MORE RESEARCH REPORT 06.03

Modeling, Optimization Research and Education (MORE) Institute
Systems & Industrial Engineering
University of Arizona, Tucson, AZ

Forthcoming in *Operations Research Letters*

ON STOCHASTIC LOT-SIZING PROBLEMS WITH RANDOM LEAD TIMES

KAI HUANG AND SİMGE KÜÇÜKYAVUZ

Abstract We give multi-stage stochastic programming formulations for lot-sizing problems where costs, demands and order lead times follow a general discrete-time stochastic process with finite support. We characterize the properties of an optimal solution and give a dynamic programming algorithm, polynomial in the scenario tree size, when orders do not cross in time.

Key words: stochastic programming, lot-sizing, semi-Wagner-Whitin property, polynomial algorithm, random lead times.

1. Introduction

In this paper, we consider lot-sizing problems where the costs, demands and order lead times follow a discrete-time stochastic process with finite probability space. Traditionally, the methods developed for such problems assume that the underlying stochastic process is stationary and homogeneous and/or that the costs, demands and lead times are independent random variables [4, 5, 10]. In this paper, we assume that the stochastic process is very general, i.e., cost, demand and lead time distributions are non-stationary and are correlated. For example, consider a manufacturer of a seasonal product. If the lead time is large, then the demand from customers is usually smaller since the customers may be lost to competitors whose lead times are smaller. We use a multi-stage stochastic program to model this problem. Utilizing the property of an optimal solution we propose a dynamic programming (DP) algorithm that is polynomial in the size of the scenario tree for the case where orders do not cross in time.

Relevant literature. The deterministic lot-sizing problem and its extensions have been studied extensively in the literature. Wagner and Whitin [16] characterize the properties of an optimal solution to the basic deterministic uncapacitated lot-sizing problem, known as the Wagner-Whitin property. They also give an $\mathcal{O}(T^2)$ algorithm for the deterministic lot-sizing problem, where T is the number of time periods, which is later improved to $\mathcal{O}(T \log T)$ [1, 6, 15].

In the inventory control literature, Kaplan [10] and Ehrhardt [5] consider periodic review policies for the stochastic inventory control problem with time-independent demand distribution and a particular stationary lead time distribution with no order crossing in time. Liberatore

Date: Submitted September 18, 2006. Revised July 23, 2007. Accepted October 19, 2007.

Kai Huang: Department of Operations, Planning, Accounting and Control, Department of Technology Management, Technology University of Eindhoven, P.O. Box 513, 5600MB, Eindhoven, The Netherlands `k.huang@tue.nl`.
Simge Küçükyavuz: Department of Systems and Industrial Engineering, University of Arizona, P.O. Box 210020, Tucson, AZ 85721-0020, USA `simge@sie.arizona.edu`.

[11], Nevison and Burstein [14] and Alp et al. [3] give algorithms for dynamic lot/batch sizing with deterministic demands and stochastic lead times when orders do not cross in time. In particular, Nevison and Burstein [14] show that for the lot-sizing problem with deterministic dynamic demand and stochastic non-stationary lead times, each positive order in an optimal solution exactly satisfies demands of a sequence of consecutive periods when orders do not cross in time. This is a natural extension of the Wagner-Whitin property. In this paper, we also let demands be discrete non-stationary random variables that are possibly correlated with lead times. In this case, a simple extension of the Wagner-Whitin property does not hold anymore.

In the stochastic programming literature, Ahmed et al. [2] study the *stochastic lot-sizing problem* in which demand follows a stochastic process described by a scenario tree. They show that the Wagner-Whitin property does not hold for the stochastic case. Guan et al. [7] propose a branch-and-cut algorithm to solve the stochastic lot-sizing problem with zero lead times. Lulli and Sen [12] give a branch-and-price algorithm for the stochastic batch-sizing problem. In an independent work, Guan and Miller [8] study stochastic lot sizing with zero lead times and give a dynamic programming algorithm that is polynomial in the scenario tree size using a different analysis.

Outline. In Section 2 we formally define the stochastic lot-sizing problem with random lead times. We first study the special case of stochastic lot-sizing with zero lead times in Section 3, which is interesting in its own right. In Section 4 we generalize our results in presence of non-zero lead times. In Section 4.1 we show that an important property of optimal solutions that holds for deterministic lot sizing with dynamic lead times, but does not hold for stochastic lot sizing with random lead times. In Section 4.2 we characterize the property of optimal solutions for the stochastic problem called the semi-Wagner-Whitin property, a stochastic counterpart of the Wagner-Whitin property. Using this property we propose a dynamic programming algorithm that runs in polynomial time in input size, given by the number of nodes in the scenario tree.

2. Stochastic lot-sizing problem with random lead times

Given the finite-support discrete-time stochastic process describing the costs, demands and order lead times over a finite horizon, the (uncapacitated) *stochastic lot-sizing problem with random lead times* (RLTSLSP) is to determine the order and inventory quantities in each period so that demand for a single product in each time period is met over a finite horizon and the expected total cost of order and holding over the horizon is minimized.

RLTSLSP can be modelled as a multi-stage stochastic program in which the stochastic process is described by a scenario tree. Let the length of the planning horizon be T. Then the scenario tree, $\mathcal{T} = (\mathcal{V}, \mathcal{A})$, is given by a directed layered tree with T layers, where each layer corresponds to a stage (period) of the multi-stage stochastic program. Let $N = |\mathcal{V}|$. The tree is rooted at node 1. A node $n \in \mathcal{V}$ in layer t_n corresponds to a realization of the stochastic process at period t_n and the probability of this realization is p_n. For $n \in \mathcal{V} \setminus \{1\}$, let a_n be the unique predecessor of node n. Clearly, a_n lies in layer $t_n - 1$. Let $\mathcal{L}$ be the set of leaf nodes of the tree. Then a path from the root node to node $n \in \mathcal{L}$ represents a scenario (a realization of the process over the entire horizon). Let $\mathcal{T}_n = (\mathcal{V}_n, \mathcal{A}_n)$ be the directed subtree rooted at $n \in \mathcal{V}$ and let $\mathcal{L}_n$ be the leaf nodes of $\mathcal{T}_n$. Also let $\mathcal{P}_n$ denote the nodes in the unique path from the root node to node n. Let $\mathcal{C}_n^k$ be the set of nodes that are at the $(k+1)$th layer of the subtree $\mathcal{T}_n$. In other words, the nodes in $\mathcal{C}_n^k$ are the kth generation children of node n and $\mathcal{C}_n^0 = \{n\}$. See Figure 1 for a representation of the notation.

