A LAGRANGEAN RELAXATION ALGORITHM FOR MULTI-ITEM LOT-SIZING PROBLEMS WITH JOINT PIECEWISE LINEAR RESOURCE COSTS[#]

Nafee Rizk, Alain Martel*

Centor, Université Laval, Québec, Canada

Amar Ramudhin

École de technologie supérieure, Montréal, Canada

Centor

(*Network Enterprise Technology Research Center*)

Université Laval, Québec, Canada

Working Paper DT-2002-AM-1

(Submitted for publication in November 2002, Revised in August 2004)

(Accepted for publication in IJPE in February 2005)

[#] This research was supported by the Natural Sciences and Engineering Council of Canada, by CAPS LOGISTICS, and by FORAC.

* Corresponding author: Faculté des sciences de l'administration, Université Laval, Québec, QC, Canada, G1K 7P4 (alain.martel@centor.ulaval.ca)

ABSTRACT

In this paper we study a class of multi-item lot-sizing problems with dynamic demands, as well as lower and upper bounds on a shared resource with a piecewise linear cost. The shared resource might be supply, production or transportation capacity. The model is particularly applicable to problems with joint shipping and/or purchasing cost discounts. The problem is formulated as a mixed-integer program. Lagrangean relaxation is used to decompose the problem into a set of simple sub-problems. A heuristic method based on subgradient optimization is then proposed to solve a particular case often encountered in the consumer goods wholesaling and retailing industry. Our tests show that the heuristic proposed is very efficient in solving large real-life supply planning problems.

KEYWORDS: Multi-item lot-sizing, Dynamic demand, Piecewise linear costs, Discounts, Mixed integer programming, Lagrangean relaxation.

1. Introduction

In the consumer goods wholesaling and retailing industry, a large number of stock keeping units must be managed on a regular basis. The items are typically purchased in families, each family being associated to a specific external vendor, and usually they are shipped together. The transportation costs thus incurred account for a significant part of the items price, and their structure may incorporate economies of scale, as well as constraints on the total load which can be carried. In some cases, the transportation costs are included in the item prices and they are not paid directly by the buyer. In such cases, however, the vendor often offers discounts on the total amount of the order and imposes constraints on its size. In either case, this give rise to the class of dynamic multi-item lot sizing problems with joint piecewise linear resource costs studied in this paper. Piecewise linear cost functions can also be used to model several production lot sizing situations and the methodology proposed can be used to solve these problems.

The problem studied is a generalization of several of the dynamic lot-sizing problems that have been treated in the literature. Much work has been done to find exact and heuristic methods to solve the numerous versions of the dynamic lot-sizing problem. The complexity of the problem depends largely on whether it deals with a single item or several items, on the resource constraints imposed on the lot sizes (unconstrained or joint lower bounds and/or joint upper bounds) and on the type of cost structure which applies.

A dynamic programming algorithm to solve the basic single-item case was published by Wagner and Whitin (1958). Comparisons of the numerous heuristics that have been developed to solve the single-item problem can be found in Zoller and Robrade (1988). Bitran and Yanasse (1982) provide a review and analysis of the problem under capacity constraints. When several items are involved, additional complexity is introduced by cost interactions and/or by resource interactions. The simplest multi-item case is the one where the interaction is due only to a joint setup (or order) cost. Dynamic programming algorithms to solve this problem were proposed by Zangwill (1966), Veinott (1969), Kao (1979), Silver (1979) and Haseborg (1982). Atkins and Iyogun (1988) developed a heuristic procedure. Maes and Van Wassenhove (1986) reviewed the case where the interaction is due only to capacity constraints and where costs are stationary. They also provide a comparison of three "simple" heuristics. An integer programming approach with strong cutting planes was proposed by Barany, Van Roy and Wolsey (1984), Pochet, and. Wolsey (1991), Belvaux, and Wolsey (2001) and Lagrangean relaxation procedures were developed by Thizy and Van Wassenhove (1985), Gelder, Maes and VanWassenhove (1986), Trigeiro, Thomas and McClain (1989), Diaby, Bahl, Karwan and Zionts (1992) and Sox and Gao (1999). A number of set partitioning and column generation heuristics are also proposed by Cattrysse, Maes and VanWassenhove (1990).

To the best of our knowledge, little work was done on the case where the interaction is caused by joint order cost functions as well as by restrictions on the amount of resources used/available. Such situations are very common in the consumer goods wholesaling and retailing industry and occur when shipment costs depend on the total amount of transportation resources used by all the items, and/or when all-units or incremental discounts are offered by vendors on the total invoice value, instead of on the quantity ordered for individual items, as is usually assumed in the literature. A particular case with joint piece-wise linearly increasing freight rates is discussed by van Norden and van de Velde (2005).

In practice, shipping costs depend on several factors. Pricing mechanisms are not the same for private fleets and for public carriers. In the former case, for our purposes, only variable costs related to specific shipments are relevant. In the latter case, freight-rate structures depend on the mode used, the shipping distance and weight, the type of shipment (TL or LTL), and the commodity class of the item shipped. For this reason, few studies have attempted to model freight-rate structures explicitly. Buffa and Munn (1989) and Swenseth and Godfrey (2002) provide reviews of the work done in this area. Most of the currently available models that incorporate a realistic representation of freight-rate structures deal only with the single-item, constant-demand case.

Little work has been done on the modeling of joint discounts based on total invoice value. Some aspects of the problem are studied by Chakravarty (1984) and by Silver and Peterson (1985) for the unconstrained constant-demand case. To the best of our knowledge, however, no previous work is available on joint discounts for the multi-item dynamic-demand problem treated in this paper.

The paper is organized as follows. In section 2 the problem is defined and formulated as a mixed-integer program. In section 3 Lagrangean relaxation is used to decompose it in a set of simple sub-problems. A heuristic method based on subgradient optimization is then proposed. Finally, section 4 presents the experiments performed to assess the solution method and it discusses computational results.

2. Problem Definition and Formulation

For most practical situations, joint shipping cost, price discount or production capacity cost structures, can be represented by a general (i.e., not necessarily concave, nor continuous) piecewise linear function $z(S)$ of the amount $S \in [0, S_{max}]$ of a shared resource (measured in cwt, cube, \$, hours, etc.) associated to a supply program. S_{max} is an upper bound on the availability of this resource in a given time period. In shipping and purchasing contexts, this complex cost structure is required to model the various types of discounts offered by public carriers. It also accounts for the situation where one switches from one mode of transportation to another based on volumes, and for the constraints imposed by carriers to qualify for certain types of discounts. It can also adequately model the costs incurred when a private fleet is used, as well as incremental or all units cost discount structures. In production contexts, a convex piecewise linear cost function can be used to model overtime costs, while a concave function would model economies of scale realized when one uses more efficient technologies as production volumes increase. Some situations that are encountered often in practice are illustrated in Figure 1.

