The stochastic lot-sizing problem with quantity discounts

This item was submitted to Loughborough University's Institutional Repository by the/an author.


Additional Information:

- This paper was published in the journal Computers & Operations Research and the definitive published version is available at https://doi.org/10.1016/j.cor.2016.11.014.

Metadata Record: https://dspace.lboro.ac.uk/2134/26456

Version: Accepted for publication

Publisher: © Elsevier

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
The Stochastic Lot-Sizing Problem with Quantity Discounts

Wen Jiao\textsuperscript{1}, Ju-Liang Zhang\textsuperscript{2}, and Hong Yan\textsuperscript{1}

\textsuperscript{1}Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Kowloon, Hong Kong, China
\textsuperscript{2}Department of Logistics Management, School of Economics and Management, Beijing Jiaotong University, Beijing, 100044, China

Abstract: This paper addresses the stochastic lot-sizing problem with quantity discounts. In particular, we examine the uncapacitated finite-period economic lot-sizing problem in which the parameters in each period are random and discrete. When an order is placed, a fixed cost is incurred and an all-unit quantity discount is awarded based on the quantity ordered. The lead time is zero and the order is delivered immediately. First we study the case with overstocks by which the excess inventory incurs a holding cost. The objective in this case is to minimize the expected total cost including ordering and holding costs. The stochastic dynamics is modeled with a scenario tree. We characterize properties of the optimal policy and propose a polynomial time algorithm with complexity \(O(n^3)\) for single discount level, where \(n\) is the number of nodes in the scenario tree. We extend the results to cases allowing stockout and multi-discount levels. Numerical experiments are conducted to evaluate the performance of the algorithm and to gain the management insights.

Keywords: inventory, stochastic lot-sizing problem, all-unit quantity discounts, polynomial time algorithm

\textsuperscript{*}This work is supported in part by the National Natural Science Foundation of China (Grant No. 71132008, 71390334) and supported by the MOE Project of Key Research Institute of Humanities and Social Sciences at Universities (11JJD630004). This work is also supported by the NSFC/RGC Joint Research Scheme (N_PolyU531/16, 7161101015).

\textsuperscript{†}Correspondence to: Wen Jiao (wendy.j@connect.polyu.hk)
1 Introduction

The objective of the dynamic economic lot-sizing (ELS) problem is to find a manufacturer’s optimal purchase (production) plan for a single item over $T$ periods such that all demands are satisfied at a minimal cost. The dynamic and deterministic demands and costs (e.g. setup cost, unit ordering cost and inventory cost) in the planning horizon are known by the manufacturer when a plan is made. For this problem, Wagner and Whitin (1958) presented an $O(T^2)$ forward dynamic programming algorithm and showed that the optimal replenishment policy has the Zero-Inventory-Property, i.e., production only starts when the inventory is zero. Since then, the ELS problem has attracted extensive academic interest and many extensions of this problem have been studied including capacitated problem (?), multi-level ELS problem (?), uncapacitated problem with quantity discounts (Federgruen and Lee 1990), and stochastic economic lot-sizing problem (Ahmed et al. 2003). Zipkin (2000), Karimi et al. (2003) and Brahimi et al. (2006) presented extensive reviews of ELS problems.

Due to the economies of scale, suppliers often offer quantity discounts to encourage buyers to order more at one time. There are generally two types of quantity discounts, all-unit discounts and incremental discounts. The incremental discounts scheme refers to the discount applied to the quantity beyond the prespecified points and the unit price of additional quantity decreases. In an all-unit discounts scheme, the discounts are offered to every unit and as a result, the per-unit price decreases as the ordering quantity increases. The focus of this paper is on all-unit discounts. Such pricing practice is common in the transportation industry (Li et al. 2004), grocery bulk buying (Anand and Aron 2003) and others. From a logistics point of view, the incorporation of quantity discounts into ordering price complicates the decision of the optimal purchase plan, especially in a stochastic environment. ?? showed the optimality of a generalized $(s, S)$ policy in a stochastic dynamic inventory model with a concave increasing ordering cost function and a class of demand distributions. Altintas and Tayur (2008) studied an inventory system with quantity discounts under demand uncertainty with set-up cost paid by the supplier. Recently, ? introduced the generalized $(R, S)$ policy for the stochastic lot sizing problem with piecewise linear concave ordering costs. In this paper, we address the stochastic lot-sizing problem with all-unit quantity discounts by which the setup costs are paid by the buyer (manufacturer). Our study is closely related to the following three streams of research.

The first one consists of studies on the uncapacitated lot-sizing problem with quantity discounts. Federgruen and Lee (1990) considered the ELS problems with
an all-unit discount structure and an incremental discount cost structure and
developed algorithms of complexity $O(T^3)$ and $O(T^2)$ to solve these two problems
respectively (see Xu and Lu 1998 for further modifications on these algorithms).
Chan et al. (2002) demonstrated the NP-hardness of the ELS problem with a
piecewise linear all-unit discount cost function and showed that the zero-inventory-
ordering policy performs well. Li et al. (2004) proposed algorithms to solve the
batch-ordering ELS problem with time complexity $O(T \log T)$ and the truckload
discount ELS problem with time complexity $O(T^3 \log T)$. Other ELS problems
pertinent to quantity discounts include: dynamic quantity discounts ELS problem
with resales (Sohn and Hwang 1987), coordinative replenishment dynamic ELS
problem with quantity discounts (Chung et al. 2000) and quantity discount ELS
problem in material requirement planning (?). All these studies about quantity
discounts assume deterministic demand and costs.

The second stream of research associated with our study is algorithms for
the deterministic ELS problem with exact solutions. In the early 1990s, Feder-
gruen and Tzur (1991), Wagelmans et al. (1992) and Aggarwal and Park (1993)
independently obtained the same time complexity $O(T \log T)$ algorithms for the
ELS problem through different approaches. Federgruen and Tzur (1991) explored
the monotone optimal policy to develop a simple forward algorithm. Wagelmans
et al. (1992) utilized the geometric interpretation of minimal cost versus cumu-
lative demand to identify effective production periods in an optimal production
plan. Aggarwal and Park (1993) used the properties of Monge arrays to improve
the performance of algorithms. Following these studies, other related problems
such as capacititated ELS problem, ELS problem with backlog, multi-level ELS
problem have also been discussed. Ganas and Papachristos (2005) proposed a
polynomial-time $O(T^2)$ algorithm for the ELS problem with backlog based on the
convexity of the closed-form total optimal cost function. Ahuja and Hochbaum
(2008) studied the capacititated ELS problem with linear production costs and
solved the problems with or without backorder in $O(T \log T)$ time. ? solved EL-
S problem under stochastic and non-stationary demand through three heuristic
methods which are evolutionary computation and swarm intelligence, namely par-
ticle swarm optimization, differential evolution and harmony search. ? considered
the multi-level ELS problem with production capacities and general concave costs.
They defined a basic path in terms of time and stage in a supply chain to propose
a polynomial-time algorithm.

The third stream of research related to the current study concerns the stochas-
tic economic lot-sizing (SELS) problem. Ahmed et al. (2003) explored a multi-
period stochastic capacity expansion problem and proved that the Zero-Inventory-
Property does not apply to the stochastic situation. Halman et al. (2009) showed that the single-item stochastic inventory problem is NP-complete in $T$. Guan and Miller (2008) adopted a scenario tree to model the uncapacitated SELS problem, showed a production-path property for the optimal policy and characterized the value functions in terms of breakpoint. Based on the properties of the value functions, a backward dynamic programming algorithm was devised for this problem in polynomial time $O(n^3 \log \xi)$, where $n$ is the number of nodes of the scenario tree and $\xi$ is the maximum number of children of each node in the tree. Guan (2011) extended the uncapacitated and capacitated SELS problems to the backlogging case. Other papers about SELS problems include: SELS problem with random lead times (Huang and Küçükyavuz 2008), SELS problem with multi-item (?). However, the above studies about the SELS problem have excluded a quantity discount cost structure.