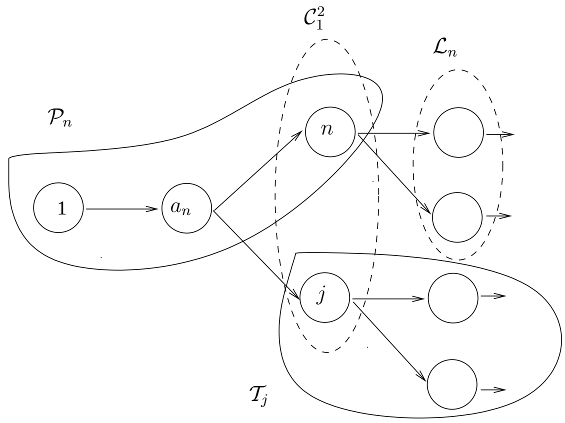

FIGURE 1. Scenario tree notation.

The stochastic data are given as follows:

d_n	:	demand ($d_n \geq 0$).
f_n	:	fixed order (setup) cost ($f_n \geq 0$).
c_n	:	variable order cost ($c_n \geq 0$).
h_n	:	variable inventory holding cost ($h_n \geq 0$).
ℓ_n	:	order lead time ($\ell_n \geq 0$, integer).

Note that the vector $(d_n, f_n, c_n, h_n, \ell_n)$ constitutes a complete realization of the stochastic process at node n. Therefore, all the descendants of n share the same information of n. For example, if $\ell_n > 0$ for some n and we order one unit at node n, then this one unit will arrive at *every* node $j \in \mathcal{C}_n^{\ell_n}$.

Let y_n denote the order quantity in time period t_n under realization n, and z_n denote the inventory quantity at the end of period t_n under realization n, respectively. Also let x_n be the fixed-charge variable for order in period t_n under realization n. Finally, let $\mathcal{R}_n = \{j \in \mathcal{P}_n : n \in \mathcal{C}_j^{\ell_j}\}$ be the set of all nodes in path $\mathcal{P}(n)$ whose orders arrive exactly at n. RLTSLSP can be formulated as:

$$\begin{array}{lll} \min & \sum_{n \in \mathcal{V}} p_n(f_n x_n + c_n y_n + h_n z_n) & \\ \text{s.t.} & z_{a_n} + \sum_{j \in \mathcal{R}_n} y_j = z_n + d_n & \text{for } n \in \mathcal{V} \\ & y_n \leq M_n x_n & \text{for } n \in \mathcal{V} \\ & x_n \in \{0,1\}, y_n, z_n \geq 0 & \text{for } n \in \mathcal{V} \\ & z_{a_1} = 0, & \end{array} \tag{1}$$

where M_n, $n \in \mathcal{V}$ is a large constant. In particular, $M_n = \max_{m \in \mathcal{L}_n}\{D_m\} - D_{a_n}$, where $D_n = \sum_{j \in \mathcal{P}_n} d_j$ represents the cumulative demand from the root node to node n. Note that the assumption of no initial inventory is made without loss of generality. Given a positive initial inventory level, we can construct an equivalent problem with no initial inventory by using this initial inventory to satisfy the demands in the first, second, ... layers until it is depleted.

3. Stochastic Lot-Sizing Problem with Zero Lead Times

To study problem (1), we start from the following basic (uncapacitated) *stochastic lot-sizing problem* (SLSP) with zero lead times, i.e. $\ell_n = 0$ for all $n \in \mathcal{V}$,

$$\begin{array}{lll} \min & \sum_{n\in\mathcal{V}} p_n(f_n x_n + c_n y_n + h_n z_n) & \\ \text{s.t.} & z_{a_n} + y_n = z_n + d_n & \text{for } n \in \mathcal{V} \\ & y_n \le M_n x_n & \text{for } n \in \mathcal{V} \\ & x_n \in \{0,1\}, y_n, z_n \ge 0 & \text{for } n \in \mathcal{V} \\ & z_{a_1} = 0. & \end{array} \tag{2}$$

3.1. **Semi-Wagner-Whitin property.** Deterministic basic uncapacitated lot sizing with zero lead times is a special case of SLSP with $|\mathcal{L}| = 1$. The Wagner-Whitin property [16] implies that there exists an optimal solution for deterministic basic uncapacitated lot sizing with zero lead times in which $z_{a_n} y_n = 0$ for all $n \in \mathcal{V}$. Unfortunately, Ahmed et al. [2] show that the Wagner-Whitin property does not hold for the stochastic case, when $|\mathcal{L}| > 1$. However, it is interesting that a relaxation of the Wagner-Whitin property still holds for certain segments of sample paths in a scenario tree. We call this property the semi-Wagner-Whitin property.

Proposition 1. [Semi-Wagner-Whitin property of SLSP]
There exists an optimal solution (x', y', z') of SLSP such that if $y'_n > 0$ for $n \in \mathcal{V}$, then

$$z'_{a_n} + y'_n = D_m - D_{a_n}, \tag{3}$$

for some $m \in \mathcal{V}_n$.

We skip the proof of this proposition, because we prove a more general form of it in the presence of lead times in Section 4. This property was first observed by [13] where it was called the production path property. It is also used in [9] to design an $\mathcal{O}(N^2)$ algorithm for a stochastic lot-sizing problem with no fixed charges.

