Algorithms for Lot-sizing with Supplier Selection

Diego Klabjan
Department of Industrial Engineering and Management Sciences
Northwestern University
Evanston, IL
d-klabjan@northwestern.edu

June 20, 2008

Abstract

The traditional lot-sizing problem is to find the least cost production lot-sizes in several time periods. We consider the lot-sizing model together with simultaneous selection of suppliers, which have variable and fixed cost. We first show that the resulting problem is polynomially solvable in presence of equal capacities on production and supply. We also develop a fully polynomial time approximation scheme.

1 Introduction

The lot-sizing problem has been extensively studied in the past. Wagner and Whitin's dynamic programming algorithm for solving the single-item single-stage lot-sizing problem was one of the first important results on the subject, Wagner and Whitin (1958). Since this seminal work, the model has been enhanced mostly in the direction of considering multi-item multi-stage problems.

In this work we consider the single-item single-stage lot-sizing problem with supplier selection. We assume that a set of suppliers is given and in each time period we decide lot-sizes and a subset of suppliers to use. With each supplier we associate the variable cost corresponding to the actual cost of the material and the fixed cost of using a particular supplier. Let $T = \{1, \dots, t\}$ be the set of production periods and let $N = \{1, \dots, n\}$ be the set of suppliers. The single-item lot-sizing problem with supplier selection (LSSS) is formulated as the following mixed integer program

$$\min \sum_{i \in T} h_i s_i + \sum_{i \in T} \sum_{j \in N} (p_i + c_{ji}) w_{ji} + \sum_{i \in T} fp_i y_i + \sum_{i \in T} \sum_{j \in N} fs_{ji} z_{ji}$$

$$\begin{aligned}
s_{i-1} + x_i &= d_i + s_i && i \in T \\
x_i &\le C_i y_i && i \in T && (1) \\
x_i &= \sum_{j \in N} w_{ji} && i \in T && (2) \\
w_{ji} &\le K_{ji} z_{ji} && i \in T, j \in N && (3) \\
s_0 &= s_t = 0 \\
x \ge 0, w &\ge 0, s \ge 0 \\
&\text{y binary, z binary.}
\end{aligned}$$

Here, x_i, s_i represent the lot size and stock in period i, y_i indicates whether a production set-up cost must be incurred in period i, w_{ji} represents the amount sourced from supplier j in period i, and z_{ji} indicates whether a fixed sourcing cost must be incurred with supplier j in period i. Quantities h_i, p_i, fp_i, and d_i are the holding cost, variable production cost, production set-up cost, and demand in period i, respectively. Values c_{ji} and fs_{ji} represent the variable and fixed sourcing set-up cost for supplier j in period i. C's and K's are production and supplying capacities, which without loss of generality we assume are integral. This model assumes that for each unit we need one unit of supply. Note that this is without loss of generality since otherwise we scale w and adjust K's accordingly. We assume that d_i is a positive integer for each $i \in T$. If $C_i = \infty$ for every i, we say that the problem is production uncapacitated. Similarly, if $K_{ji} = \infty$ for every i and j, we say that the problem is supplier uncapacitated.

This model is clearly an extension of the single-item single-stage model. On the other hand, it is a special case of the two stage model, where inventory at the second stage is not present. We first present a polynomial algorithm for the case where all production capacities are equal and all supplier capacities are identical. We also develop a fully polynomial time approximation scheme (FPTAS). The scheme is an extension of the FPTAS for lot-sizing given by Van Hoesel and Wagelmans (2001). Several non-trivial extensions are required since the sourcing problem itself is NP-hard. We first give conditions under which a lot-sizing problem with non-polynomially computable production and holding cost functions exhibits an FPTAS. Next we argue that these conditions hold for our problem.

Our model without the fixed production cost, and supplier and production uncapacitated is studied in Aghezzaf and Wolsey (1994). Bhatia and Palekar (2001) give a description of the extreme vertices for the same case. Belvaux and Wolsey (2001) introduced models for various practical lot-sizing problems and a specialized branch-and-cut optimization system. Federgruen and Tzur (1991), Wagelmans *et al.* (1992), and Aggarwal and Park (1993) give improved algorithms with a $O(t \log t)$ running time for the general uncapacitated problem, where t is the number of time periods. Although the general case is NP-hard as shown in Florian *et al.* (1980), Florian and Klein (1971) and van Hoesel and Wagelmans (1996) show that there exists a polynomial algorithm if the order capacities are constant.

