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

TECHNICAL REPORT NO. 642

December 1984

**A 98%-EFFECTIVE LOT-SIZING RULE FOR A
MULTI-PRODUCT, MULTI-STAGE
PRODUCTION/INVENTORY SYSTEM**

by

Robin Roundy

This research was partially supported by National Science Foundation Grants
ECS 84-04641 and ECS 80-22027; and also by the General Motors Corporation.

A 98%-EFFECTIVE LOT-SIZING RULE FOR A MULTI-PRODUCT, MULTI-STAGE PRODUCTION/INVENTORY SYSTEM

by Robin Roundy

Abstract

We study a multi-product, multi-stage production/inventory system in continuous time. In-process and finished goods are all referred to as products and inventories of a single product held at different locations are treated as different products. External demand can occur for any or all of the products at a constant, product-dependent rate. There is no circuit of products each of which consumes its predecessor.

When an order is placed for a product it is instantly delivered (production and transportation are instantaneous) and the required amounts of all products consumed in producing the given product are simultaneously withdrawn from their respective inventories. There is a set-up cost of placing an order that is a function of the set of products being ordered at that point in time. There is also a holding-cost rate per unit stored per unit time for each product.

The goal is to schedule orders for each of the products over an infinite horizon so as to minimize the long-run average cost. Although there is an extensive literature on problems of this type, only a few cases have been solved. For an infinite horizon the single-product series system in discrete time and the single-product two-facility series system in continuous time have been solved. For a finite horizon there are methods for solving arborescence distribution

and assembly systems but they have exponential running times. Polynomial-time algorithms exist only for series systems.

The **effectiveness** of a policy is defined to be 100% times the ratio of the infimum of the average cost over all policies to the average cost of the policy in question. The effectiveness of an optimal policy is, of course, 100%. None of the many existing heuristics has been shown to have a worst-case effectiveness greater than zero. Most of these heuristics assume both stationarity and nestedness. The worst-case effectiveness of an optimal nested policy is zero.

We define a new class of policies in which each product uses a stationary interval of time between successive orders, and the ratio of the order intervals of any two products is an integer power of two. An optimal policy within this class is shown to be 98%-effective in the worst case. This extends a similar result for the one-warehouse, multi-retailer system. For assembly systems we can compute such a policy in $O(N \log N)$ time. For more general systems, the major computational task in computing this policy is solving a nonlinear network-flow problem. This can be done efficiently for very large systems by solving a sequence of max-flow, min-cut problems.

98%-EFFECTIVE POWER-OF-TWO LOT-SIZING FOR MULTI-PRODUCT MULTI-STAGE PRODUCTION/INVENTORY SYSTEMS

by Robin Roundy

1. INTRODUCTION.

We study a multi-product, multi-stage production/inventory system in continuous time. In-process and finished goods are all referred to as products, and inventories of a single product held at different locations are treated as different products. External demand can occur for each of the products at a constant, product-dependent rate. These demands must be met as they occur over an infinite horizon without shortages or backlogging.

It is assumed that there is no circuit of products, each of which is consumed in producing its successor. When an order is placed for a product, it is instantly delivered (production and transportation are instantaneous) and the required amounts of all other products consumed in producing the given product are simultaneously withdrawn from their respective inventories. A generalization of the problem that has non-zero lead times is discussed in Section 9.

Setup costs on the placement of orders are associated with families of products. A **family** is any subset of the set of products, and a product can belong to any number of different families. The nonnegative setup cost associated with a family is incurred whenever one or more of the products in the family is ordered. For example in

a grocery store the set of families might include the set of all products, the set of all meat products, the set of all fresh produce, and the the singleton sets consisting of one product each. In a manufacturing environment the set of products made on a certain machine, or which use a certain tool, may constitute a family.

In addition to the setup costs, there is a holding-cost rate per unit stored per unit time for each product. The goal is to schedule orders for each of the products over an infinite horizon so as to minimize the long-run average cost.

Complexity of Optimal Policies.

Although there is an extensive literature on problems of this type, only two cases have been solved. They are the series system in discrete time [Lo72] and the two-facility series system in continuous time [Sc73]. Even if optimal policies for more general systems could be computed efficiently, they are too complex to be attractive. They typically call for both the order quantities and the intervals of time between successive orders of each product to vary over time [Ro84].

Effectiveness.

Since optimal policies are difficult to find and are usually very complex, we are led to seek policies with high guaranteed effectiveness. The **effectiveness** of a policy is 100% times the ratio of the infimum of the average cost over all policies to the average cost of the policy in question.

Lot-Sizing Rules with High Worst-Case Effectiveness.

In the sequel we develop two lot-sizing rules for this problem that have worst-case effectiveness 94% and 98%, respectively. The lot-sizing rules produce policies with a special structure that is easy to implement. Orders are placed at equal intervals for each of the products in the system, and for every pair of products the ratio of the lengths of time between successive orders is an integer power of two. This extends our earlier result for one-warehouse multi-retailer systems [Ro84].

Computations Required.

Let Π be the number of products in the system. For one-warehouse multi-retailer systems, a 98%-effective policy can be found in $O(\Pi \log \Pi)$ time [Ro84]. We will show herein that this can also be done for assembly systems. For more complex systems, the major computational task to be performed in implementing our lot-sizing rules is solving a nonlinear network-flow problem. Using an algorithm due to Maxwell and Muckstadt, this can be done very efficiently by solving a number of max-flow, min-cut problems [MM83]. The number of such problems that must be solved is bounded by, and is usually much smaller than, the number of products in the system. The size of the network on which the computations are done depends on the configuration of the system. While there can be as many as $O(2^\Pi)$ nodes in the network, for realistic configurations both the number of nodes and the number of arcs are small multiples of the product of Π and the number of end products in the system, where an **end product** is a product with positive external demand.

Overview.

In the remainder of this section we survey other methods that have been used to solve this and related problems. In Section 2 we define the class of power-of-two policies, which is used by our lot-sizing rules. In Section 3 we introduce the product network, a network which represents the flow of goods through the system. We compute the average cost of a power-of-two policy in Section 4. In Section 5 we relax the constraint that the ratios of the order intervals be powers of two. We then formulate the problem of minimizing the relaxed average cost of a power-of-two policy as a nonlinear network-flow problem. This network-flow problem is solved in Section 6. In Section 7 we prove that the solution to the network-flow problem is in fact a lower bound on the average cost of any feasible schedule over any finite horizon. In Section 8 we describe how 94%-effective and 98%-effective power-of-two policies are found once the network-flow problem has been solved. In Section 9 we discuss lead times and other extensions and limitations of the model. In Section 10 we discuss other problems that are equivalent to minimizing the relaxed average cost of a power-of-two policy. Several algorithms developed for these problems can be used to solve this one, including an $O(\Pi \log \Pi)$ algorithm for assembly systems. We then draw our conclusions in Section 11.

Dynamic-Programming Algorithms for Optimizing Discrete-Time Systems.

The first dynamic-programming algorithms for non-stationary, discrete-time, finite-horizon, multi-stage economic-lot-size problems

were developed by Zangwill [Za66,68]. Dynamic-programming algorithms exist for series systems [Za69,Lo72], arborescence distribution systems [Ve69,Ka70,EMV80], and assembly systems [CW73]. Unfortunately, only the algorithms for series systems have polynomial running times. For the other systems, it is unlikely that problems with more than about 15 products and 15 time periods could be solved efficiently.

Dynamic-Programming Algorithms for Suboptimization in Discrete Time.

A **nested** policy is one in which each facility orders every time any of its suppliers orders. Veinott [Ve69] developed a polynomial-time dynamic-programming method for computing an optimal nested policy for an arborescence distribution system. Nested policies often perform very well, but they can also perform very poorly.

Lot-Sizing Rules for Continuous-Time, Infinite-Horizon Systems.

Numerous researchers have proposed lot-sizing rules for stationary, continuous-time, infinite-horizon multi-stage production/inventory systems [TS70, CWH72, JK72, CWW73, Sc73, SS75, GS77, Wi81, JMM83, MM83, Wi83, Ro84]. These authors deal with many different types of systems, but they all make similar assumptions about the types of policies they allow.

A **stationary** policy is one in which each facility uses a fixed order quantity and a fixed interval of time between successive orders. All of the above papers restrict attention to policies which are both stationary and nested, called **stationary nested**. Unfortunately, the worst-case effectiveness of an optimal nested policy is zero [Ro84].

The work of Maxwell and Muckstadt is especially relevant [MM83]. Retaining the assumptions of stationarity and nestedness, they developed a lot-sizing rule which is similar in spirit to ours and applies to multi-stage, multi-product systems in which every family consists of a single product. Their theorem stating that their lot-sizing rule is always within 6% of the average cost of an optimal nested policy was motivated by this work; however the Network-Partitioning Algorithm that we use is due to them. The work in [JMM83] on the joint-replenishment problem is also closely related, and the result on worst-case effectiveness given therein was motivated by our corresponding result for one-warehouse multi-retailer systems [Ro84].

2. POWER-OF-TWO POLICIES.

In this section we introduce the class of power-of-two policies. Power-of-two policies combine the advantages of computational tractability, high effectiveness, and easy implementation.

Definition of a Power-of-Two Policy.

A **power-of-two policy** is a sequence $T = (T_n : n \in N)$ of positive numbers with the following three properties. First, orders for product n are placed once every $T_n > 0$ units of time beginning at time zero. Second,

$$(1) \quad T_n = 2^{k_n \beta} \text{ for all products } n \text{ and for some } 1 \leq \beta < 2$$

where k_n is an integer. Finally, the **Zero-Inventory Property** holds, i.e., an order is placed for a product only when the inventory of that product is zero [Ve69, Sc73].

Policies in which the order intervals are restricted to powers-of-two times some time period β have been used for the economic-lot-scheduling problem [FH73, E178, MS81] and for systems with dynamic demand and/or with multiple products [CM81, HLV82, MM83]. The class of power-of-two policies used herein differs from past approaches primarily in that the assumptions of stationarity and nestedness are relaxed.

Comparison of Power-of-Two Policies and Stationary Nested Policies.

A power-of-two policy is stationary nested if and only if $T_m/T_n \geq 1$ whenever product m is consumed in producing product n . If this condition fails, the policy is neither stationary nor nested, and the order quantities of some of the products will be periodic. To see this, consider a warehouse ($n = 0$) that supplies two retailers ($n = 1$ and $n = 2$). The order intervals are $T_0 = 1$, $T_1 = 1/2$,

Figure 1. Timing of Orders in a Simple Power-of-Two Policy.

and $T_2 = 2$, and the demand rates for the retailers are both one. By the Zero-Inventory Property the order quantities at the warehouse are 3, 1, 3, 1,

3. THE PRODUCT NETWORK.

Let N be the set of products in the system. For each product $n \in N$ let π_n be the set of direct **predecessors** of product n , i.e., the set of products consumed in producing product n . We define the **product network** to be the network with node set N and arc set $\{(m,n) : m \in \pi_n\}$.

For example, consider a manufacturer of two-piece pajamas. The products in our simplified system are material (product 1), tops (product 2), pants (product 3), and packaged pajamas (product 4). Material is used in both the tops and the pants. The product network for this system is shown in Figure 2. In this example $\pi_4 = \{2,3\}$ and $\pi_2 = \pi_3 = \{1\}$.

A second example illustrates that the product network need not be connected. Consider a table manufacturer who produces both a large table, that can be expanded by inserting leaves, and a small table. The products in the system are small table tops (product 1), finished small tables (product 2), end pieces to large table tops (product 3),

Figure 2. The Product Networks for Pajamas and Tables.

The Pajama Manufacturer.

The Table Manufacturer.

leaves for large tables (product 4), and finished large tables (product 5).

The product network of the table manufacturer is also shown in Figure 2. In this case $\pi_2 = \{1\}$ and $\pi_5 = \{3, 4\}$. The two components of the table manufacturer's product network will subsequently be linked together by setup costs.

It is assumed that the product network is circuitless, i.e., there is no circuit of products each of which consumes its predecessor. However undirected cycles may exist. The product network of the pajama manufacturer contains such a cycle.

Routes and Preroutes in the Product Network.

Much of the subsequent analysis is based on routes and preroutes in the product network. A **preroute** is a sequence of products $r = \langle n_1, n_2, \dots, n_I \rangle$ such that n_{i-1} is a predecessor of n_i for all i . We say that a unit of product n_1 **follows** a preroute r if it is processed into higher-level products in the sequence $n_1, n_2, \dots, n_I$. If $m \in \pi_n$, let λ_{mn} be the amount of product m consumed in producing a unit of product n . Then $\lambda_r \equiv \prod_{i=1}^{I-1} \lambda_{n_i n_{i+1}}$ is the amount of product n_1 that must follow r in order to produce one unit of product n_I .

