An $O(n^2)$ Algorithm for Lot Sizing with Inventory Bounds and Fixed Costs

Alper Atamtürk[†] and Simge Küçükyavuz

September 4, 2007

Abstract

We present an $O(n^2)$ dynamic programming algorithm for lot sizing with inventory bounds and fixed costs, where n is the number of time periods. The algorithm utilizes a hierarchy of two layers of value functions and improves the complexity bound of an earlier $O(n^3)$ algorithm for concave-cost and bounded inventory.

Key words: Lot-sizing, dynamic programming, extreme points.

October 2006; Revised June 2007

1 Introduction

For a finite planning horizon n, given the demand d_t, variable order cost p_t, and fixed order cost f_t, inventory capacity u_t, variable inventory holding cost h_t, and fixed inventory holding cost g_t for time periods $t \in \{1, \dots, n\}$, the *lot-sizing problem with bounded inventory* (LSBI) is to determine the order quantity and inventory in each period so that the sum of order and inventory holding costs over the horizon is minimized.

The lot-sizing problem lies at the core of production/order and inventory planning. Wagner and Whitin [7] give an $O(n^2)$ algorithm for the lot-sizing problem with no bounds on order quantity or inventory (*uncapacitated lot-sizing problem*), where n is the number of time periods. Federgruen and Tzur [3], Wagelmans et al. [6], Aggarwal and Park [1] give $O(n \log n)$ algorithms for this problem. Love [4] describes an $O(n^3)$ algorithm for lot sizing with concave cost and bounded inventory, which is more general than LSBI with fixed charges. Atamtürk and Küçükyavuz [2] give a polyhedral analysis of LSBI.

[*]Alper Atamtürk: Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA 94720-1777, USA. Email: `atamturk@berkeley.edu`.
Simge Küçükyavuz: Department of Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721-0020, USA `simge@sie.arizona.edu`
[†]Corresponding author.

In this paper we give an $O(n^2)$ dynamic programming algorithm for LSBI. The key element of the algorithm is to define a hierarchy of two layers of value functions to solve the problem efficiently.

While capacity and fixed charge on inventories have received limited attention in the literature, there are emerging applications, for which tight storage capacity and high fixed cost of specialized storage are so critical that the traditional infinite capacity and variable cost assumptions on inventory that have been prevalent in the literature are inappropriate. Below we describe a couple of emerging applications, in which highly specialized storage is the main activity of interest.

Bio-pharmaceutical storage

A bio-pharmaceutical supply chain concerns the movement of a variety of highly perishable bio-pharmaceutical products, such as antibiotics, vaccines, plasma, from the manufacturers to the retailers through distributors following intermediary storage steps. The products typically have elaborate storage requirements on temperature control, sterilization, and security. Maintaining a clean, secure, and sufficiently insured storage facility may prove to be prohibitive unless the product volume is sufficiently large. Therefore, outsourcing of bio-pharmaceutical storage has been gaining popularity. Companies, such as *BioStorage Technologies*, provide secure storage services that are climate-controlled and environmentally monitored. The charge for outsourced storage is a periodic fixed rate depending on the storage class.

Data processing and storage services in a computing utility

Explosive growth in digital data, concerns over privacy and security, and the burden of regulatory compliance have led organizations to outsource their data storage to storage service providers (SSPs) enabling secure, reliable, and fast access to data. The market for SSPs is growing rapidly as the public is increasingly concerned with privacy and security, in response to the growing number of high-profile thefts of social security and credit card information. SPPs provide data storage services often at a negotiated fixed rate per period [5].

2 The model

Now we describe the mathematical formulation of LSBI for clarity. Let y_t denote the order quantity in time period t, i_t denote the inventory at the end of period t. Also let x_t and z_t be the fixed-charge variables for order and inventory in period t, respectively. Then LSBI can be formulated as

$$
\begin{array}{llll}
 & \min & \sum_{t=1}^{n}(f_t x_t + p_t y_t + g_t z_t + h_t i_t) & \\
 & \text{s.t.} & i_{t-1} + y_t - i_t = d_t, & t \in [1,n] \\
(\text{LSBI}) & & 0 \le i_t \le u_t z_t, & t \in [0,n] \\
 & & 0 \le y_t \le (d_t + u_t) x_t, & t \in [1,n] \\
 & & z_0 = z_n = 0, & \\
 & & y \in \mathbb{R}^n,\ x \in \{0,1\}^n,\ i \in \mathbb{R}^{n+1},\ z \in \{0,1\}^{n+1}, &
\end{array}
$$

where we use $[i,j] := \{t \in \mathbb{Z} : i \le t \le j\}$. We assume that the data of the problem (f,p,g,h,d,u) consists of nonnegative real numbers. Although no explicit upper bound is specified for the order, for given demand, there is an implicit upper bound $d_t + u_t$ on the order quantity y_t due to the inventory capacities. Without loss of generality, we make the following assumption:

(A1) $u_{t-1} \le d_t + u_t$ for $t \in [1,n]$.