An equivalent representation of Proposition 1 is as follows:

Corollary 2. *There exists an optimal solution (x^*, y^*, z^*) of problem (2) such that for any node $n \in \mathcal{V}$, the outgoing inventory z^*_n takes a value:*

$$D_m - D_n \quad \text{for some } m \in \mathcal{V} \text{ and } D_m \ge D_n. \tag{4}$$

Proof. We use an inductive argument on each layer, which first applies the semi-Wagner-Whitin property to the root node 1. We can choose not to order (possible only if $D_1 = 0$) or to order. If we order, then we must have $y_1 = D_m$ for some $m \in \mathcal{V}_1 = \mathcal{V}$ and we have $z_1 = D_m - D_1$. If we do not order, then we have $z_1 = z_{a_1} - D_1 = 0$. In other words, z_1 can only take at most N values, each value corresponding to a node in the scenario tree. Now consider any node n such that $z^*_{a_n} = D_i - D_{a_n}$ for some $i \in \mathcal{V}$ and $D_i - D_{a_n} \ge 0$. If $y^*_n = 0$, then according to feasibility, $D_i - D_{a_n} \ge d_n$ and $z^*_n = D_i - D_{a_n} - d_n = D_i - D_n$. Otherwise, $y^*_n > 0$ and according to Proposition 1, there exists $m \in \mathcal{V}_n$ such that $z^*_{a_n} + y^*_n = D_m - D_{a_n}$. Then $z^*_n = z^*_{a_n} + y^*_n - d_n = D_m - D_n \ge 0$. □

Note that in (4) when $y^*_n = 0$, we may have $m \in \mathcal{V} \setminus \mathcal{V}_n$, but when $y^*_n > 0$ we must have $m \in \mathcal{V}_n$ according to Proposition 1. Next we describe an efficient dynamic programming algorithm that makes use of Proposition 1 and Corollary 2.

3.2. **DP algorithm for Stochastic Lot Sizing with Zero Lead Times.** To facilitate future discussion, we re-index the nodes $n \in \mathcal{V}$ according to an increasing order of their cumulative demands so that $D_1 \le D_2 \le \cdots \le D_N$. In this indexing, for $i \in \mathcal{P}_{a_j}$ with $D_i = D_j$ we let $i < j$. Therefore, node 1 is still the root node. Throughout we let $b^+ = \max\{0, b\}$.

With this indexing, we define the value function at node n as $v_n([k]) : \mathcal{V} \to \mathbb{R}$, where $[k]$ denotes that the incoming inventory at node n is $(D_k - D_{a_n})^+$, i.e., the sum of all orders placed in $\mathcal{P}_{a_n}$ exactly satisfies the cumulative demand in node $k \in \mathcal{V}$. (From Proposition 1, k exists.)

Note that for feasibility k must take a value such that $D_k \geq D_{a_n}$. The function $v_n([k])$ gives the optimal objective value of the subproblem of (2) with respect to subtree $\mathcal{T}_n$, given the incoming inventory at n is $(D_k - D_{a_n})^+$. Note, again from the semi-Wagner-Whitin property, that there exists an optimal solution in which the order quantity at node n takes a value $(D_j - D_k)^+$ for some $j \in \mathcal{V}_n \cup \{k\}$.

For $i, n \in V$ and $j \in \mathcal{V}_n \cup \{i\}$ such that $j \geq i \geq a_n$ we define a function $g_n([i],[j]) : \mathcal{V}^2 \to \mathbb{R}$ that gives the objective value of the subproblem corresponding to subtree $\mathcal{T}_n$ when the incoming inventory is $z_{a_n} = D_i - D_{a_n}$ and the order decision is $y_n = D_j - D_i$. We let $j = i$, to represent not ordering in n. If $y_n > 0$, then we must have $j > i$ and the outgoing inventory $z_n = z_{a_n} + y_n - d_n = (D_i - D_{a_n}) + (D_j - D_i) - d_n = D_j - D_n$. On the other hand if $y_n = 0$, then we have $j = i$ and $z_n = z_{a_n} + y_n - d_n = (D_i - D_{a_n}) - d_n = D_i - D_n$. Hence if $D_j > D_i$,

$$g_n([i],[j]) = p_n f_n + p_n c_n (D_j - D_i) + p_n h_n (D_j - D_n) + \textstyle\sum_{m \in \mathcal{C}_n^1} v_m([j]), \tag{5}$$

and if $D_j = D_i$,

$$g_n([i],[j]) = p_n h_n (D_i - D_n) + \textstyle\sum_{m \in \mathcal{C}_n^1} v_m([i]). \tag{6}$$

Note that for $n \in \mathcal{L}$, we have $\mathcal{C}_n^1 = \emptyset$ and so $\sum_{m \in \mathcal{C}_n^1} v_m([i]) \equiv 0$ in equations (5)–(6). Given $g_n([\cdot],[\cdot])$, the function $v_n([k])$ is

$$v_n([k]) = \min_{j \in \mathcal{V}_n \cup \{k\} : j \geq \max\{k,n\}} \{g_n([k],[j])\}. \tag{7}$$

The optimal solution of (2) is given by $v_1([0])$, where $[0]$ denotes the 0 incoming inventory at root node 1. (See Algorithm 1 for the description of the dynamic programming algorithm.)

Algorithm 1 A dynamic programming algorithm for SLSP

Initialization: Re-index nodes according to their cumulative demands so that $D_1 \leq D_2 \leq \cdots \leq D_N$. In this indexing, if $i \in \mathcal{P}_{a_j}$ and $D_i = D_j$, then $i < j$. Set $n = N$.
while $n \geq 1$ **do**
  Let $k = a_n$
  while $k \leq N$ **do**
    Let $j = \max\{k, n\}$.
    while $j \leq N$ **do**
      if $j \in \mathcal{V}_n \cup \{k\}$ **then**
        Calculate $g_n([k],[j])$ from (5) and (6).
      end if
      $j \leftarrow j + 1$.
    end while
    Calculate $v_n([k])$ from (7).
    $k \leftarrow k + 1$.
  end while
  $n \leftarrow n - 1$.
end while
return $v_1([0])$.

Theorem 3. *SLSP can be solved in $\mathcal{O}(BN^3)$ time, where B is the maximum number of branches of non-leaf nodes, i.e., $B = \max_{n \in \mathcal{V}} |\mathcal{C}_n^1|$.*

Proof. For a given $n \in \mathcal{V}$ and $k \in \mathcal{V}$ with $k \geq a_n$, we calculate $g_n([k],[j])$ according to (5) and (6) for a given $j \in \mathcal{V}_n \cup \{k\}$. This calculation can be done in $\mathcal{O}(B)$ time. As this calculation is done for each n, k and j, the complexity of Algorithm 1 is $\mathcal{O}(BN^3)$. Note that the calculation of $v_n([k])$ in equation (7) will not add more complexity, because this calculation can be incorporated while computing $g_n([k],[j])$.

□

In an independent work, Guan and Miller [8] fully characterize a related value function $f_n(z_{a_n}) : \mathbb{R} \to \mathbb{R}$ for any incoming inventory to node n, $z_{a_n} \geq 0$ and give an $\mathcal{O}(\log BN^3)$ algorithm for this problem. Here, we compute the value function $v_n([k])$ for at most N values of k.