A **route** is a preroute that ends at an end product. We say that a unit of product n_1 **flows along** a route $r = \langle n_1, n_2, \dots, n_I \rangle$ if it follows the route and is then used to satisfy the external demand for n_I . We say that r is a **superroute** of route q if $r = (p, q)$ for some (possibly vacuous) preroute p , and that q is a

subroute of r if r is a superroute of q . The superroute-of relation partially orders the routes. Each arc in the Hasse diagram of this partial order joins a route r to its **successor** formed by

Figure 3. Hasse Diagram of Routes for Pajama and Table Manufacturers.

The Pajama Manufacturer.

The Table Manufacturer.

removing the first product in r therefrom. The Hasse diagrams derived from the product networks of Figure 2 are shown in Figure 3.

4. THE AVERAGE COST OF A POWER-OF-TWO POLICY.

In this section we derive an expression for the average cost of a power-of-two policy. Using echelon inventories and incremental holding cost rates, the average holding cost of a stationary nested policy can be expressed as a linear function of the order intervals of the products [MM83]. This is not true of non-nested power-of-two policies. We will show that the average holding cost of a power-of-two policy is instead a linear function of the order intervals of the routes where the order interval of route r is $\max_{n \in r} T_n$, the maximum of the order intervals of the products in the route.

Demand and Holding-Cost Rates.

The **demand rate** and the **holding-cost rate** per unit time for product n are $d_n \geq 0$ and $h'_n > 0$, respectively. The **incremental holding cost rate** for product n is $h_n \equiv h'_n - \sum_{m \in \pi_n} \lambda_{mn} h'_m$. We assume that h_n is positive for each product n . This assumption is both a common and a natural one. Let P_{mn} (resp., R_{mn}) be the set of preroutes (resp., routes) that begin at product m and end at product n . It is easily verified that $h'_n = \sum_m \sum_{p \in P_{mn}} \lambda_p h'_m$.

Inventories on Routes.

The length of time that a unit of product m will be in the system depends on the route along which the unit will flow. For this reason we will find it helpful to distinguish inventories by route rather than by product. We assign a unit of product m to the inventory on route $r \in R_{mn}$ if it will flow along r . In this way each unit of the inventory of m is assigned to a unique route r that starts at m . Note that when an item leaves the inventory on route r , it enters the inventory on the successor of r .

It is convenient to choose the unit of measure for the inventory on $r \in R_{mn}$ to be the amount of product m that must flow along r to satisfy the external demand for product n for one-half of a unit of time. This is $.5\lambda_r d_n$ units of product m . The holding cost rate per unit time for inventory on $r \in R_{mn}$ is $H'_r \equiv .5\lambda_r d_n h'_m$. Define the **incremental holding cost rate** H_r of route r by $H_r \equiv .5\lambda_r d_n h_m$. It is easily verified that the holding cost rate H'_r at route r is the sum of the incremental holding cost rates H_q over all superroutes q of r .

The Echelon Method of Computing Holding Cost Rates.

Evidently the rate at which holding costs are accumulating at any point in time can be found by multiplying the holding cost rate H_r^I at route r and the physical inventory on r , and summing over all routes. It is convenient to express this in terms of echelon inventories. The **echelon inventory** E_r^t on route r at time t is the sum of the inventories on all subroutes of r . Clearly the rate at which holding costs are accumulating at time t is

$$(2) \quad \sum_r H_r E_r^t.$$

Note that in deriving (2) we make no assumptions about the demand process or the type of policy being used. Also note that if $r \in R_{mn}$, then E_r^t is twice the length of time during which the echelon inventory on r is sufficient to satisfy the external demand for product n without replenishment of product m .

The Average Holding Cost of a Power-of-Two Policy.

To find the average holding cost incurred by the power-of-two policy T , we need to find the average value of the echelon inventory E_r^t on route r . In order to gain insight into the way goods flow along a route, we refer to the route $r = \langle 1, 2, 4 \rangle$ of the pajama manufacturer. Let $T_1 = 8$, $T_2 = 1$ and $T_4 = 2$. As is illustrated in Figure 4, product two is ordered simultaneously with every order for product four. By the Zero-Inventory Property, units of product two that will flow along $\langle 2, 4 \rangle$ are ordered only once every $T_2 \vee T_4 = 2$

Figure 4. Inventories Along a Route of the Pajama Manufacturer.

The Timing of Orders for Each Product in the Route $r = \langle 1, 2, 4 \rangle$.

The Inventory on $\langle 1, 2, 4 \rangle$.

The Inventory on $\langle 4 \rangle$.

The Echelon Inventory on $\langle 1, 2, 4 \rangle$.

units of time. Similarly, units of product one that will flow along $\langle 1,2,4 \rangle$ are ordered only once every $T_1 \vee (T_2 \vee T_4) = 8$ units of time. Because product two is ordered every time product four is, the inventory on $\langle 2,4 \rangle$ is zero.

Let $r \in R_{mn}$. The echelon inventory on r increases only when units of product m are ordered that will flow along r , and we have seen that this happens once every $\max_{n \in r} T_n$ units of time starting at time zero. At these points in time an order is simultaneously placed for every product in route r , so by the Zero-Inventory Property the inventory of each product in r is zero before ordering. Therefore the echelon inventory on r is also zero before ordering at these times. This implies that the echelon inventory E_r^t on r follows the familiar saw-toothed inventory pattern with period $\max_{n \in r} T_n$. Because of our choice of units, E_r^t decreases at a constant rate of two per unit time. Therefore the average value of E_r^t is $\max_{n \in r} T_n$. By (2) the average holding cost per unit time of the power-of-two policy T is $\sum_r H_r \max_{n \in r} T_n$.

Let R be the union of the set of all routes and the set of all products. If n is not an end product, define $H_{\langle n \rangle} \equiv 0$. Then the average holding cost can be written as $\sum_{r \in R} H_r \max_{n \in r} T_n$. The reason for including all of N in R will become apparent later on.

The Average Setup Cost.

The average setup cost incurred by the power-of-two policy $T = (T_n : n \in N)$ is computed as follows. Associated with every family of products $f \subseteq N$ is a nonnegative setup cost K_f . This cost is incurred at every point in time at which any of the products in f is

ordered. If $T_m \leq T_n$, then product m is ordered every time product n is. Therefore the frequency with which the setup cost K_f is incurred per unit time is $1/(\min_{n \in f} T_n)$. If $F \equiv N \cup \{f \subseteq N : K_f > 0\}$, then the average setup cost per unit time incurred by $\mathbf{T}$ is $\sum_{f \in F} K_f / (\min_{n \in f} T_n)$. The reason we include products $n \in N$ for which $K_n = 0$ in F will become apparent later on.

The Average Cost of a Power-of-Two Policy.

The average cost of the power-of-two policy $\mathbf{T}$ is therefore

$$(3) \quad c(\mathbf{T}) = \sum_{f \in F} K_f / (\min_{n \in f} T_n) + \sum_{r \in R} H_r \max_{n \in r} T_n.$$

Since K_f and H_r are nonnegative, $c(\mathbf{T})$ is a sum of maximums of convex functions and so is a convex function of $\mathbf{T} \geq 0$.

5. A NETWORK-FLOW PROBLEM FOR MINIMIZING THE RELAXED AVERAGE COST.

We now relax the power-of-two constraint (1) and minimize $c(\mathbf{T})$ over all vectors $\mathbf{T} \geq 0$. We are interested in minimizing the relaxed average cost $c(\mathbf{T})$ subject to $\mathbf{T} \geq 0$ for two reasons. The first reason is that the minimum relaxed average cost is subsequently proven to be a lower bound on the long-run average cost of any feasible schedule. This result makes possible the error bounds given herein. Second, the value of $\mathbf{T}$ that minimizes the relaxed average cost will be used to find effective power-of-two policies.

The problem of minimizing the relaxed average cost of a power-of-two policy is formulated as the dual of a nonlinear network-flow problem on what we will call the cost network. The nodes of the cost

network correspond to the routes in R and the families in F . A flow in the cost network can be interpreted as a way of distributing the holding cost rate H_r (resp., the setup cost K_f) among the individual products in route r (resp., family f). We will use such an allocation to split the cost of an arbitrary policy into the sum of the costs of feasible policies for Π single-item problems, one for each product. This allows a lower bound on the cost of an optimal schedule to be established.

In this section we define the cost network, formulate the problem of minimizing the relaxed average cost as a dual network-flow problem, and find the primal network-flow problem.

The Cost Network.

The node set of the cost network is $J \equiv R \cup F$. Note that $R \cap F = \emptyset$. We will use the letters j , i , and occasionally k to denote general nodes in J . When we wish to emphasize that a node belongs to N (resp., R , F) we denote the node by n (resp., r , f).

We define a partial order on J as follows. If $i, j \in J$, we say that $i \geq j$ if either $i, j \in R$ and i is a superroute of j , or if $i, j \in F$ and $i \subseteq j$, or if $i \in R$ and $j \in F$ and $i \cap j \neq \emptyset$. The arcs A of the cost network are the arcs in the Hasse diagram corresponding to this partial order. The cost network has the following properties. First, there are exactly two arcs leading out of each node $r \in R_{mn}$ where $m \neq n$, one to m and one to the route found by removing m from r . Second, the cost network is circuitless, and arcs lead from routes to shorter routes or individual products, from individual products to families, and from families to

larger families. Finally, there is a directed path in the cost network from $r \in R \setminus N$ to n (resp., from n to $f \in F \setminus N$) if and only if $n \in r$ (resp., $n \in f$) .

We assume that every product belongs to a family with positive setup cost and to a route whose terminus is an end product. Therefore every node in the cost network has a predecessor $r \in R$ for which $H_r > 0$ and a successor $f \in F$ for which $K_f > 0$.

As an example consider the systems shown in Figure 2. The cost networks of these systems are shown in Figure 5. The routes in R are denoted by the string of digits corresponding to the products in the route. The families in F containing two or more products are denoted by the underlined string of digits corresponding to the products in the family. The pajama manufacturer has a major setup cost associated with the family $\{2,3\}$. The table manufacturer has major setup costs associated with the wood shop (family $\{1,3,4\}$) and with finishing operations (family $\{2,5\}$) .

Figure 5. The Cost Networks for the Pajama and Table Manufacturers.

The Pajama Manufacturer

The Table Manufacturer

Reducing the Size of the Cost Network.

If a node $n \in N$ has a unique predecessor in the cost network and also satisfies $H_n = 0$, meaning that n is not an end product, then we can collapse node n into its predecessor. For the pajama manufacturer this condition holds for products 2 and 3, and for the table manufacturer this condition holds for products 1, 3, and 4. In assembly systems this condition is satisfied for all products in the system except the end product.

The Dual Network-Flow Problem.

We now formulate the problem of minimizing the relaxed average cost of a power-of-two policy as a dual network-flow problem. Let $T \geq 0$ and let $\mathbf{t} = (t_j : j \in J)$ satisfy $t_n = T_n$ for all $n \in N$, $t_r = \max_{n \in r} T_n$ for all $r \in R \setminus N$, and $t_f = \min_{n \in f} T_n$ for all $f \in F \setminus N$. Then $c(T) = g(\mathbf{t})$ where

$$(4) \quad g(\mathbf{t}) \equiv \sum_{f \in F} K_f / t_f + \sum_{r \in R} H_r t_r .$$

Furthermore

$$(5) \quad t_j \geq 0 \text{ for all } j \in J \text{ and } t_i \geq t_j \text{ for all } i \rightarrow j \in A$$

where $i \rightarrow j$ is the arc from i to j in the cost network. The dual network-flow problem D is to minimize (4) subject to (5).

Clearly the dual network-flow problem is equivalent to minimizing the relaxed average cost $c(T)$ of a power-of-two policy subject to $T \geq 0$. Note that in an optimal solution to the dual network-flow problem we have

$$(6) \quad t_r = \max_{n \in r} t_n \quad \text{if } r \in R \quad \text{and} \quad t_f = \min_{n \in f} t_n \quad \text{if } f \in F .$$

The Primal Network-Flow Problem.

The conjugate dual P of D is to maximize

$$(7) \quad f(\mathbf{v}) \equiv \sum_j 2(K_j v_j)^{.5}$$

subject to:

$$(8) \quad \sum_{k \in s_j} x_{jk} - \sum_{i \in p_j} x_{ij} + v_j = H_j \quad \text{for all } j \in J ,$$

$$\mathbf{x} \geq 0 , \quad \mathbf{v} \geq 0$$

where p_j (resp., s_j) is the set of direct predecessors (resp., successors) of node j in the cost network, $\mathbf{x} = (x_{ij})$, and $\mathbf{v} = (v_j)$.