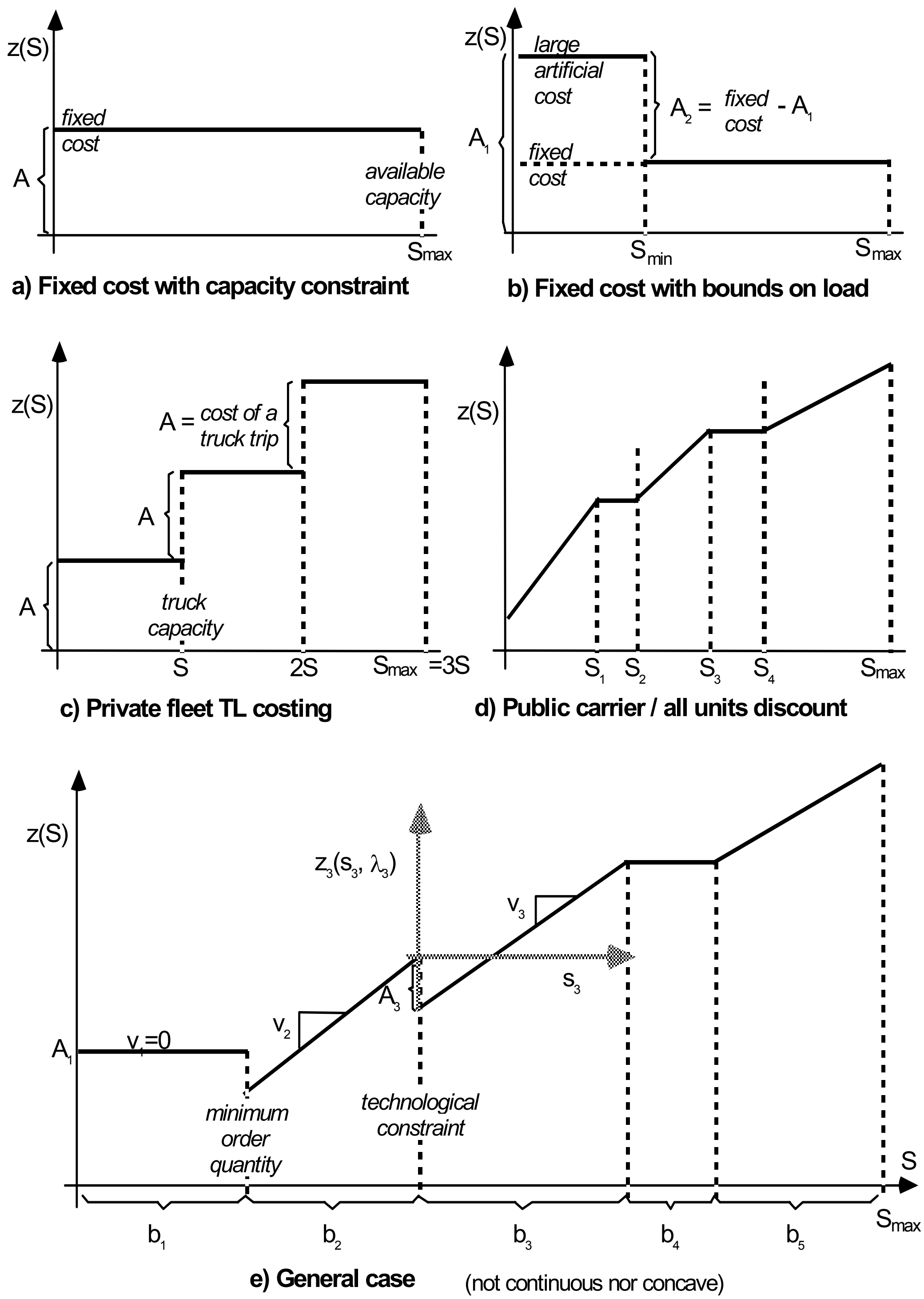

Figure 1: Piecewise Linear Resource Consumption Cost Functions

Let b_j, $j = 1,...,\gamma$, denote the length of the j^{th} interval defined on the S-axis by the linear segments of z(S) and define new linear functions,

$$z_j(s_j, \lambda_j) = A_j\lambda_j + v_js_j, \qquad \lambda_j \in \{0, 1\}, \quad s_j \in [0, b_j], \qquad j = 1, ..., \gamma,$$

over each of these intervals as illustrated in Figure 1e). Then, assuming that we want to minimize costs, the amount of resource used is given by $S = \sum_{j=1}^{\gamma} s_j$, where s_j is the amount procured at rate v_j, and the shared resource consumption cost by

$$z(S) = \sum_{j=1}^{\gamma} z_j(s_j, \lambda_j) = \sum_{j=1}^{\gamma} (A_j\lambda_j + v_js_j),$$

provided that,

$$b_j\lambda_{j+1} \le s_j \le b_j\lambda_j, \qquad \lambda_j \in \{0, 1\}, j = 1, ..., \gamma, \lambda_{\gamma+1} = 0.$$

To simplify the notation, we assume without loss of generality that the intervals on the S-axis of the resource consumption cost function do not change from period to period.