In this paper, we combine the above streams of research to propose a realistic model for solving the stochastic economic lot-sizing problem with all-unit quantity discounts. In particular, we study the uncapacitated finite-period SELS problem in which the demand and costs in each period are random and discrete. When an order is placed at a certain period, a fixed cost is incurred and an all-unit quantity discount is awarded based on the ordering quantity. The leadtime is zero and the order is delivered immediately. The excess inventory incurs a holding cost. The objective of this problem is to minimize the expected total cost including ordering and inventory costs. We adopt the technique from Guan and Miller (2008) and model the problem by using a scenario tree to simulate the stochastic process. First we show the property of the optimal policy. By analyzing the continuity and the number of breakpoints of the objective function, we propose a polynomial time algorithm in terms of the input size. The algorithm complexity is $O(n^3)$ for single discount level with or without backlog and is $O(n^{\log_2(m+1)+2})$ for $m$-discount levels with backlog, where $n$ is the number of total nodes in the scenario tree. Numerical experiments are conducted to analyze the impact of parameters on the efficiency of the algorithm and to gain certain insights into the management.

This study is closely related to Guan and Miller (2008). The main difference is that an all-unit discount based on the ordering quantity is included in our study, since quantity discounts are very common in practice. In addition, we obtain a tighter upper bound on the number of breakpoints of the objective function for the undiscounted SELS problem, $O(n)$ in our paper while $O(n^2)$ in Guan and Miller’s paper. Third, we conduct the actual implementation of the proposed dynamic programming algorithm. The computational results indicate that the number of time periods has the greatest influence on the running time of the algorithm while
the number of discount levels has the least impact.

The remainder of the paper is organized as follows. In Section 2 the basic model and the notation are introduced. Our preliminary findings of the model are reported in Section 3. A polynomial time algorithm for the basic model is presented in Section 4 and the time complexity analysis is discussed in Section 5. The results are then extended to settings with backlog and multi-discount levels in Section 6. In Section 7 numerical experiments are conducted to evaluate the performance of the algorithm and to gain insights into the management. Conclusions and further discussions are provided in Section 8.

2 Basic Model

We study a single item, $T$-period stochastic economic lot-sizing model with quantity discounts in which the manufacturer faces uncertain dynamic demand and costs. We focus on the all-unit quantity discount scheme that is commonly used in practice. The definitions of the notation used in this paper are summarized in Table 1.

Given the information of the current period (demand and associate cost parameters), assume that the uncertain parameters evolve as discrete time and state-dependent stochastic process with a finite probability space. We capture the uncertainty through specifying a number of representative scenarios and then update the scenarios of the following periods by including the latest information. We use a scenario tree with $T$ periods (Figure 1) to simulate the parameters on the time evolution of the process. For any node in the scenario tree except the root node, there is a unique parent node while from the current node $i$ and there are several children, such as node $j$ in Figure 1, representing possible states. Each state $j$ occurs at period $t(j)$ with probability $p_j$. To facilitate the analysis, we assume that each non-leaf node has at least two children, $|\beta(i)| \geq 2$. This assumption is reasonable if there is only one child from a non-leaf node, we simply add a virtual node with zero demand and zero probability. The probability of the children must satisfy the conditions for the stochastic process, i.e., the probability of a parent node equals to the sum of the probabilities of its children, $\sum_{j \in \beta(i)} p_j = p_i$, and the sum of the probabilities of all nodes at the same period should equal to one, $\sum_{t(i)=t} p_i = 1$. For notation brevity, let the cost parameters $c_{ij}$, $f_i$, $h_i$ include the probability $p_i$ of each node $i$ (Guan and Miller 2008).

The sequence of events at each period is listed as follows: (1) the demand realization and cost coefficients of current period are observed; (2) the scenario tree is updated based on the information up to the current period, i.e., keeping the
Table 1: The list of the notation

<table>
<thead>
<tr>
<th>Decision variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_i$</td>
<td>the ordering quantity at node $i$.</td>
</tr>
<tr>
<td>$y_i$</td>
<td>the indicator variable whether ordering or not at node $i$.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>the total number of periods.</td>
</tr>
<tr>
<td>$t(i)$</td>
<td>the time period corresponding to node $i$.</td>
</tr>
<tr>
<td>$p_i$</td>
<td>the probability of being in state $i$ which will happen at period $t(i)$.</td>
</tr>
<tr>
<td>$V(i)$</td>
<td>the subtree rooted at node $i$.</td>
</tr>
<tr>
<td>$L(i)$</td>
<td>the set of leaf nodes of node $i$.</td>
</tr>
<tr>
<td>$\beta(i)$</td>
<td>the set of branches of node $i$, $</td>
</tr>
<tr>
<td>$f_i$</td>
<td>the fixed ordering cost at node $i$.</td>
</tr>
<tr>
<td>$Q_i$</td>
<td>the single discount level at node $i$.</td>
</tr>
<tr>
<td>$c_{i0}$</td>
<td>the original ordering price at node $i$.</td>
</tr>
<tr>
<td>$c_{i1}$</td>
<td>the discounted ordering price at node $i$, $c_{i1} &lt; c_{i0}$.</td>
</tr>
<tr>
<td>$d_i$</td>
<td>the demand at node $i$.</td>
</tr>
<tr>
<td>$d_{ik}$</td>
<td>the total demand from node $i$ to node $k$, where $t(k) &gt; t(i)$.</td>
</tr>
<tr>
<td>$h_i$</td>
<td>the unit inventory holding cost at node $i$.</td>
</tr>
<tr>
<td>$n$</td>
<td>the total number of nodes of the scenario tree, i.e., $n =</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objective functions and associated variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_i$</td>
<td>the inventory level at the beginning of time period $t(i)$ before ordering, shorten as $s$ when there is no confusion.</td>
</tr>
<tr>
<td>$G(i, s)$</td>
<td>the minimum expected total cost from node $i$ to leaf nodes when the inventory level at the beginning of period $t(i)$ is $s$.</td>
</tr>
<tr>
<td>$G_O(i, s)$</td>
<td>the minimum expected cost of ordering at the price $c_{ij}$ at node $i$ with the inventory level $s$.</td>
</tr>
<tr>
<td>$G_N(i, s)$</td>
<td>the minimum expected cost of not ordering at node $i$.</td>
</tr>
<tr>
<td>$G_Q(i, s)$</td>
<td>the minimum expected cost of ordering the discount level $Q_i$ at node $i$.</td>
</tr>
<tr>
<td>$G'(i, s)$</td>
<td>the minimum between $G_N(i, s)$ and $G_Q(i, s)$.</td>
</tr>
<tr>
<td>$B(i)$, $B'(i)$, $B_O(i)$, $B_N(i)$, $B_Q(i)$</td>
<td>the breakpoint set of $G(i, s)$, $G'(i, s)$, $G_O(i, s)$, $G_N(i, s)$ and $G_Q(i, s)$, respectively.</td>
</tr>
</tbody>
</table>
realized node, deleting other nodes of current period and updating the probabilities of nodes in the following periods; (3) the ordering decision is made; (4) the order is delivered instantly and satisfies the demand with excess inventory incurring a holding cost.

The SELS problem with all-unit quantity discounts can be formulated as the following optimization problem:

$$\min \sum_{i \in V(0)} (x_i C_i(x_i) + y_i f_i + h_i s_i)$$  \hspace{1cm} (1)

s.t. \hspace{1cm} s_i + x_i = s_j + d_i \hspace{1cm} \forall i \in V(0), j \in \beta(i) \hspace{1cm} (2)

$$y_i = \begin{cases} 1 & \text{if } x_i > 0, \forall i \in V(0), \\ 0 & \text{if } x_i = 0, \forall i \in V(0), \end{cases}$$  \hspace{1cm} (3)

$$C_i(x_i) = \begin{cases} c_{i0} & \text{if } 0 \leq x_i < Q_i, \forall i \in V(0), \\ c_{i1} & \text{if } x_i \geq Q_i, \forall i \in V(0), \end{cases}$$  \hspace{1cm} (4)

$$x_i \geq 0, s_i \geq 0 \hspace{1cm} \forall i \in V(0).$$  \hspace{1cm} (5)

The objective of the model (1) is to minimize the expected total cost including ordering and holding costs. Constraint (2) indicates the inventory balance and constraint (3) denotes a fixed ordering cost that only occurs when an order is placed. Constraint (4) defines the variable ordering cost function under an all-unit discount scheme. Constraint (5) ensures the nonnegativity of the ordering quantity and inventory level.
3 Preliminaries

In this section, we characterize the properties of the optimal policy and the structure of objective function. Let \((x^*, y^*, s^*)\) be an optimal policy where \(x^*, y^*, s^*\) represent the optimal ordering quantities, ordering setup decision and inventory levels for all nodes, respectively.

3.1 Property of the Optimal Policy

**Theorem 3.1.** For the SELS problem with all-unit quantity discounts, there exists an optimal policy \((x^*, y^*, s^*)\) satisfying the following property: if \(x^*_i > 0\), either \(x^*_i = Q_i\) or there exists a node \(k \in V(i)\) such that \(x^*_i = d_{ik} - s^*_i\).