In Section 2 we give a polynomial algorithm for the equal capacities case. The FPTAS is given in Section 3.

2 Polynomial Algorithm for Production and Supplier Equal Capacitated Case

We start with two observations that establish the computational complexity of LSSS.

Proposition 1. *If LSSS is production uncapacitated and $K_{ji} = K_j$ for every time period i, i.e. the suppliers capacities do not vary with time, then LSSS is NP-hard.*

Proof. Given rational vectors u, v, a and a rational number b, the single node fixed-charge problem reads

$$\begin{aligned}\min \sum_{j\in M} u_j w_j + \sum_{j\in M} v_j z_j & \\ \sum_{j\in M} w_j = b & \\ w_j \le a_j z_j \qquad & j \in M \\ w \ge 0, z \text{ binary.} &\end{aligned} \tag{4}$$

This problem is NP-hard, see, e.g., Klose (2008).

Consider now a single time period, i.e. $t = 1$. Then the production uncapacitated LSSS problem is equivalent to the single node fixed-charge problem. In (4) it suffices to consider $M = N$, $a_j = K_j$, $b = d_1$, and $u_j = p_1 + c_{j1}, v_j = fs_{j1}$. □

Proposition 2. *If LSSS is supplier uncapacitated but production capacitated, the problem is NP-hard.*

Proof. Consider now a single supplier. Then the supplier uncapacitated LSSS problem is equivalent to the single-item lot-sizing problem with production cost $p_i + c_{1i}$ during time period i and production setup cost $fp_i + fs_{1i}$ for every time period i. This problem is NP-hard as shown in Florian *et al.* (1980). □

In the reminder of this section, we focus on the case where $C_i = C$ for all i and $K_{ji} = K$ for all j and i. If $K > C$, then we can set $K = C$ without affecting optimality. Thus, without loss of generality, we assume $K \le C$. For any $i \in T, j \in T, j \ge i$ we denote $d_{ij} = \sum_{k=i}^{j} d_k$.

Our polynomial algorithm relies on the Wagner-Whitin algorithm and it is based on dynamic programming. It is also an extension of the equal capacity dynamic program presented in Florian and Klein (1971), i.e. in their work $K = \infty$ or $n = 1$. Florian and Klein's algorithm is based on dynamic programming and a network representation of the myopic production cost. We follow this framework; however, the myopic production cost problem in our case is more complicated and special treatment is required.

Definition 1. *Time periods $u, v, u < v$ form a production sequence, denoted by P_{uv}, if every optimal production schedule in time periods $u, u+1, \ldots, v$ with $s_u = s_v = 0$ has positive inventory in the intermediate time periods, i.e., $s_{u+1} > 0, s_{u+2} > 0, \ldots, s_{v-1} > 0$.*

A regeneration point is a time period u with $s_u = 0$ for an optimal solution. It is clear that the production decision for periods after a regeneration point is independent of the production decision prior to that regeneration point. Every regeneration point essentially decomposes the problem into two sub-problems. If u is a regeneration point, then an optimal solution to the problem can be found independently by finding solutions to the problem for the first u periods and the last $t - u$ periods. Furthermore, every optimal solution can be decomposed to several production sequences (each production sequence starts and finishes with a regeneration point). Let F_u be the cost associated with an optimal solution over periods $u+1, \ldots, t$, and b_{uv} be the cost associated with a production sequence over periods $u+1, \ldots, v$. We have the following recursion:

$$\begin{aligned} F_t &= 0 \\ F_u &= \min_{u<v\le t} \{b_{uv} + F_v\} \qquad u = 0, \ldots, t-1. \end{aligned}$$

If we can calculate b_{uv} in polynomial time for given u and v, we can unwind this recursion to calculate F_0, which yields an optimal solution to LSSS. The Wagner-Whitin algorithm and the algorithm from Florian and Klein (1971) use the same recursive relationship. In these two algorithms it is relatively easy to derive a polynomial algorithm for evaluating b_{uv}. In our case, due to the presence of several suppliers, it is more difficult to derive such an algorithm. Next we present a polynomial algorithm for calculating b_{uv}.