Using the notation defined in Figure 2, the dynamic lot-sizing problem with joint piecewise linear resource costs can be formulated as follows:

$$\text{(P)} \qquad \min \sum_{t=\tau+1}^{T} \{ \sum_{j=1}^{\gamma} (A_{tj}\lambda_{tj} + v_{tj}s_{tj}) + \sum_{i=1}^{m} [C_{it}(R_{it}) + h_{it}I_{it}] \}$$

subject to

$$\text{(1)} \qquad R_{it} + I_{i,t-1} - I_{it} = d_{it}, \qquad \forall i, \; t = \tau+1, ..., T \;\; (I_{i\tau} = 0)$$

$$\text{(2)} \qquad \sum_{j=1}^{\gamma} s_{tj} - \sum_{i=1}^{m} r_{it}R_{it} = 0, \qquad t = \tau+1, ..., T$$

$$\text{(3)} \qquad s_{tj} - b_j\lambda_{tj} \le 0, \qquad \forall j, \; t = \tau+1, ..., T$$

$$\text{(4)} \qquad -s_{tj} + b_j\lambda_{t,j+1} \le 0, \qquad \forall j, \; t = \tau+1, ..., T \;\; (\lambda_{t,\gamma+1} = 0)$$

$$\text{(5)} \qquad R_{it}, I_{it}, s_{tj} \ge 0, \qquad \lambda_{tj} \in \{0, 1\} \qquad \forall i, j, t = \tau+1, ..., T$$

i	: an item type, $i \in F$, $F = \{1, \ldots, m\}$
t	: a planning period, $t \in H$, $H = \{\tau+1, \ldots, T\}$
τ	: supply lead-time (in planning periods) for the family of items
d_{it}	: effective demand for item i during period t
r_{it}	: resource absorption rate for item i in period t (in cwt, cube, \$, hours...)
A_{tj}	: fixed cost/saving associated to the j^{th} interval on the resource consumption cost function in period t
v_{tj}	: j^{th} unit resource consumption cost for period t
b_j	: maximum amount which can be supplied at a rate v_{tj}, for all t (in cwt, cube, \$, hours...)
$C_{it}(\ )$	: procurement/production cost function for item i in period t (excluding the cost of the shared resource)
h_{it}	: inventory holding cost for item i in period t
R_{it}	: quantity of item i supplied for the beginning of period t
I_{it}	: inventory of item i on hand at the end of period t
S_t	: amount of resource absorbed by the items supplied for period t (in cwt, cube, \$, hours...)
s_{tj}	: amount of resource procured at rate v_{jt} for period t (in cwt, cube, \$, hours...)
λ_{tj}	: binary variable associated with the j^{th} interval on the resource consumption cost function for period t

Figure 2: Basic Notation

The objective function of (P) simply says that we want to minimize the total of resource consumption, procurement/production and inventory costs. Constraints (1) are the inventory balance equations. They insure that all the effective demands are met. Constraints (2) establish the relationship between the items supplied R_{it} and the variables s_{tj} defined over the intervals of the resource consumption cost functions. Constraint sets (3) and (4) enforce the piecewise-linear nature of the resource consumption costs.

3. Lagrangean Relaxation

If we multiply the constraints (2) in (P) by Lagrange multipliers υ_t, and add the resulting expressions to the objective function of (P), a Lagrangean relaxation, LR(υ), of (P) is obtained. LR(υ) can be decomposed into the following two primal Lagrangean sub-problems:

$$\text{(SP1)} \qquad \min \sum_{t=\tau+1}^{T} \sum_{i=1}^{m} [C_{it}(R_{it}) + \upsilon_t r_{it} R_{it} + h_{it} I_{it}]$$

subject to

$$R_{it} + I_{i,t-1} - I_{it} = d_{it}, \qquad \forall i,\ t = \tau+1, \dots, T \quad (I_{i\tau} = 0)$$

$$R_{it}, I_{it} \geq 0 \qquad \forall i,\ t = \tau+1, \dots, T$$

and

$$\text{(SP2)} \qquad \min \sum_{t=\tau+1}^{T} \sum_{j=1}^{\gamma} [A_{tj}\lambda_{tj} + (v_{tj} - \upsilon_t) s_{tj}]$$

subject to

$$s_{tj} - b_j \lambda_{tj} \leq 0, \qquad \forall j,\ t = \tau+1, \dots, T$$

$$-s_{tj} + b_j \lambda_{t,j+1} \leq 0, \qquad \forall j,\ t = \tau+1, \dots, T \quad (\lambda_{t,\gamma+1} = 0)$$

$$s_{tj} \geq 0,\ \lambda_{tj} \in \{0, 1\} \qquad \forall j,\ t = \tau+1, \dots, T.$$

Clearly, (SP1) decomposes into the following m unconstrained single-item lot-sizing problems:

$$\text{(SP1-i)} \qquad L1_i(\upsilon) = \min \sum_{t=\tau+1}^{T} [C_{it}(R_{it}) + \upsilon_t r_{it} R_{it} + h_{it} I_{it}]$$

subject to

$$\text{(6)} \qquad R_{it} + I_{i,t-1} - I_{it} = d_{it}, \qquad t = \tau+1, \dots, T \quad (I_{i\tau} = 0)$$

(7) $R_{it}, I_{it} \geq 0 \quad t = \tau+1, \ldots, T$

Program (SP1-i) is a network flow problem. When $C_{it}(R_{it})$ is linear, it can be solved efficiently as a transportation LP. When $C_{it}(R_{it})$ is concave, it can be solved with Wagner-Whitin's dynamic programming algorithm (Wagner and Whitin, 1958). The later case includes the familiar fixed plus linear cost structure often assumed when studying Capacitated Lot Sizing Problems (CLSP), i.e. the cost function $C_{it}(R_{it}) = a_i\delta(R_{it}) + c_{it}R_{it}$, where a_i is an item set-up cost, c_{it} an item variable production cost and the function $\delta(R) = 0$ when $R = 0$ and 1 when $R > 0$.

Program (SP2) decomposes into the $(T-\tau)$ independent sub-problems:

(SP2-t) $L2_t(\upsilon) = \min \sum_{j=1}^{\gamma} [A_{tj}\lambda_{tj} + (v_{tj}-\upsilon_t)s_{tj}]$

subject to

(8) $s_{tj} - b_j\lambda_{tj} \leq 0, \quad \forall j$

(9) $-s_{tj} + b_j\lambda_{t,j+1} \leq 0, \quad \forall j \ (\lambda_{t,\gamma+1} = 0)$

(10) $s_{tj} \geq 0, \ \lambda_{tj} \in \{0, 1\} \quad \forall j.$

Problem (SP2-t) can be solved efficiently with the $O(\gamma)$ time algorithm proposed by Diaby and Martel (1993). They showed that (SP2-t) is equivalent to the following pure integer programming model (SP2-t)'

(SP2-t)' $L2_t(\upsilon) = \min \sum_{j=1}^{\gamma} [A_{tj}\lambda_{tj} + (v_{tj}-\upsilon_t)\,\alpha_{tj}\,b_j]$

subject to

(11) $\lambda_{t,j+1} \leq \alpha_{tj} \leq \lambda_{tj} \quad \forall j.$

(12) $\alpha_{tj}, \lambda_{tj} \in \{0, 1\} \quad \forall j.$

They also showed that (SP2-t)' can be solved optimally by making one forward pass through the intervals j ($0 \le j \le \gamma$) to determine, in a greedy manner, which one improves most the value of the objective function $L2_t(\upsilon)$. It is important to observe that the coefficient matrix of (SP2-t)' is totally unimodular, which means that the integrality requirements for the variables λ_{tj} and α_{tj} can be dropped.