*Proof.* The proof is presented in Appendix A. \(\square\)

**Remark:** When we consider SELS without quantity discounts, the optimal ordering policy satisfies the property that there is a node \(k \in V(i)\) such that \(x^*_i = d_{ik} - s^*_i\). This is the optimal policy reported by Guan and Miller (2008). When the parameters in each period are deterministic, \(d_{ik}\) represents the total demand from period \(t(i)\) to period \(t(k)\), and Theorem 3.1 is reduced to Theorem 1 in Federgruen and Lee (1990).

Theorem 3.1 implies that the optimal ordering policy is very simple. At each node, when the optimal decision is to order, either ordering the discount level or ordering up to the total demand from current node to some descendant. This reduces the complexity of ordering operations greatly and we take this advantage to propose a polynomial algorithm for our model.

3.2 Structure of the Objective Function

Let \(G(i, s)\) be the minimal expected total cost from period \(t(i)\) to the last period when the initial inventory before ordering is \(s\) and an optimal policy is adopted. From Theorem 3.1, there are three possible options for each node \(i\): (I) order-up-to \(d_{ik}\) for some node \(k \in V(i)\); (II) order the discount level \(Q_i\); or (III) order nothing. The expected total costs corresponding to these three options are represented by \(G_O(i, s), G_Q(i, s)\) and \(G_N(i, s)\), respectively.

Option (I): Given the inventory level \(s\) before ordering, the ordering decision should be greater than \(s\) for node \(k \in V(i)\) such that \(d_{ik} > s\). \(G_O(i, s)\) includes: (1) the fixed and variable ordering costs and inventory cost at this node; and (2) the cost incurred at later periods if the inventory at the beginning of period \(t(i)+1\)
is \( d_{ik} - d_i \), which is \( \sum_{l \in \beta(i)} G(l, d_{ik} - d_i) \). Then,

\[
G_O(i, s) = f_i + \min_{k \in V(i) : s < d_{ik}} \{ (d_{ik} - s)C_i(d_{ik} - s) + h_i(d_{ik} - d_i) \}
+ \sum_{l \in \beta(i)} G(l, d_{ik} - d_i) \}.
\]

From (6), we know that \( G_O(i, s) \) is piecewise linear with respect to variable \( s \).
To further differentiate the slopes of variable \( s \), we consider two cases based on different variable ordering costs.

\[
G_O(i, s) = \min \{ G_O^1(i, s), G_O^0(i, s) \}.
\]

where

\[
G_O^1(i, s) = \min_{k \in I_j} \{ M_j(i, k) \} - c_{ij}s,
\]

\[
M_j(i, k) = f_i + d_{ik}c_{ij} + h_i(d_{ik} - d_i) + \sum_{l \in \beta(i)} G(l, d_{ik} - d_i),
\]

\[
I_1 = \{ k \in V(i) | s \leq d_{ik} - Q_i \}, I_0 = \{ k \in V(i) | d_{ik} - Q_i < s \leq d_{ik} \}.
\]

\( G_O^1(i, s) \) is the minimum expected cost of ordering larger than or equal to \( Q_i \) with the variable ordering price \( c_{i1} \). \( G_O^0(i, s) \) is the minimum expected cost of ordering less than \( Q_i \) with the variable ordering cost \( c_{i0} \).

Option (II): Ordering the discount level will lead to stockout when \( s < d_i - Q_i \). \( G_Q(i, s) \) is well defined only when \( s \geq d_i - Q_i \).

\[
G_Q(i, s) = f_i + Q_i c_{i1} + h_i(Q_i + s - d_i) + \sum_{l \in \beta(i)} G(l, Q_i + s - d_i).
\]

Option (III): Nothing is ordered. In this case, \( s \) must be larger than or equal to \( d_i \) since we assume that stockout is not allowed. \( G_N(i, s) \) comprises of two items: (1) inventory cost at this period and (2) the cost incurred in later periods if the inventory level at the beginning of period \( t(i) + 1 \) is \( s - d_i \).

\[
G_N(i, s) = h_i(s - d_i) + \sum_{l \in \beta(i)} G(l, s - d_i).
\]

To facilitate the analysis, let \( G'(i, s) \) be the the minimum between \( G_N(i, s) \) and \( G_Q(i, s) \). \( G_N(i, s) \) is only effective when \( s \geq d_i \) since stock-out is not allowed in the assumption. Thus when \( (d_i - Q_i)^+ \leq s < d_i \), \( G'(i, s) = G_Q(i, s) \).

\[
G'(i, s) = \begin{cases} 
G_Q(i, s), & \text{if } (d_i - Q_i)^+ \leq s < d_i, \\
\min \{G_Q(i, s), G_N(i, s)\}, & \text{if } d_i \leq s < \max_{k \in V(i)} d_{ik}, 
\end{cases}
\]
where \((x)^+ = \max\{0, x\}\).

The cost function \(G(i, s)\) can therefore be written as follows.

\[
G(i, s) = \begin{cases} 
G_O(i, s), & \text{if } 0 \leq s < (d_i - Q_i)^+ , \\
\min\{G_O(i, s), G'(i, s)\}, & \text{if } (d_i - Q_i)^+ \leq s < \max_{k \in V(i)} d_{ik}, \\
G_N(i, s), & \text{if } s \geq \max_{k \in V(i)} d_{ik}. 
\end{cases}
\] (11)

4 An Optimal Algorithm

In this section, we prove that \(G(i, s)\) is piecewise linear and achieves the minimum at breakpoints of objective function (where a breakpoint is a discontinuous point or a non-differentiable point of objective function). We then propose an algorithm for our model by computing the breakpoints.

**Lemma 4.1.** For each node \(i\), \(G(i, s)\) is a piecewise linear and lower semi-continuous function with respect to \(s\) and achieves its minimum in the interval \([0, \max_{k \in V(i)} d_{ik}]\).

**Proof.** The proof is presented in Appendix B. 

Due to the piecewise linearity and lower semi-continuity, the objective functions can be represented by breakpoints and their slopes. We compute the objective function backward from leaf nodes.

For leaf node \(i\),

\[
G(i, s) = \begin{cases} 
f_i + (d_i - s)c_{i1}, & \text{if } 0 \leq s \leq (d_i - Q_i)^+ , \\
f_i + Q_i c_{i1} + h_i(Q_i + s - d_i), & \text{if } (d_i - Q_i)^+ < s \leq d'_i, \\
f_i + (d_i - s)c_{i0}, & \text{if } d'_i < s < d_i, \\
h_i(s - d_i), & \text{if } s \geq d_i, 
\end{cases}
\] (12)

where \(d'_i = d_i - Q_i \frac{c_{i1} + h_i}{c_{i0} + h_i}\), \(d'_i\) is the intersection between \(G_Q(i, s)\) and \(G'_Q(i, s)\). The objective function (12) of leaf node is illustrated in Figure 2.

Next, we compute \(G(i, s)\) for each non-leaf node in the following steps:

**Step 1:** Compute \(\sum_{l \in \beta(i)} G(l, s)\). Merge the breakpoints of \(G(l, s)\) for each \(l \in \beta(i)\) into a single ordered list.

**Step 2:** Compute \(G_N(i, s)\) and \(G_Q(i, s)\). For \(G_N(i, s)\), we can move \(\sum_{l \in \beta(i)} G(l, s)\) to right by \(d_i\) units and add the function \(h_i(s - d_i)\). For \(G_Q(i, s)\), we move \(G_N(i, s)\) to left by \(Q_i\) units and increase the intercept by \(f_i + Q_i c_{i1}\).

**Step 3:** Compute \(G'(i, s)\). Merge the breakpoints of \(G_N(i, s)\) and \(G_Q(i, s)\) to a single ordered list. Between each interval in the merging list, choose the
minimum piece between $G_N(i,s)$ and $G_Q(i,s)$. If a new breakpoint exists, update the associated values and delete the column(s) of the replaced breakpoint(s).

**Step 4:** For different ordering prices $c_{ij}$, compute and store $M_j(i,k) = f_i + d_{ik}c_{ij} + h_i(d_{ik} - d_i) + \sum_{l \in \beta(i)} G(l, d_{ik} - d_i)$, for each node $k \in \beta(i)$ and $j = 0,1$.

