_n}$. Furthermore, observe that the magnitude of most of these reorder intervals is similar, indicating that more operations might share common reorder intervals in the solution to Problem (34).

This solution to Problem (35) shows that the operations should initially be divided into 7 distinct groups. Problem (34)'s solution indicates that the operation should be combined further. A base planning period of a day was used. The optimal solution indicates that the operations in G^{++} should be on a cycle that is 128 days long. Those in G^{+-++} and G^{+-+-} should have reorder intervals equal to 64 days. The operations in sets G^{--}, G^{-+-}, G^{-++}, and G^{+--} should all have the same reorder interval of 32 days.

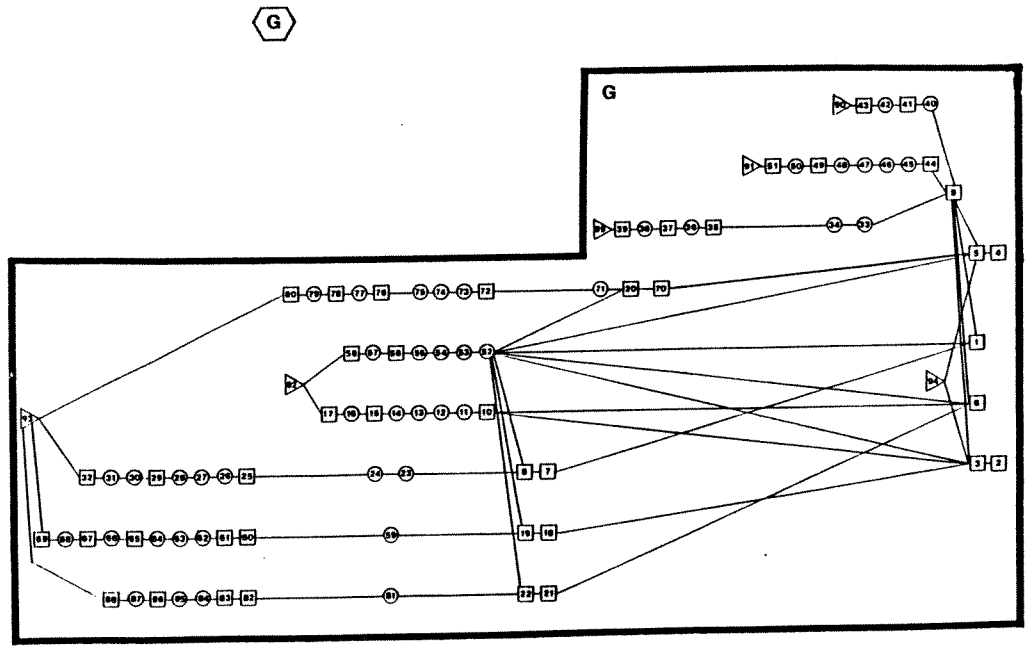


FIGURE 8

Product Structure Graph G

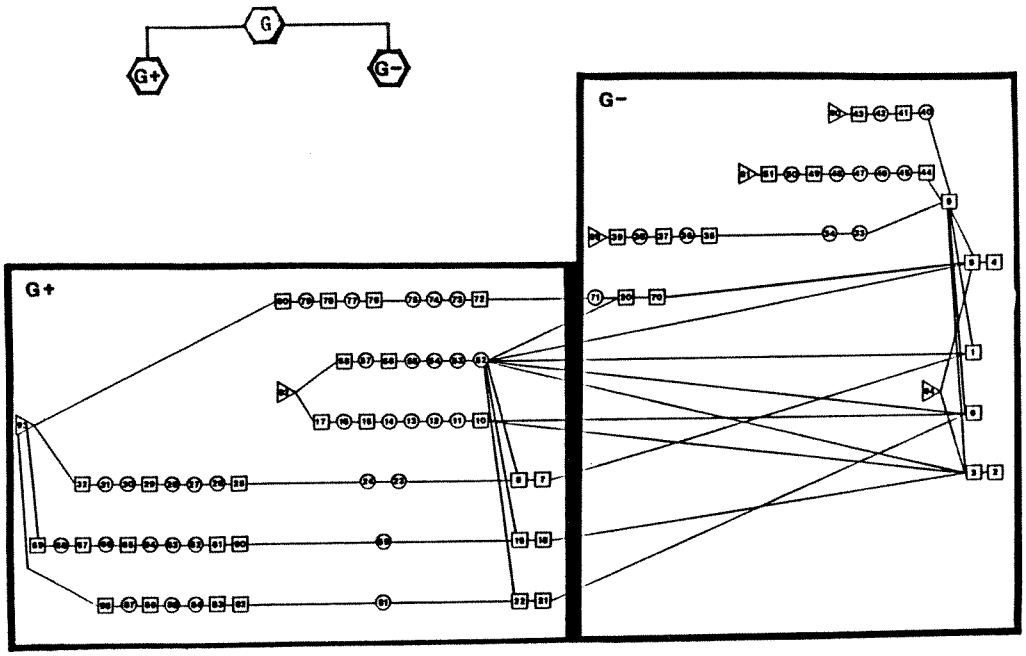


FIGURE 9

Graphs G^+ and G^-

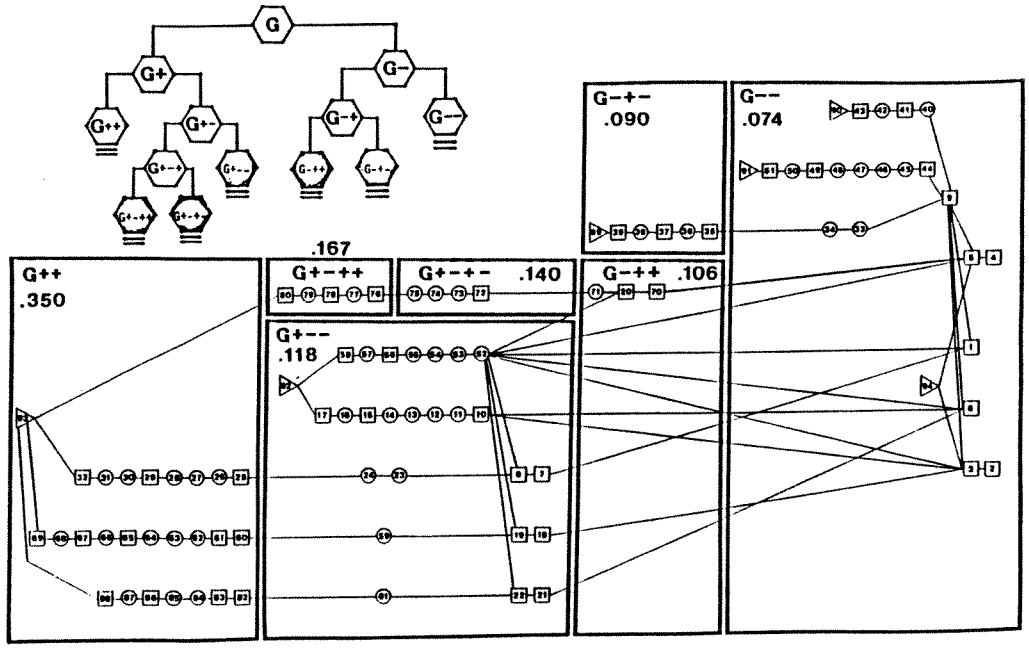


FIGURE 10

Optimal Solution to the Relaxed Problem

6.5 Summary

In this section, we studied the problem of determining consistent and realistic reorder intervals in a general production–distribution system environment. A mathematical model was developed whose special structure permits its solution using a standard network flow algorithm. As in earlier sections, attention was restricted to policies that are nested, stationary, and a powers–of–two multiple of a base planning period. As before, if we follow a powers–of–two policy, the total cost cannot be more than 6% higher than can be achieved following any other nested policy.

7. CONSTRAINED WORK CENTERS

To this point we have assumed that the reorder intervals were chosen independent of available resource levels. In this section we consider planning problems occurring when the production facility is divided into work centers, each of which is assumed to have a limited production capacity, expressed in total annual standard hours. Since the total run time of all jobs processed through the work center is unaffected by lot sizing/reorder interval decisions we deduct the projected total annual run time from the production capacity. The result is an estimate of the total annual standard hours available for performing other work center functions, in particular, for performing setups within the work center. The remainder of the assumptions on which we base the following model are the same as those made for the models developed earlier in section 2 of this chapter. Notational differences are minor and will be identified as we proceed.

We assume that each operation i in the production/distribution graph is associated with one and only one work center. Following the concepts and the notation of section 3, we assume that the work centers form a partition of the node set of the graph G. Let W_h denote the subgraph of G formed from the node set $N(W_h)$, consisting of all the operations in work center h, and the arc set $A(W_h)$, consisting of all arcs of the form (i,j) where

$i,j \in N(W_h)$ and $(i,j) \in A(G)$, for $h = 1, \ldots, H$, where H is the number of distinct work centers. Let $\eta(i)$ index the work center associated with operation i.

Let L_i denote the setup time required by operation i, measured in standard hours, for all $i \in N(G)$. We also assume that the setup cost for operation i is directly proportional to L_i. This assumes that work center groupings are closely aligned with labor classifications and that material and energy costs involved in setting up operations are not significant. Let c_h denote the setup labor rate in work center h. By assumption, then, $K_i = c_h L_i$ for all operations i in work center h, $h = 1, \ldots, H$. Finally, let b_h represent the total annual time available for performing setups in work center h, $h = 1, \ldots, H$.

The general (relaxed version) multiple work center capacitated planning problem that we examine is given by

$$\text{minimize} \quad \sum_{i \in N(G)} [c_{\eta(i)} L_i / T_i + g_i T_i]$$

$$\text{subject to} \quad T_i \geq T_j \geq 0, \ (i,j) \in A(G), \tag{42}$$

$$\sum_{i \in N(W_h)} L_i / T_i \leq b_h, \ h = 1, 2, \ldots, H.$$

The powers–of–two formulation of this problem includes restrictions of the form $T_i = 2^{\ell} T_L$, $\ell \in \{0, 1, \ldots\}$, $i \in N(G)$.

Before examining this problem, we make an observation that is the basis of the solution procedure we will discuss. Recall the necessary and sufficient conditions given in section 3.3 for solving the following problem:

$$\text{minimize} \qquad \sum_{i \in N(G)} (K_i/T_i + g_i T_i) \tag{43}$$

$$\text{subject to} \qquad T_i \geq T_j \geq 0, \text{ for all } (i,j) \in A(G).$$

Now suppose we have a new problem with $\bar{K}_i = \alpha K_i$ and $\bar{g}_i = \beta g_i$ for each $i \in N(G)$, $\alpha > 0$, $\beta > 0$. Suppose we have an optimal solution to Problem (43), that is, an optimal partition $(G_1, \ldots, G_N)$ of G. This partition is also optimal for the new problem as well. This result easily follows from the conditions of Theorem 2 presented in section 3.3. We call this the invariance property of the optimal partition of G. Thus the optimal partition of G is invariant under positive scaling of both setup and holding costs. Thus uniform increases or decreases in labor rate, overhead or interest rates do not change the optimal partition. It may, however, change the value of the optimal reorder intervals.

7.1 Single Work Center

The case of a single work center is particularly easy to solve because of the invariance property. The relaxed version of the capacitated single work center problem can be written as follows:

$$\text{minimize} \qquad \sum_{i \in N(G)} [cL_i/T_i + g_i T_i]$$

$$\text{subject to} \qquad T_i \geq T_j \geq 0, \ (i,j) \in A(G), \tag{44}$$

$$\sum_{i \in N(W)} L_i/T_i \leq b,$$

where the subscript h indexing different work centers has been dropped. We assume $c > 0$.

Let α denote the Lagrange multiplier for the capacity constraint. Dualizing this constraint, and letting $\alpha' = \alpha + c$, yields the following relaxed problem:

$$\text{minimize} \quad \sum_{i \in N(G)} [\alpha' L_i / T_i + g_i T_i]$$

$$\text{subject to} \quad T_i \geq T_j \geq 0, \ (i,j) \in A(G). \tag{45}$$

We seek a value of $\alpha' \geq c$ $(\alpha \geq 0)$ such that the solution to this uncapacitated problem satisfies the omitted capacity constraint and satisfies it with equality if $\alpha' > c$ $(\alpha > 0)$.

We find the optimal solution as follows. First, find the optimal partition for Problem (45) with $\alpha' = c$. If the associated solution satisfies the capacity constraint, then stop; the solution is optimal for the capacitated problem. Otherwise, the capacity constraint is binding. Based on the invariance property, the same partition is optimal for all positive values of α'. Suppose the partition is $(G_1, \ldots, G_N)$. The optimal reorder intervals for Problem (45) as a function of α' are given by

$$T^*(k,\alpha') = (\alpha' L(k)/g(k))^{1/2}, \text{ where} \tag{46}$$

$L(k) = \sum_{i \in N(G_k)} L_i$ and $g(k) = \sum_{i \in N(G_k)} g_i$, $k = 1, \ldots, N$. The capacity used as a function of α' is given by

$$\sum_{k=1}^{N} \sum_{i \in N(G)} L_i / T(k,\alpha') = \sum_{k=1}^{N} L(k) / [\alpha' L(k)/g(k)]^{1/2}$$

$$= \sum_{k=1}^{N} \frac{[L(k)g(k)]^{1/2}}{\sqrt{\alpha'}}.$$

Substituting for the left hand side in the capacity constraint and solving for α' yields

$$\alpha'^{*} = \left[\frac{\sum_{k=1}^{N} [L(k)g(k)]^{1/2}}{b} \right]^2 . \tag{47}$$

Thus, a closed form solution is available for the single work center problem, Problem (45), given the optimal partition for Problem (45). Note that α'^{*} must be greater than c in the above expression because, by assumption, the constraint is violated at $\alpha' = c$. Clearly the capacity used is a decreasing function of α', $\alpha' \geq c$.

Let U denote the capacity utilization ratio when $\alpha' = c$ $(\alpha = 0)$:

$$U = \frac{\sum_{k=1}^{N} \sum_{i \in N(G_k)} L_i / T^{*}(k,c)}{b}$$

$$= \sum_{k=1}^{N} \frac{[L(k)g(k)]^{1/2}}{\sqrt{c} \cdot b} . \tag{48}$$

Rewriting (47) in terms of the utilization ratio, U:

$$\alpha'^{*} = cU^2,$$

or (49)

$$\alpha^{*} = cU^2 - c.$$

Similarly, the optimal reorder intervals for the capacitated problem, $\{T^{*}(k); k = 1, \ldots, N\}$, satisfy

$$T^*(k) = T^*(k,\alpha'^*) = T^*(k,c) \cdot U \,. \tag{50}$$

The optimal reorder intervals for the relaxed version of the single work center constrained problem are simply the natural $(\alpha = 0)$ reorder intervals scaled by the capacity utilization ratio (when $U > 1$).