From Lagrangean relaxation theory (Fisher, 1981), we know that

$$(11) \qquad L_D(\upsilon) = \sum_{i=1}^{m} L1_i(\upsilon) + \sum_{t=\tau+1}^{T} L2_t(\upsilon)$$

is a lower bound on the cost of the optimal solution of (P). We also know that the greatest lower bound attainable with our Lagrangean relaxation is provided by the multipliers obtained by solving the following Lagrangean dual problem (LD):

$$(LD) \qquad L_D = \max_{\upsilon} L_D(\upsilon)$$

Problem (LD) can be solved efficiently by a subgradient optimization procedure (see Bazarra and Goode (1979) or Fisher (1981)). This provides the basis for the elaboration of a good branch-and-bound algorithm, as well as a heuristic procedure, for the solution of (P). Although, a branch-and-bound algorithm involving a branching strategy based on feasibility considerations, a bounding scheme based on Lagrangean relaxation and a heuristic procedure to generate upper bounds at each node examined was developed to solve (P), we focus our attention in what follows on the case when the procurement/production costs are linear, i.e. $C_{it}(R_{it}) = c_{it}R_{it}$. This is the prevailing situation in the consumer goods wholesaling and retailing industry.

From Largrangean relaxation theory, we know that the optimal solution value of (LD) provides a lower bound that is at least as good as the one obtained by solving the linear relaxation of (P). However, in practice, the subgradient optimization procedures implemented usually terminate before an optimal solution of (LD) is reached. Hence, there is no guarantee that the lower bound obtained using the subgradient optimization procedure to solve (LD) is better than the one obtained by solving the linear relaxation of (P). When. $C_{it}(R_{it})$ is linear, the Lagrangean relaxation program LR(υ) has the integrality propriety. This means that, in this case, the lower bound obtained by solving (LD) can, at most, be as good as the lower bound obtained by solving the linear relaxation of (P). This suggest that, for this particular case, the branch and bound algorithms, based on the linear relaxation of the problem, found in commercial solvers such as Cplex should perform very well. This was confirmed experimentally when we compared the solution times of Cplex 9.0 with those of our branch-and-bound algorithm for this particular case.

The main advantage of using a Lagrangean relaxation is that it, usually, preserves most of the original problem structure. This makes it easier to use the relaxed problem solutions to generate good quality feasible solutions for the original problem. This suggests that a very efficient heuristic method to solve (P) could be obtained by applying the subgradient optimization procedure and checking, at each iteration, if the solution provided by the primal sub-problems is a feasible solution of (P), i.e. if $\sum_{j=1}^{\gamma} s_{tj} - \sum_{i=1}^{m} r_{it}R_{it} = 0$. If it is, then this solution is also optimal for (P). Otherwise, this solution can be modified using a perturbation procedure to generate a feasible solution for (P). A detailed description of a heuristic procedure based on this idea is presented in the following section. In section 5, its performance is compared to those of the Cplex solver for realistic size problems.

4. Solution Method for the Linear Procurement/Production Costs Case

4.1 Subgradient Heuristic

Our overall solution method to solve (P) is a modified subgradient optimization procedure. At each iteration of this procedure, a lower bound and a feasible solution for (P) are generated. A detailed description of the procedure is found in Figure 3.

Step 1 Initialize the subgradient procedure parameters:

$\upsilon_t^0 = 0$ $t = \tau+1, \ldots, T$ (Lagrange multipliers)

$k = 1$ (iterations counter)

$\Delta_0 = 1$ (multiplier)

$L_D = 0$ (Lower bound value)

$\overline{L_D} = M$ (Upper bound value) (where M is a large number)

repeat until $k = k_{max}$ (Maximum number of iterations)

Step 2 Solve the Lagrangean subproblems, (SP1-i) and (SP2-t)' with $\upsilon_t = \upsilon_t^{k-1}$

Step 3 Compute the new lower bound value:

$$L_D(\upsilon^{k-1}) = \sum_{i=1}^{m} L1_i(\upsilon^{k-1}) + \sum_{t=\tau+1}^{T} L2_t(\upsilon^{k-1})$$

Step 4 if the Lagrangean solution (R^{k-1*}, s^{k-1*}) is feasible for (P)

(i.e. $\sum_{j=1}^{\gamma} s_{tj}^{k-1*} - \sum_{i=1}^{m} r_{it} R_{it}^{k-1*} = 0$) then:

Step 5 Update the lower and upper bound values: $\overline{L_D} = L_D = L_D(\upsilon^{k-1})$.

stop: the Lagrangean solution is an optimal solution for (P)

else begin

Step 6 if $L_D(\upsilon^{k-1}) > L_D$ then update L_D: $L_D = L_D(\upsilon^{k-1})$

Step 7 Generate a feasible solution for (P) using a *perturbation procedure*

Step 8 Compute the objective function value of the new feasible solution:

$\overline{L_D}(\upsilon^{k-1})$

Step 9 if $\overline{L_D}(\upsilon^{k-1}) < \overline{L_D}$ then update $\overline{L_D}$: $\overline{L_D} = \overline{L_D}(\upsilon^{k-1})$

Step 10 Update the Lagrangean multipliers:

$$\upsilon_t^k = \upsilon_t^{k-1} + \beta_{k-1} \left[\left(\sum_{i=1}^{m} r_{it} R_{it}^{k-1*} - \sum_{j=1}^{\gamma} s_{tj}^{k-1*}\right)\right]$$

where β_{k-1} is the subgradient step size

endelse

Step 11 Increment the iterations counter: $k = k+1$

endrepeat

Figure 3: Subgradient Heuristic Method

The procedure works as follows. At iteration k, if the Lagrangean solution (R^{k-1*}, s^{k-1*}) is not feasible for (P), this solution is modified using the perturbation procedure described in the next sub-section to generate a new feasible solution for (P) (Step7). If the value of the new feasible solution is better than the incumbent upper bound of (P) ($\overline{L_D}$) then the new value becomes the incumbent upper bound (Step9).