For each node $k \in \beta(i)$, use binary search to find the first element in the ordered breakpoint list that is greater than $d_{ik} - d_i$ and obtain the intercept and slope of $\sum_{l \in \beta(i)} G(l, s)$ at the first element. Compute the function value of $\sum_{l \in \beta(i)} G(l, d_{ik} - d_i)$ and the value of $M_j(i,k)$.

**Step 5:** Compute $G_O(i,s)$. For each ordering price $c_{ij}$ where $j = 0,1$, construct the intercept $M_j(i,k)$ and breakpoints $d_{ik} - Q_i$ ($d_{ik}$) to represent the function $G_O^j(i,s)$. Sort $M_j(i,k)$ in an ascending order for each $j$. Next choose the minimum piece between $G_O^1(i,s)$ and $G_O^0(i,s)$ under their domains.

**Step 6:** Compute $G(i,s)$. Choose the minimum piece of $G_O(i,s)$ and $G'(i,s)$.

## 5 Algorithm Analysis

$G(i,s)$ is piecewise linear and lower semi-continuous so the total number of breakpoints determines the computational complexity of the algorithm. We further analyze the number of breakpoints of the objective functions, and derive the time complexity of the algorithm.
5.1 The Total Number of Breakpoints

\(B(i), B'(i), B_O(i), B_N(i), B_Q(i)\) are the breakpoint set of \(G(i,s), G'(i,s), G_O(i,s), G_N(i,s)\) and \(G_Q(i,s)\), respectively. We calculate the breakpoints in backward recursion from leaf nodes. For each leaf node \(i \in L(0)\), \(G(i,s)\) has at most four breakpoints as (12) shows. For each non-leaf node \(i\), we consider \(G_N(i,s), G_Q(i,s), G_O(i,s)\) and \(G(i,s)\), respectively.

(1) The breakpoints of \(G_N(i,s)\) and \(G_Q(i,s)\)

The breakpoints of \(G_N(i,s)\) are obtained by adding \(d_i\) units to each breakpoint of \(\sum_{l \in B(i)} \Delta(G(l,s))\). Thus, for each non-leaf node \(i\),

\[
|B_N(i)| \leq \sum_{l \in B(i)} |B(l)|.
\]

(13)

The breakpoints of \(G_Q(i,s)\) are derived by moving the breakpoints of \(G_N(i,s)\) to left by \(Q_1\) units, so \(G_Q(i,s)\) has the same number of breakpoints as \(G_N(i,s)\).

(2) The breakpoints of \(G_O(i,s)\)

According to (6), \(G_O(i,s)\) is piecewise linear and \(B_O(i)\) contains \(d_{ik} - Q_1\) and \(d_{ik}\) for nodes \(k \in V(i)\).

**Lemma 5.1.** (1) Among the breakpoints of \(G_O(i,s)\), there are at most \(|V(i)| - 1\) breakpoints being neither \(d_{ik} - Q_1\) nor \(d_{ik}\).

(2) If the breakpoints \(d_{ik}\) and \(d_{ik} - Q_1\) of \(G_O(i,s)\) belong to \(B(i)\), these breakpoints also belong to \(B'(i)\).

**Proof.** (1) By Theorem 3.1, there must exist a node \(k \in V(i)\) such that the ordering quantity is \(d_{ik} - s\) since ordering the discount level has already been considered in \(G_O(i,s)\). From (6) and (7), \(d_{ik}\) (\(d_{ik} - Q_1\)) are the only possible breakpoints of \(G_O^0(i,s), G_O^1(i,s)\). Sequence \(\{d_{ik} - Q_1\}\) is a nondecreasing order such that \(d_{ik} - Q_1 = d_{i[1]} - Q_1 \leq d_{i[2]} - Q_1 \leq \cdots \leq d_{i[|V(i)|]} - Q_1\).

Note that \(G_O(i,s) = \min\{G_O^0(i,s), G_O^1(i,s)\}\). \(G_O(i,s)\) has at most one new breakpoint being neither \(d_{ik} - Q_1\) nor \(d_{ik}\) in the interval \((d_{i[1]} - Q_1, d_{i[|V(i)|]} - Q_1)\). The existence of a new breakpoint implies that the line of \(G_O^1(i,s)\) in \((d_{i[1]} - Q_1, d_{i[|V(i)|]} - Q_1)\) intersects with the line of \(G_O^0(i,s)\) in \((d_{i[1]} - Q_1, d_{i[|V(i)|]} - Q_1)\) since the slopes of \(G_O^0(i,s)\) and \(G_O^1(i,s)\) are \(-c_{i1}\) and \(-c_{i0}\) respectively (see Figure 3). There are \(|V(i)| - 1\) subintervals in \((d_{i[1]} - Q_1, d_{i[|V(i)|]} - Q_1)\), so at most \(|V(i)| - 1\) breakpoints are neither \(d_{ik} - Q_1\) nor \(d_{ik}\).

(2) Assume that \(d_{ik}\) for a node \(k \in V(i)\) belongs to \(B_O(i)\) and \(B(i)\), but not in \(B'(i)\). Recall that \(G(i,s) = \min\{G_O(i,s), G'(i,s)\}\) when \((d_i - Q_1)^+ \leq s < \max_{k \in V(i)} d_{ik}\), we have \(G(i,d_{ik}) = G_O(i,d_{ik})\) since \(d_{ik}\) is not in \(B'(i)\). But
Figure 3: Objective function $G_O(i, s)$ of a non-leaf node

$G_O(i, d_{ik}) - G_N(i, d_{ik}) = f_i > 0$. Thus, $G(i, d_{ik}) > G_N(i, d_{ik})$. This finding contradicts the minimum definition of $G(i, s)$.

Similarly, assume $d_{ik} - Q_i$ for a node $k \in V(i)$ is included in $B_O(i)$ and $B(i)$, but not in $B'(i)$. Because $d_{ik} - Q_i$ belongs to $B_Q(i)$, and $G'(i, s) = \min\{G_N(i, s), G_Q(i, s)\}$, we have

$$G'(i, d_{ik} - Q_i) = G_N(i, d_{ik} - Q_i).$$  \hfill (14)

According to the assumption that $d_{ik} - Q_i$ does not belong to $B'(i)$,

$$G(i, d_{ik} - Q_i) = \min\{G_O(i, d_{ik} - Q_i), G'(i, d_{ik} - Q_i)\} = G_O(i, d_{ik} - Q_i).$$  \hfill (15)

From (14), we derive $G_N(i, d_{ik} - Q_i) < G_Q(i, d_{ik} - Q_i)$ and from (15), $G_O(i, d_{ik} - Q_i) < G_N(i, d_{ik} - Q_i)$. But $G_O(i, d_{ik} - Q_i) = G_Q(i, d_{ik} - Q_i)$, which also yields a contradiction.

**Lemma 5.2.** $G(i, s)$ has at most $2(|B_N(i)| + |V(i)|) - 1$ breakpoints.

**Proof.** The proof is presented in Appendix C.

**Theorem 5.1.** $G(i, s)$ has at most $O(|V(i)|^2)$ breakpoints for each non-leaf node $i$.

**Proof.** The proof is presented in Appendix D.

### 5.2 Time Complexity Analysis

**Theorem 5.2.** The SELS problem with quantity discounts can be solved in $O(n^3)$ time.
Proof.

Step 1: Compute $\sum_{l \in B(i)} G(l, s)$. Choose the minimum breakpoint in $O(\log \beta) = O(1)$ time since $\beta$ is a known constant. The total time of merging is $O(\sum_{l \in B(i)} |B(l)|) = O(|V(i)|^2)$. We spend $O(1)$ time to perform the summation for each breakpoint in the merging list. $\sum_{l \in B(i)} |B(l)|$ also has at most $O(|V(i)|^2)$ breakpoints. Hence, the total time of sum operations is $O(|V(i)|^2)$. The upper bound of this step is therefore $O(|V(i)|^2)$.

Step 2: Compute $G_N(i, s)$ and $G_Q(i, s)$. Because $O(1)$ time is spent on conducting the basic operations for each breakpoint in the ordered list, the total running time of this step will be $O(|V(i)|^2)$.

Step 3: Compute $G'(i, s)$. The minimization between $G_N(i, s)$ and $G_Q(i, s)$ at each breakpoint takes $O(1)$ time. According to Lemma 5.2, there are at most $2|B_N(i)|$ breakpoints for $G'(i, s)$ so this step completes in $O(|V(i)|^2)$ time.

Step 4: Compute $M_j(i, k)$. For each node $k$, the time complexity of binary search is $O(\log |V(i)|)$. There are $|V(i)|$ elements in the set of $V(i)$, so the time complexity of this step is $O(|V(i)| \log |V(i)|)$.