4. DP Algorithm for Stochastic Lot Sizing with Random Lead Times

4.1. **Deterministic Lot Sizing with Dynamic Demands and Lead Times.** Consider the special case of RLTSLSP where $|\mathcal{L}| = 1$. Hence $T = N$, $\mathcal{V} = \{1, 2, \ldots, T\}$, $a_j = j - 1$ and $t_j = j$ for all $j \in \mathcal{V}$, and $C_j^{\ell_j} = \{j + \ell_j\}$. We call this problem the *deterministic uncapacitated lot-sizing problem with lead times* (DLSPLT). We state the following proposition without proof as it is a trivial extension of the Wagner-Whitin Property for deterministic lead times.

Proposition 4. [Wagner-Whitin Property for DLSPLT] *There exists an optimal solution (x', y', z') of DLSPLT such that if $y'_k > 0$ for $k \in \mathcal{V}$, and $n = k + \ell_k$, then*

(i) *For some $m \in \mathcal{V}_n$*
$$z'_{n-1} + \sum_{j \in \mathcal{R}_n} y'_j = D_m - D_{n-1}, \tag{8}$$

(ii) *and*
$$k = \text{argmin}_{j \in \mathcal{R}_n}\{f_j + c_j(D_m - D_{n-1})\} \text{ and } y'_j = 0 \text{ for all } j \in \mathcal{R}_n \setminus \{k\}.$$

From Proposition 4, the dynamic program given in [16] for deterministic lot sizing with zero lead times can easily be extended to solve DLSPLT efficiently. Recall that the Wagner-Whitin property (i) in Proposition 4 does not hold in the stochastic case even when all lead times are zeroes. In the next section, we show that property (ii) in Proposition 4 also does not hold in the stochastic case. As before, we assume that the nodes of the scenario tree are re-indexed so that $D_1 \leq D_2 \leq \cdots \leq D_N$ and for $i \in \mathcal{P}_{a_j}$ with $D_i = D_j$ we let $i < j$.

4.2. **Semi-Wagner-Whitin Property for RLTSLSP.**

Example 1. Consider the example in Figure 2 in which the node numbers also represent the demands at these nodes. Note that the nodes are also numbered in increasing cumulative demands from the root node. The only random event is in the third stage where there are two possible outcomes with equal probability. Suppose that the lead time vector is $\ell = (0, 1, 2, 0, 1, 1)$, the fixed cost vector is $f = (100, 1, 100, 0, 5, 5)$ and the variable cost vector is $c = (1, 1, 1, 0, 1, 1)$. Suppose that all holding costs are one. As the lead time for node 2 is one, the order placed in node 2 arrives in nodes 3 and 4. We depict the order arrivals by the non-tree arcs. The optimal solution is $y = (3, 8, 0, 2, 0, 0)$ with a total expected cost of 119.5. Note $y_2, y_4 > 0$ in the unique optimal solution, and property (ii) in Proposition 4 is violated in this stochastic case.

Throughout this section, we assume that there is *no order crossing in time.* In other words, an order placed with the supplier at node j in period t_j does not arrive later than another order placed at node $k \in \mathcal{V}_j \setminus \{j\}$, i.e., $t_j + \ell_j \leq t_k + \ell_k$ for all $k \in \mathcal{V}_j \setminus \{j\}$. Note that when there are zero lead times, this assumption is trivially satisfied. This is a realistic assumption in practice when there is a single supplier. This assumption is also necessary in the periodic review policies proposed in [5, 10] when order lead times in every period are identically distributed and are independent of demands.

Proposition 5. [Semi-Wagner-Whitin Property for RLTSLSP] *There exists an optimal solution (x', y', z') of RLTSLSP such that if $y'_n > 0$ for $n \in \mathcal{V}$, then for some $i \in \mathcal{C}_n^{\ell_n}$,*
$$z'_{a_i} + \sum_{j \in \mathcal{R}_i \cap \mathcal{P}_n} y'_j = D_m - D_{a_i}, \tag{9}$$
for some $m \in \mathcal{V}_i$, when orders do not cross in time.

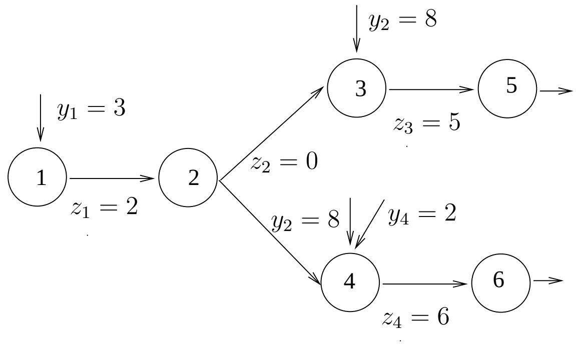

FIGURE 2. Stochastic lot-sizing example.

Proof. Let (x^*, y^*, z^*) be an optimal solution to RLTSLSP, and n be any node such that $y_n^* > 0$. We first claim that there exists $m \in \mathcal{C}_n^{\ell_n}$, such that $y_k^* = 0$ for all $k \in (\mathcal{P}_m \setminus \mathcal{P}_n) \cap \mathcal{R}_m$. If this is not the case, i.e., for each $m \in \mathcal{C}_n^{\ell_n}$ there exists some $k_m \in (\mathcal{P}_m \setminus \mathcal{P}_n) \cap \mathcal{R}_m$ with $y_{k_m}^* > 0$, then we compare two costs $\alpha = p_n c_n$ and $\beta = \sum_{m \in \mathcal{M}} p_{k_m} c_{k_m}$, where $\mathcal{M} \subset \mathcal{C}_n^{\ell_n}$ such that $\cup_{m \in \mathcal{M}} \mathcal{C}_{k_m}^{\ell_{k_m}} = \mathcal{C}_n^{\ell_n}$, and for all $i, j \in \mathcal{M}$, $i \neq j$, $\mathcal{C}_{k_i}^{\ell_{k_i}} \cap \mathcal{C}_{k_j}^{\ell_{k_j}} = \emptyset$. If $\alpha \geq \beta$, then we can eliminate the ordering at node n and obtain a new solution (x', y', z') such that $y'_n = 0$ and $y'_{k_m} = y^*_{k_m} + y^*_n$ (for all $m \in \mathcal{M}$) with nonincreasing objective. If $\alpha < \beta$, then we can eliminate the ordering at a specific node k_u (in fact, $u = \text{argmin}_{m \in \mathcal{M}} y^*_{k_m}$) and obtain a new solution (x', y', z') such that $y'_n = y^*_n + y^*_{k_u}$ with decreasing objective. This procedure is repeated until our claim holds.