The derivation of the primal P from the dual D is outlined in Appendix A.

6. SOLVING THE NETWORK-FLOW PROBLEM.

In this section we show how to solve efficiently the primal network-flow problem P . The solution to P is subsequently proven to be a lower bound on the long-run average cost of any feasible schedule. An optimal flow for P will be used to establish this fact, and the value of t that solves the dual D will be used to find effective power-of-two policies.

The method we present is for systems with general structure. For one-warehouse multi-retailer systems and assembly systems, there are more efficient algorithms that solve P and have $O(\Pi \log \Pi)$ running

time. An algorithm for one-warehouse, multi-retailer systems was developed by Roundy [Ro84]. An algorithm for assembly systems was developed by Thompson [Th62] and, in a somewhat different and a more general form, by Lawler [La78]. For a description of how the problems that motivated these last two authors relate to P , see Section 10.

The dual D of P is essentially the same problem that Maxwell and Muckstadt [MM83] solved in order to minimize the relaxed average cost of a stationary nested policy. Because they use stationary nested policies, their computations are done on the original product network rather than on the cost network. Furthermore, in their model all families consist of a single product. Consequently, they have $K_j, H_j > 0$ for all j . However their algorithm can be used to solve P . It is based on the following observation.

In (8) we can interpret $v_j - H_j$ as the demand at node j . The objective function of P depends only on these demands, not on the flows $\mathbf{x}$ in the cost network. If $S \subseteq J$ we define v_S (resp., H_S, K_S) to be $\sum_{j \in S} v_j$ (resp., $\sum_{j \in S} H_j, \sum_{j \in S} K_j$). A necessary condition on the demands is that they sum to zero, i.e.,

$$(9) \quad v_J = H_J$$

Overview.

Maxwell and Muckstadt's algorithm can be outlined as follows. First minimize $f(\mathbf{v})$ subject to (9). This generates demands at the nodes of the cost network. Then use the max-flow min-cut algorithm to search for a feasible flow $\mathbf{x}$ in the cost network that is consistent

with these demands. If a feasible flow is found, $\mathbf{x}$ and $\mathbf{v}$ must solve P . If not, a maximum-capacity cut is used to split P into two independent subproblems of the same type. This procedure is then applied to each of the subproblems.

Generating Demands at the Nodes in the Cost Network.

It is easily verified that the value of $\mathbf{v}$ that minimizes $f(\mathbf{v})$ subject to (9) is $\bar{\mathbf{v}} \equiv (K_j/\tau, j \in J)$ where $\tau \equiv K_J/H_J$.

Splitting the Cost Network in Two.

If there is a nonnegative flow $\mathbf{x}$ that satisfies (8) with $\mathbf{v} = \bar{\mathbf{v}}$, then $\mathbf{x}$ and $\mathbf{v}$ solve P . If not, there is a maximum-capacity cut (U, V) . Without loss of generality we assume that $v_U - H_U > 0$ and $v_V - H_V < 0$. Since the flows in all of the arcs have zero as a lower bound and have infinite upper bounds, there are no arcs from V to U , and the capacity of the cut is $(v_U - H_U) - (v_V - H_V) = 2(K_U/\tau - H_U) = 2(H_V - K_V/\tau) > 0$.

Maxwell and Muckstadt observed that for all arcs $i \rightarrow j$ from U to V , the constraint $t_i \geq t_j$ is redundant in an optimal solution to D . Therefore $x_{ij} = 0$ in an optimal solution to P . Thus we will not change the optimal solution to P if we delete all such arcs from the cost network. This splits the cost network into two disconnected subnetworks whose node sets are respectively U and V . When we remove the variables x_{ij} , $i \in U$, $j \in V$ from (8), P splits into two independent subproblems on these two subnetworks. These subproblems have the same form as the original problem. Consequently we can apply the same procedure to the subproblems that we have just applied to P . This gives rise to the following algorithm.

The Network-Partitioning Algorithm.

Step 1: Search for a Feasible Flow.

Set the demand at node j of the cost network equal to $\bar{v}_j - H_j$ where $\bar{v}_j \equiv K_j/\tau$. Find either a feasible flow $\mathbf{x}$ or a maximum-capacity cutset (U, V) . If there is a feasible flow, $\mathbf{x}$ and $\mathbf{v} = \bar{\mathbf{v}}$ solve P . Otherwise, go to Step 2.

Step 2: Split the Network in Two.

Set $x_{ij} = 0$ for all $i \rightarrow j$ with $i \in U$ and $j \in V$. Then repeat Step 1 twice, first using the subnetwork whose node set is U in place of the cost network and then using the subnetwork whose node set is V .

Justification.

The network-partitioning algorithm splits the cost network into subnetworks called **clusters**. Initially there is only one cluster, but the number L of clusters grows by one during each iteration of the algorithm in which Step 2 is executed. After the algorithm terminates, we index the clusters so that if in some earlier iteration of Step 2 the nodes in cluster k were in U and the nodes in cluster ℓ were in V , then $k < \ell$. Since there are no arcs from V to U , there are no arcs from cluster ℓ to cluster k for any $k < \ell$. Let J^ℓ be the node set of cluster ℓ and let $K^\ell \equiv K_{J^\ell}$, $H^\ell \equiv H_{J^\ell}$ and $t^\ell \equiv (K^\ell/H^\ell)^{.5}$.

Lemma 1. (Positivity, Finiteness, and Monotonicity of the Order Intervals). For all clusters ℓ we have $K^\ell > 0$ and $H^\ell > 0$; and if $k < \ell$, then $t^k \geq t^\ell$.

Proof. The proof is by induction on the iterations of the algorithm. Suppose that we have just completed an iteration of the algorithm and that at each iteration of Step 1 we have always had $H_j > 0$ and $K_j > 0$. We now index the current clusters as described in the previous paragraph. Every time we have executed Step 2 the capacity of the cut has been positive, meaning that $K_U > 0$ and $H_V > 0$. This implies that K^ℓ and H^ℓ cannot both be zero for any of the current clusters, so $t^\ell \in [0, \infty]$ is well-defined. By Theorem 1 in [MM83], $t^k \geq t^\ell$ for all $k < \ell$. But since every node in the cost network has a predecessor in R and a successor in F , $H^1 > 0$ and $K^L > 0$ where L is the highest current cluster index. Therefore $\infty > t^1 \geq t^\ell \geq t^L > 0$, so $K^\ell > 0$ and $H^\ell > 0$ for all ℓ . Consequently we will have $K_j > 0$ and $H_j > 0$ at the next iteration of Step 1. $\square$

Theorem 1. (Justification of the Network-Partitioning Algorithm).

The values of $\mathbf{x}$ and $\mathbf{v}$ computed by the Network-Partitioning Algorithm are optimal for P . An optimal solution to D is given by

$$(10) \quad t_j = t^\ell \equiv (K^\ell/H^\ell)^{.5} \quad \text{for all } j \in J^\ell.$$

Furthermore

$$(11) \quad v_j = K_j/t_j^2 = K_j H^\ell / K^\ell \quad \text{for all } j \in J$$

and

$$(12) \quad B \equiv g(\mathbf{t}) = f(\mathbf{v}) = \sum_{\ell} 2(K^\ell H^\ell)^{.5}.$$

Proof. Clearly the $\mathbf{x}$ and $\mathbf{v}$ produced by the algorithm are feasible for P . By Lemma 1, the $\mathbf{t}$ given by (10) is feasible for

D . Equation (11) follows directly from Step 1 of the algorithm and from (10). Equation (12) follows from (11) and the fact that $K^\ell/t^\ell = H^\ell t^\ell = (K^\ell H^\ell)^{.5}$. The optimality of $\mathbf{x}$, $\mathbf{v}$, and $\mathbf{t}$ now follows from (12) and the theory of conjugate duality [Ro70]. $\square$

The Running Time of the Network-Partitioning Algorithm.

The major part of the computational effort comes from the max-flow min-cut computations in Step 2. Step 2 is executed $L - 1$ times where L is the number of clusters. The following lemma implies that there is a product in every cluster, so $L \leq \Pi$ where Π is the number of products in the network.

Lemma 2. Each cluster contains a node of N .

Proof. By the Network-Partitioning Algorithm there is a feasible flow in cluster ℓ with a demand of $K_j/(t^\ell)^2 - H_j$ at each node j in the cluster. If $j \in R \setminus N$ (resp., $F \setminus N$) , then the demand at j is negative (resp., positive). Let $j \in J^\ell$ have nonpositive demand. Then either $j \in N$ or the demand at j is negative. In the latter case there is a path in cluster ℓ from j to some node that has positive demand. But any such path must contain a node in N . $\square$

A max-flow min-cut computation requires $O(V^3)$ operations where there are V nodes in the cost network [Ka74,MPM78,Hu82]. Therefore the total computational effort is $O(\Pi V^3)$.

The number of nodes in the cost network is $|(F \cup R) \setminus N|$. The largest value $|F|$ could possibly take is $2^\Pi - 1$. Consider a system in which for each $n \in N$, the production of a unit of product n consumes one unit of each product $m < n$. Then the product network

is the complete network on Π nodes where the arc between m and $n > m$ is oriented from m to n . If the external demand for every product is positive, every subset of N is a route in R . In this case $|R \setminus N| = 2^\Pi - \Pi - 1$. Therefore $V = O(2^\Pi)$, and the worst-case running time of the network-partitioning algorithm as a function of the number of products in the system is $O(\Pi 8^\Pi)$.

Fortunately, for real-world production/inventory systems the running time is much smaller. In practice the number of routes is a small multiple of $\Pi\Phi$ where Φ is the number of end products. The reason for this is that in such systems $|R_{mn}| = O(1)$. In assembly and distribution systems, $|R_{mn}|$ is always either zero or one. In complex production systems, it is not uncommon for a product to be used in two distinct intermediate components, neither of which is used in the other and both of which are used in the same end product. This happened in the pajama manufacturer of Figure 2 and shows that $|R_{mn}|$ can be greater than one. However, the number of distinct routes from m to n in the product network is seldom, if ever, large. Furthermore, in multi-product systems there are a great many combinations of m and n for which $|R_{mn}| = 0$, meaning that product m is not used in end product n . Consequently the number of routes can be considered to be $O(\Pi\Phi)$ in practice. As an example, an actual industrial system illustrated in [MM83] has 50 products, 4 end products, and 113 routes, meaning that the average value of $|R_{mn}|$ is 0.565. The solution had only seven clusters.

Since the number of families is usually not much larger than Π , the total number of nodes in the cost network is $O(\Pi\Phi)$. In addition

the number of clusters is usually only a small fraction of Π , so in practice the running time of the algorithm is between $O(\Pi^3 \Phi^3)$ and $O(\Pi^4 \Phi^3)$.

7. THE LOWER-BOUND THEOREM.

In this section we prove that the minimum relaxed average cost computed in Section 6 is a lower bound on the average cost of any feasible policy over any finite horizon. This result enables us to prove our results on worst-case effectiveness. While we use the dual P of D given in Section 5 for computational purposes, there is another closely-related dual of D that is more useful for theoretical purposes.

The Dual P' of D .

Let A_R (resp., A_F) be the set of arcs in the cost network that are incident to a member of $R \setminus N$ (resp., $F \setminus N$). The dual P' of D is to maximize

$$(13) \quad \sum_n 2(K_n^* H_n^*)^{.5}$$

subject to

$$(14) \quad \sum_{g \in s_f} y_{fg} - \sum_{g \in p_f} y_{gf} + K_f = 0 \quad \text{for all } f \in F \setminus N,$$

$$(15) \quad \sum_{q \in p_r} z_{qr} - \sum_{q \in s_r} z_{rq} + H_r = 0 \quad \text{for all } r \in R \setminus N,$$

$$(16) \quad K_n^* = K_n + \sum_{f \in s_n} y_{nf} \quad \text{and} \quad H_n^* = H_n + \sum_{r \in p_n} z_{rn},$$

and

$$\mathbf{y} \geq 0, \quad \mathbf{z} \geq 0$$

where $\mathbf{y} = (y_{fg})$ and $\mathbf{z} = (z_{fg})$. The duality computations are given in Appendix B.