Step 5: Compute $G_O(i, s)$. Based on Lemma 5.1, $G_O(i, s)$ has at most $|V(i)| - 1$ breakpoints that are neither $d_{ik} - Q_i$ nor $d_{ik}$ and at most $2|V(i)|$ breakpoints $(d_{ik} - Q_i$ and $d_{ik})$. The sorting step requires $O(|V(i)| \log |V(i)|)$ time and the minimization between $G^0_O(i, s)$ and $G^1_O(i, s)$ takes $O(|V(i)|)$ time. The running time of this step is $O(|V(i)| \log |V(i)|)$.

Step 6: Compute $G(i, s)$. The minimization between $G_O(i, s)$ and $G'(i, s)$ at each breakpoint takes $O(1)$ time so the complexity of this step is $O(|V(i)|^2)$. Among all the six steps, the maximum complexity is $O(|V(i)|^2)$. We need to run all steps for each node and therefore the complexity of this algorithm is bounded by $O(n^3)$.

\[\square\]

Remark: For the SELS problem without quantity discounts studied by Guan and Miller (2008), the function $G_Q(i, s)$ will not be considered and therefore $|B'(i)| = |B_N(i)|$. In Lemma 5.1, there are at most $|V(i)|$ breakpoints generated by $G_O(i, s)$ and then we have $|B(i)| \leq |B_N(i)| + |V(i)|$. By induction, $|B(i)| \leq (T - t(i) + 2)|V(i)| - 2^{T-t(i)+2} + T - t(i) + 3 \leq (T - t(i))|V(i)| + T -$
t(i) + 1. The dominant term of |B(i)| would be O(|V(i)|). Therefore, G(i, s) has at most O(|V(i)|) breakpoints. The maximum complexity of the six steps then becomes O(|V(i)| log |V(i)|) and the time complexity of the SELS problem will be O(n^2 log n), which is less than the complexity O(n^3) in the Corollary 2 of Guan and Miller (2008) for the case |β(i)| = β ≥ 2. For |β(i)| ≥ 1, an O(n^2 max{β, log n}) algorithm is derived for the optimal production solution of general stochastic lot-sizing problem rather than the full characterization of objective function (the focus of our paper).

6 Extensions

6.1 The SELS Problem with Backlogging

In this subsection, we extend the results above to the backlogging case, i.e., unsatisfied demand is allowed and is backlogged with the incurrence of a unit penalty cost b_i. The objective of the model (1) changes to minimize the expected total costs including ordering cost, holding cost and backlogging cost, min \[ \sum_{i \in V(0)} (x_i C_i(x_i) + y_i f_i + \max\{h_i s_i, -b_i s_i\}) \] and s_i can be negative. Similar to Theorem 3.1, the following theorem can be proved.

**Theorem 6.1.** For the SELS problem with all-unit quantity discounts and backlogging, there exists an optimal policy \((x^*, y^*, s^*)\) satisfying the following property: if \(x_i^* > 0\), either \(x_i^* = Q_i\) or there exists a node \(k \in V(i)\) such that \(x_i^* = d_{ik} - s_i^*\).

**Proof.** The proof is provided in Appendix E. \hfill \qedsymbol

Relaxing the nonnegativity of the inventory level, \(G_Q(i, s)\) and \(G_N(i, s)\) are defined as follows.

\[
G_Q(i, s) = f_i + Q_i C_i(x_i) + \max\{h_i(Q_i + s - d_i), -b_i(Q_i + s - d_i)\} \\
+ \sum_{l \in \beta(i)} G(l, Q_i + s - d_i).
\]

\[
G_N(i, s) = \max\{h_i(s - d_i), -b_i(s - d_i)\} + \sum_{l \in \beta(i)} G(l, s - d_i).
\]

In this case, \(G'(i, s) = \min\{G_Q(i, s), G_N(i, s)\}\), when \(s < \max_{k \in V(i)} d_{ik}\) and \(G(i, s)\) becomes

\[
G(i, s) = \begin{cases} 
\min\{G_Q(i, s), G'(i, s)\}, & \text{if } s < \max_{k \in V(i)} d_{ik}, \\
G_N(i, s), & \text{if } s \geq \max_{k \in V(i)} d_{ik}.
\end{cases}
\]
$G(i, s)$ is still a piecewise linear and lower semi-continuous function with respect to $s$. For the algorithm in Section 4.2, one change is the computation of $G_N(i, s)$. In the backlogging case, the function $\max\{h_i(s - d_i), -h_i(s - d_i)\}$ will be added into $\sum_{l \in B(i)} G(l, s)$. The rest of the algorithm remains unchanged.

Table 2: $|B(i)|$ in the SELS problem with backlogging

| Interval                          | $|B(i)|$ | Note                                      |
|----------------------------------|---------|-------------------------------------------|
| $[\max_{k \in V(i)} d_{ik}, +\infty)$ | 1       | $\max_{k \in V(i)} d_{ik}$               |
| $(-\infty, \max_{k \in V(i)} d_{ik})$ | $|V(i)|$ | $G'(i, s) = G_O(i, s)$                     |
| Total                            | $2|B_N(i)| + |V(i)| - 2$ | $G'(i, s) \neq G_O(i, s)$ |

The total number of breakpoints is shown in Table 2. When $s < \max_{k \in V(i)} d_{ik}$ and $G'(i, s) \neq G_O(i, s)$, $G'(i, s)$ has at most $2|B_N(i)| - |V(i)| - 1$ breakpoints, while $G_O(i, s)$ has the same upper bound of the number of breakpoints, $2|V(i)| - 1$. Add all breakpoints in each interval together, $|B(i)| \leq 2(|B_N(i)| + |V(i)|) - 1$ for node $i$. For each leaf node $k \in L(0)$, $|B(k)| \leq 5$. Similar analysis in Theorem 5.1, we have $|B(i)| \leq 2|V(i)|^2 + 2|V(i)| + 1$ by induction. $G(i, s)$ has at most $O(|V(i)|^2)$ breakpoints for each non-leaf node. In this case, the SELS problem with backlogging can still be solved in $O(n^3)$ time.

6.2 The SELS Problem with Multi-discount Levels and Backlogging

In this subsection, we discuss the problem when the variable ordering price of each node $i$ has $m$ discount levels $Q_{ij}(j = 1, 2, \cdots, m)$.

**Theorem 6.2.** For the SELS problem with $m$-discount levels and backlogging, there exists an optimal solution $(x^*, y^*, s^*)$ satisfying the property: if $x^*_i > 0$, then $x^*_i = Q_{ij}$ or there exists a node $k \in V(i)$ such that $x^*_i = d_{ik} - s^*_s$.

**Proof.** The details of the proof are provided in Appendix F. □

$G_N(i, s)$ is derived similarly as that in the single discount level. Meanwhile, $G_O(i, s)$ is the minimum of $G_{Q_1}(i, s), G_{Q_2}(i, s), \cdots, G_{Q_m}(i, s)$ with regard to $m$ discount levels. $G_O(i, s)$ is still piecewise linear, but we need to compute $M_j(i, k) = f_i + d_{ik} c_{ij} + h_i (d_{ik} - d_i) + \sum_{l \in B(i)} G(l, d_{ik} - d_i)$ with respect to different ordering prices $c_{ij}$. Construct the intercept $M_j(i, k)$ and the slope $-c_{ij}$ with respect to the breakpoints $d_{ik} - Q_{ij}$ for each $G^j_O(i, s)(j > 0)$.

Next select the minimum piece among $G^j_O(i, s)$ under its domain. $G_O(i, s)$ has at most $m(|V(i)| - 1)$ breakpoints being neither $d_{ik} - Q_{ij}$ nor $d_{ik}$. The total
Table 3: $|B(i)|$ in the SELS problem with multi-discount levels and backlogging

| Interval                        | $|B(i)|$ | Note                                           |
|---------------------------------|--------|------------------------------------------------|
| $[\max_{k \in V(i)} d_{ik}, +\infty)$ | 1      | $\max_{k \in V(i)} d_{ik}$                     |
| $(-\infty, \max_{k \in V(i)} d_{ik})$ | $m|V(i)|$ | $G'(i, s) = G_O(i, s)$ |
|                                 | $(m+1)(|B_N(i)| - 1) + |V(i)|$ | $G'(i, s) \neq G_O(i, s)$ |
| Total                           | $(m+1)(|B_N(i)| + |V(i)| - 1) + 1$ | |

number of breakpoints is shown in Table 3. When $s < \max_{k \in V(i)} d_{ik}$ and $G'(i, s) \neq G_O(i, s)$, $G'(i, s)$ has at most $(m+1)|B_N(i)| - m|V(i)| - 1$ breakpoints, while $G_O(i, s)$ has at most $(m+1)|V(i)| - m$ breakpoints. In this case, we have $|B(i)| \leq (1+m)(|B_N(i)| + |V(i)| - 1) + 1$. For each leaf node $k \in L(0)$, $|B(k)| \leq 2m + 3$. By induction, the dominant term of $|B(i)|$ is $(m+1)^{T-t(i)+1}(2|V(i)| - 2^{T-t(i)+1} + 2) \leq (m+1)^{T-t(i)+1}|V(i)|$. Thus, $G(i, s)$ has at most $O(|V(i)|^{\log_2(m+1)+1})$ breakpoints for each node $i$. We then arrive at the following theorem.