7.2 The Powers–of–Two Single Work Center Problem

Before extending the solution technique to handle multiple work centers, we consider the relation between the relaxed version of the single work center problem and the powers–of–two version. The powers–of–two formulation of the single work center planning problem is given by:

$$\text{minimize} \qquad \sum_{i \in N(G)} [cL_i/T_i + g_i T_i]$$

$$\text{subject to} \qquad \sum_{i \in N(G)} L_i/T_i \leq b,$$

$$T_i \leq T_j, \ (i,j) \in A(G),$$

$$T_i = 2^{\ell_i} T_L, \ \ell_i \in \{0, 1, 2, \ldots\}, \ i \in N(G) \,.$$

By analogy with the uncapacitated problem discussed earlier, consider the powers–of–two solution given by $T_i = T(k)$ for all $i \in N(G_k)$, $k = 1, \ldots, N$, and let $T(k) = 2^{\ell(k)} T_L$, where $\ell(k)$ is the smallest value of ℓ satisfying

$$2^{\ell} \geq \frac{1}{T_L} \left[\frac{\alpha'^* L(k)}{2g(k)} \right]^{1/2} . \tag{51}$$

We now investigate the feasibility and cost performance of this solution using some of the results developed previously.

By (46) and (51), $\ell(k)$ satisfies

$$2^{\ell(k)} \geq \frac{T^*(k)}{\sqrt{2} \cdot T_L} > 2^{\ell(k)-1},$$

so that

$$\frac{T^*(k)}{\sqrt{2}} < T(k) \leq \sqrt{2} \cdot T^*(k). \tag{52}$$

We examine the cost performance of the powers–of–two solution first. Let

$$f_k(T) = \sum_{i \in N(G_k)} [cL_i/T + g_i T]$$

$$= cL(k)/T + g(k)T,$$

the cost associated with a common reorder interval T for subgraph G_k. By convexity, $f_k(T(k)) \leq \max\{f_k(T^*(k)/\sqrt{2}), f_k(\sqrt{2}T^*(k))\}$. Assuming the capacity constraint is binding ($U \geq 1$), we have

$$f_k(\sqrt{2}T^*(k)) = \frac{cL(k)}{\sqrt{2}T^*(k)} + g(k)\sqrt{2}T^*(k)$$

$$= \frac{cL(k)}{\sqrt{2}T^*(k,c) \cdot U} + g(k)\sqrt{2}T^*(k,c) \cdot U$$

$$= \sqrt{cL(k)g(k)} \left[\frac{1 + 2U^2}{\sqrt{2}U} \right] \tag{53}$$

and

$$f_k(T^*(k)/\sqrt{2}) = \sqrt{cL(k)g(k)} \left[\frac{2 + U^2}{\sqrt{2}U} \right] \tag{54}$$

$$\leq f_k(\sqrt{2}T^*(k)),$$

since $U \geq 1$.

Let F_b denote the cost associated with the powers–of–two solution given a capacity of b:

$$F_b = \sum_{k=1}^{N} \sum_{i \in N(G_k)} \left[\frac{cL_i}{T(k)} + g_i T(k) \right]$$

$$= \sum_{k=1}^{N} f_k(T(k))$$

$$\leq \sum_{k=1}^{N} \sqrt{cL(k)g(k)} \left[\frac{1 + 2U^2}{\sqrt{2}U} \right]. \tag{55}$$

Suppose $\{T(k); k = 1, \ldots, N\}$ is feasible for the powers–of–two single work center problem. Then F_b is an upper bound on the minimal cost for that problem. Let Z_b denote the minimal cost of the relaxed version of the single work center problem:

$$Z_b = \sum_{k=1}^{N} \sum_{i \in N(G_k)} \left[\frac{cL_i}{T^*(k)} + g_i T^*(k) \right]$$

$$= \sum_{k=1}^{N} f_k(T^*(k))$$

$$= \sum_{k=1}^{N} \sqrt{cL(k)g(k)} \left[\frac{1 + U^2}{U} \right]. \tag{56}$$

Z_b can be shown to be a lower bound on the minimal cost for the powers–of–two single work center problem using the method developed in section 3. If follows that

$$\frac{F_b}{Z_b} \leq \frac{(1 + 2U^2)}{\sqrt{2}\ (1 + U^2)}. \tag{57}$$

As $U \to \infty$ this bound converges to $\sqrt{2}$. That is, the proposed powers–of–two solution cannot exceed the solution to the relaxed version of the single work center problem by more than 41%. Table 1 tabulates the bound for a variety of values of U. Observe that for a 20% overutilization in the constrained case $(U = 1.2)$ the powers–of–two cost is bounded to within approximately 12% of the cost of the solution to the relaxed problem.

U	Bound on F_b/Z_b
1.0	1.061
1.1	1.094
1.2	1.124
1.5	1.197
2.0	1.273
5.0	1.387

TABLE 1

Cost Bound for the Powers–of–Two Solution

Next, let us turn our attention to the feasibility of the proposed powers–of–two solution. Observe that the second condition of Theorem 2 implies that the constraints $T_i \geq T_j$ $(i,j) \in A(G)$ will be satisfied by the solution $\{T(k); k = 1, \ldots, N\}$. Using the fact that $T(k) \geq T^*(k)/\sqrt{2}$, $k = 1, \ldots, N$, worst case analysis reveals that

$$\sum_{k=1}^{N} \sum_{i \in N(G_k)} L_i/T(k) \leq \sqrt{2} \cdot b \,. \tag{58}$$

That is, the powers–of–two solution can exceed capacity by at most 41%. This analysis assumes that for each set G_k in the partition, $T(k)$ takes on the smallest possible value. In reality, we would expect that $T(k)$ would be larger than $T^*(k)$ for some of the sets G_k and smaller than T_k for others. If the number of sets in the partition is large, we would anticipate that the powers–of–two solution would utilize roughly the same amount of capacity as the solution to the relaxed problem.

To illustrate, suppose $T(k)$ is uniformly distributed over the interval $[T^*(k)/\sqrt{2}, \sqrt{2}T^*(k)]$. Then the expected amount of capacity required by the powers–of–two solution is

$$E\left[\sum_{k=1}^{N} \sum_{i \in N(G_k)} L_i/T(k)\right] = \sum_{k=1}^{N} L(k)E[1/T(k)]$$

$$= \left[\sum_{k=1}^{N} L(k)/T^*(k)\right]\sqrt{2}\log 2$$

$$= b(\sqrt{2}\log 2), \tag{59}$$

since $\{T^*(k); k = 1, \ldots, N\}$ satisfies the constraint exactly. Assuming the $\{T(k); k = 1, \ldots, N\}$ are independent, then the strong law of large numbers implies

$$\lim_{N\to\infty} \sum_{k=1}^{N} \sum_{i \in N(G_k)} L_i/T(k) = \sqrt{2}\log 2 \cdot b$$

$$\approx .980 \cdot b .$$

Hence, in a probabilistic sense, for a large number of sets in the optimal partition, we would anticipate that the powers–of–two solution would be feasible. A more rigorous analysis of this conjecture appears impractically difficult. Computational experience with the algorithm, described by Jackson, Maxwell, and Muckstadt (1988), suggests that for practical problems the powers–of–two solutions are close to feasible.

Roundy (1986) describes an extension to this algorithm that guarantees feasibility of the powers–of–two solution by adjusting the length of the base planning period, T_L. For his algorithm, the cost of the powers–of–two solution cannot exceed the solution to the relaxed version by more than 44%. In practice, the base planning period is often determined by the information system reporting cycle and in these cases it is not easily subject to change. For example, in most manufacturing facilities visited by the authors we observed MRP type planning systems in which T_L was either a week or a month. That is, these planning systems generated production requirements for each week or month over some time horizon. Thus reorder intervals were constrained to be some multiple of these time intervals.

7.3 The Multiple Work Center Planning Algorithm

There are several ways in which the multiple work center capacitated planning problem could be solved. We propose a technique that takes advantage of the ease by which single work center problems can be solved. The idea behind the algorithm is that there are two useful Lagrangian relaxations of the multiple work center problem. We propose an iterative algorithm that alternates between these two relaxations.

One approach to using Lagrangian relaxation on the multiple work center problem is to dualize all the capacity constraints. Let α_n denote the Lagrange multiplier on the capacity constraint for work center h, $h = 1, \ldots, H$. If $\alpha'_h = \alpha_h + c_h$, $h = 1, \ldots, H$, and α' denotes the vector of coefficients $(\alpha'_1, \ldots, \alpha'_H)$, then one possible relaxation of the multiple work center is given by

$$\text{minimize} \quad \sum_{i \in N(G)} [\alpha'_{\eta(i)} L_i / T_i + g_i T_i]$$

$$\text{subject to} \quad T_i \geq T_j \geq 0, \ (i,j) \in A(G). \tag{60}$$

We seek a vector $\alpha' \geq (c_1, \ldots, c_H)$ such that the capacity constraints which are omitted from Problem (60) are satisfied by the solution to Problem (60). Furthermore, if $\alpha'_h > c_h$, the constraint for work center h must be satisfied with equality. Unfortunately, the optimal partition for this relaxed problem is not necessarily invariant to the choice of α' because the scaling factors are not common across all operations. Let the optimal partition for this problem be denoted $(G_1^{\alpha'}, \ldots, G_{N(\alpha')}^{\alpha'})$ for a given vector α'. Let the optimal set of reorder intervals be given by $\{T^*(k,\alpha'); k = 1, \ldots, N(\alpha')\}$ and let $U_h^{\alpha'}$ denote the corresponding capacity utilization ratio for work center h:

$$U_h^{\alpha'} = \frac{\sum_{k=1}^{N} \left[\sum_{i \in N(W_h) \cap N(G_k^{\alpha'})} L_i \right] / T^*(k,\alpha')}{b_h}, \tag{61}$$

for $h = 1, \ldots, H$.

An alternative approach to using Lagrangian relaxation on the multiple work center problem is to dualize all the consistency constraints that link different work centers together. That is, eliminate constraints of the form $T_i \geq T_j$, where $(i,j) \in A(G)$ and $\eta(i) \neq \eta(j)$. Let θ_{ij} denote the Lagrange multiplier on such a linking consistency constraint and let

$$\kappa_i(\theta) = \sum_{\substack{j \in N(G) \\ (j,i) \in A(G) \\ \eta(j) \neq \eta(i)}} \theta_{ji} - \sum_{\substack{j \in N(G) \\ (i,j) \in A(G) \\ \eta(i) \neq \eta(j)}} \theta_{ij}.$$

Then, an alternative Lagrangian relaxation of the multiple work center capacitated problem is given by the following problem:

$$\text{minimize} \quad \sum_{i \in N(G)} [c_{\eta(i)} L_i / T_i + g_i T_i + \kappa_i(\theta) T_i] \tag{62}$$

$$\text{subject to} \quad T_i \geq T_j \geq 0, \ (i,j) \in A(W_h),$$

$$\sum_{i \in N(W_h)} L_i / T_i \leq b_h, \ h = 1, 2, \ldots, H.$$

For a given vector of multipliers, $\theta = (\theta_{ij})$, this latter relaxation is separable by work center. The single work center subproblem, for a given vector θ, is given by:

$$\text{minimize} \quad \sum_{i \in N(W_h)} [c_h L_i / T_i + (g_i + \kappa_i(\theta)) T_i]$$

$$\text{subject to} \quad T_i \geq T_j \geq 0, \ (i,j) \in A(W_h), \tag{63}$$

$$\sum_{i \in N(W_h)} L_i / T_i \leq b_h,$$

for work center h, $h = 1, \ldots, H$.

Observe that for a given vector θ, this single work center subproblem is identical in form to the single work center problem examined previously. Letting $\bar{\alpha}_h$ denote the Lagrange multiplier for the single work center capacity constraint and letting $\bar{\alpha}'_h = \bar{\alpha}_h + c_h$, the *uncapacitated* version of the single work center subproblem can be written:

$$\text{minimize} \qquad \sum_{i \in N(W_h)} [\bar{\alpha}'_h L_i / T_i + (g_i + \kappa_i(\theta)) T_i]$$

$$T_i \geq T_j \geq 0, \ (i,j) \in A(W_h). \tag{64}$$

Therefore, by the invariance property, given an optimal partition to Problem (64) for any value of $\bar{\alpha}'$, a closed form solution to the capacitated subproblem, Problem (63), is available immediately. Denote the optimal value of the Lagrange multiplier by $\bar{\alpha}^*_h(\theta)$, a function of θ since the definition of the subproblem depends on the vector θ.

In the following, we note that if an optimal partition is available for Problem (60), which was used to define the vector θ, then this partition induces an optimal partition on each of the single work center uncapacitated subproblems.

Theorem 3. For a given vector of Lagrange multipliers α' on the work center capacity constraints, let $(G_1^{\alpha'}, \ldots, G_{N(\alpha')}^{\alpha'})$ be an optimal partition of (60) and let θ be a corresponding vector of optimal Lagrange multipliers on the consistency constraints linking work centers. Let $RP_h(\theta)$ be the single work center capacitated problem of type (62) for work center h which is formed using this vector θ. Then, the ordered partition of G, $(G_1^{\alpha'}, \ldots, G_{N(\alpha')}^{\alpha'})$, induces an optimal partition on the subgraph W_h for problem $RP_h(\theta)$, $h = 1, \ldots, H$.

The proof of this theorem can be found in Jackson, Maxwell, and Muckstadt (1988).

If an optimal partition is available for Problem (60), then the solution to Problem (62) can be obtained in closed form without explicit knowledge of the vector of multipliers, θ, on the linking consistency constraints. That is, under the conditions of the proposition, for $h = 1, \ldots, H$,

$$\bar{\alpha}_h^*(\theta) = \max \{0,\ \alpha_h'[U_h^{\alpha'}]^2 - c_h\} \ . \qquad (65)$$

The proof of this fact can be found in Jackson, Maxwell, and Muckstadt (1988).