In the solution method, initial values for the Lagrange multipliers are set to zero:

$\upsilon_t^0 = 0 \qquad t = \tau+1, \ldots, T$

At iteration k, the Lagrange multipliers are changed (Step10) according to the formula:

$$\upsilon_t^k = \upsilon_t^{k-1} + \beta_{k-1} [(\sum_{i=1}^{m} r_{it} R_{it}^{k-1*} - \sum_{j=1}^{\gamma} s_{tj}^{k-1*})]$$

where (R^{k-1*}, s^{k-1*}) is the solution to the Lagrangean sub-problem at iteration k-1, and β_{k-1} is the Lagrangean step size given by:

$$\beta_{k-1} = \Delta_{k-1} (\overline{L_D} - L_D(\upsilon^{k-1})) / [\sum_{t=\tau+1}^{T} (\sum_{i=1}^{m} r_{it} R_{it}^{k-1*} - \sum_{j=1}^{\gamma} s_{tj}^{k-1*})^2],$$

where $\Delta_{k-1} \in (0, 2]$; $\overline{L_D}$ is an upper bound on the value of the Lagrangean dual problem, and $L_D(\upsilon^{k-1})$ is the value of the Lagrangean sub-problem at iteration k-1. We start with $\Delta_0 = 1$ and multiply Δ_{k-1} by 0.45 if the improvement in the Lagrangean bound (L_D) is not more than 0.25% in two consecutive trials. We stop the procedure if the Lagrangean solution is feasible for (P), if the improvement in the Lagrangean bound is not more than 0.25% in 35 consecutive trials (to simplify, this stopping criterion is not included in Figure 3), or if the total number of iterations reaches a predetermined value k_{max}.

4.2 Perturbation Procedure

At each iteration of the sub-gradient procedure, a feasible solution to (P) is obtained by perturbing the optimal solution of (SP1). The solution of (SP1) may not be feasible in (P) as the total quantity of resources absorbed in period t ($S_t = \sum_{i=1}^{m} r_{it} R_{it}$) may not fall within period t transportation capacity limits. The perturbation procedure proposed involves a backward and a forward pass that reduce lot sizes to avoid transportation capacity shortages. At each iteration the missing capacity is calculated for each period. Based on procurement and inventory cost considerations, the shortage is eliminated by shifting supply quantities among periods. The result is a feasible solution and an upper bound for (P). This feasible solution is further improved by a heuristic procedure that seeks higher gains through additional period shifting adjustments. A detailed description of the backward procedure is presented in Figure 4.

---

for t = T downto (τ+2) do
begin
Step 1 $gap_t = S_t - S_{max}$;
if $gap_t > 0$ then *{reduce lot size}*
repeat
Step 2 $\varphi = \{\theta < t \mid S_\theta < S_{max}\}$;
if ($\varphi \neq \phi$) then
begin
Step 3 Choose a product i_0 in F and a period $t_0 < t$ in φ based on the following cost criterion:

$$(c_{i_0t_0} + \sum_{\theta=t0}^{t-1} h_{i_0\theta} - c_{i_0t}) = \min_{\alpha\in\varphi} \{ \min_{\substack{j\in F \\ R_{jt}>0}} ((c_{j\alpha} + \sum_{\theta=\alpha}^{t-1} h_{j\theta} - c_{jt}))\};$$

Step 4 Compute the shipment quantity Δ of product i_0 to shift from period t to period t_0 as follows:

$$\Delta = \min (R_{i_0t}, \lceil gap_t/r_{i_0t} \rceil, \lfloor (S_{max} - S_{t_0})/r_{i_0t_0} \rfloor);$$

Step 5 Update R_{i_0t}, $R_{i_0t_0}$, and gap_t:

$$R_{i_0t} = R_{i_0t} - \Delta, \quad R_{i_0t_0} = R_{i_0t_0} + \Delta, \quad gap_t = gap_t - (r_{i_0t}\Delta)$$

endif
until ($gap_t \leq 0$ or $\varphi = \phi$)
endif
endfor

Figure 4: Backward Pass of Perturbation Procedure

The backward pass procedure, starting from period T and ending at period (τ+2), verifies for each period t whether the quantity of resource (S_t) is greater than the maximum capacity allowed (S_{max}), by computing $gap_t = S_t - S_{max}$ (step 1). An iterative procedure is then used to eliminate the "capacity shortage" by shifting a quantity Δ of a product i_0 supply from period t to a period $t_0 < t$. The product i_0 and the period t_0, if any, are chosen in step 3 so as to have the smallest net increase of the procurement and holding costs ($c_{i_0t_0} + \sum_{\theta=t_0}^{t-1} h_{i_0\theta} - c_{i_0t}$), and the quantity Δ is determined in step 4 in a way not to generate infeasibility at period t_0 ($\Delta \leq \lfloor (S_{max} - S_{t_0})/r_{i_0t_0} \rfloor$) and not to be larger than needed ($\Delta \leq \lceil gap_t/r_{i_0t} \rceil$). This process is repeated until the shortage capacity is eliminated ($gap_t \leq 0$) or no quantity of any product i_0 can be moved from period t to any period $t_0 < t$ so as to reduce capacity shortage without violating problem (P) constraints.

Starting from period τ+1 and ending at period T-1, the forward pass procedure follows the same logic as the backward procedure, but this time, the surplus quantity in period t is shifted to a period $t_0 > t$.

```
        for t = T downto (τ+2) do
        begin
step 1.   if S_t > 0 then
          begin
              for t_e = t-1 downto (τ+1) do
              begin
                  if S_max > S_te then
                  begin
                      for i =1 to m do
                      begin
step 2.                   Compute the shipment quantity of product i to shift from period t
                          to period t_e as follows:
                              Δ = min ( R_it , ⌊S_t / r_it⌋, ⌊(S_max – S_te) / r_ite⌋ );
                          if Δ > 0 then
                          begin
step 3.                       Compute the net increase on the objective function if the quantity Δ
                              of product i is shifted from period t to the period t_e as follows:
                                  ao = CT' – CT, {where
                                      CT:  is the objective function value of the current solution
                                      CT': is the objective function value of the solution obtained
                                           from the current solution by shifting a quantity Δ of
                                           product i from period t to the period t_e}
                              if ao < 0 then
                              begin
step 4.                           R_it = R_it - Δ,    R_ite = R_ite + Δ,
                              endif
                          endif
                      endfor
                  endif
              endfor
          endif
        endfor
```

Figure 5: Backward Pass of Improvement Procedure

To improve the quality of the feasible solution found by the perturbation procedure, a backward-forward improvement algorithm is then used. A description of this algorithm is given in Figure 5. Starting from period T and ending at period (τ+2), the backward pass of the improvement algorithm verifies for each period t whether the quantity of resources adsorbed (S_t) is greater than 0 (step 1). If $S_t > 0$, for each period t_e and for each product i, the quantity Δ of product i that can be shifted from period t to period t_e without violating the solution feasibility is calculated in step 2. The net increase in the objective function if the quantity Δ of product i is shifted from period t to period t_e is calculated in step 3. If the net increase is below 0, the lot sizes of product i in period t (R_{it}) and in period t_e (R_{it_e}) are respectively decreased and increased by the quantity Δ.