The dual P' of D is a network-flow problem on two disjoint networks that are tied together by a joint flow-cost function. The flow $\mathbf{y}$ is on the network with node set F and arc set A_F . In this network there is a demand K_f at node $f \in F$ and a variable supply K_n^* at node $n \in N$. The flow $\mathbf{z}$ is on the network with node set R and arc set A_R . This network has a supply H_r at node $r \in R$ and a variable demand H_n^* at node $n \in N$. The objective function is a nonseparable function of K_n^* and H_n^* , $n \in N$.

Relationship Between P and P' .

Because v_j in (8) and (11) is not proportional to K_j in (14), there is no direct correspondence between feasible flows for P and feasible flows for P' . However there is a direct correspondence between optimal flows for P and optimal flows for P' . It is given by the following lemma.

Lemma 3. An optimal solution to P' is given by

$$y_{fg} \equiv x_{fg} t_g^2 \quad \text{for all } f \rightarrow g \in A_F \quad \text{and}$$

$$z_{qr} \equiv x_{qr} \quad \text{for all } q \rightarrow r \in A_R$$

where $\mathbf{x}$ solves P and $\mathbf{t}$ solves D . Furthermore

$$\sum_{n \in N^L} K_n^* = K^L, \quad \sum_{n \in N^L} H_n^* = H^L, \quad t_n^2 H_n^* = K_n^*, \quad \text{and}$$

$$\sum_{n \in N^L} 2(K_n^* H_n^*)^{.5} = 2(K^L H^L)^{.5}$$

where $N^L \equiv N \cap J^L$.

The proof is given in Appendix C.

P' as a Cost Allocation Scheme.

The flows $\mathbf{y}$ and $\mathbf{z}$ can be interpreted as a way of allocating the holding cost rates H_r for route r and the setup costs K_f for family f among the individual products in the route or family. This allocation is the key to the proof that the solution to P is a lower bound on the average cost of any feasible schedule. The holding and setup costs incurred by an arbitrary schedule on a finite horizon will be allocated to individual products according to the values of $\mathbf{y}$ and $\mathbf{z}$. We then show that the sum of all costs allocated to product n is the cost of a feasible policy for the single-product lot-size problem on the finite horizon with setup cost K_n^* , holding cost rate H_n^* , and demand rate two. Moreover the sum of the average costs of the optimal solutions to these single-product problems is at least as large as the minimum relaxed average cost B .

The Proof of the Lower-Bound Theorem.

The following lemma is used to prove the Lower-Bound Theorem.

Lemma 4. Let $\mathbf{y}, \mathbf{z}$ be feasible for P' . If $\alpha_f \leq \alpha_g$ for all $f \rightarrow g \in A_F$, then $\sum_f \alpha_f K_f \geq \sum_n \alpha_n K_n^*$. Similarly, if $\beta_q \geq \beta_r$ for all $q \rightarrow r \in A_R$, then $\sum_r \beta_r H_r \geq \sum_n \beta_n H_n^*$.

Proof. Multiplying (14) by α_f and the first part of (16) by α_n and summing, we have

$$\sum_n K_n^* \alpha_n \equiv \sum_f K_f \alpha_f + \sum_{f \rightarrow g \in A_F} y_{fg} (\alpha_f - \alpha_g) \leq \sum_f K_f \alpha_f .$$

Similarly, by (15) and (16) we have

$$\sum_n H_n^* \beta_n \equiv \sum_r H_r \beta_r + \sum_{q \rightarrow r \in A_R} z_{qr} (\beta_r - \beta_q) \leq \sum_r H_r \beta_r . \quad \square$$

We are now ready to state and prove the Lower-Bound Theorem.

The Lower-Bound Theorem. The minimum relaxed average cost B is a lower bound on the average cost of any feasible schedule over any finite horizon.

Proof. Let C be the sum of all costs incurred by a feasible schedule on the time horizon $[0, \tau)$. We will show that $C \geq B\tau$.

The Total Setup Cost.

Let I_n be the number of times in $[0, \tau)$ that an order is placed for product n . Then the total number of times in $[0, \tau)$ that the setup cost K_f is incurred is at least $\max_{n \in f} I_n$, and consequently the sum of all setup costs incurred in $[0, \tau)$ is at least $\sum_f K_f (\max_{m \in f} I_m)$. By Lemma 4 with $\alpha_f \equiv \max_{m \in f} I_m$, the total setup cost incurred is at least $\sum_n K_n^* I_n$.

The Rate at which Holding Costs Accumulate.

In Section 3 we saw that the rate at which holding costs accumulate at time t is $\sum_r H_r E_r^t$. If $r \in R_{mn}$, the echelon inventory

E_r^t on r is twice the length of time after time t during which the inventory on r is sufficient to meet the external demand for product n without replenishment of product m . If $n \in N$ is not an end product, define E_n^t to be the minimum of the echelon inventories of the routes that start at product n . Then $E_q^t \geq E_r^t$ for all $q \rightarrow r \in A_R$. By (2) and Lemma 4 with $\beta_r \equiv E_r^t$, the rate at which holding costs accumulate at time t is at least $\sum_n H_n^* E_n^t$.

The Total Cost Incurred in $[0, \tau)$.

Therefore $C \geq \sum_n (K_n^* I_n + \int_0^\tau H_n^* E_n^t dt)$. Note that for each product n , E_n^t is nonnegative, increases only when an order is placed for product n , and decreases with a slope of -2 otherwise. Therefore $I_n K_n^* + \int_0^\tau H_n^* E_n^t dt$ is the cost incurred by a feasible schedule for the single-facility lot-sizing problem on $[0, \tau)$ with setup cost K_n^* , holding cost rate H_n^* , and demand rate two. The optimal solution to this problem calls for orders to be placed at equal intervals, resulting in an average cost of at least $2(K_n^* H_n^*)^{.5}$ [CH62]. Therefore $C \geq \sum_n 2\tau(K_n^* H_n^*)^{.5} = B\tau$. $\square$

Note that the proof of the Lower Bound Theorem depends only on the fact that $\mathbf{y}, \mathbf{z}$ are feasible solutions to P' , not on their optimality. Thus the dual of the problem of minimizing the relaxed average cost of a power-of-two policy is the problem of allocating setup and holding costs to individual products so as to maximize the lower bound B .

8. COMPUTING EFFECTIVE POLICIES.

An **order-preserving policy** T is a power-of-two policy whose order intervals are nondecreasing in their optimal relaxed values t , i.e., that satisfies $T_m \geq T_n$ whenever $t_m \geq t_n$ and $m, n \in N$. In this section we prove that an order-preserving optimal power-of-two policy exists. We then show how to compute both an optimal power-of-two policy and a policy that is optimal among all power-of-two policies with fixed base time period β . The worst-case effectiveness of these two policies is at least 98% and 94%, respectively, relative to the lower bound B . The methods of computing policies and the proofs given in this section are similar to the ones given in [Ro84] for one-warehouse multi-retailer systems.

The Average Cost of an Order-Preserving Policy.

All products in a given cluster have the same optimal relaxed order interval. Therefore they will have the same order interval in an order-preserving policy. Thus we can treat each cluster as if it were a single product.

The next lemma states that the average cost of an order-preserving policy can be expressed as a sum of average cost functions of the single-facility lot-sizing type, one for each cluster. The lower bound B is the sum of the minima of these average cost functions. Therefore, the effectiveness of an order-preserving policy can be bounded from below by comparing each of the single-facility average-cost functions with its minimum.

Lemma 5. The average cost of an order-preserving policy $\mathbf{T}$ is

$$(17) \quad c(\mathbf{T}) = \sum_{\lambda} (K^{\lambda}/T^{\lambda} + H^{\lambda}T^{\lambda}) ,$$

where $T^{\lambda} = T_n$ if product n is in cluster λ .

Proof. If $r \in R \cap J^{\lambda}$, by (6) and (10) $t^{\lambda} = t_r = \max_{n \in r} t_n$. This implies that $T^{\lambda} = \max_{n \in r} T_n$. Similarly, $T^{\lambda} = \min_{n \in f} T_n$ for all $f \in F \cap J^{\lambda}$. The result now follows from (3). $\square$

Note that by (11) the average cost $K^{\lambda}/T^{\lambda} + H^{\lambda}T^{\lambda}$ associated with cluster λ is minimized by the cluster's optimal relaxed order interval t^{λ} , and that the minimum is $M^{\lambda} \equiv 2(K^{\lambda}H^{\lambda})^{.5}$. By (12) we have

$$(18) \quad B = \sum_{\lambda} M^{\lambda} .$$

If we define $e(q) \equiv 2/(q + q^{-1})$ and $q^{\lambda} \equiv T^{\lambda}/t^{\lambda}$, then since $K^{\lambda}/t^{\lambda} = H^{\lambda}t^{\lambda} = M^{\lambda}/2$ we can write (17) as

$$(19) \quad c(\mathbf{T}) = \sum_{\lambda} M^{\lambda}/e(q^{\lambda}) .$$

Observe that if $\mathbf{T} = \mathbf{t}$, then (19) reduces to (18) because $e(1) = 1$. Also note that if we consider only the costs associated with cluster λ , the effectiveness of $\mathbf{T}$ is $e(q^{\lambda})$. Thus the effectiveness of $\mathbf{T}$ at each cluster depends only on the quotients q^{λ} of the order intervals that $\mathbf{t}$ and $\mathbf{T}$ use there, and is otherwise independent of the cost and demand data. For this reason we will call q^{λ} an **effectiveness quotient**.

It follows from the Lower-Bound Theorem that the effectiveness of an order-preserving integer-ratio policy $\mathbf{T}$ is at least

$$(20) \quad \frac{B}{c(\mathbf{T})} = \left[\sum_{\ell} (M^{\ell}/B)/e(q^{\ell}) \right]^{-1} \geq \min_{\ell} e(q^{\ell}) .$$

Thus a sufficient condition for $\mathbf{T}$ to have effectiveness at least ϵ , say, is that $\mathbf{T}$ have at least that effectiveness at each cluster.

94%-Effective β -Optimal Policies.

We now show how to find an order-preserving power-of-two policy with $e(q^{\ell}) \geq e(\sqrt{2})$ for all clusters ℓ . By (20) the effectiveness of such a policy is at least $e(\sqrt{2}) > .94$. To that end, note that $e(\alpha)$ is strictly quasiconcave, achieves its maximum at $\alpha = 1$, and satisfies $e(\alpha) = e(1/\alpha)$ for all $\alpha > 0$. Therefore $e(q^{\ell}) \geq e(\sqrt{2})$ if and only if q^{ℓ} is in the **root-two interval** $[\sqrt{.5}, \sqrt{2}]$.

Let $\beta > 0$. We will temporarily restrict ourselves to order-preserving policies in which T_n is a power of two times β . For every cluster ℓ there is a unique positive number $T^{\ell} > 0$ such that $\sqrt{.5} \leq q^{\ell} < \sqrt{2}$ and T^{ℓ}/β is an integer power of two. Clearly T^{ℓ} is given by

$$(21) \quad T^{\ell} = 2^k \beta \quad \text{if} \quad 2^{-.5-k} \leq \beta/t^{\ell} < 2^{.5-k}$$

and k is an integer.

It is easily verified that $\mathbf{T}$ as given by (21) is order-preserving.

The 94% Theorem. For every $\beta > 0$ there is a power-of-two policy with effectiveness at least $200\% / (\sqrt{2} + \sqrt{.5}) > 94\%$ whose order intervals are all powers of two times β .

Proof. The theorem follows immediately from (20) and (21). $\square$

A policy that is optimal among all power-of-two policies for which T_n is an integer power of two times β is called **β -optimal**. The next lemma states that the order-preserving policy given by (21) is in fact β -optimal.

Lemma 6. If $\beta > 0$ and T satisfies (21), then T is β -optimal. Further, if $\bar{T} = (\bar{T}_n : n \in N)$ is β -optimal and $K_n^* > 0$, then $\bar{q}_n \equiv \bar{T}_n/t_n$ is in the root-two interval.

The proof is found in Appendix D.

Overview of the Method for Computing 98%-Effective Policies

We now show how to find an optimal power-of-two policy given t in $O(\Pi \log \Pi)$ time and prove that this policy is at least 98%-effective. We outline the method briefly as follows. By Lemma 6 there is an optimal power-of-two policy that satisfies (21) for some base period β . Furthermore we may assume that $\beta \in [\sqrt{.5}, \sqrt{2})$. Consider the class of ratio sets $(T^\ell/\beta : 1 \leq \ell \leq L)$ such that T satisfies (21) for some $\sqrt{.5} \leq \beta < \sqrt{2}$. We will show that there are at most $L + 1$ distinct ratio sets in this class. For any one of these $L + 1$ ratio sets, we easily can compute an optimal base period β . Consequently, the best of these $L + 1$ policies is an optimal power-of-two policy.