To this point we have specified two relaxations of the multiple work center capacitated problem. The first relaxed problem, Problem (60), can be solved by the modified network algorithm described in Maxwell and Muckstadt (1985), which was discussed in section 6, for any given vector $\alpha = (\alpha_1, \ldots, \alpha_H)$. A by–product of that algorithm is a vector of Lagrange multipliers on all the consistency constraints $\{T_i \geq T_j;\ (i,j) \in A(G)\}$. That is, the solution to Problem (60) implies the existence of a vector of Lagrange multipliers on the constraints linking the work centers: $\theta = (\theta_{ij};\ (i,j) \in A(G),\ \eta(i) \neq \eta(j))$. These multipliers can be used to define an alternative relaxation of the problem, referred to as Problem (62). This relaxed problem is separable and can be used to define a new vector of multipliers, $\bar{\alpha}(\theta)$, on the capacity constraints. The subproblems of Problem (62) and Problem (63) can be solved using Equation (65) without explicit knowledge of the vector θ. The following algorithm is iterative in nature; it alternates between solving (60) to get a new vector of capacity utilization ratios, $\alpha' = (U_1^{\alpha'}, \ldots, U_H^{\alpha'})$, and solving (62) to get a new vector of capacity multipliers, α.

The Multiple Work Center Planning Algorithm.

STEP 0. Pick an initial nonnegative vector α^0.

STEP 1. On iteration n, for vector α^n, solve Problem (60) using the algorithm described in section 6. Obtain an optimal partition, $(G_1^{\alpha^n}, \ldots, G_{N(\alpha^n)}^{\alpha^n})$, and a solution $\{T_i^{*\alpha^n}\}$. Let θ^n denote a vector of Lagrange multipliers on the consistency constraints associated with this solution. It is not necessary to obtain this vector explicitly.

STEP 2. For each work center h, compute the capacity utilization ratio:

$$U_h^{\alpha^n} = \frac{\sum_{i \in N(W_h)} L_i / T_i^{*\alpha^n}}{b_h} . \tag{66}$$

For a pre–specified tolerance ϵ, if $U_h^{\alpha^n} - 1 < \epsilon$ for all h such that $\alpha_h^n = 0$ and $|U_h^{\alpha^n} - 1| < \epsilon$ for all h such that $\alpha_h^n > 0$, h = 1, ... , H, then stop; the solution from Step 1 is ϵ–optimal.

STEP 3. For each work center h, h = 1, ... , H, let

$$\alpha_h(\theta^n) = \max \{0, (\alpha_h^n + c_h)[U_h^{\alpha^n}]^2 - c_h\} . \tag{67}$$

Let $\alpha(\theta^n)$ denote the resulting vector of Lagrange multipliers on the capacity constraints.

STEP 4. Let $\alpha^{n+1} = \alpha^n + \gamma^n(\alpha(\theta^n) - \alpha^n)$ where γ^n is an appropriately chosen step size parameter $(0 < \gamma^n \leq 1)$. Set $n \leftarrow n + 1$ and go to Step 1.

Step 1 of the algorithm requires most of the computational effort. The computational complexity of this problem depends on the structure of the graph G. For arbitrary acyclic graphs the computational effort is at worst $O(n^4)$, where n is the number of nodes in the graph. If G has a special structure, such as the ones described in other sections of this chapter, computation time may be proportional to n log n.

There are many possible strategies for choosing the step size γ^n in Step 4. To date, we have been unable to identify a strategy that guarantees convergence of the algorithm. However, the naive strategy of halving the current step size after a pre–specified number of iterations was effective in the test cases reported in Jackson, Maxwell, and Muckstadt (1988).

8. JOINT ORDER COSTS

In all earlier sections we have assumed that there is an order cost K_i for stage i in the system, and that if S is the set of items being ordered at a given point in time, then the total order cost incurred at that point in time is $\sum_{i \in S} K_i$, i.e., the order cost is a modular function of the set of items being ordered. In many situations this is not a realistic assumption. In this section we survey several different models in which the total order cost incurred depends in a more complicated way of the set of items being ordered at a given point in time.

8.1 The Joint Replenishment Problem

Perhaps the simplest way in which the cost of ordering can fail to be modular is through a joint order cost. Consider a single facility that stocks a number of different items. Each item i has a traditional order cost K_i, a holding cost rate h_i, and deterministic demand which occurs at a constant continuous rate λ_i. However, in addition, there is a joint order cost K_0 which is incurred every time an order is placed for any item. This order cost is in addition to the order costs for the individual items being ordered. Therefore, if at a given point in time an order is placed for the items in the set $S \neq \phi$, the total order cost incurred is $K_0 + \sum_{i \in S} K_i$. The objective is to schedule orders for the items over an infinite time horizon so as to minimize the long-run average order and holding cost, while meeting the demand for all items without shortages.

The joint replenishment system can be viewed as a special case of the assembly system discussed in section 4. This is accomplished by modeling the joint order cost K_0 as a separate stage. Consider the assembly system illustrated in Figure 11. In this system stages i, $i \geq 1$ correspond to the items of the joint replenishment system, and stage 0 corresponds to the joint order cost. The order costs and the holding cost coefficients for stages $i \geq 1$ are K_i and $g_i = \frac{1}{2}\lambda_i h_i$, respectively. For stage 0 the order cost is K_0 and the holding cost coefficient is $g_0 = 0$.

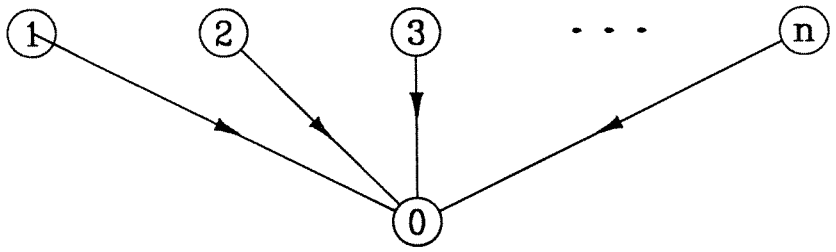


FIGURE 11

Joint Replenishment System

For assembly systems nested policies are optimal. For the joint replenishment system of Figure 11, the nestedness constraint implies that an order must be placed at node 0 (i.e., the joint order cost must be incurred) whenever any of the items is ordered. Therefore for the joint replenishment system, the nestedness constraint forces the joint order cost to be incurred at the appropriate points in time, and is a necessary condition for feasibility.

If the algorithms of section 4 are applied to the system of Figure 11, they will compute policies whose cost is within 6% of optimal if a fixed base planning period is used, and within 2% of optimal if the base planning period is a variable (Jackson, Maxwell, and Muckstadt (1985)). However, for these systems a much simpler and more efficient algorithm is available for solving the relaxation of the planning problem – Problem (20) of section 4. The algorithm is given below. In Step 2 we set $K/g = \infty$ if $g = 0$.

ALGORITHM 4. SIMPLE JOINT REPLENISHMENT SYSTEMS

STEP 1. Sort the items so that $K_i/g_i \leq K_{i+1}/g_{i+1}$ for all $i \geq 1$. Set $K \leftarrow K_0$, $g \leftarrow 0$, and $k \leftarrow 0$.

STEP 2. If $k = n$ or $K/g < K_{k+1}/g_{k+1}$, then go to Step 4. Otherwise go to Step 3.

STEP 3. Set $k \leftarrow k + 1$. Set $K \leftarrow K + K_k$ and $g \leftarrow g + g_k$. Go to Step 2.

STEP 4. Set $T_i^* = \sqrt{K/g}$ for all $i \leq k$, and set $T_i^* = \sqrt{K_i/g_i}$ for all $i > k$. The optimal clusters are $\{0,1,...,k\}$, $\{k + 1\}$, $\{k + 2\}$, ... , $\{n\}$.

The running time of this algorithm is limited by the time required to perform the sort in Step 1. This can be done in $O(n \log n)$ time. The algorithm is based on the idea that the joint order cost will be incurred once every T_0 time units. There are two types of items: those that order every T_0 time units and those that order less frequently. The items that order less frequently do so because their natural reorder interval, based on their own order and holding costs, is longer than T_0. The items that order every time the joint order cost is incurred have natural reorder intervals shorter than T_0. If $\{i: 1 \leq i \leq k\}$ is

the set of items that order once every T_0 time units then local optimality implies that $T_0 = [(K_0 + \sum_{1 \le i \le k} K_i)/(\sum_{1 \le i \le k} g_i)]^{1/2}$.

Via the transformation of section 5, the problem of finding a good nested policy for a one–warehouse, multi–retailer inventory system can be transformed into the problem of finding a good policy for a two–stage assembly system. The only essential difference between a two–stage assembly system and a joint replenishment system is that assembly systems typically have a positive holding cost coefficient associated with node 0. This can be compensated for by changing the statement $g \leftarrow 0$ in Step 1 to $g \leftarrow g_0$. With this change, the above algorithm can be used to find nested polices in one–warehouse, multi–retailer distribution systems, or to find policies for two–level assembly systems. The performance bounds obtained in sections 4 and 5 apply to two–level assembly and distribution systems, respectively.

8.2 The Family Model

In this sub–section we consider an extension of the general lot sizing model of section 6. The system considered there is modeled by an acyclic, directed graph representing a bill of material network G with node set $N(G)$ and arc set $A(G)$. Node $i \in N(G)$ corresponds to stage i of the system. Part i is stocked at stage i. Part i has order cost $K_i \ge 0$, echelon holding cost rate $h_i \ge 0$, and demand rate $\lambda_i \ge 0$. Associated with each arc $(i,j) \in A(G)$ is a *gozinto parameter* γ_{ij}. Whenever an order for q units of part j is placed it is instantly delivered, and $\gamma_{ij}q$ units of part i are simultaneously consumed.

This system, which we studied in section 6, is now enhanced by allowing order costs to be associated with families of components. To illustrate, consider an injection molding machine on which we produce Lego blocks. The blocks come in three different sizes and in three different colors, making nine parts in all. There are three molds, one for each size of block. Whenever we finish a batch of one of the nine parts the cavity in the mold needs to

be cleaned. The cost of doing this is K_c. When we switch the machine from one block size to another, the mold currently on the machine needs to be removed and another one must be mounted. The cost of doing this is K_m.

Let i index the parts, let k index the molds, and let I_k be the set of parts that are produced using mold k. Let part i be produced once every T_i days, and let mold k be loaded onto the machine once every T_k days, where the T's are powers of two. If the batches are sequenced appropriately, the average order cost incurred per day is $\sum_i K_c/T_i + \sum_k K_m/T_k$. The order intervals are constrained to satisfy $T_k \leq T_i$ for all $i \in I_k$.

This example motivates the following extension to the model of section 6. Let F be a collection of sets $f \subset N(G)$ called *families*. We assume that $|f| \geq 2$ for all $f \in F$. Associated with each family $f \in F$ is an order cost $K_f > 0$. We assume that the order cost K_f is incurred every time an order is placed for one or more of the items $i \in f$.

We model the joint setup costs K_f, $f \in F$ as follows. Given the bill of material network G of section 6, we define the *extended bill of material network* G' to be the network with node set $N(G') = N(G) \cup F$ and arc set $A(G') = A(G) \cup \{(i,f): i \in f, f \in F\}$. The nodes $f \in F$ are referred to as *order cost nodes*, and the arcs (i,f), $i \in f$, $f \in F$ are referred to as *order cost arcs*. The nodes in $N(G)$ are referred to as *inventory nodes*, and the arcs in $A(G)$ are *inventory arcs*. Associated with each order cost node $f \in F$ is an order cost K_f, an echelon holding cost $h_f = 0$, and an echelon demand rate $\lambda_f = 0$. Associated with each order cost arc (i,f) is a gozinto parameter $\gamma_{ij} = 0$. Since the order cost K_f must be incurred every time any of the items $i \in f$ is ordered, all feasible policies satisfy

$$T_f \leq T_i \text{ for each } i \in f, \ f \in F. \tag{68}$$

We say that a policy is *nested on order cost arcs* if it satisfies (68). All feasible polices are nested on order cost arcs.

Following section 6, we define a *stationary nested powers-of-two policy* for the extended bill of material network G' to be a a vector $\mathscr{T} = (T_i: i \in N(G'))$ satisfying

$$T_i = 2^{\ell_i} T_L, \ \ell_i \text{ integer}, \ T_L \text{ positive for all } i \in N(G') \tag{69}$$

and

$$T_i \geq T_j \text{ for all } (i,j) \in A(G'). \tag{70}$$

An optimal stationary nested powers–of–two policy is found by solving

$$\text{minimize:} \qquad c(\mathscr{T}) = \sum_{i \in N(G')} \left[\frac{K_i}{T_i} + g_i T_i \right]. \tag{71}$$

subject to: (69), (70).

This problem is clearly of the same form as the corresponding problem is section 6, and the algorithms presented there can be applied to it. They compute a policy whose cost is known to be within 2% or 6% of the cost of an optimal nested policy, depending on how the roundoff operation is performed. See Roundy (1986).

In the extended bill of material network, order cost nodes have holding cost coefficients of zero. This leads to the concern that the solution to the relaxation of (71) may have $T_i^* = \infty$ for some i. If one or more stages i has order cost $K_i = 0$, then there is also a possibility that the solution to the relaxation of (71) may have $T_i^* = 0$. The

following mild conditions on an extended bill of material network G' guarantee that all reorder intervals in the solution to the relaxation of (71) are positive and finite.

1. G is an acyclic directed network.
2. If no arcs emanate from node i in G, then $K_i > 0$.
3. If no arcs lead into node i in G then $g_i > 0$.

We conclude this subsection by pointing out that order cost families can be combined with the capacity-constrained models of section 7. To return to the injection molding machine we discussed at the beginning of the subsection, let us now suppose that a mold can be used on either of two different machines. It is used to make parts 1 and 2 on machine a, and it is used to make parts 3 and 4 on machine b. We wish to model the cost of mounting the mold on each of the machines, and also to model the fact that the mold is only available for a limited amount of time. We can accomplish this by creating two families, $\{1, 2\}$ and $\{3, 4\}$. The cost of mounting the mold on machine a is associated with family $\{1, 2\}$, and the cost of mounting it on machine b is associated with $\{3, 4\}$. A constrained workcenter is created to model the availability of the molds. If the time required to clean the mold is negligible, the workcenter consists of the families $\{1, 2\}$ and $\{3, 4\}$. If the time required to clean the mold is significant, the workcenter includes the nodes 1, 2, 3, and 4 in addition to the families $\{1, 2\}$ and $\{3, 4\}$.

8.3. Multi–Item Distribution Systems

There are two interesting and important special cases of the model of subsection 8.2 for which extremely efficient algorithms have been developed. The first of these is a one–warehouse, n-retailer distribution system through which I items are distributed. Each item is stocked at each location, i.e., at the warehouse and at each retailer. Each

item–location pair has its own order cost, echelon holding cost rate, and demand rate. In addition there is a joint order cost associated with each of the retailers. The order cost for retailer j is incurred whenever an order is placed at retailer j, regardless of which items comprise the order.