Lemma 1. *For each production sequence P_{uv} there exists an optimal solution such that there is at most one time period i with $u+1 \leq i \leq v$ in which x_i does not equal either $0, C$, or some multiple of K.*

Proof. First we model production sequence P_{uv} as a network, see Figure 1. Nodes $u+1, \dots, v$ correspond to the production periods, and nodes $SP_1, \dots, SP_n$ correspond to the suppliers. The flow differences in the former nodes $u+1, \dots, v$ equal to $d_{u+1}, \dots, d_v$. For each period $i, u+1 \leq i \leq v$, we add node i' and arc (i', i) to the network. The flow on such an arc corresponds to the production lot x_i. The production period nodes are connected in the usual way. Every node SP_j is connected with every node i', and the flow on this arc corresponds to the amount of supply w_{ji}. Next we add to the network a node 0 connected to all SP_j nodes with $d_{u+1,v}$ amount of flow going out of it. The remaining nodes must preserve flow. The cost of every arc is assigned according to the underlying decision variable.

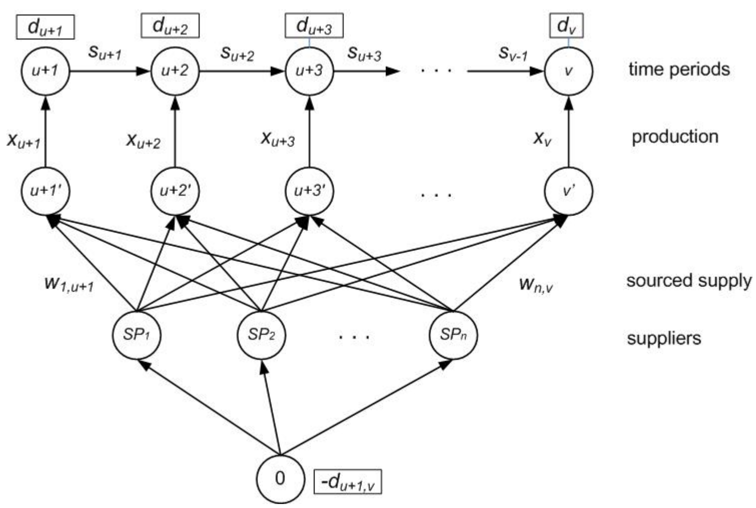

Figure 1: Network

We consider an optimal production sequence P_{uv}. Let x^*, w^*, s^*, y^*, z^* be the corresponding optimal solution. Arcs (i', i) with $y_i^* = 1$ have capacity C and arcs (SP_j, i') with $z_{ji}^* = 1$ have capacity K. All remaining (i', i) and (SP_j, i') arcs have 0 capacity. All other arcs have an infinite capacity. Flow implied by s^*, x^* and w^* corresponds to a solution to the minimum cost network flow problem given in Figure 1.

There must exist an extreme or basic feasible solution to this minimum cost network flow model, and therefore there exists an optimal solution in which x and w correspond to a cycle free solution. Since P_{uv} is a production sequence, the corresponding inventory levels satisfy $s_{u+1} > 0, s_{u+2} > 0, \dots, s_{v-1} > 0$. Suppose we have two periods i and j, $i < j$ with x_i and x_j strictly between 0 and C, and K divides neither x_i nor x_j. Since $x_i = \sum_{k \in N} w_{ki}$, it follows that K does not divide $\sum_{k \in N} w_{ki}$. Together with $0 \leq w_{ki} \leq K$ for each k, we conclude that there exists a supplier a such that $0 < w_{ai} < K$. Similarly, there exists a supplier b such that $0 < w_{bj} < K$. By definition of a production sequence positive flow is present on all arcs corresponding to s_p, $i \leq p < j$ (the subset of inventory arcs for time periods p). The cycle consisting of arcs $(0, SP_a)$, (SP_a, i'), (i', i), $(p, p+1)$ for $i \leq p < j$, (j', j), (SP_b, j') and $(0, SP_b)$ is composed entirely of arcs neither at the lower bound nor at the upper bound. This is a contradiction to the cycle free property. □