In order to ease the exposition, we add one more layer of dummy nodes to the original scenario tree. One dummy node is added as the descendent of each leaf node of $\mathcal{T}$. Let $\mathcal{L}' = \{i' : i \in \mathcal{L}\}$ and define one dummy arc (i, i') for every $i \in \mathcal{L}$. Each dummy node i' has 0 demand and costs. For $i \in \mathcal{V}$ with $t_i + \ell_i > T$, let $\mathcal{C}_i^{\ell_i} = \mathcal{L}'_i$, where $\mathcal{L}'_i = \{j' : j \in \mathcal{L}_i\}$.

Now for any $m \in \mathcal{V}_n \setminus \{n\}$, define $\mathcal{H}_n(\{m\}) = (\cup_{q \in \mathcal{C}_m^{\ell_m}} \mathcal{P}_{a_q}) \setminus (\cup_{u \in \mathcal{C}_n^{\ell_n}} \mathcal{P}_{a_u})$, which includes all the nodes such that order placed at node n has arrived, while order placed at node m (if any) has not yet arrived. Similarly, for any $\mathcal{M} \subset \mathcal{V}_n \setminus \{n\}$, we define $\mathcal{H}_n(\mathcal{M}) = \cup_{m \in \mathcal{M}} \mathcal{H}_n(\{m\})$. Let $\mathcal{B}_n := \{m \in \mathcal{V}_n \setminus \{n\} : y_m^* > 0, \quad y_i^* = 0 \quad \forall i \in \mathcal{P}_{a_m} \setminus \mathcal{P}_n\} \cup \{m \in \mathcal{L}'_n : y_i^* = 0 \quad \forall i \in \mathcal{P}_{a_m} \setminus \mathcal{P}_n\}$. Then

$$\mathcal{H}_n(\mathcal{B}_n) = \cup_{m \in \mathcal{B}_n} [(\cup_{q \in \mathcal{C}_m^{\ell_m}} \mathcal{P}_{a_q}) \setminus (\cup_{u \in \mathcal{C}_n^{\ell_n}} \mathcal{P}_{a_u})]$$

represents the set of nodes whose demands must be covered by orders placed in $\mathcal{P}_n$, as the orders placed later (in $\mathcal{B}_n$) will not arrive on time to satisfy these demands. (See Figure 3 for an illustration of $\mathcal{H}_n(\mathcal{B}_n)$ where $\mathcal{B}_n = \{n_1, n_2\} = \mathcal{C}_n^{\ell_n} = \mathcal{C}_n^1$.) Note that according to the claim in the first paragraph of the proof, $\mathcal{H}_n(\mathcal{B}_n)$ is *nonempty*. Clearly, for any $m \in \mathcal{H}_n(\mathcal{B}_n) \cap \mathcal{V}_i$ where $i \in \mathcal{C}_n^{\ell_n}$, we must have $z'_{a_i} + \sum_{j \in \mathcal{R}_i \cap \mathcal{P}_n} y'_j \geq D_m - D_{a_i}$ by feasibility. Assume (9) does not hold, then $z'_{a_i} + \sum_{j \in \mathcal{R}_i \cap \mathcal{P}_n} y'_j > D_m - D_{a_i}$ for all such m, i.e., $z_m^* > 0$.

Next, we define the following costs: $\alpha = p_n c_n + \sum_{i \in \mathcal{H}_n(\mathcal{B}_n)} p_i h_i$ and $\beta = \sum_{m \in \mathcal{B}_n \setminus \mathcal{L}'_n} p_m c_m$, where α is the sum of the expected variable cost of ordering one unit at node n, plus the expected holding cost of carrying it as inventory at all nodes $m \in \mathcal{H}_n(\mathcal{B}_n)$ and β is the expected total variable cost of ordering one unit at each node $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$ (when $\mathcal{B}_n \setminus \mathcal{L}'_n = \emptyset$ we have $\beta = 0$).

On the one hand, if $\alpha \geq \beta$, then we can shift a positive amount of ordering from node n to nodes $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$. Let $\delta = \min\{y_n^*, \min_{j \in \mathcal{H}_n(\mathcal{B}_n)} z_j^*\} > 0$. Let $y'_n = y_n^* - \delta$, $y'_m = y_m^* + \delta$ for all $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$ and $z'_k = z_k^* - \delta$ for all $k \in \mathcal{H}_n(\mathcal{B}_n)$. All other order and inventory values are unchanged. Clearly, if $\delta = y_n^*$, then $y'_n = 0$, and there is no need to check (9) at node n anymore. Otherwise, $z'_{a_i} + \sum_{j \in \mathcal{R}_i \cap \mathcal{P}_n} y'_j = (z^*_{a_i} + \sum_{j \in \mathcal{R}_i \cap \mathcal{P}_n} y^*_j) - \delta = (D_m - D_{a_i} + \delta) - \delta = D_m - D_{a_i}$,

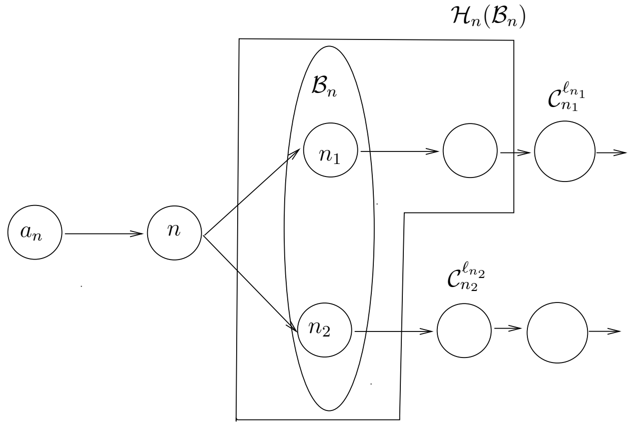

FIGURE 3. Illustration of $\mathcal{H}_n(\mathcal{B}_n)$ for an instance with $\ell_{n_1} = 2$ and $\ell_{n_2} = 1$.

where $m = \text{argmin}_{j\in\mathcal{H}_n(\mathcal{B}_n)}\{z_j^*\}$ for $i \in \mathcal{C}_n^{\ell_n}$ such that $i \in \mathcal{P}_m$. Hence, (9) holds at node n. Note that in this shifting operation, the objective value is nonincreasing.