The extended bill of material network G of this system is illustrated in Figure 12. Stages 1, 2, and 3 correspond to the inventories of items 1, 2, and 3, respectively, at the warehouse. Stages 4, 5, and 6 correspond to the inventories of the three items at retailer one, and items 7 through 12 correspond to the inventories at retailers two and three. These nodes are all inventory nodes with positive order costs, positive echelon holding costs, and positive demand rates. The arcs connecting them are inventory arcs that represent the movement of material through the system. Their gozinto parameters are all equal to one.

Stage 13 is an order cost node that represents the joint order cost at retailer one. Its demand rate and echelon holding cost rate are both zero, and its order cost is the joint order cost. The arcs adjacent to 13 are order cost arcs with gozinto parameters of zero. Stages 14 and 15 are similar.

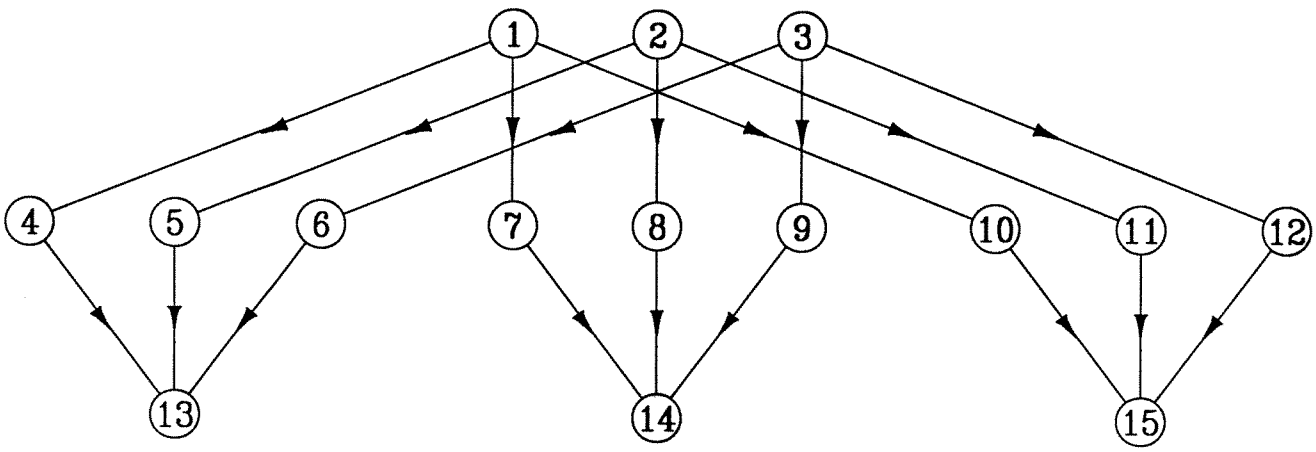


FIGURE 12

A Three Item, Three Retailer System

This model is clearly a special case of the family model of section 8.2, and the algorithm of that section can be used to find a good nested policy for it. We now describe a much more efficient $O(n \log n)$ algorithm that finds the optimal clusters for this problem. As before, once the clusters have been found, Steps 2 and 3 of Algorithm 1 are used to obtain the reorder intervals. See Muckstadt and Roundy (1987).

Let $W \subset N(G)$ be the set of nodes corresponding to the inventories of the I items held at the warehouse (nodes 1, 2, 3 in Figure 12). Let $R \subset N(G)$ be the set of nodes corresponding to the inventories held at the different retailers (nodes 4 through 12 in Figure 12), and let $J \subset N(G)$ be the set of joint order cost nodes (nodes 13, 14, and 15). Recall that s_i (resp., p_i) is the set of direct successors (resp., predecessors) of i in G.

The following observations are important in understanding the algorithm. The first two observations concern the one–warehouse, multi–retailer subsystems and the joint replenishment subsystems of RP. For a given node $j \in J$, consider the joint replenishment subsystem whose graph is the subgraph of G induced by the node set $\{j\} \cup p_j$. Consider the clusters that are optimal for this subsystem, when it is considered in isolation from the rest of G. Algorithm 4 can be used to compute these clusters. Among these clusters, let L_j be the cluster that contains node j. Each node $i \in p_j - L_j$ is in a singleton cluster $\{i\}$. Let $\tau^j = (\sum_{i \in L_j} K_i)/(\sum_{i \in L_j} g_i)$, and let $\tau^r = K_r/g_r$ for all $r \in R$. The key property of L_j is the following.

Observation 1. If $r \in L_j - \{j\}$ and $i \in P_j - L_j$, then $\tau^r \leq \tau^j < \tau^i$.

Similarly, consider the clusters that are optimal for the one–warehouse, multi–retailer subsystem whose graph is the subgraph induced by $\{w\} \cup s_w$. The clusters that are optimal for this subsystem are also computed using Algorithm 4 via the transformation of section 5. Let V_w be the cluster that contains w, and let $\tau^w = (\sum_{i \in V_w} K_i)/(\sum_{i \in V_w} g_i)$. The following observation holds.

Observation 2. If $r \in S_w - V_w$ and $i \in V_w - \{w\}$, then $\tau^r < \tau^w \leq \tau^i$.

The final observation is the key to understanding the algorithm used to solve RP.

Observation 3. The optimal clusters for the graph G of Figure 12 can be classified into four types. There is a special cluster called the major cluster, abbreviated MC. Let

$$\tau_{MC} = (\sum_{i \in MC} K_i)/(\sum_{i \in MC} g_i). \tag{72}$$

The other three cluster types are: The one–warehouse, multi–retailer clusters V_w, $w \in W$ for which $\tau^w > \tau_{MC}$; The joint replenishment clusters L_j, $j \in J$ for which $\tau^j < \tau_{MC}$; The one–node clusters $\{r\}$ for which $r \in R$, $\tau^r > \tau_{MC}$, $r \notin V_w$ for any $w \in W$; and the one–node clusters $\{r\}$ for which $r \in R$, $\tau^r < \tau_{MC}$, $r \notin L_j$ for any $j \in J$.

Observation 3 implies that the optimal clusters and reorder intervals for G can easily be determined once τ_{MC} is known. One way to interpret what the algorithm does is to view it as a parametric search for τ_{MC} and MC. It first determines what nodes would be in MC if τ_{MC} were infinitely large (as determined by Observation 3). The value of τ_{MC} is then continuously lowered from ∞ to its optimal value, and the algorithm performs the sequence of changes in the membership of MC that Observation 3 requires. Every time the membership of MC changes, the right–hand side of (72) is updated. The algorithm determines that the optimal value of τ_{MC} has been found when (72) holds. The algorithm has then established the optimal membership of MC.

The algorithm is given below. Step 2 requires that we solve the joint replenishment subproblems and the one–warehouse, multi–retailer subproblems referred to in Observations 1 and 2 above.

ALGORITHM 5. MULTI–ITEM, ONE–WAREHOUSE MULTI–RETAILER SYSTEMS

STEP 1. Let $\tau^r = K_r/g_r$, $r \in R$. Sort these values and list them from the largest to the smallest.

STEP 2. For all $j \in J$, solve the joint–replenishment problem formed by the graph $G(\{j\} \cup p_j)$. For all $w \in W$, solve the corresponding one–warehouse, multi–retailer problem over the graph $G(\{w\} \cup s_w)$. Let $\tau^j = (\sum_{i \in L_j} K_i)/(\sum_{i \in L_j} g_i)$ for all $j \in J$ and $\tau^w = (\sum_{i \in V_w} K_i)/(\sum_{i \in V_w} g_i)$ for all $w \in W$.

STEP 3. Insert τ^j and τ^w, $j \in J$, $w \in W$, into the previously generated list of τ^r, $r \in R$. If $\tau^r = \tau^w$, place τ^r before τ^w in the list. If $\tau^j = \tau^r$, put τ^j in the list ahead of τ^r.

STEP 4. Set the major cluster $MC = W$.

STEP 5. Set $\tau_{MC} = (\sum_{i \in MC} K_i)/(\sum_{i \in MC} g_i)$. If either the list is empty or the largest remaining number τ^n on the list satisfies $\tau^n \leq \tau_{MC}$, go to Step 6. Otherwise:

a. Remove the largest remaining number τ^n from the list.

b. If $n \in R$ and $(w,n) \in A(G)$ then:

i. If $n \in V_w$, set $MC \leftarrow MC \cup \{n\}$.

ii. If $n \notin V_w$, create the cluster $\{n\}$ and set $MC \leftarrow MC - \{n\}$. (Note: n may have entered MC via step 5d, or n may not be in MC.

c. If $n \in W$, create the cluster V_n and set $MC \leftarrow MC - V_n$.

d. If $n \in J$, set $MC \leftarrow MC \cup L_n$.

e. Go to Step 5.

STEP 6. The major cluster MC has been determined. If τ^j remains on the list for some $j \in J$, L_j is a cluster. If τ^j remains on the list for some $r \in R \cap P_j$ and $r \notin L_j$, then $\{r\}$ is a separate cluster. The remaining τ^n on the list do not correspond to clusters.

The optimal clusters are MC, plus the clusters identified in Steps 4bii, 4c, and 6. As before, once the clusters have been found the reorder intervals are computed using Steps 2 and 3 of Algorithm 1.

Some explanatory comments are in order. All $w \in W$ are initially in MC. Nodes $w \in W$ leave MC only through Step 5c. By Observation 2, whenever Step 5a is executed, we have $w \in MC$. Nodes $r \in R$ leave MC only through Steps 5c and 5bii. By Observation 2 and Step 5b, whenever we execute Step 5c we have $V_n \subset MC$. Nodes $r \in R$ join MC only in Steps 5bi and 5d. When Step 5bii is executed, either node n has joined MC via Step 5d, or node n is not in MC and the composition of MC does not change.

We now give a numerical example. The system consists of three items stocked at three retailers and one warehouse. The graph for this system is displayed in Figure 12 and the data for the problem are given in Table 2.

node i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
K_i	1	4	3	2	1	1	1	1	2	1	1	3	2	5	1
g_i	8	1	1	1	3	5	1	1	1	2	6	1	0	0	0

TABLE 2

Problem Data

The solutions to the one–warehouse, three–retailer problems and joint replenishment problems are as follows.

$n = 1$: $V_1 = \{1,4,7,10\}$, $\tau^1 = 5/12$.

$n = 2$: $V_2 = \{2\}$, $\tau^2 = 4$.

$n = 3$: $V_3 = \{3,12\}$, $\tau^3 = 3$.

$n = 13$: $L_{13} = \{13,5,6\}$, $\tau^{13} = 1/2$.

$n = 14$: $L_{14} = \{14,7,8,9\}$, $\tau^{14} = 3$.

$n = 15$: $L_{15} = \{15,11\}$, $\tau^{15} = 1/3$.

The list generated in Step 3 is:

n	2	14	12	3	9	4	8	7	13	10	1	15	5	6	11
τ^n	4	3	3	3	2	2	1	1	1/2	1/2	5/12	1/3	1/3	1/5	1/6

The optimal clusters are:

$N(G_1) = \{11,15\}$ with $T^*(1) = \sqrt{1/3}$,

$N(G_2) = \{5,6,10,13\}$ with $T^*(2) = \sqrt{1/2}$,

$MC = N(G_3) = \{1,4,7,14\}$ with $T^*(3) = \sqrt{9/10}$,

$N(G_4) = \{8\}$ with $T^*(4) = 1$,

$N(G_5) = \{9\}$ with $T^*(5) = \sqrt{2}$,

$N(G_6) = \{3,12\}$ with $T^*(6) = \sqrt{3}$, and

$N(G_7) = \{2\}$ with $T^*(7) = 2$.

The second system for which an efficient solution algorithm has been developed generalizes the multi–item, one–warehouse, multi–retailer system. In this system a number of items are distributed through a distribution network. The locations of the distribution network are indexed by ℓ, $0 \leq \ell < L$. All goods enter the system at location $L-1$. Each location $\ell < L-1$ receives its supply of all items it stocks from a unique *predecessor*

location $\varphi(\ell) > \ell$. Identifying locations with nodes and predecessor–successor relationships with arcs yields the location arborescence illustrated in Figure 13.

The set of items stocked in the system is $\mathcal{J}$. At location ℓ, each item $J \in \mathcal{J}$ has a constant demand rate (possibly zero) and a linear echelon holding cost rate. At each location, external demand can occur for any or all of the items. When an order for a set of items is placed at a location ℓ, the required amounts of inventory are instantly transferred from location $\varphi(\ell)$ to location ℓ.

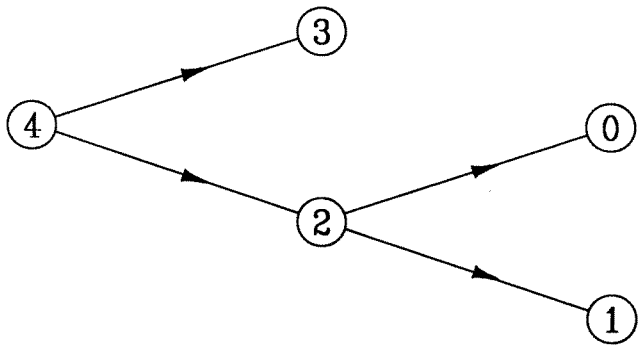


FIGURE 13

The Location Arborescence

Order costs are associated with subsets f(i), $1 \leq i \leq I$ of $\mathcal{J}$ called *families*. We assume that

$$\text{if } f(i) \cap f(i') \neq \phi \text{ then either } f(i) \subset f(i') \text{ or } f(i') \subset f(i) \tag{73}$$

Since items interact only through families, we assume without loss of generality that family $F(I)$ contains all of the items. For convenience we also assume that each individual item constitutes a separate family.

To illustrate, suppose that the items stocked are 1 (ice cream), 2 (frozen meat), and 4 (canned peas). The cost of providing a truck is incurred every time an order is placed. This cost is associated with the family $\{1, 2, 4\}$. If either or both of items 1 and 2 are ordered, the truck will have to be refrigerated. The incremental cost of providing a refrigerated truck is associated with the family $\{1, 2\}$. In addition, there is an administrative cost for each distinct item included in the shipment. This cost is associated with each of the families $\{1\}$, $\{2\}$, and $\{4\}$.

For each family $F(j)$, we define the *successor* $\beta(j)$ of j by letting $F(\beta(j))$ be the minimal family that properly contains the family $F(j)$. The uniqueness of $\beta(j)$ follows from (73). The family arborescence is defined by associating nodes with families and arcs with predecessor–successor relationships. Figure 14 shows the family arborescence for the simple, three–item example described above.