If $u_{t-1} > d_t + u_t$ for $t \in [1,n]$, then u_{t-1} can be reduced to $d_t + u_t$ without changing the feasible set of solutions.

Definition 1. Let $\mathcal{Q}$ denote the feasible set of LSBI. For a given point (y,x,i,z) in $\mathcal{Q}$, a consecutive sequence of time periods $[j,\ell]$ is called a *block* if $i_{j-1} \in \{0, u_{j-1}\}$, $i_\ell \in \{0, u_\ell\}$, and $0 < i_t < u_t$ for all $t \in [j, \ell-1]$.

We represent the four types of blocks by $[j,\ell]^\alpha_\beta$, where $\alpha \in \{0, u_{j-1}\}$ and $\beta \in \{0, u_\ell\}$. It follows from the network structure of the problem that (y,x,i,z) is an extreme point of $\mathcal{Q}$ if and only if there is at most one period with positive order quantity in every block $[j,\ell]^\alpha_\beta$. Consequently, the order quantity in such a period equals $d_{j\ell} + \beta - \alpha$. Partitioning of extreme points into $O(n^2)$ blocks leads to the $O(n^3)$ dynamic programming algorithm of Love [4].

The following observation states that if a block has an outgoing inventory, then the order period must be the last one, and if the block has no incoming inventory, then the order period must be the first one. It is useful for computing block costs efficiently in the next section.

Observation 1. For any block $[j,\ell]^\alpha_\beta$ of an extreme point (y,x,i,z) of $\mathcal{Q}$,

1. if $\beta = u_\ell$ and $y_k > 0$ for some $k \in [j,\ell]$, then $k = \ell$;
2. if $\alpha = 0$ and $y_k > 0$ for some $k \in [j,\ell]$, then $k = j$.

Both parts of Observation 1 follow from the fact that there is at most one order period in a block of an extreme point. For part (1) if $y_k > 0$, then $y_{k+1} = \cdots = y_\ell = 0$, implying $i_k = d_{k+1} + \cdots + d_\ell + u_\ell \le u_k$. However, from assumption **(A1)** we also have $u_k \le d_{k+1} + \cdots + d_\ell + u_\ell$. Thus, $i_k = u_k$ and $k < \ell$ contradicts with the definition of a block. For part (2) if $y_k > 0$, then $y_\ell = \cdots = y_{k-1} = 0$. Then, $i_{j-1} = 0$ implies $i_{k-1} = 0$ and $j < k$ contradicts with the definition of a block.

3 The algorithm

In this section we describe an $O(n^2)$ dynamic programming algorithm for solving LSBI. First, by substituting $i_t = \sum_{j\in[1,t]}(y_j - d_j)$, we eliminate the inventory variables in the objective. Thus, the reformulated objective has $h'_t = 0$ and $p'_t = p_t + \sum_{j\in[t,n]} h_j$ for $t \in [1,n]$ and a constant term. Inventory fixed charges for each block are handled explicitly by the algorithm. For $k \le \ell$, let $g_{k\ell} = \sum_{t=k}^{\ell} g_t$ and $d_{k\ell} = \sum_{t=k}^{\ell} d_t$; and for $k > \ell$, let $g_{k\ell} = d_{k\ell} = 0$. At the beginning of the algorithm we calculate and store g_{1k} and d_{1k} for all $k \in [1,n]$ in $O(n)$ with a single linear pass. Afterwards, computing each g_{jk} or d_{jk} takes constant time.