Identify the Potentially-Optimal Sets of Ratios.

We wish to describe T^ℓ as given by (21) as a function of β . Let the integer k^ℓ and the scalar b^ℓ be given by $\sqrt{.5} \leq 2^{.5-k^\ell} t^\ell \equiv b^\ell < \sqrt{2}$. Then by (21)

$$(22) \quad T^\ell/\beta = \begin{cases} 2^{k^\ell} & \text{if } \sqrt{.5} \leq \beta < b^\ell \\ 2^{k^\ell-1} & \text{if } b^\ell \leq \beta < \sqrt{2} . \end{cases}$$

As β increases from $\sqrt{.5}$ to $\sqrt{2}$, T^ℓ/β only changes once for each cluster ℓ . Therefore there are only $L + 1$ different potentially-optimal ratio sets $(T^\ell/\beta, 1 \leq \ell \leq L)$.

Finding an Optimal Power-of-Two Policy.

We sort the b^ℓ 's, list them in nondecreasing order, and denote the m^{th} number in the list by b_m . On setting $b_0 \equiv \sqrt{.5}$ and $b_{L+1} \equiv \sqrt{2}$, if $\sqrt{.5} \leq \beta < \sqrt{2}$ then we have $b_m \leq \beta < b_{m+1}$ for some $0 \leq m \leq L$. By (17), if $b_m \leq \beta < b_{m+1}$ and if (22) holds then $c(T) = K'_m/\beta + H'_m\beta$ where

$$K'_m \equiv \sum_{b^\ell \leq b_m} K^\ell 2^{1-k^\ell} + \sum_{b^\ell > b_m} K^\ell 2^{-k^\ell} \quad \text{and} \\ H'_m \equiv \sum_{b^\ell \leq b_m} H^\ell 2^{k^\ell-1} + \sum_{b^\ell > b_m} H^\ell 2^{k^\ell} .$$

For this ratio set the optimal value of β is

$$(23) \quad \beta = (K'_m/H'_m)^{.5} ,$$

and the average cost of the corresponding policy is $2(K'_m H'_m)^{.5}$.

Therefore an optimal power-of-two policy is given by (22) and (23), where m is chosen to minimize $K'_m H'_m$.

The computational effort required to find an optimal power-of-two policy can be analyzed as follows. The number of operations required to calculate k^λ and b^λ for all $1 \leq \lambda \leq L$ and to calculate K'_0 and H'_0 is $O(\Pi)$. The number of computations required to calculate K'_m and H'_m given K'_{m-1} and H'_{m-1} is $O(1)$. The hardest part of the procedure is sorting the b^λ 's, which requires $O(L \log L) \leq O(\Pi \log \Pi)$ comparisons. Therefore given K^λ , H^λ , and J^λ for all $1 \leq \lambda \leq L$, the computational effort required to calculate an optimal power-of-two policy is $O(\Pi \log \Pi)$.

Effectiveness.

The next theorem gives us a very tight lower bound on the worst-case effectiveness of an optimal power-of-two policy.

The 98% Theorem. The effectiveness of an optimal power-of-two policy is at least $\sqrt{2} \ln 2 \times 100\% > 98\%$.

Proof. For each base period $\beta \in [\sqrt{.5}, \sqrt{2})$, let $C(\beta)$ be the average cost of the policy $\mathbf{T}$ given by (22). By Lemma 6 the cost of the optimal power-of-two policy is $\min_{\sqrt{.5} \leq \beta < \sqrt{2}} C(\beta)$.

We will use as an upper bound on $\min_{\sqrt{.5} \leq \beta < \sqrt{2}} C(\beta)$ the weighted average of $C(\beta)$ over the interval $[\sqrt{.5}, \sqrt{2})$, weighted with respect to the density function $1/(\beta \ln 2)$. Recall from (19) that $C(\beta) = \sum_{\lambda} M^\lambda / e(q^\lambda)$ where $q^\lambda \equiv T^\lambda / t^\lambda$ as a function of β is given by (22). We will show that the weighted average of $1/e(q^\lambda)$ as β runs from $\sqrt{.5}$ to $\sqrt{2}$ is equal to the weighted average of $1/e(\alpha)$ as α

runs from $\sqrt{.5}$ to $\sqrt{2}$ and is thus independent of the input data. The density function $1/(\beta \ln 2)$ was chosen to make this property hold.

It follows from (22) that $q^\lambda = \sqrt{2} \beta / b^\lambda$ if $\sqrt{.5} \leq \beta < b^\lambda$ and $q^\lambda = \sqrt{.5} \beta / b^\lambda$ if $b^\lambda \leq \beta < \sqrt{2}$. Therefore

$$\begin{aligned} & \int_{\sqrt{.5}}^{\sqrt{2}} \frac{1/e(q^\lambda)}{\beta \ln 2} d\beta \\ &= \int_{1/b^\lambda}^{\sqrt{2}} \frac{1/e(\alpha)}{\alpha \ln 2} d\alpha + \int_{\sqrt{.5}}^{1/b^\lambda} \frac{1/e(\alpha)}{\alpha \ln 2} d\alpha \\ &= \frac{1}{\ln 2} \int_{\sqrt{.5}}^{\sqrt{2}} \left(\frac{1}{2\alpha^2} + \frac{1}{2} \right) d\alpha = \frac{1}{\sqrt{2} \ln 2}. \end{aligned}$$

Therefore by (18),

$$\min_{\sqrt{.5} \leq \beta \leq \sqrt{2}} C(\beta) \leq \int_{\sqrt{.5}}^{\sqrt{2}} C(\beta) \frac{d\beta}{\beta \ln 2} = \frac{B}{\sqrt{2} \ln 2}. \quad \square$$

9. LEAD TIMES AND OTHER EXTENSIONS AND LIMITATIONS OF THE MODEL.

Of the assumptions made in our model, one of the most severe is the requirement that when an order is placed, it is instantly delivered. We discuss two different ways in which this assumption can be relaxed, some of the difficulties that arise in so doing, and conditions under which our results on worst-case effectiveness still hold. We also discuss conditions under which the power-of-two constraint can be modified.

Lead Times.

Deterministic lead times can be accommodated, but often at the cost of losing our bounds on worst-case effectiveness. As an example consider the system illustrated in Figure 6. Let $\Lambda_{mn} > 0$ be the lead time for production of product m and for shipment from m to n , and let $\Lambda_{13} = 1$, $\Lambda_{14} = 4$, $\Lambda_{23} = 2$, and $\Lambda_{24} = 5$. We initiate production of the first lot of product 1 (resp., 2, 3, 4) at time 1 (resp., 0, 2, 5). The production of subsequent lots is timed according to the order intervals computed by our model. The first shipments of products 1 and 2 become available for processing into product 3 (resp., 4) at time 2 (resp., 5), precisely when production of the first lot of product 3 (resp., 4) is initiated. In this ideal case lead times can be easily accommodated without causing any unnecessary waiting.

Figure 6. A Simple Product Network.

We are not always so fortunate. If instead $\Lambda_{24} = 4$, each shipment of product 2 that arrives for processing into product 4 will apparently have to wait for one time period until the shipment of product 1 arrives. In the meantime extra holding costs are incurred. The magnitude of these extra holding costs can depend on the order intervals. For example, if the order interval for product 2 is 1

and we modify the schedule by omitting the first shipment from 2 to 4 , then all goods can be used immediately upon arrival.

If there are families of two or more products, lead times can cause extra setup costs to be incurred as well as extra holding costs. Recall that the setup cost K_f is incurred every time any subset of the products in family f is ordered. In compensating for lead times, we want to avoid creating a situation in which orders for two products in the same family that should have occurred simultaneously occur separately. This would result in an extra setup cost.

We will give sufficient conditions under which lead times can easily be compensated for without incurring any extra setup or holding costs. We also describe how to economically compensate for lead times when these conditions are only approximately met.

Compensating for Lead Times in a Balanced Network.

Define the **lead-time network** to be the network obtained by appending to the product network a node for every multi-product family $f \in F \setminus N$, and an arc with zero lead time from every product $n \in f$ to f . A **path** is a sequence of nodes $\{k_1, k_2, \dots, k_I\}$ in the lead-time network such that either (k_i, k_{i+1}) or (k_{i+1}, k_i) is an arc for all $1 \leq i < I$. A **forward arc** is an arc of the form (k_i, k_{i+1}) and a **backward arc** is an arc of the form (k_{i+1}, k_i) . The **transit time** of a path is the sum of the lead times of the forward arcs in the path, minus the sum of the lead times of the backward arcs in the path. A **cycle** is a path in which $k_1 = k_I$.

A multi-item multi-stage production/inventory system is called **balanced** if all cycles in the lead time network have zero transit

times. We will show that in a balanced system, 98%-effective and the 94%-effective policies can be implemented with no loss of effectiveness.

We assume that we are free to choose the point in time t^n at which we begin to meet the external demand for product n . This assumption is made for convenience only. As long as t^n is finite, the long run average cost of an optimal policy is independent of t^n .

We also assume that there are uniform, finite upper bounds on both the total number of orders that can be placed in any one unit of time, and on the total amount of inventory that can be in the system at any one point in time. This implies that in the long run the average amount per unit time of product m that flows along the arc $m \rightarrow n \in A$ is $V_{mn} \equiv \sum_k \sum_{r \in R_{nk}} \lambda_r d_k$, independent of the schedule chosen. The pipeline holding cost incurred per unit of product m that flows along arc $m \rightarrow n$ is $E_{mn} \equiv h_m^i \tau_{mn}$. Therefore in the long run the total average pipeline holding cost per unit time incurred in the system is $D \equiv \sum_{m \rightarrow n \in A} V_{mn} E_{mn}$.

Given a balanced system with lead times, called the **given system**, we define offsets as follows. The **offset** τ^1 for node 1 is defined to be zero. The **offset** τ^k for node $k \neq 1$ is defined to be the transit time of an arbitrary path from 1 to k . Since the given system is balanced, τ^k is uniquely defined.

From the given system we obtain the **zero-lead-time** system by setting all of the lead times equal to zero. Given any schedule for the zero-lead-time system, the **corresponding** schedule for the given system is found by moving the schedule of orders for each product n

forward in time by τ^n time units. Conversely, if we have a schedule for the given system, the **corresponding** schedule for the zero-lead-time system is found by moving the schedule of orders for each product n backward in time by τ^n time units.

Lemma 7. A schedule S is feasible for the given system if and only if the corresponding schedule S^* is feasible for the zero-lead-time system. Furthermore, if $A(S^*)$ is the long-run average cost of S^* , then the long-run average cost of the corresponding schedule for the given system is $A(S^*) + D$.

The proof is given in Appendix E.

Then Balanced-System Theorem. If the given system is balanced and T is a 94%-effective (resp, 98%-effective) power-of-two policy for the zero-lead-time system then the effectiveness of the corresponding schedule for the given system is at least 94% (resp., 98%).

Proof. By Lemma 7 and the Lower Bound Theorem, the long-run average cost of any feasible schedule for the given system is at least $B + D$. If the long-run-average cost of T is $A(T)$, then the effectiveness of the corresponding schedule for the given system is at least $(B + D)/(A(T) + D) \geq B/A(T) \geq .94$ (resp., .98). $\square$

Balanced systems are more common than one might think. Because the lead-time networks of single-product assembly systems [CW72, CWH72, CWW73, SS77], arborescence distribution systems [GS75], and conjoined assembly-arborescence systems [Wi81, Wi82] are acyclic,

these systems are balanced. A more general class of balanced multi-item systems, which includes most multi-item distribution systems and many multi-item assembly systems, is the following. There are a number of storage locations, at each of which any of several different items may be held. We construct a network in which the storage locations are nodes and there is an arc from location i to location j if any one item is routed directly from location i to location j . If this network is acyclic, if all families consist entirely of products held at a single location, and if the transit time from any one location to another is the same for all items, then the system is balanced.

We can determine whether or not a system with lead times is balanced by using a shortest-path algorithm as follows. The arc $m \rightarrow n$ in the lead-time network is assigned length Δ_{mn} , and for every such arc we add another arc from n to m with length $-\Delta_{mn}$. Clearly the system is balanced if and only if there are no negative cycles in this expanded network.

The existence of a negative cycle can be tested in $O(NE)$ time where N is the number of nodes and E is the number of arcs in the lead-time network [Ye70]. The number of nodes is the sum of the number of products and the number of families with more than one member. For most real-world systems, the number of arcs is small. For example, in the industrial system illustrated in [MM83], the number of nodes in the expanded network is 50 and the number of arcs is 124.