We index the location–family pair $(F(i),\ell)$ by $n \equiv i + \ell I$. Associated with each such pair is an order cost $K'_n = K'_{i+\ell I}$. The cost $K'_{i+\ell I}$ is incurred at every point in time at which an order is placed for any item in $F(i)$ at location ℓ. Thus, if $S \subset \mathcal{I}$ is the set of items ordered at location ℓ at time t, the total order cost incurred at location ℓ is $\sum_{F(i)\cap S\neq\phi} K'_{i+\ell I}$. The extended bill of material network G of this system is defined as follows. The node set of G is $N(G) \equiv \{i + \ell I: 1 \leq i \leq I,\ 0 \leq \ell < L\}$, the set of all location–family pairs. The arc set $A(G)$ of G consists of all *inventory arcs* of the form

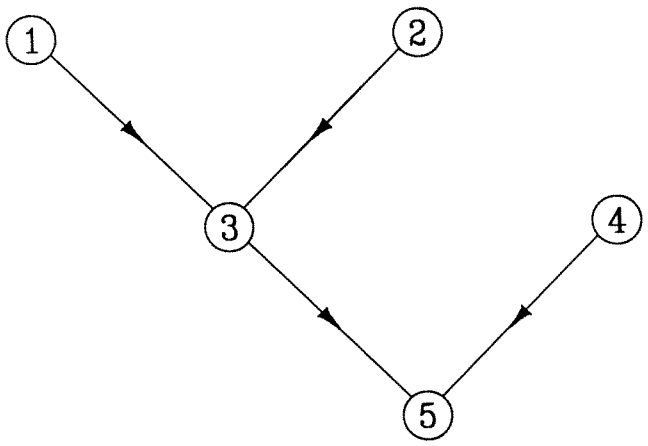


FIGURE 14

The Item Arborescence

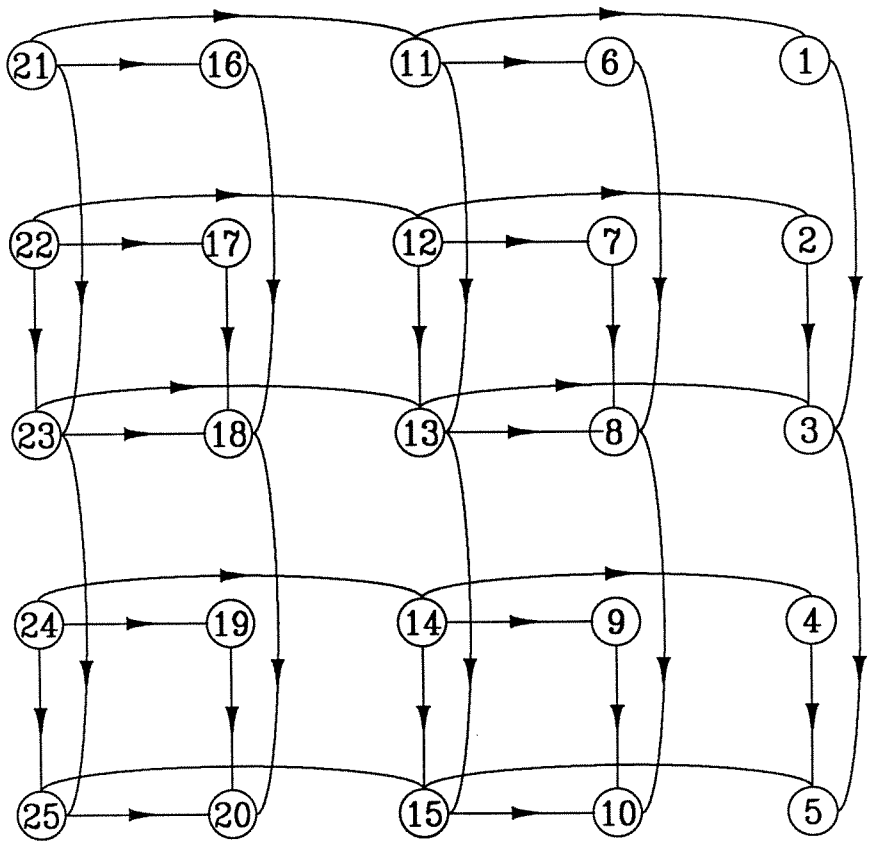


FIGURE 15

The Extended Bill of Material Network G

$(i + \varphi(\ell)I, i + \ell I)$, and of all order cost arcs of the form $(i + \ell I, \beta(i) + \ell I)$. G is illustrated in Figure 15. Since $I = 5$ and $13 = 3 + 2 \times 5$, node 13 corresponds to item 3 and location 2. Note that the nodes in each column of Figure 15 correspond to a location and that the nodes in each row correspond to a family.

For this system an O(LID log LI) algorithm has been developed where D is the depth of the item arborescence depicted in Figure 14 (Roundy (1987)). For most real–world systems D will seldom exceed five.

8.4 Models with Submodular Ordering Costs

Given a finite set of stages N^* let $P(N^*)$ be the set of all subsets of N^*. The function $\bar{K}: P(N^*) \to \mathbb{R}$ is said the be *submodular* if $\bar{K}(S) + \bar{K}(T) \geq \bar{K}(S \cap T) + \bar{K}(S \cup T)$ for all $S,T \in P(N^*)$. $\bar{K}$ is said to be *monotone* if $0 = \bar{K}(\phi)$ and $\bar{K}(S) \leq \bar{K}(T)$ whenever $S \subset T$. In this section we consider a model in which the cost of placing an order is a monotone submodular function of the set of items being ordered.

The model we consider is an extension of the models of section 6 and subsection 8.2. We are given a circuitless bill of material network G with node set N(G) and arc set A(G). Each node corresponds to a stage at which a single part is stocked. As before, echelon holding cost rates and demand rates are associated with each stage. The assumptions that time is continuous, orders are instantly delivered, demand is deterministic and constant, and stockouts are not allowed apply to this model as well.

The model of this subsection differs from our earlier models in that our order costs are determined by a submodular function $\bar{K}$ mapping P(N(G)) into $\mathbb{R}$. Whenever an order is placed the total order cost incurred is $\bar{K}(S)$ where S is the set of items being ordered at that point in time. By contrast, the systems we studied in sections 3 through 7 had order costs of the form $K(S) = \sum_{n \in S} K_n$. These order costs satisfy $K(S) + K(T) =$

$K(S \cup T) + K(S \cap T)$, i.e., they are *modular*. The systems considered in sections 8.1 and 8.3 are special cases of the family model of section 8.2. In this model $\bar{K}(S) = \sum_{f \in F: f \cap S \neq \phi} K_f$. Because $K_f \geq 0 \ \forall$ f, this is a submodular order cost function. Other interesting examples of submodular order costs are given by Zheng (1987).

The task of finding a nested powers–of–two policy for systems of this type can be formulated as a nonlinear integer program (NIP) similar to the ones we have formulated in earlier sections. The approach used to compute a policy is also similar to the approach we have used in earlier sections, and the same results on the relative cost of the policies computed apply (Queyranne (1985), Zheng (1987), Federgruen, Queyranne and Zheng (1989)). We first relax the integrality constraints in NIP and solve a continuous optimization problem. We then round off the resulting order intervals to powers–of–two using Step 3 of Algorithm 1. The continuous relaxation of the NIP is solved using a direct analog of the divide and conquer algorithm of section 6. In the divide and conquer algorithm of section 6, a maximum flow problem was solved in each iteration. For this model a maximal polymatroidal network flow problem is solved at each iteration. This polymatroidal network flow problem has special structure, and Zheng (1987) has developed a special-purpose algorithm for solving it. The algorithm has a running time of $O(n^4 d)$. Here d is the time required to answer the following question: Given j, $1 \leq j \leq n$, and a vector $x = (x_i \geq 0: 1 \leq i \leq n)$ that satisfies $\sum_{i \in S} x_i \leq \bar{K}(S)$ for all $S \subset N(G)$, find the largest amount by which x_j can be increased without violating any of the inequalities $\sum_{i \in S} x_i \leq \bar{K}(S)$, $S \subset N(G)$. In practice one would hope that the systems being modeled have special structure which allows this computation to be performed easily.

For the special case of a joint replenishment system with submodular costs, in which the network G has no arcs, a much more efficient algorithm exists. Before describing the algorithm we consider what the average cost of a powers–of–two policy is. Suppose that we

are given clusters $C_1, C_2, \ldots, C_N$, that all nodes $i \in C_k$ place orders once every $T(k)$ days, that $T(k) = 2^{\ell(k)}T_L$, $\ell(k)$ integer, and that $T(k) \leq T(k+1)$ for all k. The average holding cost incurred per day is clearly $\sum_{k=1}^{N} g(C_k)T(k)$ where, as before, $g(C_k) = \sum_{i \in C_k} g_i$. Let $T(N+1) = \infty$ and let $\bar{C}_k = \bigcup_{\ell \leq k} C_\ell$. On the average, once every $\left[\frac{1}{T(k)} - \frac{1}{T(k+1)}\right]^{-1}$ days an order is placed for the items in $\bar{C}_k$ and no others. Therefore, the average order cost incurred is $\sum_{k=1}^{N} \bar{K}(\bar{C}_k)\left[\frac{1}{T(k)} - \frac{1}{T(k+1)}\right]$. Thus the average cost of the policy is

$$\sum_k [(\bar{K}(\bar{C}_k) - \bar{K}(\bar{C}_{k-1}))/T(k) + g(C_k)T(k)]$$

where $\bar{C}_0 = \phi$.

As before, we initially ignore the integrality constraints and compute clusters by solving a convex, nonlinear optimization problem. In the solution to this problem, the reorder interval for nodes in cluster C_k is

$$T^*(k) = [(\bar{K}(\bar{C}_k) - \bar{K}(\bar{C}_{k-1}))/g(C_k)]^{1/2}. \tag{74}$$

For a given set of clusters $C_1, \ldots, C_N$, $\bar{C}_\ell = \bigcup_{k \leq \ell} C_k$, and for each node set $S \subseteq C_\ell$, we define $\bar{K}_\ell(S) = \bar{K}(S \cup \bar{C}_{\ell-1}) - \bar{K}(\bar{C}_{\ell-1})$.

The function $\bar{K}_\ell$ is itself a monotone submodular function on the subsets of C_ℓ. The algorithm used to compute the clusters is similar to the Divide and Conquer Algorithm of section 6. It starts with all nodes in a single cluster, and successively splits clusters into smaller clusters, until the final set of clusters has been identified. The algorithm follows.

ALGORITHM 6. JOINT REPLENISHMENT SYSTEMS WITH SUBMODULAR ORDER COSTS

STEP 0. Set $N \leftarrow 1$, $\ell \leftarrow 1$, $C_1 \leftarrow \{1,2,\ldots,n\}$.

STEP 1. Find a maximal vector $X = (X_i : i \in C_\ell)$ satisfying $\sum_{i \in S} X_i \leq \bar{K}_\ell(S)$ for all $S \subseteq C_\ell$ and $X_i \leq U_i^\ell$ for all $i \in C_\ell$, where $U_i^\ell = g_i \bar{K}_\ell(C_\ell)/g(C_\ell)$.

STEP 2. Set $A \leftarrow \text{sat}(X)$ where $\text{sat}(X)$ is the (unique) maximal subset of C_ℓ satisfying $\sum_{i \in \text{sat}(X)} X_i = \bar{K}_\ell(\text{sat}(X))$.

STEP 3. If $A = C_\ell$ then set $\ell \leftarrow \ell + 1$ and go to Step 4 (C_ℓ is one of the clusters). If $A \neq C_\ell$ then set $N \leftarrow N + 1$, $C_{m+1} \leftarrow C_m$ for each $m > \ell$, $C_{\ell+1} \leftarrow C_\ell \backslash A$, and $C_\ell \leftarrow A$.

STEP 4. If $\ell = N + 1$ then go to Step 5. Otherwise go to Step 1.

STEP 5. Compute $T^*(k)$ for each cluster k using (74). Then compute the reorder intervals T_i of the items using Step 3 of Algorithm 1.

The running time of this algorithm depends on the amount of time required to perform Steps 1 and 2. In practice one would hope that the function $\bar{K}$ has special structure which enables these computations to be performed easily. For a more complete discussion of this model and algorithm, see Zheng (1987).

9. NON–NESTED POLICIES

In sections 3 through 8, we have restricted ourselves to stationary nested powers–of–two policies. For the systems of sections 3 and 4, nested policies were optimal. However for the other systems we have considered, optimal nested policies can be seriously sub–optimal. Suppose for example that we are modeling a three–stage distribution system. Stage one (the factory) supplies two outlets, stages two and three. The factory is in Albany, outlet one is in New York City, and outlet two is in Singapore. The problem data is in

Table 3 below. The time units are weeks. The demand rate λ at the Singapore outlet is very low, and the order cost K there is very high.

	Stage 1 Factory	Stage 2 New York	Stage 3 Singapore
Order Cost	1	1	K
Demand Rate	0	2	λ
Echelon Holding Cost	1	1	1

TABLE 3

Problem Data

A reasonable policy for this system is to place orders at the factory and at the New York outlet on a weekly basis, and to place orders from Singapore approximately once every $\sqrt{K/\lambda} > 1$ weeks. However, if a nested policy were followed, an order could not be placed at the factory unless a simultaneous order were placed in Singapore. A nested policy would incur either high inventory costs at the factory due to a long reorder interval there, or high order costs at the Singapore outlet due to a low reorder interval there, or both. For several sets of problem parameters, the effectiveness of an optimal nested policy is illustrated in Table 4. The effectiveness is the ratio of the cost of an optimal policy to the cost of an optimal nested policy.

	K = 2	K = 32	K = ∞
$\lambda = 1/2$	96%	75%	58%
$\lambda = 1/32$	82%	44%	17%
$\lambda = 0$	73%	29%	0%

TABLE 4

Effectiveness of Optimal Nested Policies

As this example illustrates, nested polices tend to become seriously sub–optimal when relatively high order costs are combined with relatively low demand rates. In this model, n retailers with order costs of K and demand rates of λ have the same effect as one retailer with an order cost of nK and a demand rate of $n\lambda$. Therefore no individual retailer needs to have an uncommonly high order cost for nested policies to perform poorly.

9.1 Algorithm for One–Warehouse, Multi–Retailer Systems

Consider a single warehouse (stage 0) which supplies a number of retailers (stages 1 through n). The order cost at stage i is K_i, the echelon holding cost rate at stage i is h_i, and the demand rate at retailer $i \geq 1$ is λ_i. The bill of material network is illustrated in Figure 16. As before, we seek to minimize the total order and inventory costs subject to the constraint that all demand must be met without stockouts.