Let P_{uv} be a production sequence and $X_j = \sum_{i=u+1}^{j} x_i$, $j = u+1, \ldots, v$. We assume we have a solution satisfying the property in Lemma 1. A consequence of Lemma 1 is that each X_j can only take on a finite number of values. More importantly, this finite number is polynomial with respect to $v - u$ and n. The total production $d_{u+1,v}$ equals to $Cm + K\overline{m} + \epsilon$ for unknown integers $m, \overline{m}$, and ϵ with $0 \leq m \leq v - u$, $0 \leq \overline{m} \leq n(v-u)$, $0 \leq \epsilon < K$. Observe that $\epsilon = (d_{u+1,v} - Cm) \mod K$. Since $0 \leq m \leq v - u$, there are at most $v - u$ possible values of ϵ, which depend on m. We write $\epsilon_m = (d_{u+1,v} - Cm) \mod K$ to reflect this. We conclude that there is an optimal production schedule that produces C in m time periods, K in $\overline{m}$ time periods, and ϵ_m in a single time period. All other lot sizes are 0.

Let L_j be the set of feasible values for X_j. This set includes all values of the form $Cm + Km' + \epsilon_{\bar{m}}$ such that $0 \leq m \leq j - u$, $0 \leq m' \leq n(j-u)$, and $0 \leq \bar{m} \leq v - u$.

In order to calculate b_{uv}, we construct an auxiliary acyclic network in which $s-t$ paths represent feasible solutions to the production sequence P_{uv} that satisfy the property in Lemma 1. If the length of an $s - t$ path equals to the total cost of the corresponding production feasible solution, then solving the shortest path problem on this auxiliary network yields an optimal solution to the production sequence P_{uv}.

There is a one-to-one correspondence between each node in the auxiliary network and $X_j = l$, where $l \in L_j, u+1 \leq j \leq v$. We label this node as (j, l). For each node (j, l) with $l = C \cdot m + K \cdot m'$, we add an arc from (j, l) to $(j+1, l')$ for each

$$\begin{aligned} l' \in \{C(m+1) + K \cdot m'\} &\cup \{C \cdot m + K(m'+i) | 0 \leq i \leq n\} \\ &\cup \{C \cdot m + K(m'+i) + \epsilon_{\bar{m}} | 0 \leq i \leq n, 0 \leq \bar{m} \leq j - u\} . \end{aligned}$$

The first option corresponds to the case of producing C units in time period $j+1$, and the remaining two options correspond to the case when the lot size is less than C. In the latter case, we can either produce a multiple of K (the second case), or a multiple of K and the fractional part $\epsilon_{\bar{m}}$ for an unknown $\bar{m}, 0 \leq \bar{m} \leq j - u$ (the third case). For each node (j, l) with $l = C \cdot m + K \cdot m' + \epsilon_{\bar{m}}$ for some $\bar{m}$, we add an arc from (j, l) to $(j+1, l')$ for

$$l' \in \{C(m+1) + K \cdot m' + \epsilon_{\bar{m}}\} \cup \{Cm + K(m'+i) + \epsilon_{\bar{m}} | 0 \leq i \leq n\} .$$

The first case corresponds to producing exactly C units and the second case corresponds to producing less than C, which implies that the production must be a multiple of K. Since l already includes $\epsilon_{\bar{m}}$, it means that the "fractional" production $\epsilon_{\bar{m}}$ has already occurred before time period j. Therefore by Lemma 1 in the remaining time periods we can produce only C or a multiple of K.

To complete the construction, we add to the network a source node s and a sink node t. Node s has outgoing arcs to every node $(u+1, l)$, $l \in L_{u+1}$. In order to obtain $s-t$ paths that correspond to feasible production sequences, there is an arc from $(v, l), l \in L_v$, to node t only if $l = Cm + Km' + \epsilon_{\bar{m}}$

with $m' = \lfloor \frac{d_{u+1,v} - Cm}{K} \rfloor$ (this guarantees that the total production is $d_{u+1,v}$). This network has at most $O(nt^4)$ nodes (there are at most t options for j and nt^3 possible values for l) and therefore at most $O(n^2t^8)$ edges. By construction, this network is acyclic.

If a node (j, l) with $l = C \cdot m + K \cdot m' + \epsilon_{\bar{m}}$ for some $\bar{m}$ is on an $s - t$ path, any subsequent node $(j', l'), j' > j$ along the path will have $l' = C(m + i) + K(m' + i') + \epsilon_{\bar{m}}$ for some i, i'. Thus our construction guarantees that each $s - t$ path has at most one node $(j, C \cdot m + K \cdot m' + \epsilon_{\bar{m}})$ and therefore each $s - t$ path corresponds to a solution satisfying the property in Lemma 1 of the production sequence P_{uv}.