Compensating for Lead Times in Non-Balanced Systems.

Suppose we have a nonbalanced system and a feasible policy for the corresponding zero-lead-time system. We wish to implement this policy in the given system by simply translating the schedule of orders for product n forward by τ_n units of time. We assume that for every arc (m,k) in the product network, the units of product m that are shipped to k at time t are used at k at time $t + \tau_n - \tau_m$. In order for such a policy to be feasible we must have $\tau_n - \tau_m \geq \Delta_{mn}$ for all arcs $m \rightarrow n$.

We define the **waiting time** on arc (m,k) to be $w_{mn} \equiv \tau_n - \tau_m - \Delta_{mn} \geq 0$. The average number of units of product m that will flow along (m,n) per unit time is $\lambda_{mn} \sum_k \sum_{r \in R_{nk}} \lambda_r d_k$. Consequently the average cost per unit time of holding the goods that flow along (m,n) during the w_{mn} days that they spend waiting is $w_{mn} h'_m \lambda_{mn} \sum_k \sum_{r \in R_{nk}} \lambda_r d_k \equiv H'_{mn} w_{mn}$. The problem of finding the amounts τ_n that the schedules should be translated by in order to minimize the average cost of waiting per unit time is to minimize

$$(91) \quad \sum_{(m,n)} H'_{mn} w_{mn}$$

subject to

$$(92) \quad \tau_n - \tau_m - w_{mn} = \Delta_{mn} \quad \text{for all arcs } (m,n)$$

$$(93) \quad w_{mn} \geq 0.$$

If we eliminate w_{mk} from the above formulation by substituting (92) into (91) and (93) we get the dual of a linear transportation problem.

This method will work well when the waiting times are small relative to the order intervals and the fraction of the total setup costs that are associated with multi-product families is small. When this is not the case the schedule generated above will be feasible, but may be far from optimal. The problem of compensating for lead times in such systems is an interesting and an important one, and warrants further investigation.

Finite Production Rates.

Suppose that the production rate must be finite, i.e., the lead time has a component that is linear in the lot size. This is very common in practice. It is well known that in the single-item, single-stage case, one can account for the effect of a finite production rate by multiplying the holding cost rate by a scalar [St72]. In a multi-stage system the total holding cost depends on the order intervals in a more complicated way. The form of this dependence is a function of how goods move from one stage to the next. This problem has been studied under different assumptions by Schwarz and Schrage [SS75] for assembly systems, by Bigham and Mogg [BM78, BM79] for assembly systems, by Szendrovits [Sz75] for series systems, and by Singer [Si78] for single-warehouse, multi-retailer systems.

Another, perhaps more serious, problem that arises with the introduction of finite production rates is that of scheduling. Our model may call for lots of several different products to be produced simultaneously on a single machine. In this case one must find some way of scheduling production of the products so that only one of them is being produced at any given point in time. For a one-stage system

this is known as the economic lot-scheduling problem [E178, MS81]. It has been observed by Elmagraby [E178] and others that the scheduling problem is usually much easier with power-of-two policies than with other types of policies.

Modifying the Power-of-Two Constraint.

We have assumed that T_m/T_n is an integer power of two for all pairs of products m, n . This assumption is computationally convenient, and we have seen that resulting loss of effectiveness is very small. However there are situations in which order-interval ratios other than powers of two are desirable. Within the framework we have established, the power-of-two constraint can be replaced with the following condition. For every route $r \in R$ and every product $n \in r$, $(\max_{m \in r} T_m)/T_n$ is an integer, and for every family $f \in F \setminus N$ and every product $n \in f$, $T_n/(\min_{m \in f} T_m)$ is an integer.

Following the analysis in Section 4 it is easily verified that the average cost of such a policy is given by (3). If, in addition, the policy is order-preserving and q_n is in the root-two interval for all products n , the 94% Theorem holds. Unfortunately, the condition is difficult to check. A simpler sufficient condition is that T be order-preserving and that $T^\ell/T^{\ell+1}$ be either an integer or the reciprocal of an integer for all clusters ℓ .

10. OTHER METHODS OF COMPUTING POWER-OF-TWO POLICIES.

Veinott [Ve71] has shown that there is a flow $\mathbf{v}, \mathbf{x}$ depending on $\mathbf{d} = (d_j : j \in J)$ that simultaneously minimizes

$$(24) \quad \sum_j d_j f(v_j/d_j)$$

subject to (8) and $\mathbf{x} \geq 0$ for every convex function f . Taking $d_j = K_j$, $f(v) = -\sqrt{v}$ if $v \geq 0$, and $f(v) = \infty$ if $v < 0$, (24) becomes (7).

By Veinott's theorem we can take $f(v) = v^2$ and solve the problem as a separable quadratic program with network constraints. However the special-purpose quadratic-programming algorithms that exist do not exploit the full structure of the problem. Consequently the Network-Partitioning Algorithm should be much more efficient.

Dembo recently developed an algorithm for solving convex-cost network-flow problems [De83]. He tested it on five problems that have separable cost functions. Typical results include solving a problem with 64 nodes and 116 arcs in 5.1 seconds, a problem with 666 nodes and 906 arcs in 99 seconds, and an ill-conditioned problem with 1117 nodes and 2230 arcs in 7560 seconds. He used a DEC 20/60 in his tests.

By comparison, Glover, Klingman, Mote, and Whitman report solving max-flow min-cut problems with 1000 arcs and 5000 nodes in an average running times between .4 seconds and 1.3 seconds, depending on the topology of the network [GKMW80]. They used a CDC6600.

The number of clusters produced by the network-partitioning algorithm is small compared to the total number of products, and as the algorithm progresses the size of the networks decreases. Thus network-partitioning algorithm should be much faster than Dembo's algorithm. Of course this is to be expected because Dembo's algorithm was designed to solve a much more general problem.

Problems equivalent to P have arisen and useful algorithms have been developed for solving both P and special cases of P in at least three very different contexts. The first of these is isotonic regression, the problem of finding maximum-likelihood estimators under order restrictions [BBBB72, pp. 24-25; Ve71]. The isotonic regression problem is to minimize $\sum_j w_j(t_j - c_j)^2$ subject to $t_i \geq t_j$ for all $i \rightarrow j \in A$. The dual of this problem is to minimize $\sum_j v_j^2/2w_j$ subject to (8) and $x \geq 0$ where $H_j = 2w_j c_j$. This problem is equivalent to P by Veinott's theorem with $f(v) = v^2$ and $d_j = 2w_j$. A number of algorithms have been developed for isotonic regression. In particular, if the cost network is a rooted tree, there is an algorithm that can be implemented in $O(V \log V)$ time where V is the number of vertices in the cost network [Th62; BBBB72, pp. 13-15]. This result applies to assembly systems. In this case the cost network is the product network, so $V = \Pi$. Consequently the overall running time for finding a 98%-effective power-of-two policy for single-product assembly systems is $O(\Pi \log \Pi)$.

A second context in which a problem equivalent to P has arisen is multi-dimensional scaling [Kr64a]. Given a finite number of items and a measure of the dissimilarity between pairs of items, we wish to

find a point in euclidean n -space for each item such that the distances between pairs of points coincide as nearly as possible with the dissimilarities between the corresponding items. As a subproblem in an algorithm for multi-dimensional scaling, Kruskal minimizes $\sum_j (t_j - c_j)^2$ subject to $t_i \geq t_j$ for all $i \rightarrow j \in A$ for the special case in which the cost network is a simple chain [Kr64b]. This corresponds to the single-product series system. The running time of the algorithm is $O(\Pi)$.

A third context in which a problem equivalent to P has arisen is the problem of scheduling jobs on a single machine so as to minimize the total weighted completion time subject to precedence constraints. This problem is NP-complete, but a specific instance of the problem can usually be split up into a number of smaller subproblems [Si75]. The problem of finding the finest admissible partition of the original problem into subproblems is shown to be equivalent to P in Appendix F.

Lawler has developed two algorithms for this problem, one for the general case and one for the case in which the cost network is a series-parallel digraph [La78]. The running time for the latter algorithm is $O(\Pi \log \Pi)$. This generalizes the above-mentioned result for rooted trees. Unfortunately other than assembly systems, multi-stage production/inventory systems whose cost networks are series-parallel digraphs are extremely rare. Assuming that the data is all integer and uniformly bounded in magnitude, Lawler's algorithm for the general case can be implemented in $O(\Pi V^3 \log V)$ time. His algorithm is similar in spirit to Maxwell and Muckstadt's $O(\Pi V^3)$ algorithm.

11. CONCLUSION.

The effectiveness and simplicity of power-of-two policies make them very attractive. In most practical applications the errors inherent in the data and in the model will be much more significant than the 2% difference between the average cost of an optimal power-of-two policy and the minimum relaxed average cost. The lot-sizing rules are efficient enough to be used in multi-stage multi-product production/inventory systems that are both very large and very complex. For assembly systems, the lot-sizing rule can be implemented in $O(\Pi \log \Pi)$ time.

It is hoped that this work will prove to be useful as a practical method of lot-sizing in many multistage systems, and that it will provide insight that will be helpful in developing effective methods of controlling capacity-constrained, stochastic, and nonstationary multi-stage production/inventory systems.

ACKNOWLEDGMENTS

I am greatly indebted to my advisor, Professor Arthur F. Veinott, Jr. The encouragement and sense of accomplishment he provided during the investigation were invaluable. His insightful suggestions have led to many valuable improvements in the exposition of the paper, its organization, and in the notation used. He also made suggestions which led to a considerably simpler and much more elegant proof of the Lower-Bound Theorem. I am also indebted to Professor Christos Papadimitriou and Joe Mitchell for the helpful comments they made on earlier versions of this work.

I would like to thank the National Science Foundation and General Motors Corporation for the financial support they provided. I would also like to thank the Operations Research Department at Stanford University and the School of Operations Research and Industrial Engineering at Cornell for the use of their Philips 2001 word-processors, on which this document was prepared.

APPENDIX A. CALCULATION OF THE DUAL P OF D .

In this appendix we calculate the conjugate dual P of D . Let $K_j \equiv 0$ for all $j \notin F$ and $H_j \equiv 0$ for all $j \notin R$. If $K_j > 0$, we define $K_j(t_j) = K_j/t_j$ for all $t_j > 0$, and $K_j(t_j) = \infty$ for all $t_j \leq 0$. If $K_j = 0$ we define $K_j(t_j) \equiv 0$. Then we can write (4) as

$$(4^*) \quad \sum_j [K_j(t_j) + H_j t_j] .$$

In (5), the constraint $t_j \geq 0$ is redundant for all nodes j that have a successor in the cost network. If j has no successors then $K_j > 0$, so the constraint $t_j \geq 0$ is reflected in the cost function (4*). Therefore the nonnegativity constraint in (5) can be ignored if we use (4*) in place of (4).

We use the standard, right-hand-side perturbation of D , namely

$$\phi(\mathbf{t}, \boldsymbol{\xi}) = \sum_j (K_j(t_j) + H_j t_j) + \sum_{i \rightarrow j \in A} I(t_i - t_j \geq \xi_{ij}) .$$

The conjugate dual of D is to choose $\mathbf{x} \equiv (x_{ij} : i \rightarrow j \in A)$ to minimize

$$\phi(\mathbf{x}) \equiv \sup_{\mathbf{t}, \boldsymbol{\xi}} [\boldsymbol{\xi} \mathbf{x} - \phi(\mathbf{t}, \boldsymbol{\xi})] .$$

First note that if $x_{ij} < 0$, by letting $\xi_{ij} \rightarrow -\infty$ we have $\phi(\mathbf{x}) = \infty$. Furthermore if $\mathbf{x} \geq 0$, we can assume $\xi_{ij} = t_i - t_j$. Thus

$$\phi(\mathbf{x}) = \sup_{\mathbf{t}} \sum_j [(\sum_{k \in s_j} x_{jk} - \sum_{i \in p_j} x_{ij} - H_j)t_j - K_j(t_j)] .$$

We will refer to the term in brackets as $f_j(t_j, \mathbf{x})$. We now show that for each fixed $\mathbf{x}$, either $\phi(\mathbf{x}) = \infty$ or for every $j \in J$ there is a $t_j \in (0, \infty]$ that maximizes $f_j(t_j, \mathbf{x})$ and satisfies

$$(25) \quad \sum_{k \in s_j} x_{jk} - \sum_{i \in p_j} x_{ij} - H_j + K_j/t_j^2 = 0 .$$

Suppose that $K_j = 0$. Then either $\sup_{t_j} f_j(t_j, \mathbf{x}) = \infty$, meaning that $\phi(\mathbf{x}) = \infty$, or else $f_j(t_j, \mathbf{x})$ is independent of t_j , meaning that (25) holds for all $t_j > 0$. On the other hand, if $K_j > 0$ and $\sup_{t_j} f_j(t_j, \mathbf{x})$ is finite, then the supremum is attained at some $t_j \in (0, \infty]$ because f_j is concave. Furthermore the partial derivative of f_j with respect to t_j must be zero there, i.e., (25) must hold.