We use $\varphi^{10}_{j,k,\ell}$ to denote the objective value of the block $[j,\ell]_0^{u_{j-1}}$ with order in period $k \in [j,\ell]$. Thus,

$$\varphi^{10}_{j,k,\ell} := \begin{cases} f_k + p'_k(d_{j\ell} - u_{j-1}) + g_{j\ell-1}, & \text{if } d_{j\ell} > u_{j-1} \ge d_{jk-1} \ \&\ u_k \ge d_{k\ell} \\ g_{j\ell-1}, & \text{if } d_{j\ell} = u_{j-1} \\ \infty, & \text{otherwise.} \end{cases}$$

Similarly we use $\varphi^{11}_{j,k,\ell}$ to denote the objective value of the block $[j,\ell]_{u_\ell}^{u_{j-1}}$ with order in period $k \in [j,\ell]$. So,

$$\varphi^{11}_{j,k,\ell} := \begin{cases} f_k + p'_k(d_{j\ell} + u_\ell - u_{j-1}) + g_{j\ell}, & \text{if } k=\ell,\ d_{j\ell}+u_\ell > u_{j-1} \ge d_{jk-1} \ \&\ u_k \ge d_{k\ell}+u_\ell \\ g_{j\ell}, & \text{if } u_{j-1} = d_{j\ell} + u_\ell \\ \infty, & \text{otherwise.} \end{cases}$$

Observe that $d_{j\ell} + u_\ell - u_{j-1} \ge 0$ by Assumption **(A1)**.

Also we use $\varphi^{00}_{j,k,\ell}$ and $\varphi^{01}_{j,k,\ell}$ to denote the objective values of the blocks $[j,\ell]_0^0$ and $[j,\ell]_{u_\ell}^0$ with order in period $k \in [j,\ell]$. Then,

$$\varphi^{00}_{j,k,\ell} := \begin{cases} f_k + p'_k d_{k\ell} + g_{j\ell-1}, & \text{if } j = k \ \&\ u_k \ge d_{k\ell} > 0 \\ 0, & \text{if } j = \ell \ \&\ d_{j\ell} = 0 \\ \infty, & \text{otherwise.} \end{cases}$$

and

$$\varphi^{01}_{j,k,\ell} := \begin{cases} f_k + p'_k(d_{k\ell} + u_\ell) + g_{j\ell}, & \text{if } j = k = \ell \ \&\ u_k \ge d_{k\ell} + u_\ell > 0 \\ 0, & \text{if } j = \ell \ \&\ d_{j\ell} + u_\ell = 0 \\ \infty, & \text{otherwise.} \end{cases}$$

Note that $\varphi_{j,k,\ell}$ has three indices. In the $O(n^2)$ algorithm we do not compute $\varphi_{j,k,\ell}$ for all $1 \le j \le k \le \ell$ explicitly, as this would require $O(n^3)$ time.

Let v^0_k be the optimal value of LSBI over the periods $[k,n]$ given that $i_{k-1} = 0$ and v^1_k be the optimal value over the periods $[k,n]$ given that $i_{k-1} = u_{k-1}$. Hence, the optimal value of LSBI equals v^0_1.

In order to compute the value functions v^0_k and v^1_k efficiently, we do not relate them directly to φ using the block partitioning of the solutions. Rather, we first define an intermediate layer of value functions.

Toward this end, let $w^{10}_{j,k}$ denote the optimal value of LSBI over periods $[j, n]$ with first order period k in a block of type $[j,\ell]_0^{u_{j-1}}$; thus,

$$w^{10}_{j,k} := \min_{k\le\ell\le n} \left\{\varphi^{10}_{j,k,\ell} + v^0_{\ell+1}\right\},$$

where $v^0_{n+1} := 0$. Let $w^{11}_{j,k}$ denote the optimal value over periods $[j, n]$ with first order period k in a block of type $[j,\ell]_{u_\ell}^{u_{j-1}}$. So,

$$w^{11}_{j,k} := \min_{k\le\ell\le n} \left\{\varphi^{11}_{j,k,\ell} + v^1_{\ell+1}\right\},$$

where $v^1_{n+1} := 0$. Similarly, we also define

$$w^{00}_{j,k} := \min_{k\le\ell\le n} \left\{\varphi^{00}_{j,k,\ell} + v^0_{\ell+1}\right\}$$

and

$$w^{01}_{j,k} := \min_{k\le\ell\le n} \left\{\varphi^{01}_{j,k,\ell} + v^1_{\ell+1}\right\}.$$