**Theorem 6.3.** The SELS problem with $m$-discount levels and backlogging can be solved in $O(n^{\log_2(m+1)+2})$ time.

With multi-discount levels and backlogging, the algorithm is still polynomial time concerning the total number of nodes in the scenario tree.

## 7 Numerical Results

### 7.1 Impact of Parameters on the Efficiency of Algorithm

In this subsection, we conduct numerical experiments to investigate the impact of the parameters (the number of periods $T$, the number of branches $\beta$ and the number of discount levels $m$) on the efficiency of the algorithm. From Sections 5 and 6, the time complexity of our algorithm with $m$-discount levels is $O(n^{\log_2(m+1)+2})$, where $n \in [2^T - 1, \beta^T - \frac{1}{\beta-1}]$. When $n$ is fixed, $T$ and $\beta$ affect the efficiency of the algorithm adversely. The larger $\beta$ is, the smaller $T$ is. Theoretically, when the total number of nodes, $n$, of the scenario tree is fixed, $m$ has the greatest impact on the efficiency of the algorithm followed by $\beta$ with $T$ affecting the efficiency least.

We consider 12 instances with $T$ ranging from 3 to 9, $\beta$ varying from 2 to 5 and $m$ changing from 2 to 5. The cost parameters are generated randomly from uniform distributions, $f \sim [8, 15], c \sim [2, 4], h \sim [1, 2]$. Demand patterns are randomly generated from an uniform distribution with a mean of 30. We generate
the scenario tree through conditional probability \( p_i \), satisfying \( \sum_{j \in \beta(i)} p_j = p_i \). We coded the program in MATLAB and ran it in Lenovo PC with an Intel Core i5 3.4GHz processor and 4G RAM. We randomly generate 10 replications for each instance with constant \( T, \beta, m \), and the average of the run-time was calculated. The results are summarized in Table 4.

Table 4: Running time on different scenario trees

<table>
<thead>
<tr>
<th>No.</th>
<th>Period</th>
<th>Branches</th>
<th>Discount</th>
<th>Nodes</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>31</td>
<td>0.72</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>63</td>
<td>2.90</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>127</td>
<td>16.37</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>255</td>
<td>96.40</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>511</td>
<td>590.08</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>31</td>
<td>0.32</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>364</td>
<td>17.63</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>1365</td>
<td>66.21</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>3906</td>
<td>191.70</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>63</td>
<td>3.45</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>63</td>
<td>4.08</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>2</td>
<td>5</td>
<td>63</td>
<td>4.64</td>
</tr>
</tbody>
</table>

It can be seen from Table 4 that: (1) the algorithm described in the previous section is efficient. As the number of nodes increases, the run-time increases gradually; (2) \( T \) has the greatest impact on the running time followed by \( \beta \) with \( m \) having the least impact. This is an interesting phenomenon as it is inconsistent with the theoretical worst-case complexity. Our explanation for \( T \) with larger impact on the computational efficiency than \( \beta \) is as follows.

When the number of periods increases, the recursion during the computation of \( G(i, s) \) grows more quickly than that when the number of branches increases although the number of nodes in the latter expands faster. From Table 4, the run-time of No.9 with \( n = 3906 \) is 191.7 seconds while the run-time of No.5 with \( n = 511 \) is 590.08 seconds. Take the computation of \( G(5, s) \) in No.1 and No.6 as examples (see Figure 4). The number of total nodes in No.1 and No.6 are the same but the run-time for No.1 is longer than that for No.6. According to (6)-(11), in order to compute \( G(5, s) \), we need to obtain \( \sum_{l \in \beta(5)} G(l, s) \). In No.6, we only compute the function of leaf nodes from \( G(26, s) \) to \( G(30, s) \) before computing \( \sum_{l \in \beta(5)} G(l, s) \). Yet in No.1, we first need to compute the function of leaf nodes from \( G(23, s) \) to \( G(26, s) \) and only through computing \( G(11, s) \) and \( G(12, s) \) can we finally obtain \( \sum_{l \in \beta(5)} G(l, s) \).
When \( T \) and \( \beta \) are fixed, the increase in \( m \) results in the change in the calculation of \( G_Q(i, s) \) and \( G_O(i, s) \). \( G_Q(i, s) \) becomes the minimum of \( G_{Q_j}(i, s) \) and \( G_O(i, s) \) becomes the minimum of \( \{M_j(i, k) - c_{ij}s\} \) for \( j = 1, 2, \cdots, m \). The running time only increases during the process of choosing the minimum. In the case with only \( \beta \) increases, we need to compute the function of new nodes \( G(L(0), s) \) for leaf nodes and for each non-leaf node, the running time of computing \( G(i, s) \) increases since it requires to compute \( \sum_{l \in \beta(i)} G(l, s) \). Thus, \( m \) has the least impact on the computational efficiency when compared to that of \( \beta \) and \( T \).

8 Conclusions

In this paper, we study the uncapacitated stochastic lot-sizing problem with an all-unit discount cost structure. We show that the optimal policy has a simple structure. Based on the structure of the optimal policy, we develop an algorithm that solves the model in \( O(n^3) \) time for single discount level and in \( O(n^{\log_2(m+1)+2}) \) time for \( m \)-discount levels with backlogging case. We also conduct numerical experiments to analyze the impact of parameters on the efficiency of the algorithm.

To simplify the analysis, this paper has made several assumptions, including all-unit quantity discounts, single product and without capacity limit. In future, we hope to investigate efficient algorithms by adding other constraint conditions, such as (1) incremental discount; (2) ordering capacity; (3) multiple products and (4) demand with time windows.
References


Federgruen, A. and M. Tzur (1991). 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(8), 909–925.


A Proof of Theorem 3.1

Proof. Assume that there exists a node \( i \) in an optimal solution \((x^*, y^*, s^*)\) such that \( x^*_i > 0, x^*_i \neq Q_i, \) and \( x^*_i \neq d_{ik} - s^*_i \) for any node \( k \in V(i) \). \( Z(i) \) is the set of the first nodes after node \( i \) in paths from node \( i \) to leaf nodes \( L(i) \) with positive ordering quantity, which does not equal to its discount level. Let \( \bar{L}(i) \) be the set of nodes in paths from node \( i \) to leaf nodes \( L(i) \) if the paths do not include any node in \( Z(i) \).

In fact, there are two kinds of paths from node \( i \):

1. \( P_{Z(i)}^i \) is the set of nodes in paths rooted in node \( i \) and ended in nodes of \( Z(i) \).
2. \( P_{\bar{L}(i)}^i \) is the set of nodes from node \( i \) to nodes of \( \bar{L}(i) \).

We only adjust the ordering quantity of nodes \( i \) and \( z \in Z(i) \), keeping the same ordering decision of nodes in \( \bar{L}(i) \). The change of expected total cost includes the changes of ordering cost for nodes \( i \) and \( z \in Z(i) \) and the changes of holding costs for nodes with positive inventory in the above paths. Let \( S(i) = \{ l \in V(i) | s_l > 0 \text{ and } l \in P_{Z(i)}^i \cup P_{\bar{L}(i)}^i \} \).

For notational brevity, use \( \omega = C_i(x^*_i + 1) + \sum_{l \in S(i)} h_l - \sum_{z \in Z(i)} C_z(x^*_z) \).