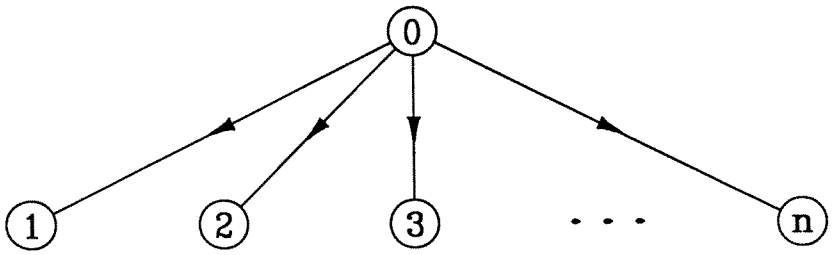


FIGURE 16

One Warehouse, Multi–Retailer System

In earlier sections we restricted attention to stationary nested powers–of–two policies which satisfy the Zero–Ordering Property. In this section we drop the assumptions of stationarity and nestedness, and study powers–of–two policies that satisfy the Zero–Ordering Property. This means that orders are placed at equal intervals of time which

are powers–of–two multiples of T_L, and that the inventory at a stage is zero every time an order is placed there.

Let orders be placed at stage i once every T_i days. Let $g_i = \frac{1}{2} h_i \lambda_i$ and let $g^i = \frac{1}{2} h_0 \lambda_i$. Let $T = (T_0, \ldots, T_n)$ be a powers–of–two policy satisfying the Zero–Ordering Property. For the purpose of computing holding costs, we separate the inventory at the warehouse into n categories, according to the retailer to which it will be shipped. Let $h_i(T_0, T_i)$ be the average annual cost of holding inventory at the warehouse that is destined to be shipped to retailer i, and of holding inventory at retailer i. We will show that

$$h_i(T_0,T_i) = g_i T_i + g^i(T_0 \vee T_i) \tag{75}$$

where $\vee$ denotes the maximum.

CASE 1: $T_i \geq T_0$. In this case the warehouse places an order simultaneously with every order at the retailer. Therefore no inventory for retailer i is ever held at the warehouse, and the only costs to be considered are the costs at the retailer. As in the familiar single–item model,

$$h_i(T_0, T_i) = (g_i + g^i)T_i ,$$

which agrees with (75) if $T_i \geq T_0$.

CASE 2: $T_i < T_0$. In this case it is convenient to use the echelon method of computing order costs. We consider only those goods that will eventually be consumed by external demand at retailer i. The total system inventory of these goods follows the familiar sawtooth inventory pattern with an order interval of T_0. The inventory at retailer i

follows a sawtooth inventory pattern with order interval T_i. Therefore the average cost of holding all inventory in the system that is destined to be consumed at retailer i is

$$h_i(T_0, T_i) = g_i T_i + g^i T_0 ,$$

which agrees with (75) when $T_i \leq T_0$.

Therefore the average cost of a powers–of–two policy is given by

$$c(\mathcal{T}) = \sum_{i \geq 0} \frac{K_i}{T_i} + \sum_{i \geq 1} [g_i T_i + g^i (T_0 \vee T_i)] . \tag{76}$$

Since we are considering powers–of–two policies satisfying the Zero Inventory Ordering Property, the only constraint on the order intervals is

$$T_i = 2^{\ell_i} T_L, \ \ell_i \text{ integer}, \ T_L > 0 . \tag{77}$$

Therefore, the problem of finding an optimal powers–of–two policy is the problem of minimizing (76) subject to (77).

We use the same basic approach that we have used in earlier sections. We first relax (77) and minimize (76) over all non–negative $\mathcal{T}$ (Roundy (1985)). We then round–off the reorder intervals thus computed to integer powers of two. We use Step 3 of Algorithm 1 if T_L is fixed, and we use the algorithm in Roundy (1985) if T_L is variable. In the former case the cost of the policy is within 6% of the minimum of (76), and in the latter case the cost of the policy is within 2% of the minimum of (76).

It is possible to show that the minimum of (76) is in fact a lower bound on the average cost of any feasible policy whatsoever. In sections 5 and 6 we computed policies whose costs were close to the cost of an optimal nested policy, but optimal nested polices can

be far from optimal. The policies we compute in this section have costs that are close to the cost of an optimal policy.

What remains is to show how (76) can be efficiently minimized. The intuition behind the algorithm is as follows. There are three kinds of retailers. Those retailers who place orders less frequently than the warehouse does are in category G. If $\mathcal{T}^* = (T_i^*: 1 \leq i \leq n)$ minimizes (76) then local optimality and (76) dictate that

$$T_i^* = \sqrt{K_i/(g_i + g^i)} > T_0^* \text{ if } i \in G. \tag{78}$$

Those retailers who place orders more frequently than the warehouse does are in category L. Local optimality and (76) dictate that

$$T_i^* = \sqrt{K_i/g_i} < T_0^* \text{ if } i \in L. \tag{79}$$

Finally, those retailers who place orders simultaneously with the warehouse are in category E. Local optimality and (76) dictate that they place orders once every

$$T_0^* = \sqrt{[K_0 + \sum_{i \in E} K_i]/[\sum_{i \in E} (g_i + g^i) + \sum_{i \in L} g^i]} \tag{80}$$

days. The algorithm searches for sets G, L, and E that satisfy (78) – (80). It is given below.

ALGORITHM 7. NON–NESTED POLICIES FOR ONE–WAREHOUSE MULTI–RETAILER SYSTEMS

STEP 1. Calculate the reorder cycles $\tau'_i = [K_i/(g^i + g_i)]^{1/2}$ and $\tau_i = (K_i/g_i)^{1/2}$ and sort them to form a nondecreasing sequence of $2n$ numbers. Call this sequence S. Label each reorder cycle with the value of i and with an indicator showing whether it is the conventional reorder cycle τ'_i or the echelon reorder cycle τ_i.

STEP 2. Set $E = G = \phi$, $L = \{1, \ldots, n\}$, $K = K_0$, and $H = \sum_{i \geq 1} g^i$.

STEP 3. Let τ be the largest element of S. If $\tau^2 \geq K/H$ and $\tau = \tau_i$ is an echelon reorder cycle, remove τ from S and update E, L, K, and H by $E \leftarrow E \cup \{i\}$, $L \leftarrow L\backslash\{i\}$, $K \leftarrow K + K_i$ and $H \leftarrow H + g_i$. Then go to Step 3. If $\tau^2 > K/H$ and $\tau = \tau'_i$ is a conventional reorder cycle, remove τ from S and update E, G, K, and H by $E \leftarrow E\backslash\{i\}$, $G \leftarrow G \cup \{i\}$, $H \leftarrow H - g^i - g_i$ and $K \leftarrow K - K_i$, and go to Step 3. Otherwise, the current sets G, L, and E are optimal. Go to Step 4.

STEP 4. Set $T_0^* = (K/H)^{1/2}$. Then $T_i^* = T_0^*$ for all retailers $i \in E$, and T_i^* for retailers not in E is given by (78) and (79).

The sort in Step 1 requires $O(n \log n)$ comparisons. All other phases of the algorithm require a number of operations that is linear in n.

9.2 The Extended Bill of Material Network

We return momentarily to the three–node example at the beginning of section 9. In this example, we could conceptually separate the inventory at the factory into two categories; the inventory that will be shipped to New York and the inventory that will be shipped to Singapore. We could then assign separate reorder intervals to these two

categories of inventory. At the factory we could order inventory bound for New York once per week, and order inventory bound for Singapore approximately once every $\sqrt{K/\lambda}$ weeks. In this way we would succeed in managing the system effectively while preserving much of the structure of stationary nested powers-of-two policies.

This is essentially what the algorithm of section 9.1 does. We can interpret it as an algorithm which computes nested policies on a modified bill of material network. The bill of material network in Figure 16 is modified in the following way. We split the warehouse into $n + 1$ different stages, labeled 0, $n + 1$, $n + 2$, ... $2n$. Stage 0 represents the joint order cost at the warehouse. Stage i, $1 \leq i \leq n$, represents retailer i, and stage $n + i$, $1 \leq i \leq n$, represents the inventory at the warehouse that will be shipped to retailer i. The bill of material network associated with this system is illustrated in Figure 17. Table 5 lists the data associated with each stage. Note that relative to the network of Figure 17, the policies of subsection 9.1 and Table 5 are nested. Also note that the sum of the average costs associated with the stages corresponds to the total average cost of the policy $\mathcal{T}$ is given by (76).

Stage Index	Order Cost	Holding Cost Coefficient	Reorder Interval	Average Cost
0	K_0	0	T_0	K_0/g_0
$i, 1 \leq i \leq n$	K_i	g_i	T_i	$K_i/T_i + g_i T_i$
$n + i, 1 \leq i \leq n$	0	g^i	$T_0 \vee T_i$	$g^i(T_0 \vee T_i)$

TABLE 5

Data for the Stages

This example illustrates our approach to overcoming the limitations of nested policies. We transform the original bill of material network into an expanded network in which nested policies are near–optimal. This is accomplished by partitioning the inventory at each stage into classes according to the way in which the goods will be routed through the system. We treat these different classes of inventories as if they were distinct stages, and allow them to be controlled by different reorder intervals. This approach allows us to compute policies that are known to be near–optimal for systems with general bill of material networks.

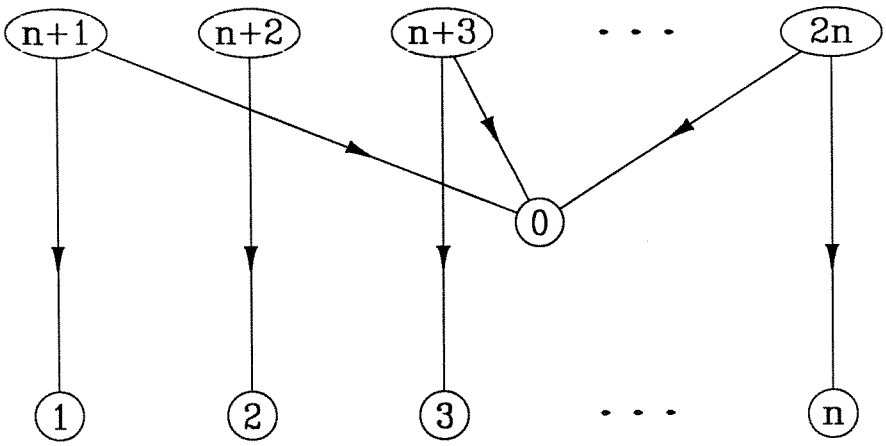


FIGURE 17

Non–Nested Policies for One–Warehouse, Multi–Retailer Systems

We now describe a procedure for transforming an arbitrary bill of material network into an expanded network on which nested policies are near–optimal. Assume that we are given a bill of material network G with node set N and arc set A. This network may contain order cost nodes and order cost arcs (see section 8.2). The following data are associated with the network. For each node $i \in N$ there is an order cost $K_i \geq 0$, an echelon holding cost $h_i \geq 0$, and a demand rate $\lambda_i \geq 0$. For each arc $(i,j) \in A$ there is a gozinto coefficient $\gamma_{ij} \geq 0$. This data is sufficient to define an instance of the production planning problems of sections 3 through 6 and section 8, with the exception of the submodular cost models of subsection 8.4. We make the following assumptions.

PROPERTY 1. G is an acyclic directed graph.

PROPERTY 2. If no arcs emanate from node i in G, then $K_i > 0$.

PROPERTY 3. If no arcs lead into node i in G, then $g_i > 0$.

As mentioned in section 8.2, these properties guarantee that the reorder intervals will be positive and finite.

In this subsection it is convenient to use the *external demand rates* $D_i, i \in N(G)$ for the nodes rather than the total demand rates $\lambda_i, i \in N(G)$ that we use elsewhere. The external and total demand rates are related via the equation

$$\lambda_i = D_i + \sum_{(i,j)\in A(G)} \gamma_{ij}\lambda_j \quad \forall\ i \in N(G) .$$

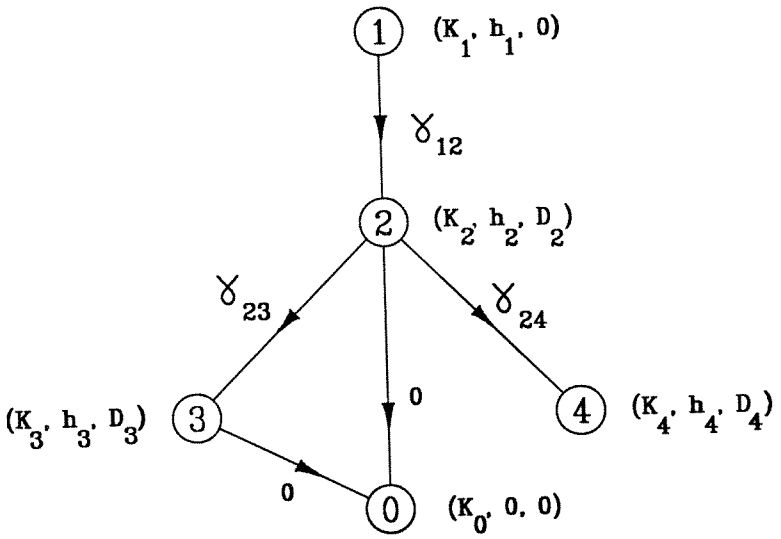
An arc $(i,j) \in A(G)$ is an *inventory arc* if $\gamma_{ij} > 0$, and it is an *order cost arc* if $\gamma_{ij} = 0$. A *common part node* is a node with more than one emanating inventory arc, or a node with one emanating inventory arc and with a positive external demand rate. The constraint that a nested policy must be followed can cause trouble at common part nodes.

The algorithm we use to transform the bill of material network is iterative in nature. We say that node i *precedes* node j in a network if there is a directed path from i to j in the network. At each iteration the procedure selects a common part node i that does not precede any other common part node in the network. It then makes local changes to the network by splitting node i into several different nodes, none of which is a common part node. The algorithm terminates when no common part nodes are left.

The iterative step of the algorithm is illustrated in Figure 18. There is an order cost associated with the family {2, 3}, which is represented by node 0. All arcs not adjacent to node 0 are inventory arcs. Initially, node 2 is the only common part node. Nodes 2, 3, and 4 have positive external demand rates.