Then, from these definitions, it follows that

$$v^0_j = \min\left\{\min_{j\le k\le n} w^{00}_{j,k}, \min_{j\le k\le n} w^{01}_{j,k}\right\}$$

and

$$v^1_j = \min\left\{\min_{j\le k\le n} w^{10}_{j,k}, \min_{j\le k\le n} w^{11}_{j,k}\right\}.$$

Proposition 1. *The optimal value v^0_1 can be computed in $O(n^2)$.*

Proof. Clearly, for fixed j, given $w_{j,k}$ for all $k \in [j, n]$, each v^0_j and v^1_j can be computed in $O(n)$. It remains to argue that all $w_{j,k}$ can be computed in $O(n^2)$ without computing all $\varphi_{j,k,\ell}$. Toward this end, we make an observation on φ:

Observation 2. For all $j \le k \le \ell \le n$ and $p, q \in \{0, 1\}$

1. if $\varphi^{pq}_{j,k,\ell} = \infty$, then $\varphi^{pq}_{j-1,k,\ell} = \infty$;
2. if $\varphi^{0q}_{j,k,\ell} \neq \infty$, then $\varphi^{0q}_{j-1,k,\ell} = \infty$;
3. if $\varphi^{1q}_{j,k,\ell} \neq \infty$, then
   (a) either $\varphi^{1q}_{j-1,k,\ell} = \infty$,
   (b) or $\varphi^{1q}_{j-1,k,\ell} = \varphi^{1q}_{j,k,\ell} + \Delta_1$, where $\Delta_1 = g_{j-1} + p'_k(u_{j-1} - u_{j-2} + d_{j-1})$,
   (c) or $\varphi^{1q}_{j-1,k,\ell} = \varphi^{1q}_{j,k,\ell} + \Delta_2$, where $\Delta_2 = g_{j-1} + f_k + p'_k(u_{j-1} - u_{j-2} + d_{j-1})$.

Note that due to Assumption **(A1)**, $u_{j-1} + d_{j-1} \ge u_{j-2}$; thus order quantity in period k may only increase to cover the reduction in incoming inventory.

Consider now computing $w^{10}_{j,k}$ for all $1 \le j \le k \le n$: Clearly, $w^{10}_{n,n} = f_n + p'_n(d_n - u_{n-1})$ if $d_n \ge u_{n-1}$, and $w^{10}_{n,n} = \infty$ otherwise, and it is computed in constant time. Now, given $w^{10}_{j,k} < \infty$, it follows from Observation 2 (as

Δ_1 and Δ_2 are independent of ℓ) that it is also possible to compute $w^{10}_{j-1,k}$ in constant time as

$$w^{10}_{j-1,k} = \begin{cases} w^{10}_{j,k} + \Delta_2, & \text{if } u_{j-2} \geq d_{j-1,k-1} \text{ \& } d_{j\ell^*} = u_{j-1}, \\ w^{10}_{j,k} + \Delta_1, & \text{if } u_{j-2} \geq d_{j-1,k-1} \text{ \& } d_{j\ell^*} > u_{j-1}, \\ \infty, & \text{otherwise,} \end{cases}$$

where ℓ^* is the minimizer in the definition of $w^{10}_{j,k}$. So, for fixed k, all $w^{10}_{j,k}$, $1 \leq j \leq k$ is computed in $O(n)$ with a backward linear pass, implying $O(n^2)$ time for computing $w^{10}_{j,k}$ for all $1 \leq j \leq k \leq n$. Other $w^{11}_{j,k}$, $w^{00}_{j,k}$, $w^{01}_{j,k}$ are computed similarly using Observation 2 in $O(n^2)$ time as well.

Hence, we have an $O(n^2)$ algorithm for computing v^0_1. $\square$

References

[1] A. Aggarwal and J. K. Park. Improved algorithms for economic lot-size problems. *Operations Research*, 41:549–571, 1993.

[2] A. Atamtürk and S. Küçükyavuz. Lot sizing with inventory bounds and fixed costs: Polyhedral study and computation. *Operations Research*, 53: 711–730, 2005.

[3] 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*, 37:909–925, 1991.

[4] S. F. Love. Bounded production and inventory models with piecewise concave costs. *Management Science*, 20:313–318, 1973.

[5] Sun Microsystems Inc. Press Release: Sun Microsystems launches innovative new programs to simplify storage acquisition and management. `http://www.sun.com/smi/Press/sunflash/2005-05/sunflash.20050503.3.xml`, 2005.

[6] 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–S156, 1992.

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