Starting from period τ+1 and ending at period T-1, the forward pass of the improvement algorithm procedure follows the same logic, but this time, the shift, if advantageous, is done from period t to a period $t_e > t$

5. Tests Results

Testing was done to compare the computational times and the quality of the solutions obtained using our Lagrangean heuristic procedure with those obtained using Cplex 9.0 libraries. We programmed our Lagrangean procedure in a Delphi 7.0 environment and implemented it on a Pentium III computer. Tests were performed on 320 problems of different sizes and with different characteristics. The maximum number of iteration parameter was set to k_{max} = 100 for all problems solved. Assuming a supply planning context, the experimental design was based on the following factors:

- *The structure of the cost function*. The transportation/purchasing cost function considered are those presented in Figure 6.

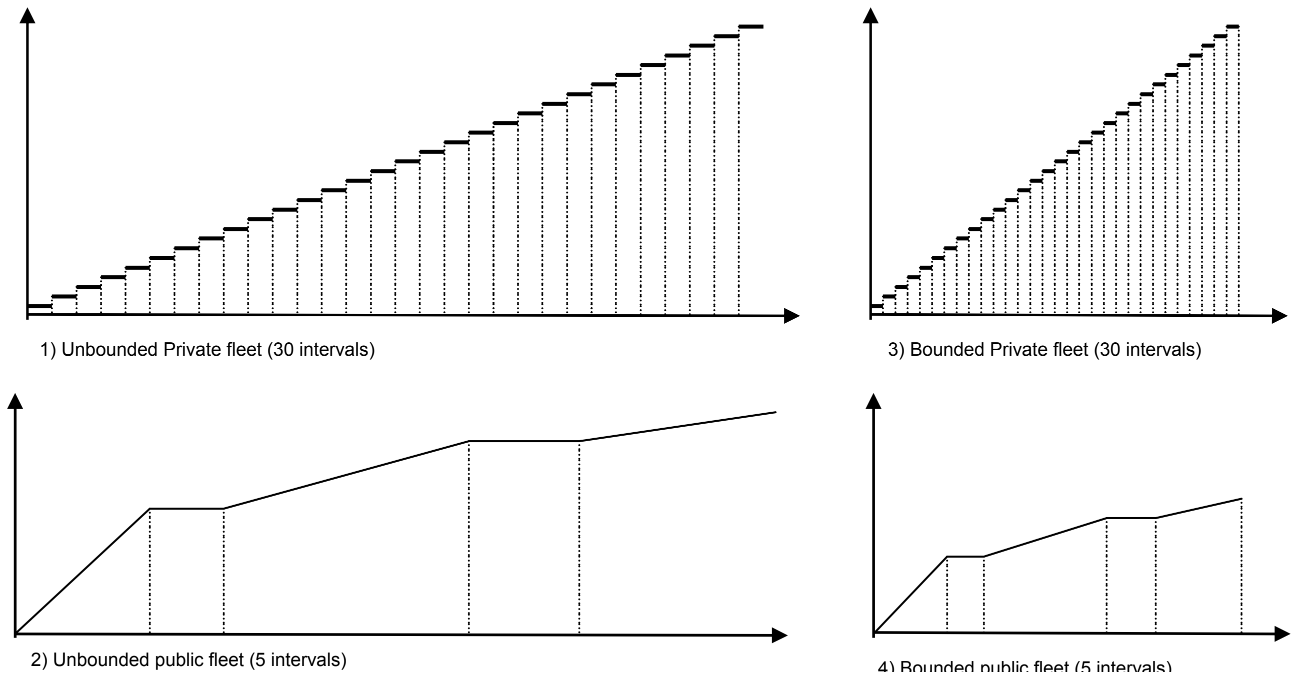

Figure 6: Structure of the cost functions used

- *The number of products*. Two groups of problems were studied. The first group has a number of products that is uniformly distributed between 50 and 100 ($m \sim U[50, 100]$). The number of products in the second group is uniformly distributed between 250 and 300 ($m \sim U[250, 300]$).
- *The number of periods*. Planning horizons of 30 and 60 periods were tested.
- *The variation of demands*. Product demands are generated from a Normal distribution ($d_i \sim N(\mu_i \sim U[10, 100]), \sigma_i = CV\mu_i$) where the coefficient of variation CV is equal to either 0.25 or 0.6 in order to evaluate the impact of demand variation amplitudes.
- *The importance of the inventory holding cost relative to the item cost*. Test problems with small inventory costs ($h_i \sim U[0.1\%c_i, 0.2\%c_i]$) and large inventory costs ($h_i \sim U[0.5\%c_i, 0.6\%c_i]$) relative to a time invariant purchasing cost c_i ($c_i \sim U[10, 40]$) are solved.

Overall we considered 4 different transportation/procurement cost functions, 2 different ranges for the number of products, 2 different planning horizon lengths, 2 different demand variations, and 2 different inventory costs scenarios. Thus, we have 64 (4×2×2×2×2) different problem instances. Since we generated five problems for each instance, 320 problems (5×64) were solved. All together, the problems solved can be classified in the 12 problem types described in Table1.

Notation	Description	Nbr. Instances
P1****	Private fleet with 30 intervals	80
P2****	Public fleet with 5 intervals	80
P3****	Private fleet with 30 intervals and an upper bound	80
P4****	Public fleet with 5 intervals and an upper bound	80
P*1***	Family of 50 to 100 products	160
P*2***	Family of 250 to 300 products	160
P**1**	30 periods horizon	160
P**2**	60 periods horizon	160
P***1*	Demands generated with CV = 0 .25	160
P***2*	Demands generated with CV = 0 .6	160
P****1	Inventory costs generated within [(0.1%)c_i, (0.2%)c_i]	160
P****2	Inventory costs generated within [(0.5%)c_i, (0.6%)c_i]	160

Table 1: Classification of the Problems Solved

With an upper time limit of 2 hours (7,200 seconds), Cplex 9.0 was not able to solve all our test problems to optimality. Therefore, to have a meaningful comparison, we present our results in two separate tables: Table 2 and Table 3. For the problems that we were able to solve to optimality with Cplex 9.0, Table 2 gives the mean and the coefficient of variation (standard deviation/$|$mean$|$) of the computational times and % deviations from the optimal solution cost. On the other end, Table 3 gives the mean and the coefficient of variation of the computational times and % deviations from the best solution cost obtained with Cplex 9.0 for the problems that the latter was not able to solve to optimality. The last column in both tables gives the percent improvement in solution time over Cplex 9.0 obtained by using the heuristics, i.e. 100[Time(Cplex) – Time(heuristics)]/Time(Cplex).