The iterative step of the algorithm is as follows. Let node i be the common part node selected (node 2 in the example). Node i is split into a number of different nodes. For every inventory arc (i,j) emanating from node i there is a new *inventory node* n(i,j) (nodes 5 and 6 in the example). The inventory of i that will be used to make part j is associated with n(i,j). Node n(i,j) inherits a copy of each arc leading into node i, and it inherits the arc (i,j) leading out of node i. The order cost and the external demand rate for node n(i,j) are both zero, and n(i,j) inherits the echelon holding cost rate of node i. If $D_i > 0$ then there is a new *demand node* n(i,d) (node 7 in the example). The inventory of i that will be used to satisfy the external demand at i is associated with n(i,d). Node n(i,d) inherits a copy of each arc leading into node i. The order cost for node n(i,d) is zero, and n(i,d) inherits the holding cost rate and the external demand rate of node i. Finally, a node n(i,K) (node 8 in the example) is used to account for the order cost. The holding cost rate and the external demand rate for node n(i,K) are zero, and n(i,K) inherits the order cost of node i. Node n(i,K) also inherits all order cost arcs leading out of node i. We create a new order cost arc from each new inventory node n(i,j) to node n(i,K), and from the new demand node n(i,d) (if there is one) to node n(i,K). Figure 18(b) shows the network at the end of the iteration.

Note that none of the nodes created during the iterative step is a common part node. However, in the example of Figure 18, node 1 has become a common part node. The next iteration will split node 1 into several nodes. Figure 18(c) displays the result. In this network there are no common part nodes, so the procedure terminates. Another example of how a bill of material network is transformed by this algorithm is given in Figure 19. In this figure, initially all arcs are inventory arcs.

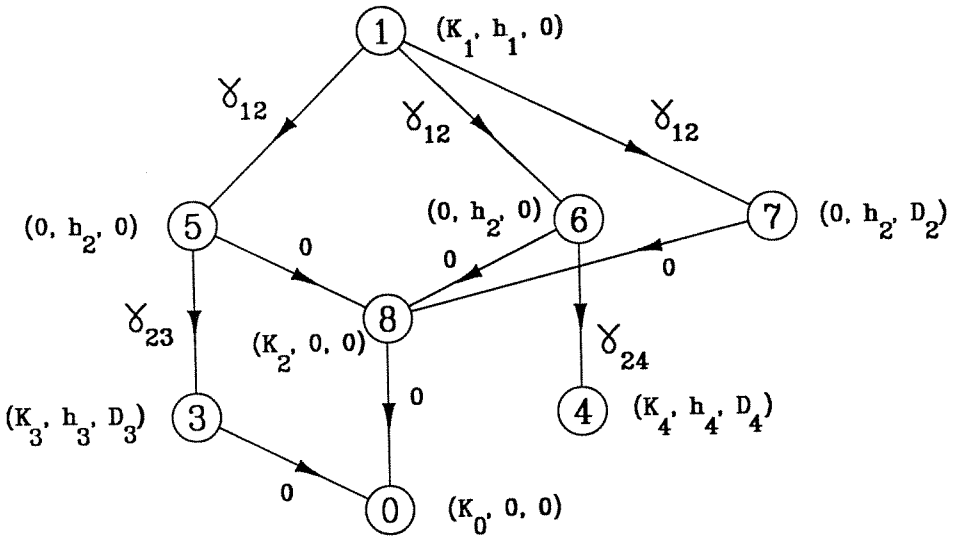


(a) G

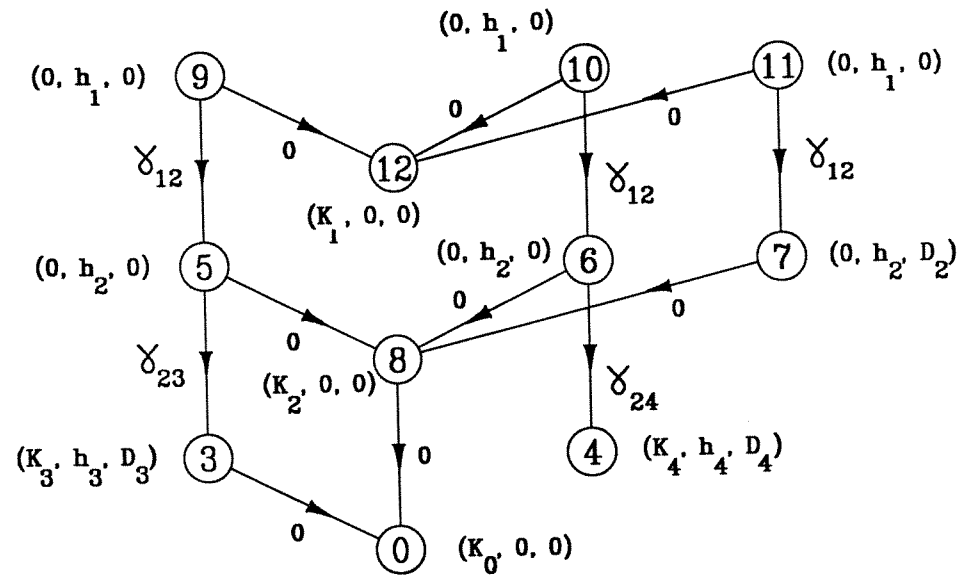
FIGURE 18

The Transformation from G to G^*

Legend: (Order Cost, Echelon Holding Cost, External Demand Rate)



(b) End of First Iteration

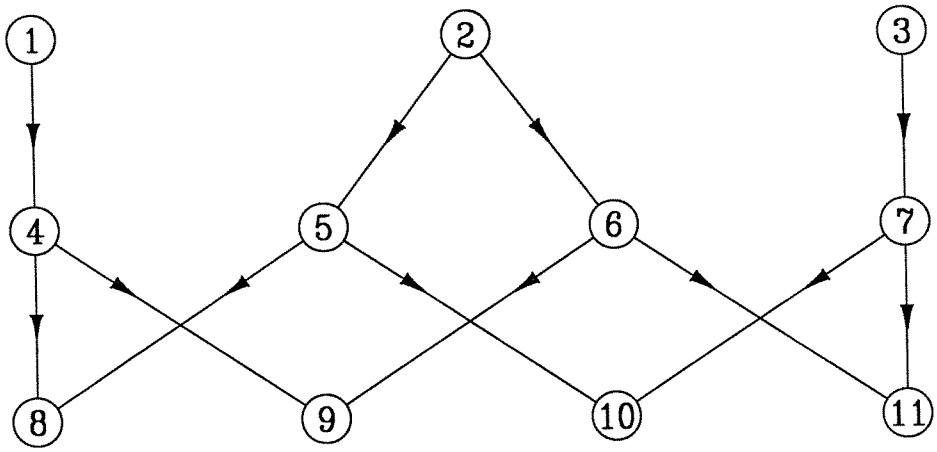


(c) G^*

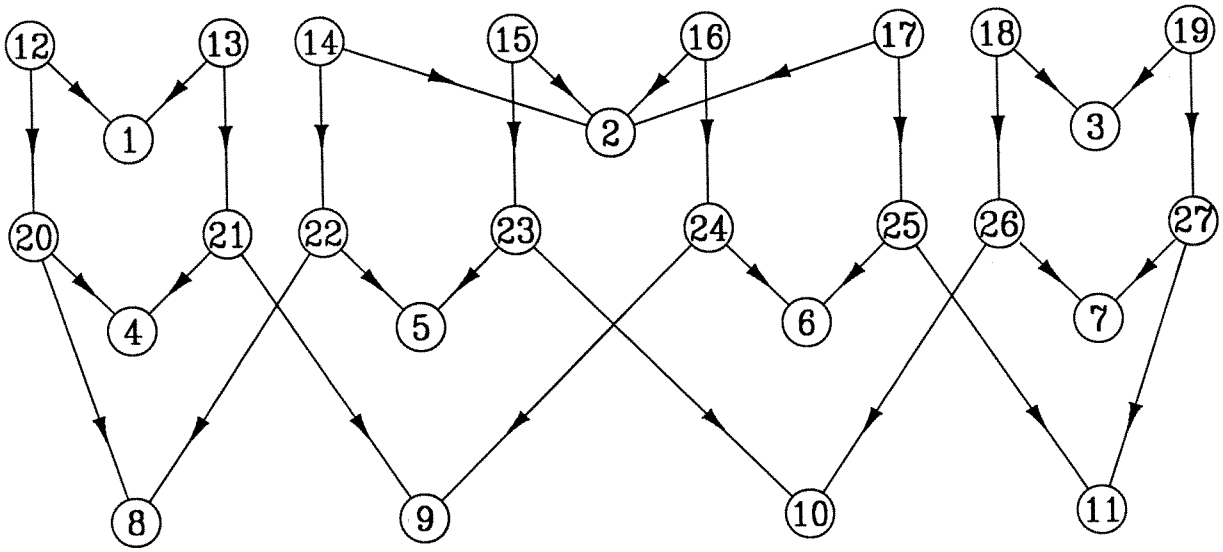
FIGURE 18

The Transformation from G TO G^*

Legend: (Order Cost, Echelon Holding Cost, External Demand Rate)



(a) G



(b) G^*

FIGURE 19

The Networks G and G^*

The iterative step of the algorithm preserves Properties $1-3$ listed above, so the algorithm produces a network and associated data that can be interpreted as a bill of material network. Let G be the original bill of material network, and let G^* be the transformed bill of material network. Recall that in all bill of material networks, policies must be nested on setup arcs in order to be feasible. We have discussed how inventories are associated with the nodes of G^*. Using this association, one can easily show that any feasible policy for G corresponds to a feasible policy for G^* that has the same cost, and vice–versa. Therefore either of these networks can be used to model the system being studied. Policies that are nested in G are nested in G^*, but not vice–versa. The advantage of G^* lies in the following fact: whereas an optimal nested policy can be far from optimal, there is always a policy that is nested in G^* and is within 6% of optimal. More precisely, the solution of the continuous relaxation of the lot sizing problem (problem (35) of section 6) for G^* is a lower bound on the cost of any feasible policy whatsoever (Roundy (1986)). Consequently, by applying the algorithms of section 6 to G^*, we can compute a policy that is within 6% of optimal if the base planning period is fixed, and within 2% of optimal if it is variable.

The network G^* is very sparse. For example, if G has no order cost families then each node of G^* has at most two emanating arcs, so $|A(G^*)| \leq 2|N(G^*)|$. However, in the worst case, $|N(G^*)|$ is exponential in $|N(G)|$. Let $F(G)$ be the set of finished products in G, i.e., the set of stages or nodes in G that have positive external demand, and let R_{ij} be the number of directed paths from node i to node j in G. It can be shown that if there are no order cost families then $|N(G^*)| \leq |N(G)| + \sum_{i \in N(G), j \in F(G)} R_{ij}$. In most real-world bill of material networks $|R_{ij}|$ is $O(1)$. For this reason, a quick and crude estimate of $|N(G^*)|$ is $|N(G)| \times (F+1)$ where F is the average over all $i \in N(G)$ of the number of finished products that require part i. For the industrial system illustrated

section 6, $|N(G)| = 94$, $|F(G)| = 4$, $|N(G^*)| = 282$, $|N(G)| \times (F + 1) = 294$, and $|R_{ij}| \leq 2$ for all $i,j \in N(G)$.

Zheng (1987) has proposed using a different network G^{**} in place of G^*. The networks G^* and G^{**} are equivalent in terms of the policies computed, their costs, and the lower bound. The advantage of G^{**} is that it is bipartite and has the same number of nodes as G^*. The disadvantage of G^{**} is that it can have as many as $\frac{1}{2} \times |N(G)| \times |N(G^*)|$ arcs, whereas G^* has fewer then $2 \times |N(G^*)|$ arcs.

Non–nested policies can easily be combined with submodular order costs of the type considered in section 8.4. See Queyranne (1985), Zheng (1987), Federgruen, Queyranne and Zheng (1989) for details.

10. OTHER EXTENSIONS

In this section we briefly describe other related research, including extensions of the models that we have discussed and improved algorithms for computations. We begin with extensions to the models.

10.1 Lead Times

Suppose that there is a positive, deterministic lead time for preparing a shipment at the originating stage, for transportation from one stage to another, and for receiving the shipment at the destination stage. We assume that these lead times are dependent only on the two stages involved. In particular they are independent of the quantity of the order. Lead times of this sort can often be incorporated without causing a drop in the performance of the policies we have discussed, or a significant amount of extra computation.

For the serial, assembly, and single–item distribution systems discussed in sections 3, 4, and 5, we handle lead times by initially ignoring them. After a schedule has been computed, the orders at each stage are simply translated forward or backward in time by an appropriate amount. The theorems on the worst–case relative cost of the resulting schedule still apply.

The multi–item distribution systems discussed in subsection 8.3 can be handled similarly. Assume that the lead time for processing and shipping an order is a function only of the two locations involved; it is independent of which items are ordered and of the order quantities. Once again, we initially ignore the lead times. After a schedule has been computed, the orders at each location are translated forward or backward in time by an appropriate amount.

For systems with arbitrary bill of material networks, such as those discussed in sections 6 and 8.2, lead times can be handled in a similar way without causing a decrease in system performance if the lead times are "balanced" (Roundy (1986)). This condition holds if the bill of material network G has no cycles, but it is likely to fail for networks with cycles unless some very special structure, such as the structure of multi–item distribution systems, is present.

10.2 Finite Production Rates

Often a stage is a manufacturing operation, and production occurs at a finite rate. Recently several researchers have constructed detailed models of the finiteness of the production rate, the way in which material is transferred from one stage to another, and the impact that this has on intervening inventories. The impact is clearly larger if the production rates are small compared to the echelon demand rates. Schwarz and Schrage (1975) consider an assembly system of the type discussed in section 4 under the assumption that material is transferred from one stage to another only after a batch is completed. Karimi (1987) studied a two–stage serial system with finite production rates and continuous transfer of material from one stage to another. Szendrovits (1975) studied serial systems in which a single reorder interval T is used at all stages, and material is transferred between stages in transfer batches of size b, $1 \leq b \leq T\lambda$, where λ is the demand rate. Atkins, Queyranne, and Sun (1989) have obtained policies that are within 6% of optimal for

assembly systems with finite production rates in which the processing rate at the successor of node n is not less than node n's processing rate, for all n.

10.3 Bounds on the Reorder Intervals

As was mentioned earlier, any of the lot–sizing models considered herein can be enhanced by adding the constraint

$$2^{\underline{\ell}} T_L \leq T_i \leq 2^{\bar{\ell}} T_L \quad \forall\ i \in N(G).$$

The only change required in the algorithms is the simple alteration to the roundoff procedure described in section 3.4. All claims of optimality and near–optimality can be shown to apply.

In many cases one would like the lower and/or upper bound on T_i to be different for different values of i. Recently Zheng (1987) has provided a general algorithm for solving this problem which is similar in spirit to the divide and conquer algorithm. He has also shown that the policies computed in this way have provably near–optimal costs. Best and Chakravarty (1987) have developed efficient algorithms for the continuous relaxation of the lot sizing problem for serial systems.