On the other hand, if $\alpha < \beta$, then we can shift a positive amount of ordering from nodes $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$ to n. Let $\delta = \min_{m\in\mathcal{B}_n\setminus\mathcal{L}'_n} y_m^* > 0$, then we let $y'_n = y_n^* + \delta$, $y'_m = y_m^* - \delta$ for all $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$ and $z'_k = z_k^* + \delta$ for all $k \in \mathcal{H}_n(\mathcal{B}_n)$. All other order and inventory values are unchanged. Note for the choice of n as the node violating the semi-Wagner-Whitin property, we can start from the largest index and follow a decreasing order. Then (9) holds for all $m \in \mathcal{B}_n \setminus \mathcal{L}'_n$. In particular, for $w = \text{argmin}_{m\in\mathcal{B}_n\setminus\mathcal{L}'_n}\{y_m^*\}$ we know that for some $i \in \mathcal{C}_w^{\ell_w}$ and $s \in \mathcal{V}_i$:

$$z_{a_i}^* + \sum_{j\in\mathcal{R}_i\cap\mathcal{P}_w} y_j^* = D_s - D_{a_i}. \tag{10}$$

Then for $u \in C_n^{\ell_n} \cap \mathcal{P}_s$ we have $z'_{a_u} + \sum_{j\in\mathcal{R}_u\cap\mathcal{P}_n} y'_j = z_{a_u}^* + \sum_{j\in(\mathcal{R}_u\cap\mathcal{P}_n)} y_j^* + \delta = (z_{a_u}^* + \sum_{j\in(\mathcal{R}_u\cap\mathcal{P}_n)} y_j^*) + y_w^*$. If $i = u$, then $(z_{a_u}^* + \sum_{j\in(\mathcal{R}_u\cap\mathcal{P}_n)} y_j^*) + y_w^* = z_{a_i}^* + \sum_{j\in\mathcal{R}_i\cap\mathcal{P}_w} y_j^*$ follows because in this case $(\mathcal{R}_u \cap \mathcal{P}_n) \cup \{w\} = \mathcal{R}_i \cap \mathcal{P}_w$. If $u \in \mathcal{P}_{a_i}$, then

$$\begin{aligned}(z_{a_u}^* + \sum_{j\in(\mathcal{R}_u\cap\mathcal{P}_n)} y_j^*) + y_w^* &= [z_{a_u}^* + \sum_{j\in(\mathcal{R}_u\cap\mathcal{P}_n)} y_j^* - (D_{a_i} - D_{a_u})] + y_w^* + (D_{a_i} - D_{a_u})\\ &= z_{a_i}^* + \sum_{j\in\mathcal{R}_i\cap\mathcal{P}_w} y_j^* + D_{a_i} - D_{a_u} \qquad (11)\\ &= D_s - D_{a_i} + D_{a_i} - D_{a_u} (= D_s - D_{a_u}). \qquad (12)\end{aligned}$$

Equality (11) follows from the fact that $\sum_{j\in\mathcal{R}_i\cap\mathcal{P}_w} y_j^* = y_w^*$ since $w \in \mathcal{B}_n \setminus \mathcal{L}'_n$, and equality (12) follows from (10). Therefore, (9) holds for node n for the new solution (x', y', z'). Moreover, the objective value of the new solution is decreasing. We repeat this argument until (9) holds for every node in the scenario tree. □

Remark 1. An equivalent representation of Proposition 5 is that there exists an optimal solution (x', y', z') of RLTSLSP such that if $y'_k > 0$ for $k \in \mathcal{V}$, then for some $i \in \mathcal{C}_k^{\ell_k}$,

$$\sum_{j\in\mathcal{P}_{a_k}} y'_j + y'_k = D_m, \tag{13}$$

for some $m \in \mathcal{V}_i$, when the orders do not cross in time.

4.3. **Algorithm Design.** We use a similar notation to Section 3.2 with the following changes. We redefine the value function at node n as $v_n([k]) : \mathcal{V} \to \mathbb{R}$, where $[k]$ denotes that the total of orders placed in nodes in $\mathcal{P}_{a_n}$ is exactly equal to the total demand D_k for some node $k \in \mathcal{V}$, i.e., $\sum_{j\in\mathcal{P}_{a_n}} y_j = D_k$. (From Proposition 5, k exists.) Note that in this redefinition $[k]$ does not represent incoming inventory to node n as in the zero lead time case, because in the presence of order lead times, some of the earlier orders may arrive after period t_n, but not after period $t_n + \ell_n$ due to the no order crossing assumption. The function $v_n([k])$ gives the optimal objective value of the subproblem of (1) with respect to node n, given that the total earlier order commitments in nodes $\mathcal{P}_{a_n}$ is D_k. Note that for a given $k \in \mathcal{V}$, if we have $D_k < D_{a_j}$ for some $j \in \mathcal{C}_n^{\ell_n}$, then the subproblem at node n is infeasible, because there could be no other order arrivals to satisfy the demand in a_j fully. Therefore, to ensure feasible solutions we only need to consider k such that $D_k \geq D_{a_j}$ for all $j \in \mathcal{C}_n^{\ell_n}$. Note, again, from Proposition 5 that there exists an optimal solution in which the order quantity at node n takes a value $(D_j - D_k)^+$ for some $j \in \bar{\mathcal{V}}_n \cup \{k\}$, where $\bar{\mathcal{V}}_n = \mathcal{V}_n \setminus (\cup_{m\in\mathcal{C}_n^{\ell_n}} \mathcal{P}_{a_m})$ represents the set of all successors of node n whose demands could be satisfied from an order placed at node n.