Case 1 \( \omega < 0 \)

Increase \( x^*_i \) by 1 and decrease \( x^*_z \) \((z \in Z(i))\) by 1. After perturbation, the change of expected total cost is

\[
\Delta = C_i(x^*_i + 1) + \sum_{l \in S(i)} h_l - \sum_{z \in Z(i)} C_z(x^*_z) - x^*_i(C_i(x^*_i + 1) - C_i(x^*_i)) \\
\leq C_i(x^*_i) + \sum_{l \in S(i)} h_l - \sum_{z \in Z(i)} C_z(x^*_z) \\
= \omega < 0.
\]

The second inequality can be explained as follows. Ordering one more unit at node \( i \) does not increase the unit ordering cost, \( C_i(x^*_i + 1) \leq C_i(x^*_i) \) and ordering one less unit at nodes \( z \) adopts the same unit ordering cost due to \( x^*_z \neq Q_z \). The negative change in the expected total cost contradicts the optimality of \((x^*, y^*, s^*)\) and therefore this case will not happen.

Case 2 \( \omega \geq 0 \)

Case 2.1 \( x^*_i > Q_i \)

Let \( \delta = \min\{x^*_i - Q_i, \min_{l \in S(i)} \{s_l\}\} \), decreasing \( x^*_i \) by \( \delta \) and increasing \( x^*_z \) \((z \in Z(i))\) by \( \delta \) would reduce the holding cost for nodes in \( S(i) \), keep the same

---

22
unit ordering cost at node $i$, $C_i(x_i^* - \delta) = C_i(x_i^*)$ and decrease or do no change
the unit ordering cost at node $z$, $C_z(x_z^* + \delta) \leq C_z(x_z^*)$. The net change is
\[
\Delta = \delta[-C_i(x_i^*) - \sum_{l \in S(i)} h_l + \sum_{z \in Z(i)} C_z(x_z^*) + \delta)
+ \sum_{z \in Z(i)} x_z^*[C_z(x_z^*) + \delta] - C_z(x_z^*)]
\leq \delta[-C_i(x_i^*) - \sum_{l \in S(i)} h_l + \sum_{z \in Z(i)} C_z(x_z^*)]
= -\delta \omega \leq 0.
\]

If $\omega < 0$, the negative change in total cost contradicts the optimality of
$(x^*, y^*, s^*)$ in the assumption. If $\omega = 0$, we could obtain an alternative solution
$(x', y', s')$ with equal cost, in which $x'_i = Q_i$ or a node $k' \in S(i) \subset V(i)$ exists
such that $x'_i = d_{ik'} - s'_i$.

**Case 2.2** $x'_i < Q_i$

An alternative solution $(x', y', s')$ can be obtained by decreasing $x'_i$ by $\delta$ and
increasing $x'_z$ ($z \in Z(i)$) by $\delta$, where $\delta = \min\{x_i^*, \min_{l \in S(i)}\{s_l\}\}$. If $\delta < x_i^*$, the
net change is the same as in Case 2.1 with a node $k' \in S(i) \subset V(i)$ such that
$x'_i = d_{ik'} - s'_i$; if $\delta = x_i^*$, the order at node $i$ is deleted. The net change is
\[
\Delta = \delta[-C_i(x_i^*) - \sum_{l \in S(i)} h_l + \sum_{z \in Z(i)} C_z(x_z^*)]
+ \sum_{z \in Z(i)} x_z^*[C_z(x_z^*) + \delta] - C_z(x_z^*) - f_i
\leq \delta[-C_i(x_i^*) - \sum_{l \in S(i)} h_l + \sum_{z \in Z(i)} C_z(x_z^*) - f_i]
= -\delta \omega - f_i < 0.
\]

By repeatedly applying the above modifications, we obtain an optimal solution
that satisfies the property and therefore the conclusion holds.

**B Proof of Lemma 4.1**

**Proof.** We prove this lemma by induction.

**Step 1:** From Figure 2, and it can be seen that $G(i, s)$ is piecewise linear and
lower semi-continuous for leaf nodes.

**Step 2:** Suppose $G(k, s)$ is piecewise linear and lower semi-continuous for
nodes $k$ satisfying $t(k) \geq t + 1$. We prove that $G(i, s)$ still has the property for
node $i$ with $t(i) = t$. 

23
For a non-leaf node $i$, from (9), $G_N(i, s)$ is the summation of piecewise linear and lower semi-continuous functions as a result of the induction assumption. Because the summation of such functions preserves the piecewise linearity and semi-continuity, $G_N(i, s)$ is piecewise linear and lower semi-continuous. From (8), we move $G_N(i, s)$ to left by $Q_i$ units and add associated positive numbers, we then obtain $G_Q(i, s)$, which is also piecewise linear and lower semi-continuous. As the minimum of piecewise linear and lower semi-continuous functions preserves the properties, $G'(i, s)$ and $G_O(i, s)$ are also piecewise linear and lower semi-continuous. Thus, $G(i, s)$ is a piecewise linear and lower semi-continuous function.

The attainment of the minimum of $G(i, s)$ can be implied from the semi-continuity of $G(i, s)$ and the compactness of interval $[0, \max_{k \in V(i)} d_{ik}]$. □

C Proof of Lemma 5.2

Proof. We calculate the number of breakpoints in the following three intervals respectively.

A. When $0 \leq s < (d_i - Q_i)^+$, $G(i, s) = G_O(i, s)$ and the minimal breakpoint of $G_O(i, s)$ is larger than or equals to $(d_i - Q_i)^+$, thus $G(i, s)$ has no breakpoint in $[0, (d_i - Q_i)^+]$.

B. When $s \geq \max_{k \in V(i)} d_{ik}, G(i, s) = G_N(i, s)$ and the maximal breakpoint of $G_N(i, s)$ is $\max_{k \in V(i)} d_{ik}$, then $G(i, s)$ has one breakpoint in $[\max_{k \in V(i)} d_{ik}, +\infty)$.

C. When $(d_i - Q_i)^- \leq s < \max_{k \in V(i)} d_{ik}, G(i, s) = \min\{G_O(i, s), G'(i, s)\}$. We proceed with the discussion from $G_O(i, s) = G'(i, s)$ and $G_O(i, s) \neq G'(i, s)$.

C(a). $G_O(i, s) = G'(i, s)$ For the breakpoints $d_{ik} - Q_i$ of $G_O(i, s)$, $G_O(i, d_{ik} - Q_i) = G_Q(i, d_{ik} - Q_i)$. If $d_{ik} - Q_i$ belong to $B(i)$, from Lemma 5.1(2), such breakpoints also belong to $B'(i)$, we then have $G(i, d_{ik} - Q_i) = G_O(i, d_{ik} - Q_i) = G'(i, d_{ik} - Q_i)$. Thus, when $G_O(i, s) = G'(i, s)$, there is no new breakpoint, and $G(i, s)$ has at most $|V(i)|$ breakpoints.

C(b). $G_O(i, s) \neq G'(i, s)$ For other breakpoints of $G_O(i, s), |B_O(i)| \leq 2|V(i)| - 1$. A new breakpoint might incur through minimization since there is a jump discontinuity at the points $d_{ik}$. $G_O(i, s)$ will have at most $2|V(i)| - 1$ breakpoints including the breakpoints $d_{ik}$ and other breakpoints being neither $d_{ik} - Q_i$ nor $d_{ik}$.

When $G_O(i, s) \neq G'(i, s), |B'(i)| \leq 2|B_N(i)| - 1 - |V(i)|$. As $G_N(i, s)$ and $G_Q(i, s)$ are both piecewise linear and lower semi-continuous with the same number of breakpoints, once a new breakpoint occurs, an old breakpoint will be deleted accordingly. Thus, $G'(i, s)$ has at most $2|B_N(i)| - 1$ breakpoints in $[(d_i - Q_i)^-, \max_{k \in V(i)} d_{ik}]$ as the last breakpoint of $G_N(i, s)$ is excluded. Exce-
t the breakpoints where \( G_O(i, d_k - Q_i) = G_Q(i, d_k - Q_i) \), \( G'(i, s) \) has at most 2\( |B_N(i)| - 1 - |V(i)| \) breakpoints when \( G_O(i, s) \neq G'(i, s) \).

Now we prove that

\[
|B(i)| \leq (|B'(i)| + |B_O(i)|)_{G_O(i,s) \neq G'(i,s)} = 2|B_N(i)| + |V(i)| - 2. \quad (16)
\]

Let \( s_1, s_2, \ldots, s_m \) be the breakpoints of \( G'(i, s) \) and \( (s_k, s_{k+1}] \) be the \( k \)th interval, where \( G'(i, s_{k+1}) \neq G_O(i, s_{k+1}). \) \(|B(i)|_k, |B'(i)|_k\) and \(|B_O(i)|_k\) represent the number of breakpoints of \( G(i, s), G'(i, s) \) and \( G_O(i, s) \) in the interval \((s_k, s_{k+1}],\) respectively.