10.4 Fixed Batch Sizes

Anily and Federgruen (1986) have studied a multi–item, two–level system in which the top level consists of a single stage which produces a single generic part in batches of fixed size. The second level consists of a step in which the generic part is transformed into any of several different parts, each of which has constant demand. Traditional holding costs apply at all stages. Order costs are also present, but the way in which they are modeled is non–standard. They present efficient algorithms for computing policies that are known to be within 6% of optimal.

10.5 Algorithms for Special Bill of Material Structures

There are several other algorithms that have been devised for solving the continuous relaxation of the lot sizing problem (Problem (35) of section 6) for different bill of material networks G. Some of these were developed for the problems of isotonic regression and precedence–constrained single–machine scheduling (Roundy (1986)). For serial networks a linear–time algorithm exists (Ayers et al. (1955), Best and Chakravarty (1987)). For series–parallel digraphs an $O(n \log n)$ algorithm exists (Lawler (1978)). For the assembly system of section 4, a linear–time algorithm based on median-finding was recently developed by Queyranne (1987). An $O(n \log n)$ algorithm for arbitrarily directed trees has been developed by Jackson and Roundy (1985).

10.6 Efficient Algorithms for Systems with Fixed Base Planning Periods

The divide and conquer algorithm of section 6 for solving the continuous relaxation (35) of the lot sizing problem (34) can be described as follows. We first assume that all of the stages are in a single cluster. We compute the reorder interval $\tau = [(\sum_i K_i)/(\sum_i g_i)]^{1/2}$ that this cluster would use. We then set the reorder intervals of all stages equal to τ and we attempt to prove that all stages are in a single cluster by finding a feasible dual solution for (35). Dual solutions correspond to feasible flows in a certain network. If we find a feasible solution, we know that we were correct in assuming that one cluster contains all of the stages. If we do not find one, we find a minimal cut which divides the nodes of G into two classes; those nodes i for which $T_i^{*2} \geq \tau$ and those nodes i for which $T_i^{*2} < \tau$, where $(T_i^*: 1 \leq i \leq n)$ solves (35). This information allows us to decompose the original problem (35) into smaller subproblems of the same type.

The computations described above can be performed for any value of τ. In particular, if T_L is given and we use $\tau = 2^{\ell+1/2}$, the algorithm will tell us which stages i satisfy $T_i^{*2} \geq \tau$ and which ones satisfy $T_i^{*2} < \tau$. By repeating this computation for

selected integer values of ℓ, we can identify integers ℓ_i, $i \in N(G)$ such that $2^{\ell_i - 1/2} T_L \leq T_i^* < 2^{\ell_i + 1/2} T_L$, without ever finding the exact value of T_i^*. The round–off procedure (Step 3 of Algorithm 1) will select $T_i = 2^{\ell_i} T_L$, which determines our policy.

This idea is due to Zheng (1987). Assuming that the number of different values of ℓ that we need to consider is uniformly bounded, he has obtained improved running times for many of the models discussed herein.

REFERENCES

Aho, A.V., Hopcroft, J.E., and Ullman, J.D. (1974), **The Design and Analysis of Computer Algorithms**, Addison–Wesley Publishing Company, Reading, Massachusetts.

Anily, S. and Federgruen, A. (1987), *Structured Partitioning Problems: Part II. Applications to Capacitated Two–Stage Multi–Item Production/Inventory Models.* Research Paper, Faculty of Commerce, University of British Columbia.

Arrow, K.J., Harris, T., and Marschak, J. (1951), *Optimal Inventory Policy.* **Econometrica** XIX, 250–272.

Arrow, K.J., Karlin, S., and Scarf, H. (1958), **Studies in the Mathematical Theory of Inventory and Production**, Stanford University Press, Stanford, California.

Atkins, D., Queyranne, M., and Sun, D. (1989), *Lot Sizing Policies for Finite Production Rate Assembly Systems.* Research Paper, Faculty of Commerce, University of British Columbia.

Ayer, M., Brunk, H.D., Ewing, G.M., Reid, W.T., and Silverman, E. (1955), *An Empirical Distribution Function for Sampling with Incomplete Information.* **Annals of Math. Stat.** 26, 641–647.

Best, M.J. and Chakravarty, N. (1987), *Active Set Algorithms for Isotonic Regression: A Unifying Framework.* Research Report CORR 87–24, Faculty of Mathematics, University of Waterloo.

Best, M.J. and Chakravarty, N. (1988), *Active Set Algorithms for Isotonic Regression when the Underlying Graph is an Arborescence.* Research Report CORR 88–18, Faculty of Mathematics, University of Waterloo.

Blackburn, J.D. and Millen, R.A. (1982), *Improved Heuristics for Multi–Stage Requirements Planning Systems.* **Management Science** 28, 44–56.

Caie, J.P. and Maxwell, W.L. (1981), *Hierarchical Machine Load Planning.* **Multi–Level Production/Inventory Control Systems: Theory and Practice** (L.B. Schwarz, ed.), North Holland, New York, New York, Chapter 5.

Clark, A. and Scarf, H. (1960), *Optimal Policies for a Multi–Echelon Inventory Problem.* **Management Science** 19, 517.

Clark, A. and Scarf, H. (1962), *Approximate Solutions to a Simple Multi–Echelon Inventory Problem.* **Studies in Applied Probability and Management Science** (K. Arrow, S. Karlin, and H. Scarf, eds.), Stanford University Press, Stanford, California, Chapter 5.

Crowston, W.B. and Wagner, M.H. (1973), *Dynamic Lot Size Models for Multi–Stage Assembly Systems.* **Management Science** 20, 14–21.

Crowston, W.B., Wagner, M.H., and Henshaw, A. (1972), *A Comparison of Exact and Heuristic Routines for Lot Size Determination in Multi–Stage Assembly Systems.* **AIIE Transactions** 4, 313–317.

Crowston, W.B., Wagner, M.H., and Williams, J.F. (1973), *Economic Lot Size Determination in Multi–Stage Assembly Systems.* **Management Science** 19, 517–527.

Dvoretzky, A., Kiefer, J., and Wolfowitz, J. (1952), *The Inventory Problem: I. Case of Known Distributions of Demand; II. Case of Unknown Distribution of Demand.* **Econometrica** XX, 187–222; 450–466.

Dvoretzky, A., Kiefer, J., and Wolfowitz, J. (1953), *On the Optimal Character of the (s,S) Policy in Inventory Theory.* **Econometrica** XXI, 586–596.

Federgruen, A., Queyranne, M., and Zheng, Y.S. (1989), *Simple Power of Two Policies are Close to Optimal in a General Class of Production/Distribution Networks with General Joint Setup Costs.* Working Paper #89–MSC–012, Faculty of Commerce, University of British Columbia.

Graves, S.C. (1981), *Multi–Stage Lot Sizing: An Iterative Procedure.* **Multi–Level Production/Inventory Control Systems: Theory and Practice** (L.B. Schwarz, ed.), North Holland, New York, New York, Chapter 4.

Graves, S.C. and Schwarz, L.B. (1977), *Single Cycle Continuous Review Policies for Arborescent Production/Inventory Systems.* **Management Science** 23, 529–540.

Hadley, G. and Whitin, T.M. (1963), **Analysis of Inventory Systems**, Prentice–Hall Inc., Englewood Cliffs, New Jersey.

Harris, F. (1915), **Operations and Cost**. (Factory Management Series). A.W. Shaw Co., Chicago, pp. 48–52.

Jackson, P.L. and Roundy, R.O. (1985), *Constructive Algorithms for Planning Production in Multi–Stage Systems with Stationary Demand.* Technical Report 632, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York.

Jackson, P.L., Maxwell, W.L., and Muckstadt, J.A. (1985), *The Joint Replenishment Problem with Powers of Two Restriction.* **AIIE Transactions** 17, 25–32.

Jackson, P.L., Maxwell, W.L., and Muckstadt, J.A. (1988), *Determining Optimal Reorder Intervals in Capacitated Production–Distribution Systems.* (Forthcoming in **Management Science**).

Jensen, P.A. and Khan, H.A. (1972), *Scheduling in a Multi–Stage Production System with Setup and Inventory Costs.* **AIEE Transactions** 4, 126–133.

Kalymon, B.A. (1972), *A Decomposition Algorithm for Arborescence Inventory Systems.* **Operations Research** 20, 860–874.

Karimi, I.A. (1987), *Optimal Cycle Times in a Two–Stage Serial System with Setup and Inventory Costs.* Technical Report, Department of Chemical Engineering, Northwestern University, Evanston, Illinois.

Lambrecht, M. and Vander Eecken, J. (1978), *A Facilities in Series Capacity Constrained Dynamic Lot–Size Model.* **European Journal of Operations Research** 2, 42–49.

Lawler, E.L. (1978), *Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints.* **Annals of Discrete Mathematics** 2, 75–90.

Lee, H.L. and Rosenblatt, M.J. (1987), *Simultaneous Determination of Production Cycle and Inspection Schedules in a Production System.* **Management Science** 33 (9), 1125–1137.

Love, S.F. (1972), *A Facilities in Series Inventory Model with Nested Schedules.* **Management Science** 18, 327–338.

Maxwell, W.L. and Muckstadt, J.A. (1981), *A Model for Planning Production in an N Stage System.* Technical Report 508, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York.

Maxwell, W.L. and Muckstadt, J.A. (1985), *Establishing Consistent and Realistic Reorder Intervals in Production–Distribution Systems.* **Operations Research** 33 (6), 1316–1341.

McLaren, B.J. (1976), *A Study of Multiple Level Lot Sizing Techniques for Material Requirements Planning Systems.* Unpublished Ph.D. Dissertation, Purdue University, West Lafayette, Indiana.

Muckstadt, J.A. (1985), *Planning Component Delivery Intervals in Constrained Assembly Systems.* **Multi–Stage Production Planning and Inventory Control** (S. Axsäter, et al., eds.), Springer–Verlag.

Muckstadt, J.A. (1988), *Establishing Reorder Intervals and Inspection Policies when Production and Inspection Processes are Unreliable.* Technical Report 774, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York.

Muckstadt, J.A. and Roundy, R.O. (1987), *Multi–Item, One–Warehouse, Multi–Retailer Distribution Systems.* **Management Science** 33 (12), 1613–1621.

Muckstadt, J.A. and Singer, H.M. (1978), *Comments on Single Cycle Continuous Review Policies for Arborescent Production/Inventory Systems.* **Management Science** 24, 1766–1768.

Porteus, E.L. (1985), *Investing in Reduced Setups in the EOQ Model.* **Management Science** 31, 998–1010.

Porteus, E.L. (1986), *Optimal Lot Sizing, Process Quality Improvement and Setup Cost Reduction.* **Operations Research** 34 (1), 137–144.

Porteus, E.L. (1987), *Setup Reduction and Increased Effective Capacity.* **Management Science** 33 (10), 1291–1301.

Queyranne, M. (1985), *A Polynomial–Time Submodular Extension to Roundy's 98% Effective Heuristic for Production/Inventory Systems.* Working Paper 1136, University of British Columbia, Vancouver, British Columbia, Canada V6T 1W5.

Queyranne, M. (1987), *Finding 94%–Effective Policies in Linear Time for Some Production/Inventory Systems.* Working Paper, University of British Columbia.

Raymond, F.E. (1931), **Quantity and Economy in Manufacture**. McGraw–Hill Book Co., New York.

Roundy, R.O. (1985a), *98% Effective Integer–Ratio Lot–Sizing for One Warehouse Multi–Retailer Systems.* **Management Science** 31 (11), 1416–1430.

Roundy, R.O. (1985b), *94%–Effective Lot–Sizing in Multi–Stage Assembly Systems.* Technical Report 674, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York.

Roundy, R.O. (1986), *A 98% Effective Lot–Sizing Rule for a Multi–Product, Multi–Stage Production Inventory Systems.* **Mathematics of Operations Research** 11, 699–727.

Roundy, R.O. (1987), *Computing Nested Reorder Intervals for Multi–Item Distribution Systems.* Technical Report 751, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York.

Scarf, H., Gilford, D., and Shelly, M. (1963), **Multistage Inventory Models and Techniques**, Stanford University Press, Stanford, California.

Schwarz, L.B. (1973), *A Simple Continuous Review Deterministic One–Warehouse N–Retailer Inventory Problem.* **Management Science** 19, 555–566.

Schwarz, L.B. and Schrage, L. (1975), *Optimal and System–Myopic Policies for Multi–Echelon Production/Inventory Assembly Systems.* **Management Science** 21, 1285–1294.

Silver, E.A. and Peterson, R. (1985), **Decision Systems for Inventory Management and Production Planning**, 2nd Edition, John Wiley and Sons, New York.

Szendrovits, A.Z. (1975), *Manufacturing Cycle Time Determination for a Multi–Stage Economic Production Quantity Model.* **Management Science** 22 (3), 298–308.

Thompson, W.A. (1962), *The Problem of Negative Estimates of Variance Components.* **Annals of Mathematical Statistics** 33, 273–289.

Veinott, A.F., Jr. (1966), *The Status Of Mathematical Inventory Theory.* **Management Science** 12 (11), 745–777.

Veinott, A.F., Jr. (1969), *Minimum Concave Cost Solution of Leontief Substitution Models of Multi–Facility Inventory Systems.* **Operations Research** 17, 262–291.

Whitin, T.M. (1953), **The Theory of Inventory Management**. Princeton University Press, Princeton, NJ.

Williams, J.F. (1981), *Heuristic Techniques for Simultaneous Scheduling of Production and Distribution in Multi–Echelon Structures: Theory and Empirical Comparisons.* **Management Science** 27, 336–352.

Williams, J.F. (1982), *On the Optimality of Integer Lot Size Ratios in Economic Lot Size Determination in Multi–Stage Assembly Systems.* **Management Science** 28, 1341–1349.

Williams, J.F. (1983), *A Hybrid Algorithm for Simultaneous Scheduling of Production and Distribution in Multi–Echelon Structures.* **Management Science** 29, 77–105.

Zangwill, W.I. (1966), *A Deterministic Multi–Product Multi–Facility Production and Inventory Model.* **Operations Research** 14, 486–507.

Zangwill, W.I. (1969), *A Backlogging Model and a Multi–Echelon Model of a Dynamic Economic Lot Size Production System – A Network Approach.* **Management Science** 15, 506–527.

Zheng, Y.–S. (1987), *Replenishment Strategies for Production/Distribution Networks with General Joint Setup Costs.* Ph.D. Thesis, Columbia University, New York, New York.