For $n \in \mathcal{V}$, let $\mathcal{H}_n(\mathcal{C}_n^1) = \cup_{m\in\mathcal{C}_n^1}[(\cup_{q\in\mathcal{C}_m^{\ell_m}} \mathcal{P}_{a_q}) \setminus (\cup_{i\in\mathcal{C}_n^{\ell_n}} \mathcal{P}_{a_i})]$ represent the set of nodes whose demands must be covered by orders placed in nodes in $\mathcal{P}_n$. For $i, n \in V$ and $j \in \bar{\mathcal{V}}_n \cup \{i\}$ such that $D_j \geq D_k$ for all $k \in \mathcal{H}_n(\mathcal{C}_n^1)$ and $D_i \geq D_{a_u}$ for all $u \in \mathcal{C}_n^{\ell_n}$ we redefine a function $g_n([i], [j]) : \mathcal{V}^2 \to \mathbb{R}$ that gives the objective value of the subproblem corresponding to subtree $\mathcal{T}_n$ when the total orders placed in $\mathcal{P}_{a_n}$ is D_i and the order decision is $y_n = (D_j - D_i)$. (We let $j = i$, to represent not ordering in n.)

For $n \in \mathcal{V}$ such that $t_n + \ell_n \leq T$, if $y_n > 0$, then we must have $j > i$ and $D_j \geq D_k$ for all $k \in \mathcal{H}_n(\mathcal{C}_n^1)$, and

$$g_n([i], [j]) = p_n f_n + p_n c_n (D_j - D_i) + \sum_{m\in\mathcal{C}_n^1} v_m([j]) + \sum_{k\in\mathcal{H}_n(\mathcal{C}_n^1)} p_k h_k (D_j - D_k), \tag{14}$$

and if $j = i$,

$$g_n([i], [j]) = \sum_{k\in\mathcal{H}_n(\mathcal{C}_n^1)} p_k h_k (D_i - D_k) + \sum_{m\in\mathcal{C}_n^1} v_m([i]). \tag{15}$$

For $n \in \mathcal{V}$ such that $t_n + \ell_n > T$, we must have $y_n = 0$ in every optimal solution, therefore, we let $j = i$ and

$$g_n([i], [j]) = 0. \tag{16}$$

Given $g_n([\cdot], [\cdot])$, the function $v_n([k])$ for $k \in \mathcal{V}$ with $D_k \geq D_{a_u}$ for all $u \in \mathcal{C}_n^{\ell_n}$ is:

$$v_n([k]) = \min_{j\in\bar{\mathcal{V}}_n\cup\{k\}: j\geq\max\{k,\bar{n}\}} \{g_n([k], [j])\}, \tag{17}$$

where $\bar{n} = \max\{j \in \mathcal{H}_n(\mathcal{C}_n^1)\}$. By a backward recursion starting from the leaf nodes, $\mathcal{L}$, we calculate the optimal solution of RLTSLSP given by $v_1([0])$, where $[0]$ denotes the incoming inventory at the root node 1. Note that we can let $z_{a_1} = 0$, without loss of generality. (See Algorithm 2 for the description of the dynamic programming algorithm.)

We illustrate this dynamic programming algorithm by applying it to Example 1.

Example 1 (cont.) In the following, we only calculate feasible $g([i], [j])$. Starting from the leaf nodes, we get $g_6([6], [6]) = v_6([6]) = 0$, $v_5([5]) = g_5([5], [5]) = v_5([6]) = g_5([6], [6]) = 0$, because $t_6 + \ell_6 = t_5 + \ell_5 = 5 > 4 = T$. Similarly, $v_3([5]) = v_3([6]) = 0$. Next, we calculate $v_4([\cdot])$. Observe that, while calculating $g_4([i], [j])$, we only need to consider $i \geq 2$ and $j = 6$ for feasibility. Therefore,

$$\begin{aligned}
g_4([2], [6]) &= \tfrac{1}{2}(f_4 + 10c_4 + 6h_4) + v_6([6]) = 3 = v_4([2]),\\
g_4([3], [6]) &= \tfrac{1}{2}(f_4 + 7c_4 + 6h_4) + v_6([6]) = 3 = v_4([3]),\\
g_4([4], [6]) &= \tfrac{1}{2}(f_4 + 6c_4 + 6h_4) + v_6([6]) = 3 = v_4([4]),\\
g_4([5], [6]) &= \tfrac{1}{2}(f_4 + 2c_4 + 6h_4) + v_6([6]) = 3 = v_4([5]).\\
g_4([6], [6]) &= \tfrac{1}{2}6h_4 + v_6([6]) = 3 = v_4([6]).
\end{aligned}$$

Algorithm 2 A dynamic programming algorithm for RLTSLSP

Initialization: Re-index nodes according to their cumulative demands so that $D_1 \leq D_2 \leq \cdots \leq D_N$. In this indexing, if $i \in \mathcal{P}_{a_j}$ and $D_i = D_j$, then $i < j$. Set $n = N$.
while $n \geq 1$ **do**
  Let $k = \max_{j \in \mathcal{C}_n^{\ell_n}} a_j$
  while $k \leq N$ **do**
    Let $j = \max\{k, \bar{n}\}$
    while $j \leq N$ **do**
      if $j \in \bar{\mathcal{V}}_n \cup \{k\}$ **then**
        Calculate $g_n([k],[j])$ from (14), (15) and (16).
      end if
      $j \leftarrow j + 1$.
    end while
    Calculate $v_n([k])$ from (17).
    $k \leftarrow k + 1$.
  end while
  $n \leftarrow n - 1$.
end while
return $v_1([0])$.

Similarly,

$$\begin{aligned}
g_2([2],[5]) &= f_2 + 8c_2 + \tfrac{1}{2}(5h_3) + v_4([5]) = 14.5\\
g_2([2],[6]) &= f_2 + 10c_2 + \tfrac{1}{2}(7h_3 + 2h_5) + v_4([6]) = 18.5, \text{ thus,}\\
v_2([2]) &= \min\{g_2([2],[5]), g_2([2],[6])\} = 14.5.
\end{aligned}$$

$$\begin{aligned}
g_2([3],[5]) &= f_2 + 5c_2 + \tfrac{1}{2}(5h_3) + v_4([5]) = 11.5\\
g_2([3],[6]) &= f_2 + 7c_2 + \tfrac{1}{2}(7h_3 + 2h_5) + v_4([6]) = 15.5, \text{ thus,}\\
v_2([3]) &= \min\{g_2([3],[5]), g_2([3],[6])\} = 11.5.
\end{aligned}$$

$$\begin{aligned}
g_2([4],[5]) &= f_2 + 4c_2 + \tfrac{1}{2}(5h_3) + v_4([5]) = 10.5\\
g_2([4],[6]) &= f_2 + 6c_2 + \tfrac{1}{2}(7h_3 + 2h_5) + v_4([6]) = 14.5, \text{ thus,}\\
v_2([4]) &= \min\{g_2([4],[5]), g_2([4],[6])\} = 10.5.
\end{aligned}$$

$$\begin{aligned}
g_2([5],[5]) &= \tfrac{1}{2}(5h_3) + v_4([5]) = 5.5\\
g_2([5],[6]) &= f_2 + 2c_2 + \tfrac{1}{2}(7h_3 + 2h_5) + v_4([6]) = 10.5, \text{ thus,}\\
v_2([5]) &= \min\{g_2([5],[5]), g_2([5],[6])\} = 5.5.\\
g_2([6],[6]) &= \tfrac{1}{2}(7h_3 + 2h_5) + v_4([6]) = 7.5, \text{ thus,}\\
v_2([6]) &= 7.5.
\end{aligned}$$