Problem type	Nbr. Instances *% of instances*	Cplex Sol: Aver. time (sec) *Coef. of var.*	Heuristic Sol: Aver. time (sec) *Coef. of var.*	Heuristic Sol: % Devi./opt.sol *Coef. of var.*	Heuristic Sol: % time imp./opt.sol *Coef. of var.*
Private fleet with 30 intervals (P1****)	60 *75.00%*	181.05 *563.95*	33.29 *34.25*	0.06 *1.06*	81.61 *0.63*
Public fleet with 5 intervals (P2****)	28 *35.00%*	2,495.42 *2,107.69*	15.23 *16.36*	0.95 *1.81*	99.39 *0.00*
Private fleet with 30 intervals & an upper bound (P3****)	80 *100.00%*	63.95 *60.41*	39.60 *43.47*	0.00 *17.94*	38.08 *2.41*
Public fleet with 5 intervals & an upper bound (P4****)	28 *35.00%*	1,946.83 *2,053.36*	15.20 *17.84*	1.10 *1.21*	99.22 *0.00*
Family of 50 to 100 products (P*1***)	100 *62.50%*	432.58 *753.70*	7.70 *8.26*	0.67 *2.31*	98.22 *0.28*
Family of 250 to 300 products (P*2***)	96 *60.00%*	1,011.50 *1,965.47*	54.66 *38.64*	0.16 *4.84*	94.60 *0.97*
30 periods horizon (P**1**)	136 *85.00%*	931.87 *1,725.55*	12.98 *13.03*	0.60 *2.43*	98.61 *0.78*
60 periods horizon (P**2**)	60 *37.50%*	227.13 *554.61*	70.87 *39.90*	0.00 *1.31*	68.80 *0.91*
Demands generated with CV = 0.25 (P***1*)	98 *61.25%*	627.84 *1,310.03*	30.79 *37.50*	0.47 *2.93*	95.10 *0.78*
Demands generated with CV = 0.6 (P***2*)	98 *61.25%*	804.43 *1,673.21*	30.61 *35.04*	0.37 *3.04*	96.19 *0.77*
Inventory costs generated within [(0.1%)c_i, (0.2%)c_i] (P****1)	105 *65.63%*	940.78 *1,826.06*	31.17 *34.63*	0.43 *3.01*	96.69 *0.62*
Inventory costs generated within [(0.5%)c_i, (0.6%)c_i] (P****2)	91 *56.88%*	456.92 *952.37*	30.16 *38.11*	0.41 *2.97*	93.40 *0.93*
All problems	196 *61.25%*	716.13 *1,505.22*	30.70 *36.29*	0.42 *2.99*	95.71 *0.78*

Table 2: Heuristic Results Compared to Cplex Optimal Solutions

The results indicate that our Lagrangean relaxation heuristic yields high quality solutions in a very short time. Table 2 shows that the heuristic solutions are obtained, on average, 23 times faster than Cplex 9.0 optimal solutions, while the cost percentage deviation from the optimal solutions is no more than 0.42% on average. For Table 3, the results show that our heuristic procedure was able to find solutions with costs that are, on average, 8.89 % lower than those of the solutions obtained with Cplex 9.0 after 2 hours of computation time. For Table 3 test problems, the heuristic procedure was, on average, 86 times faster than Cplex 9.0.

Problem type	Nbr. Instances *% of Instances.*	Cplex Sol Aver. time (sec) *Coef. of var.*	Heuristic Sol Aver. time (sec) *Coef. of var.*	% Devi./opt.sol *Coef. of var.*	% time imp./opt.sol *Coef. of var.*
Private fleet with 30 intervals (P1****)	20 *25.00%*	7,200.00	11.28 *3.31*	0.23 *0.63*	99.84 *0.00*
Public fleet with 5 intervals (P2****)	52 *65.00%*	7,200.00	96.19 *78.53*	-10.46 *1.50*	98.66 *0.01*
Private fleet with 30 intervals & an upper bound (P3****)	0 *0.00%*	N/A	N/A	N/A	N/A
Public fleet with 5 intervals & an upper bound (P4****)	52 *65.00%*	7,200.00	98.48 *82.57*	-10.83 *1.41*	98.63 *0.01*
Family of 50 to 100 products (P*1***)	60 *37.50%*	7,200.00	24.77 *11.92*	-0.31 *4.58*	99.66 *0.00*
Family of 250 to 300 products (P*2***)	64 *40.00%*	7,200.00	138.47 *78.15*	-16.93 *0.99*	98.08 *0.01*
30 periods horizon (P**1**)	24 *15.00%*	7,200.00	42.46 *5.94*	1.07 *2.15*	99.41 *0.00*
60 periods horizon (P**2**)	100 *62.50%*	7,200.00	93.29 *86.54*	-11.28 *1.37*	98.70 *0.01*
Demands generated with CV = 0.25 (P***1*)	62 *38.75%*	7,200.00	83.26 *80.92*	-8.58 *1.71*	98.84 *0.01*
Demands generated with CV = 0.6 (P***2*)	62 *38.75%*	7,200.00	83.65 *79.70*	-9.20 *1.60*	98.84 *0.01*
Inventory costs generated within $[(0.1\%)c_i, (0.2\%)c_i]$ (P****1)	55 *34.38%*	7,200.00	84.31 *78.06*	-10.00 *1.54*	98.83 *0.01*
Inventory costs generated within $[(0.5\%)c_i, (0.6\%)c_i]$ (P****2)	69 *43.13%*	7,200.00	82.77 *82.05*	-8.01 *1.76*	98.85 *0.01*
All problems	124 *38.75%*	7,200.00	83.45 *80.31*	-8.89 *1.65*	98.84 *0.01*

Table 3: Heuristic Results Compared to Cplex Best Solutions

A more detailed look at the results reveals that it is harder for Cplex to solve problems with public cost functions (LTL). Cplex was only able to solve 35% of instances with public cost functions to optimality within the time limit. However, Cplex was much more successful in solving to optimality, within the time limit, instances with private fleet cost structures (FTL). Thus, respectively 75% and 100% of instances with private fleet cost structure, without (P1****) and with (P3****) a constraining upper bound, were solved to optimality by Cplex.