We proceed our proof in the following three cases:

**Case 1:** \( |B_O(i)|_k = 0. \) \((G_O(i, s)\) does not have a breakpoint in \((s_k, s_{k+1}]\)

1. \( G_O(i, s) \) does not intersect \( G'(i, s) \) in \((s_k, s_{k+1}]. \) If \( G'(i, s) \) lies below \( G_O(i, s) \), then no new breakpoint occurs. Otherwise, \( G_O(i, s) \) lies below \( G'(i, s) \) and \( B(i) \) does not have a breakpoint in \((s_k, s_{k+1}]. \) \(|B(i)|_k \leq |B'(i)|_k = |B'(i)|_k + |B_O(i)|_k.\)

2. \( G_O(i, s) \) intersects \( G'(i, s) \) in \((s_k, s_{k+1}]. \) At most one new breakpoint is generated as two lines intersect with an old breakpoint \( s_{k+1} \) being deleted, and the total number of breakpoints remains unchanged (see Figure 5). We then have \(|B(i)|_k = |B'(i)|_k = |B'(i)|_k + |B_O(i)|_k.\)

**Case 2:** \( |B_O(i)|_k = 1. \) \((G_O(i, s)\) has the breakpoint \( s_{k+1} \) in \((s_k, s_{k+1}]\)

1. \( G_O(i, s) \) does not intersect \( G'(i, s) \) in \((s_k, s_{k+1}]. \) In this case, no new breakpoint is generated. Then \(|B(i)|_k = |B'(i)|_k < |B'(i)|_k + |B_O(i)|_k.\)

2. \( G_O(i, s) \) intersects \( G'(i, s) \) in \((s_k, s_{k+1}]. \) In this case, at most one new breakpoint is generated as two linear pieces intersect, and an old breakpoint will
be deleted, we then have \( |B(i)|_k = |B'(i)|_k + 1 = |B'(i)|_k + |B_O(i)|_k \) (see Figure 6).

**Case 3:** \(|B_O(i)|_k = 2\). \((G_O(i,s)\) has a new breakpoint and the breakpoint \(s_{k+1}\) in \((s_k, s_{k+1})\)

As in the proof of Lemma 5.1, for \(G_O(i,s)\), there is at most one breakpoint being neither \(d_{ik} - Q_i\) nor \(d_{ik}\) in the interval \((d_{i[l]} - Q_i, d_{i[l+1]} - Q_i)\). Due to the definition of \(G'(i,s)\) and the fact that \(d_{i[l]} - Q_i\) belongs to \(B_Q(i)\), the interval \((s_k, s_{k+1})\) either equals to or belongs to the interval \((d_{i[l]} - Q_i, d_{i[l+1]} - Q_i)\). Thus, \(G_O(i,s)\) has at most one breakpoint being neither \(d_{ik} - Q_i\) nor \(d_{ik}\) in \((s_k, s_{k+1})\).

(1) \((G_O(i,s)\) does not intersect \(G'(i,s)\) in \((s_k, s_{k+1})\). If the line of \(G_O(i,s)\) in \((s_k, s_{k+1})\) lies above the line of \(G'(i,s)\), then there is no new breakpoint. Otherwise, when the line of \(G'(i,s)\) dominates \(G_O(i,s)\) at the interval, one new breakpoint \(s = s'\) of \(G_O(i,s)\) is included, which does not belong to the set \(B'(i)\) as shown in Figure 7. We have \(|B(i)|_k \leq |B'(i)|_k + 1 = |B'(i)|_k + |B_O(i)|_k - 1 < |B'(i)|_k + |B_O(i)|_k\).

![Figure 7: An example of Case 3.](image)

(2) \((G_O(i,s)\) intersects \(G'(i,s)\) in \((s_k, s_{k+1})\). Whether one or two intersections appear when the two functions intersect, there are at most two new breakpoints as shown in Figure 8. We then have \(|B(i)|_k = |B'(i)|_k + 2 = |B'(i)|_k + |B_O(i)|_k\).

Add all breakpoints of \(G(i,s)\) in each interval together, then

\[
|B(i)| \leq \frac{0}{A} + \frac{1}{B} + \frac{|V(i)| + 2|B_N(i)| + |V(i)| - 2}{C(a)} + \frac{2|B_N(i)| + |V(i)| - 2}{C(b)} = 2(|B_N(i)| + |V(i)|) - 1.
\]

Summarizing the above analysis implies that Lemma 5.2 holds. \(\square\)
D Proof of Theorem 5.1

Proof. From (13) and Lemma 5.2, we have

\[ |B(i)| \leq 2(|B_N(i)| + |V(i)|) - 1 = 2\left( \sum_{l \in \beta(i)} |B(l)| + |V(i)| \right) - 1. \]  

(17)

We use induction to prove that for each non-leaf node

\[ |B(i)| \leq (3 \cdot 2^{T-t(i)+1} - 2)|V(i)| - 5 \cdot 2^{2(T-t(i))} + 3 \cdot 2^{T-t(i)+1} - 1. \]  

(18)

For leaf node \( k \in L(0) \), \( |B(k)| \leq 4 \). For each non-leaf node \( j \) at period \( T-1 \), \( |B(j)| \leq 2[4(|V(j)| - 1) + |V(j)|] - 1 = 10|V(i)| - 9. (18) holds for nodes \( j \) at period \( T-1 \). Assume that (18) holds for nodes \( l \) where \( l \in \beta(i) \). Then

\[ |B(i)| \leq 2\left( \sum_{l \in \beta(i)} |B(l)| + |V(i)| \right) - 1 \]

\[ \leq 2[(3 \cdot 2^{T-t(l)+1} - 2)|V(i)| - 5 \cdot 2^{2(T-t(l))} + 3 \cdot 2^{T-t(l)+1} - 1 \]

\[ = (3 \cdot 2^{T-t(i)+1} - 2)|V(i)| - 5 \cdot 2^{2(T-t(i))} + 3 \cdot 2^{T-t(i)+1} - 1 \]

The second inequality derives from \( \sum_{l \in \beta(i)} |V(l)| = |V(i)| - 1 \) and \(|\beta(i)| \geq 2 \). The last equality is obtained by setting \( t(l) = t(i) + 1 \).

By assumption \(|\beta(i)| \geq 2 \), we have

\[ |V(i)| \geq 1 + 2 + \cdots + 2^{T-t(i)} = 2^{T-t(i)+1} - 1. \]  

(19)

Then

\[ |B(i)| \leq \frac{1}{4}(7|V(i)|^2 + 6|V(i)| + 3). \]  

(20)

Therefore, \( G(i,s) \) has at most \( O(|V(i)|^2) \) breakpoints for non-leaf node \( i \).
E  Proof of Theorem 6.1

Proof. Let \( S^- (i) \) be the set of nodes with negative inventory in the above paths, \( S^- (i) = \{ l \in V (i) \mid s_l < 0 \text{ and } l \in P_L (i) \cup P_Z (i) \} \) and \( \omega = C_i (x^*_i) + \sum_{l \in S^- (i)} h_l - \sum_{l \in S^- (i)} b_l - \sum_{z \in Z (i)} C_z (x^*_z) \). The rest of the proof is similar to the proof of Theorem 3.1.

F  Proof of Theorem 6.2

Proof. Suppose a node \( i \) exists in an optimal solution \((x^*, y^*, s^*)\) such that \( x^*_i > 0, x^*_i \neq Q_{ij} \) and \( x^*_i \neq d_{ik} - s^*_i \) for any node \( k \in V (i) \) and \( j \in \{1, 2, \cdots, m\} \). The proof process is similar to that of Theorem 3.1, except the different \( \delta \) values in Case 2. When \( \omega \geq 0 \), if \( Q_{ij} < x^*_i < Q_{ij+1} (j = 1, 2, \cdots, m - 1) \), then let \( \delta = \min \{ \min_{j \in \{1, 2, \cdots, m-1\}} \{ x^*_i - Q_{ij} \}, \min_{l \in S (i)} \{ s_l \} \} \); if \( x^*_i < Q_{i1} \), then \( \delta = \min \{ x^*_i, \min_{l \in S (i)} \{ s_l \} \} \); Otherwise \( x^*_i > Q_{im}, \delta = \min \{ x^*_i - Q_{im}, \min_{l \in S (i)} \{ s_l \} \}. \)