We adopt the standard convention $0 \times \infty = 0$. Assuming that (25) holds for all j , we can write $\phi(\mathbf{x}) = \sum_j -2K_j/t_j$. To get a closed-form expression for ϕ , define $v_j \equiv K_j/t_j^2$ if $K_j > 0$ and $v_j \equiv 0$ if $K_j = 0$. Then $\phi(\mathbf{x}) = \sum_j -2K_j/t_j = \sum_j -2(K_j v_j)^{.5}$. The dual P of D is therefore to maximize (7) subject to (8).

APPENDIX B. CALCULATION OF THE DUAL P' OF D .

In this appendix we calculate the dual P' of D . Let $s_f \equiv 1/t_f$ for all $f \in F$. Then $s_n t_n = 1$ for all $n \in N$, and D can be written as the problem of minimizing

$$(26) \quad \sum_{f \in F} K_f s_f + \sum_{r \in R} H_r t_r$$

subject to

$$(27) \quad s_f \leq s_g \quad \text{for all } f \rightarrow g \in A_F,$$

$$(28) \quad t_q \geq t_r \quad \text{for all } q \rightarrow r \in A_R, \text{ and}$$

$$(29) \quad t_n \geq 0 \quad \text{for all } n \in N$$

where $s_n = 1/t_n$ for all $n \in N$.

The standard, right-hand-side perturbation of this problem is

$$\begin{aligned} \phi(\mathbf{s}, \mathbf{t}, \boldsymbol{\eta}, \boldsymbol{\zeta}) \equiv & \sum_{f \in F} K_f s_f + \sum_{r \in R} H_r t_r + \sum_{n \in N} I(t_n \geq 0) \\ & + \sum_{f \rightarrow g \in A_F} I(s_g - s_f \geq \eta_{fg}) + \sum_{q \rightarrow r \in A_R} I(t_q - t_r \geq \zeta_{qr}). \end{aligned}$$

The dual is to minimize

$$\phi(\mathbf{y}, \mathbf{z}) = \sup_{\mathbf{s}, \mathbf{t}, \boldsymbol{\eta}, \boldsymbol{\zeta}} [\boldsymbol{\eta} \mathbf{y} + \boldsymbol{\zeta} \mathbf{z} - \phi(\mathbf{s}, \mathbf{t}, \boldsymbol{\eta}, \boldsymbol{\zeta})].$$

Note that $\phi = \infty$ unless $\mathbf{y}, \mathbf{z} \geq 0$, and that in the latter case we can take $\eta_{fg} = s_g - s_f$ and $\zeta_{qr} = t_q - t_r$. Therefore if $\mathbf{y}, \mathbf{z} \geq 0$,

$$\begin{aligned}
\phi(\mathbf{y}, \mathbf{z}) &= \sup_{\mathbf{s}, \mathbf{t}} \left[\sum_{f \rightarrow g \in A_F} y_{fg} (s_g - s_f) + \sum_{q \rightarrow r \in A_R} z_{qr} (t_q - t_r) \right. \\
&\quad \left. - \sum_{f \in F} K_f s_f - \sum_{r \in R} H_r t_r - \sum_{n \in N} I(t_n \geq 0) \right] \\
&= \sup_{\mathbf{s}, \mathbf{t}} \left[\sum_{f \in F \setminus N} s_f \left(\sum_{g \in p_f} y_{gf} - \sum_{g \in s_f} y_{fg} - K_f \right) + \right. \\
&\quad \left. + \sum_{r \in R \setminus N} t_r \left(\sum_{q \in s_r} z_{rq} - \sum_{q \in p_r} z_{qr} - H_r \right) - \sum_{n \in N} [K_n^* s_n + H_n^* t_n - I(t_n \geq 0)] \right]
\end{aligned}$$

where

$$(16) \quad K_n^* \equiv K_n + \sum_{f \in s_n} y_{nf} \quad \text{and} \quad H_n^* \equiv H_n + \sum_{r \in p_n} z_{rn}.$$

Adopting the conventions $0/0 = 0$ and $0 \times \infty = 0$, and recalling that $s_n t_n = 1$, an optimal value of t_n is given by $t_n = (K_n^*/H_n^*)^{.5} \in [0, \infty]$. Also, $\phi(\mathbf{x}) = \infty$ unless

$$(14) \quad \sum_{g \in p_f} y_{gf} - \sum_{g \in s_f} y_{fg} - K_f = 0 \quad \text{for all } f \in F \setminus N$$

and

$$(15) \quad \sum_{q \in s_r} z_{rq} - \sum_{q \in p_r} z_{qr} - H_r = 0 \quad \text{for all } r \in R \setminus N.$$

If $\mathbf{y}, \mathbf{z} \geq 0$ and (14) and (15) hold, then since $K_n^* s_n = H_n^* t_n = (K_n^* H_n^*)^{.5}$, we have $\phi(\mathbf{y}, \mathbf{z}) = \sum_n -2(K_n^* H_n^*)^{.5}$. Thus the dual P' of D is to maximize $\sum_n 2(K_n^* H_n^*)^{.5}$ subject to (14), (15), (16), and $\mathbf{y}, \mathbf{z} \geq 0$.

APPENDIX C. PROOF OF LEMMA 3.

In this appendix we prove Lemma 3.

Lemma 3. An optimal solution to P' is given by

$$(30) \quad \begin{aligned} y_{fg} &\equiv x_{fg} t_g^2 \quad \text{for all } f \rightarrow g \in A_F \quad \text{and} \\ z_{qr} &\equiv x_{qr} \quad \text{for all } q \rightarrow r \in A_R \end{aligned}$$

where $\mathbf{x}$ solves P and $\mathbf{t}$ solves D . Furthermore,

$$\begin{aligned} \sum_{n \in N^\ell} K_n^* &= K^\ell, \quad \sum_{n \in N^\ell} H_n^* = H^\ell, \quad t_n^2 H_n^* = K_n^*, \quad \text{and} \\ \sum_{n \in N^\ell} 2(K_n^* H_n^*)^{.5} &= 2(K^\ell H^\ell)^{.5}, \end{aligned}$$

where $N^\ell \equiv N \cap J^\ell$.

Proof. Let $\mathbf{x}$ solve P and $\mathbf{t}$ solve D , and let $\mathbf{y}, \mathbf{z}$ be given by (30). Note that if $x_{ij} \neq 0$, then i and j are in the same cluster. Therefore $t_i = t_j$, so (11) and (8) imply (14) and (15). This establishes the feasibility of $\mathbf{y}$ and $\mathbf{z}$ for P' . We obtain (31) by summing (14) over all $f \in (F \setminus N) \cap J^\ell$ and (15) over all $r \in (R \setminus N) \cap J^\ell$, and by noting that $y_{fg} = 0$ (resp., $z_{qr} = 0$) unless f and g (resp., q and r) are in the same cluster. Equation (32) follows by taking $j = n$ in (8) and substituting in (11) and (30). Equation (33) follows from (31) and the facts that $(K_n^* H_n^*)^{.5} = K_n^* / t_n$, $(K^\ell H^\ell)^{.5} = K^\ell / t^\ell$, and $t_n = t^\ell$ for all $n \in J^\ell$.

The optimality of $\mathbf{y}, \mathbf{z}$ for P' now can be established as follows. We know that $\mathbf{y}, \mathbf{z}$ are feasible for P' , and by (12) the objective function of P' has the same value as the optimal solutions to D and P . By the theory of conjugate duality $\mathbf{y}$ and $\mathbf{z}$ must be optimal for P' [Ro70]. $\square$

APPENDIX D. PROOF OF LEMMA 6.

In this appendix we prove Lemma 6.

Lemma 6. If $\beta > 0$ and $\mathbf{T}$ satisfies (21), then $\mathbf{T}$ is β -optimal. Further, if $\bar{\mathbf{T}} = (\bar{T}_n : n \in N)$ is β -optimal and $K_n^* > 0$, then $\bar{q}_n \equiv \bar{T}_n/t_n$ is in the root-two interval.

Proof. Suppose that $\bar{\mathbf{T}} = (\bar{T}_n : n \in N)$ is a power-of-two policy such that $\bar{T}_n$ is a power of two times β for all products n . We will show that $c(\mathbf{T}) \leq c(\bar{\mathbf{T}})$, and that $c(\mathbf{T}) < c(\bar{\mathbf{T}})$ if $\bar{q}_n$ is not in the root-two interval and $K_n^* > 0$ for some product n . Let (21) hold and let $q_n \equiv T_n/t_n$. Since $\bar{T}_n/T_n = \bar{q}_n/q_n$ is a power of two, either $T_n = \bar{T}_n$ holds or $\sqrt{.5} \leq \bar{q}_n < \sqrt{2}$ does not hold. In either case $e(q_n) \geq e(\bar{q}_n)$, and $e(q_n) > e(\bar{q}_n)$ unless $\sqrt{.5} \leq \bar{q}_n \leq \sqrt{2}$.

Because $\bar{\mathbf{T}}$ is not necessarily order-preserving, we need to compute costs on a product-by-product basis rather than a cluster-by-cluster basis. By (3)

$$c(\bar{\mathbf{T}}) = \sum_{f \in F} K_f / (\min_{n \in f} \bar{T}_n) + \sum_{r \in R} H_r \max_{n \in r} \bar{T}_n.$$

Let $M_n \equiv 2(K_n^* H_n^*)^{.5}$. On setting $\alpha_f = 1/(\min_{n \in f} \bar{T}_n)$ and $\beta_r = \max_{n \in r} \bar{T}_n$ in Lemma 4, we have

$$c(\bar{T}) \geq \sum_n (K_n^*/\bar{T}_n + H_n^* \bar{T}_n) = \sum_n M_n/e(\bar{q}_n) \geq \sum_n M_n/e(q_n) .$$

The last inequality is strict unless $\bar{q}_n$ is in the root-two interval for all products n satisfying $M_n > 0$. But $M_n = 0$ if and only if $K_n^* = H_n^* = 0$ by Lemmas 1 and 3.

Since $q_n = q^\ell$ if product n is in cluster ℓ , and since $\sum_{n \in N^\ell} M_n = M^\ell$ by Lemma 3,

$$c(\bar{T}) \geq \sum_n M_n/e(q_n) = \sum_\ell M^\ell/e(q^\ell) = c(T) . \quad \square$$

APPENDIX E. PROOF OF LEMMA 7.

In this appendix we prove Lemma 7.

Lemma 7. A schedule S is feasible for the given system if and only if the corresponding schedule S^* is feasible for the zero-lead-time system. Furthermore, if $A(S^*)$ is the long-run average cost of S^* , then the long-run average cost of the corresponding schedule for the given system is $A(S^*) + D$.

Proof. Suppose that we are given a feasible schedule S for the given system, i.e., we know the timing and the amounts of all orders placed in the infinite horizon, and we know the points in time t^n at which we begin to satisfy the external demand for product n . The physical inventory $I_n(t)$ of product n at time t is determined by the following conditions.

- (a) $I_n(t)$ is zero before the first order for product n is received.
- (b) If we place an order of size Q for product n at time t , and if m is a predecessor of n , then we also dispatch a shipment of $\lambda_{mn}Q$ units of product m to n at time $t - \Lambda_{mn}$.
- (c) Suppose that no orders for product n are received at time t and no shipments of product n are dispatched at time t . Then $\frac{d}{dt} I_n(t) = -d_n$ if $t > t^n$ and $\frac{d}{dt} I_n(t) = 0$ if $t < t^n$.
- (d) $I_n(t)$ is right-continuous.
- (e) If either an order of product n is received at time t or a shipment of product n is dispatched at time t , then the size $I_n(t) - \lim_{s \uparrow t} I_n(s)$ of the jump in I_n at time t is the size of the order (if any) received at time t , minus the total size of all shipments dispatched at time t .

The schedule S is feasible if and only if $I_n(t)$ is nonnegative for all n and for all t .

The schedule for the zero-lead-time system that corresponds to S is S^* . Note that in S^* , the point in time at which we begin to satisfy the external demand for product n is $t^n - \tau^n$.