Finally,

$$\begin{aligned}
g_1([0],[2]) &= f_1 + 3c_1 + 2h_1 + v_2([2]) = 119.5\\
g_1([0],[3]) &= f_1 + 6c_1 + 5h_1 + 3h_2 + v_2([3]) = 125.5\\
g_1([0],[4]) &= f_1 + 7c_1 + 6h_1 + 4h_2 + v_2([4]) = 127.5\\
g_1([0],[5]) &= f_1 + 11c_1 + 10h_1 + 8h_2 + v_2([5]) = 134.5\\
g_1([0],[6]) &= f_1 + 13c_1 + 12h_1 + 10h_2 + v_2([6]) = 142.5\\
v_1([0]) &= \min_{2 \leq j \leq 6}\{g_1([0],[j])\} = 119.5.
\end{aligned}$$

Therefore, the optimal value of the objective function is 119.5. We find the optimal solution by backtracking and get $y = (3, 8, 0, 2, 0, 0)$.

Theorem 6. *RLTSLSP can be solved in* $\mathcal{O}(\max\{B, H\}N^3)$ *time when orders do not cross in time, where* $B = \max_{n \in \mathcal{V}} |\mathcal{C}_n^1| < N$ *and* $H = \max_{n \in \mathcal{V}} |\mathcal{H}_n(C_n^1)| < N$.

Proof. For each $n \in \mathcal{V}$ and $k \in \mathcal{V}$ with $k \geq a_n$, we calculate $g_n([k],[j])$ according to (14) and (15) for $j \in \bar{\mathcal{V}}_n \cup \{k\}$. Notice that (14) and (15) is evaluated in $\mathcal{O}(B+H)$ time. Therefore, the complexity of our algorithm for RLTSLSP is $\mathcal{O}(\max\{B,H\}N^3)$. Note that the calculation of $v_n([k])$ will not add more complexity, because this calculation can be done while computing $g_n([k],[j])$. □

Remark 2. Note that $H = 1$ when there are zero lead times. Hence, we get the special case given in Theorem 3.

5. Conclusions

In this paper we make the following contributions:

(1) We model lot-sizing problems with stochastic demands, costs and lead times in which the stochastic process is very general, i.e., non-stationary and nonhomogeneous. We also allow the correlations among costs, demands and lead times by modelling this problem as a multi-stage stochastic program.
(2) We characterize the properties of optimal solutions for the stochastic lot-sizing problem with stochastic lead times when orders do not cross in time.
(3) We propose algorithms for stochastic lot-sizing problems with and without stochastic lead times that run in polynomial time in the input size, which is given by the number of nodes in the scenario tree.

6. Acknowledgment

The authors gratefully acknowledge partial financial support by a contract F49620-03-1-0477 from the AFOSR/MURI to the Department of Systems and Industrial Engineering and the Department of Management and Policy at the University of Arizona.

References

[1] A. Aggarwal and J. K. Park. Improved algorithms for economic lot size problems. *Operations Research*, 41(3):549–571, 1993.
[2] S. Ahmed, A. J. King, and G. Parija. A multi-stage stochastic integer programming approach for capacity expansion under uncertainty. *Journal of Global Optimization*, 26:3–24, 2003.
[3] O. Alp, N. K. Erkip, and R. Güllü. Optimal lot-sizing/vehicle-dispatching policies under stochastic lead times and stepwise fixed costs. *Operations Research*, 51(1):160–166, 2003.
[4] K. J. Arrow, S. Karlin, and H. Scarf. *Studies in the mathematical theory of inventory and production.* Stanford University Press, 1958.
[5] R. Ehrhardt. (s,S) policies for a dynamic inventory model with stochastic lead times. *Operations Research*, 32(1):121–132, 1984.
[6] A. Federgruen and M. Tzur. A simple forward algorithm to solve general dynamic lot sizing models with n periods in $O(n \log n)$ or $O(n)$ time. *Management Science*, 37(2):909–925, 1991.
[7] Y. Guan, S. Ahmed, G. L. Nemhauser, and A. J. Miller. A branch-and-cut algorithm for the stochastic uncapacitated lot-sizing problem. *Mathematical Programming*, 105(1):55–84, 2006.
[8] Y. Guan and A. J. Miller. Polynomial time algorithms for uncapacitated stochastic lot-sizing problems. Research Report, 2006.
[9] K. Huang. *Multi-stage stochastic programming models for production planning.* PhD thesis, Georgia Institute of Technology, 2005.
[10] R. Kaplan. A dynamic inventory model with stochastic lead times. *Management Science*, 16(7):491–507, 1970.
[11] M. J. Liberatore. Planning horizons for a stochastic lead-time inventory model. *Operations Research*, 25(6):977–988, 1984.
[12] G. Lulli and S. Sen. Branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science*, 50(6):786–796, 2004.
[13] A. J. Miller. A polynomial time dynamic programming algorithm for the multi-stage stochastic uncapacitated lot-sizing problem. INFORMS Annual Meeting Presentation, 2001.
[14] C. Nevison and M. Burstein. The dynamic lot-size model with stochastic lead times. *Management Science*, 30(1):100–109, 1984.

[15] A. Wagelmans, S. van Hoesel, and A. Kolen. Economic lot sizing: An $O(n \log n)$ algorithm that runs in linear time in the Wagner-Whitin case. *Operations Research*, 40:S145–156, 1992.

[16] H. M. Wagner and T. M. Whitin. Dynamic version of the economic lot size model. *Management Science*, 5:89–96, 1958.