As could be expected, the number of instances solved to optimality with Cplex depends on the number of products and on the length of the planning horizon. The impact of the planning horizon length is more pronounced than the others. Also, instances with higher inventory costs proved to be more difficult for Cplex to solve to optimality within the time limit. Cplex was able to solve to optimality 10% more instances when inventory holding costs are low (P****1) than when they are high (P****2). On the other end, the computation time and the quality of the solution obtained with our heuristic procedure does not seem to be affected very much by the different experimental factors considered, which indicates that the procedure proposed is very robust.

5. Conclusion

We have formulated a mixed-integer linear programming model to plan the supply of a family of items under piecewise linear resource costs. The model accommodates switches between transportation modes based on volumes, various types of discounts that may be offered in practice on transportation and/or purchasing prices, as well as convex or concave production capacity costs. Mathematical properties of a Lagrangean relaxation of the model were developed and a Lagrangean heuristic procedure that exploits these properties was presented. The Lagrangean heuristic appears to be efficient and robust for solving large real-life supply planning problems.

6. References

Atkins, D. and P. Iyogun, 1988, A Heuristic with Lower Bound Performance Guarantee for the Multi-Product Dynamic Lot-Size Problem, *IIE Transactions*, 20, 369-373.

Barany, I., T. Van Roy and L. Wolsey, 1984, Strong Formulations for Multi-Item Capacitated Lot Sizing, *Management Science*, 30, 11255-11261.

Bazaraa, M. S. and J. J. Goode, 1979, A Survey of Various Tactics for Generating Lagrangian Multipliers in the Context of Lagrangian Duality, *European Journal of Operational Research*, 3, 322-338.

Belvaux, G. and L. Wolsey, 2001, Modelling Practical Lot-Sizing Problems as Mixed-Integer Programs, *Management Science*, 47, 993-1007.

Bitran, G. and H. Yanasse, 1982, Computational Complexity of the Capacitated Lot Size Problem, *Management Science*, 28, 1174-1186.

Buffa, F. and J. Munn, 1989, A Recursive Algorithm for Order Cycle-time that Minimizes Logistics Cost, *Journal of the Operational Research Society*, 40, 367-377.

Cattrysse, D., J.Maes and L.N. VanWassenhove, 1990, Set Partitioning and Column Generation Heuristics for Capacitated Dynamic Lot Sizing, *European Journal of Operational Research*, 46, 38-47.

Chakravarty, A., 1984, Joint Inventory Replenishments with Group Discounts Based on Invoice Value, *Management Science*, 30, 1105-1112.

Diaby, M., H. Bahl, M. Karwan and S. Zionts, 1992, Capacitated Lot-Sizing and Scheduling by Lagrangean Relaxation, *European Journal of Operational Research*, 59-3, 444-458.

Diaby, M., and A. Martel, 1993, Dynamic Lot-Sizing in Multi-Echelon Distribution Systems with Purchasing and Transportation Price Discounts, *Operations Research,* vol 41-1, 48-59.

Fisher, M. L., 1981, The Lagrangian Relaxation Method for Solving Integer Programming Problems, *Management Science*, 22, 1-18.

Gelders, L., J. Maes and L.N. VanWassenhove, 1986, A Branch and Bound Algorithm for the Multi-Item Single Level Capacitated Dynamic Lotsizing Problem, in Axsater, Schneeweiss and Silver (eds), *Multi-Stage Production Planning and Inventory Control*, Springer-Verlag, 92-108.

Haseborg, F. , 1982, On the Optimality of Joint Ordering Policies in a Multi-Product Dynamic Lot Size Model with Individual and Joint Set-up Costs, *European Journal of Operational Research*, 9, 47- 55.

Kao, E., 1979, A Multi-Product Dynamic Lot-Size Model with Individual and Joint Set-up Costs, *Operations Research,* 27, 279-289.

Maes, J. and L. Van Wassenhove, 1986, Multi Item Single Level Capacitated Dynamic Lotsizing Heuristics: A Computational Comparison (Part I and II), *IIE Transactions*, 18, 114-129.

Pochet, Y. and L.A. Wolsey, 1991, Solving the Multi-item Lot-sizing Problems Using Strong Cutting Planes, *Management Science*, 37, 53-67.

Silver, E., 1979, Coordinated Replenishments of Items Under Time-varying Demand: Dynamic Programming Formulation, *Naval Research Logistics Quarterly,* 26, 141-151.

Silver, E. and R. Peterson, 1985, Decision Systems for Inventory Management and Production Planning, 2nd ed., Wiley.

Sox, C.R. and Y. Gao, 1999, The Capacitated Lot Sizing Problem with Setup Carry-over, *IIE Transactions*, 31, 173-181.

Swenseth, S.R. and M.R. Godfrey, 2002, Incorporating Transportation Costs into Inventory Replenishment Decisions, *International Journal of Production Economics*, 77, 113–130.

Thizy, J. and L. Van Wassenhove, 1985, Lagrangean Relaxation for the Multi-item Capacitated Lot-Sizing Problem: A Heuristic Implementation, *IIE Transactions*, 17, 308-313.

Trigeiro, W., J. Thomas and J. McClain, 1989, Capacitated Lot Sizing with Setup Times, *Management Science*, 35, 353-366.

van Norden, L. and S. van de Velde, 2005, Multi-product lot-sizing with a transportation capacity reservation contract, *European Journal of Operational Research*, 165, 127-138.

Veinott, A. F., 1969, Minimum Concave-Cost Solution of Leontief Substitution Models of Multi-facility Inventory Systems, *Operations Research,* 17, 262-291.

Wagner, H. and T.M. Whitin, 1958, Dynamic Version of the Economic Lot Size Model, *Management Science*, 5, 89-96.

Zangwill, W.I., 1966, A Deterministic Multiproduct, Multifacility Production and Inventory Model, *Operations Research,* 14, 486-507.

Zoller, K. and A. Robrade, 1988, Efficient Heuristics for Dynamic Lot Sizing, *International Journal of Production Research*, 26, 249-265.