If we assign to arc $((j, a), (j + 1, b)), a \in L_j, b \in L_{j+1}$ a weight corresponding to the minimum cost to produce $b - a$ units in period $j + 1$, solving the shortest path problem from the source to the sink gives us an optimal solution to production sequence P_{uv}.

The cost of producing $b - a$ units in period $j + 1$ is the sum of the sourcing cost and the cost of actually manufacturing $b - a$ units. The latter equals to $h_{j+1} \cdot (b - d_{u+1,j+1}) + p_{j+1} \cdot (b - a) + \sigma_{a,b} \cdot fp_{j+1}$, where $\sigma_{a,b}$ is 1 if $b > a$ and 0 otherwise. The sourcing cost is obtained by solving

$$
\begin{aligned}
\min \sum_{k \in N} c_{k,j+1} w_{k,j+1} + \sum_{k \in N} fs_{k,j+1} z_{k,j+1} & \\
\sum_{k \in N} w_{k,j+1} = b - a & \\
w_{k,j+1} \leq K \cdot z_{k,j+1} & \qquad k \in N \\
w \geq 0, \text{z binary.} &
\end{aligned}
$$

This problem can be solved in $O(n^2)$ time, see e.g. Padberg *et al.* (1985).

Example. To illustrate the construction of the auxiliary network, consider the following example with just 2 periods in the production sequence. Let the production sequence be the periods $u + 1$ and $u + 2$ with $d_{u+1} = 6, d_{u+2} = 5$, and $C = 10, K = 4, n = 3$. Let $fs_{1,u+i} = 1$, $fs_{2,u+i} = 2$, $fs_{3,u+i} = 3$, $c_{1,u+i} = 6$, $c_{2,u+i} = 8$, $c_{3,u+i} = 10$ for $i = 1, 2$. In addition, $h_{u+1} = 3, p_{u+1} = 8, p_{u+2} = 10, fp_{u+1} = 5, fp_{u+2} = 6$. We obtain the following L_{u+1} and L_{u+2}:

$$
\begin{aligned}
L_{u+1} &= \{(1,0,0),(1,0,1),(0,1,2),(0,1,3),(0,2,0)(0,2,1),(0,2,2),(0,2,3)\}, \\
L_{u+2} &= \{(1,0,1),(0,2,3)\}.
\end{aligned}
$$

Here the triplet (m, m', ϵ) encodes $Cm + Km' + \epsilon$. We do not show elements of L_{u+1} and L_{u+2} whose value exceeds the total demand in the production sequence (11 in our case).

The complete auxiliary network is shown in Figure 2. The cost calculation is tedious and we give only numbers. Nodes $(u + 1, (0, 1, 2)), (u + 1, (0, 2, 1)), (u + 1, (0, 2, 2))$ do not have outgoing arcs since there are no arcs connecting two nodes with a positive ϵ. The highlighted path is the shortest $s - t$ path with total cost 186. This corresponds to producing 7 units in period $u + 1$ and 4 units in period $u + 2$. □

The overall running time of our dynamic programming algorithm for solving LSSS is $O(n^6t^8)$. For each b_{uv} we need $O(n^4t^8)$ steps to construct the network. The shortest path problem can be solved in the same amount of time since the network is acyclic. We need an extra $O(n^2)$ to solve the dynamic program. This algorithm is therefore polynomial.