Let $I_n^*(t) \equiv I_n(t + \tau^n)$. We will show that $I_n^*(t)$ is the inventory of product n at time t for S^* , and consequently that S^* is feasible. It suffices to show that $I_n^*(t)$ satisfies properties (a) through (e) above, with the lead times set equal to zero. It is clear that (a) and (d) hold. In schedule S , if an order for product n arrives at time t and if m is a predecessor of n , then the corresponding shipment from m is dispatched at time $t - \Lambda_{mn}$. In

schedule S^* the order arrives at time $t - \tau^n$ and the shipment is dispatched at time $t - \Lambda_{mn} - \tau^m = t - \tau^n$, which is consistent with zero lead times. Thus (b), (c), and (e) hold as well.

Finally, we show that if the long-run average cost of S is $A(S)$, and that of S^* is $A(S^*)$, then $A(S) = A(S^*) + D$. Let $V_{mn}(t)$ be the total amount of product m that flows along the arc $m \rightarrow n \in A$ in the time interval $(-\infty, t]$. Since the total amount of inventory in the system at any one point in time is uniformly bounded, $\frac{1}{t} V_{mn}(t) \rightarrow V_{mn} \equiv \sum_k \sum_{r \in R_{nk}} \lambda_r d_k$ as $t \rightarrow \infty$, independent of the schedule chosen. Recall that the pipeline holding cost incurred per unit of product m that flows along arc $m \rightarrow n$ is $E_{mn} \equiv h_m^1 \tau_{mn}$, and that $D \equiv \sum_{m \rightarrow n \in A} V_{mn} E_{mn}$.

Let $C_j(t)$ be the total cost associated with node j of the lead-time network that is incurred by S prior to time t . This includes holding costs if j is a product and setup costs if j is a family, but it excludes the pipeline holding costs. Let $C_j^*(t)$ be the corresponding cost for S^* . Then $C_j^*(t) = C_j(t + \tau^j)$. The long-run average cost for schedule S is $A(S) = \liminf_{t \rightarrow \infty} \frac{1}{t} [\sum_j C_j(t) + \sum_{m \rightarrow n \in A} E_{mn} V_{mn}(t)] = \liminf_{t \rightarrow \infty} \sum_j \{ \frac{1}{t} C_j^*(t) + \frac{1}{t} [C_j(t) - C_j(t + \tau^j)] \} + D$. But $[C_j(t) - C_j(t + \tau^j)]$ is uniformly bounded in t , so $\frac{1}{t} [C_j(t) - C_j(t + \tau^j)]$ converges to zero. Therefore $A(S^*) = \liminf_{t \rightarrow \infty} \frac{1}{t} \sum_j C_j^*(t) + D = A(S^*) + D$. $\square$

APPENDIX F. THE EQUIVALENCE OF P AND THE PROBLEM OF DECOMPOSING A PRECEDENCE-CONSTRAINED SINGLE-MACHINE SCHEDULING PROBLEM WITH DEFERRAL COSTS.

Consider the problem of scheduling precedence-constrained jobs on a single machine so as to minimize the weighted sum of the completion times. We represent the precedence constraints by an acyclic network in which the node set J is the set of jobs and there is an arc from i to j if job i is constrained to precede job j . Let H_j and K_j be respectively the processing time and the weight of job j . A nonempty proper subset S of the set of jobs is called **initial** if there are no arcs from the complement of S to S . An initial set is called **ρ^* -minimal** if it is a minimal set among all initial sets S for which K_S/H_S is maximal. Suppose that $S \neq J$ is a ρ^* -minimal set. Sidney has shown that there is an optimal schedule in which every job in S precedes every job not in S [Si75]. Thus the scheduling problem decomposes into two subproblems, one on S and one on the complement of S . The problem on S cannot be decomposed any further, but the problem on the complement of S can often be broken down further by finding a ρ^* -minimal set in $S \setminus J$.

We will show that by slightly modifying the Network-Partitioning Algorithm, the clusters produced are the subproblems of Sidney's decomposition. We modify the algorithm by making it split a network in two when there is both a feasible flow and a cutset whose capacity is zero. The max-flow min-cut algorithm can easily be modified to find such cutsets when they exist. It is readily verified that the modified Network-Partitioning Algorithm yields the same values for t as the original algorithm, and consequently still solves D .

Lemma 8. Let λ be a subset of the set of clusters generated by the modified Network-Partitioning Algorithm, and let S be a ρ^* -initial set in the network whose nodes are $\cup_{\lambda \in \lambda} J^\lambda$. Then $S = J^k$ for some $k \in \lambda$.

Proof. Let k be the largest cluster index in λ such that $Q \equiv S \cap J^k$ is nonempty. We will show that $S = J^k$. Since Q is initial in J^k , $K^k/H^k \geq K_Q/H_Q$, and $K^k/H^k = K_Q/H_Q$ only if $J^k = Q$. Otherwise the modified algorithm would have split J^k in two. Next, because S is ρ^* -minimal and $S \setminus Q$ is initial, either $Q = S$ or $K_{S \setminus Q}/H_{S \setminus Q} < K_S/H_S$. Thus $K_Q/H_Q \geq K_S/H_S$, and $K_Q/H_Q = K_S/H_S$ only if $Q = S$. Finally, let m be the lowest cluster index in λ . Since S is ρ^* -minimal and J^m is initial, $K_S/H_S \geq K^m/H^m \geq K^k/H^k$. Thus $K^k/H^k \geq K_Q/H_Q \geq K_S/H_S \geq K^k/H^k$, so $J^k = Q = S$. $\square$

References

- Barlow, R. E., D. J. Bartholomew, J. M. Bremer, and H. D. Brunk (1972). Statistical Inference under Order Restrictions. John Wiley & Sons, New York, New York, 72-75.
- Bigham, P. E. and J. M. Mogg (1977). Scheduling a Multi-Stage Production System with Startup Delays. Omega 6, 183-187.
- Bigham, P. E. and J. M. Mogg (1979). Converging-Branch, Multi-Stage Production Schedules With Finite Production Rates and Startup Delays. Journal of the Operational Research Society 30, 737-745.
- Caie, J. P. and Maxwell, W. L. (1981). Hierarchical Machine Load Planning, Chapter 5 in L. B. Schwarz (ed.), Multi-Level Production/Inventory Control Systems: Theory and Practice, North Holland, New York, N.Y.
- Carr, C. R. and C. W. Howe (1962). Optimal Service Policies and Finite Time Horizons. Management Science 9, 126-140.

- Crowston, W. B. and M. H. Wagner (1973). Dynamic Lot-Size Models for Multi-Stage Assembly Systems. Management Science 20, 14-21.
- Crowston, W. B., M. H. Wagner, and A. Henshaw (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., M. H. Wagner, and J. Williams (1973). Economic Lot-Size Determination in Multi-Stage Assembly Systems. Management Science 19, 517-526.
- Dembo, R. (1984). A Primal Truncated Newton Algorithm with Applications to Large-Scale Nonlinear Network Optimization. Yale School of Organization and Management Working Paper, Series B Number 72.
- Elmaghraby, S. E. (1978). The Economic Lot Scheduling Problem (ELSP) Review and Extension. Management Science 24, 587-598.
- Erickson, R. E., C. L. Monma, and A. F. Veinott, Jr. (1980). Minimum-Concave-Cost Network Flows. Revised August 1981. Unpublished Manuscript.
- Ferreira, A. C. and T. J. Hodgson (1973). An N-Product, Multi-Machine, Lot Size Scheduling Problem. AIIE Transactions 5, 237-244.
- Glover, F., D. Klingman, J. Mote, and D. Whitman (1980). An Extended Abstract of an Indepth Algorithmic and Computational Study for Maximum Flow Problems. Discrete Applied Mathematics 2, 251-254.
- Graves, S. C. and L. B. Schwarz (1977). Single Cycle Continuous Review Policies for Arborescent Production/Inventory Systems. Management Science 23, 529-540.
- Hodgson, T. J., M. Lambrecht, and J. Vander Eecken (1982). Production Lot Sizing and the Power of 2. Research Report No. 82-8, Department of Industrial and Systems Engineering, University of Florida.
- Hu, T. C. (1982). Combinatorial Algorithms, Addison-Wesley Publishing Company, Reading, Mass., pp. 46-60.
- Jackson, P. L., W. L. Maxwell, and J. A. Muckstadt (1983). The Joint Replenishment Problem with a Powers of Two Restriction. Technical Report No. 579, School of Operations Research and Industrial Engineering, Cornell University.
- Jensen, P. A. and H. A. Khan (1972). Scheduling in a Multistage Production System with Set-up and Inventory Costs. AIIE Transactions 4, 126-133.
- Kalyon, B. (1970). A Decomposition Algorithm for Arborescence Inventory Systems. Operations Research 20, 860-873.

- Karazanov, A. V. (1974). Determining the Maximum Flow in a Network by the Method of Preflows. Soviet Math. Dokl. 15, 434-437.
- Kruskal, J. B. (1964a). Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis. Psychometrika 29, 1-27.
- Kruskal, J. B. (1964b). Nonmetric Multidimensional Scaling: A Numerical Method. Psychometrika 29, 115-129.
- Lawler, E. L. (1978). Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints. Annals of Discrete Mathematics 2, 75-90.
- Love, S. (1972). A Facilities in Series Inventory Model with Nested Schedules. Management Science 18, 327-338.
- Malhotra, V. M., M. Pramodh Kumar, and S. N. Maheshwari (1978). An $O(v^3)$ Algorithm for Finding the Maximum Flow in a Network. Information Processing Letters 7, 277-278.
- Maxwell, W. L. and J. A. Muckstadt (1983). Establishing Consistent and Realistic Reorder Intervals in Production-Distribution Systems. Technical Report No. 561, School of Operations Research and Industrial Engineering, Cornell University.
- Maxwell, W. L. and H. Singh (1981). On Restricting Cycle Times in the Economic Lot Sizing Scheduling Problem. Technical Report No. 561, School of Operations Research and Industrial Engineering, Cornell University.
- Rockafeller, R. T. (1970). Convex Analysis, Princeton University Press, Princeton, New Jersey, 307-326.
- Roundy, R. O. (1984). 98%-Effective Integer-Ratio Lot-Sizing for One-Warehouse Multi-Retailer Systems. Chapter II herein.
- Schwarz, L. B. (1973). A Simple Continuous Review Deterministic One-Warehouse N-Retailer Inventory Problem. Management Science 19, 555-566.
- Schwarz, L. B. and L. Schrage (1975). Optimal and System-Myopic Policies for Multi-Echelon Production/Inventory Assembly Systems. Management Science 21, 1285-1294.
- Sidney, J. B. (1975). Decomposition Algorithms for Single-Machine Sequencing with Precedence Relations and Deferral Costs. Operations Research 23, 283-298.
- Singer, H. M. (1978). An Analysis of One Warehouse, N Retailer Production Inventory Systems. Technical Report No. 561, School of Operations Research and Industrial Engineering, Cornell University.

- Starr, M. K. (1972).** Production Management, Systems and Synthesis, Second Edition, Prentice-Hall, Englewood Cliffs, New Jersey, page 282.
- Szendrovits, A. Z. (1975).** Manufacturing Cycle Time Determination for a Multi-Stage Economic Production Quantity Model. Management Science 22, 298-308.
- Taha, H. A. and R. W. Skieth (1970).** The Economic Lot Sizes in Multi-stage Production Systems. AIIE Transactions 2, 157-162.
- Thompson, W. A. (1962).** The Problem of Negative Estimates of Variance Components. Annals of Mathematical Statistics 33, 273-289.
- Veinott, A. F., Jr. (1969).** Minimum Concave-cost Solution of Leontief Substitution Models of Multifacility Inventory Systems. Operations Research 17, 262-291.
- Veinott, A. F., Jr. (1971).** Least d-Majorized Network Flows with Inventory and Statistical Applications. Management Science 17, 547-567.
- 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. (1983).** A Hybrid Algorithm for Simultaneous Scheduling of Production and Distribution in Multi-Echelon Structures. Management Science 29, 77-105.
- Yen, J. Y. (1970).** An Algorithm for Finding Shortest Routes from All Sources to a Given Destination in a General Network. Quarterly of Applied Mathematics 27, 526-530.
- Zangwill, W. I. (1966).** A Deterministic Multiproduct Multifacility Production and Inventory Model. Operations Research 14, 486-507.
- Zangwill, W. I. (1968).** Minimum Concave-Cost Flows in Certain Networks. Management Science 14, 429-450.
- 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 19, 506-527.