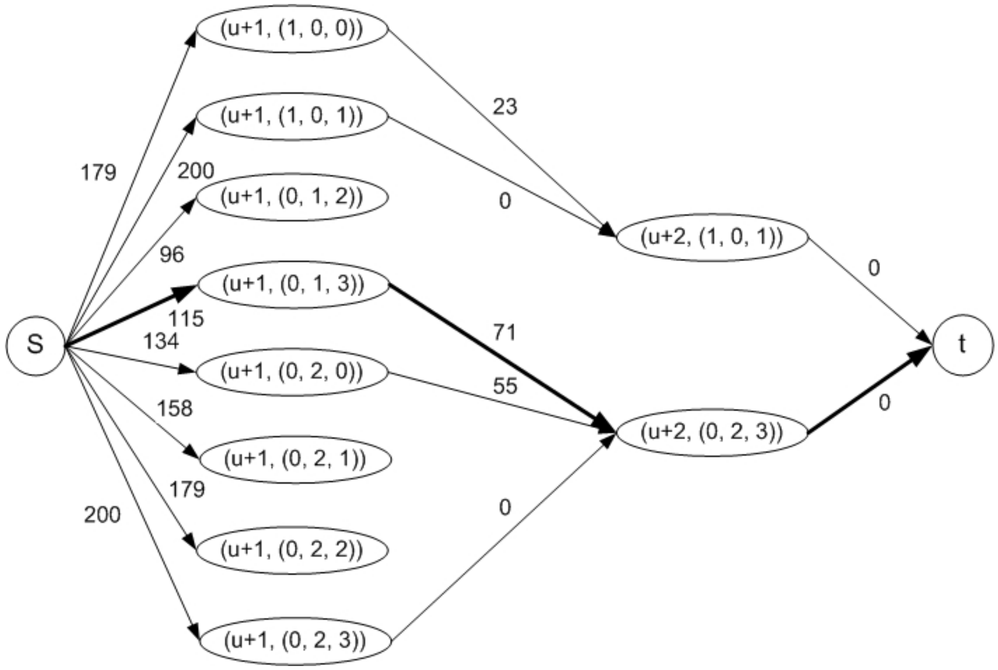

Figure 2: Auxiliary Network

3 A Fully Polynomial Time Approximation Scheme

In this section we present an FPTAS for the lot-sizing problem with supplier selection. The scheme relies on the FPTAS for standard lot-sizing developed by Van Hoesel and Wagelmans (2001). However, the extension to handle suppliers is nontrivial. The FPTAS from Van Hoesel and Wagelmans (2001) requires that given a lot size we can determine in polynomial time the production and holding costs. This is not the case for our problem. Given x, y satisfying (1), it is NP-hard to minimize an arbitrary objective function subject to constraints (2) and (3).

More recently Chubanov *et al.* (2006) developed a different FPTAS for the single-item lot-sizing problem. They also require that the production and holding costs can be determined in polynomial time and therefore their FPTAS cannot be directly applied to our variant of lot-sizing.

We first extend the FPTAS for standard lot-sizing. The new FPTAS requires only an existence of a pseudo polynomial algorithm for evaluating the production and holding costs. However, it does not allow backlogging. In the second part we show that our problem fits into this framework by exhibiting a pseudo polynomial algorithm.

The main idea behind the work of Van Hoesel and Wagelmans (2001) is to develop a dynamic program for the "dual problem" of the lot-sizing problem. The dual problem maximizes the inventory level while keeping cost below a given budget. In their algorithm, first an upper bound on the optimal objective value is obtained. Next the dual problem is solved by dynamic programming on a subset of all possible states. The resulting solution is then output.

3.1 A Fully Polynomial Time Approximation Scheme for Lot-Sizing with Pseudo Polynomial Production and Holding Cost Algorithms

Here we consider the lot-sizing problem

$$\begin{aligned} Z^* = \min \sum_{i=1}^{t} \bar{p}_i(x_i) + \sum_{i=1}^{t} \bar{h}_i(s_i) & \\ s_i = s_{i-1} + x_i - d_i & \qquad i \in T \\ 0 \le x_i \le C_i & \qquad i \in T \\ s_0 = s_t = 0 & \\ s \ge 0, x \text{ integer.} & \end{aligned}$$

We require the following assumptions.

Assumption A1. *The production cost function $\bar{p}_i$ is non-decreasing on $[0, C_i]$ and $\bar{p}_i(0) = 0$ for every $i \in T$.*

Assumption A2. *The holding cost function $\bar{h}_i$ is non-decreasing on $[0, \infty)$ and $\bar{h}_i(0) = 0$ for every $i \in T$.*

Assumption A3. *For any integer $x_i \in [0, C_i]$ and integer $M \ge 0$, the problem "Is $\bar{p}_i(x_i) \le lM$ for $l \in \mathbb{Z}_+$?" can be answered in time $\mathcal{O}(P(l))$, where P is a polynomial.*

Assumption A4. *For every integer $s_i \in [0, \infty)$ and integer $M > 0$, the problem "Is $\bar{h}_i(s_i) \le lM$ for $l \in \mathbb{Z}_+$?" can be answered in time $\mathcal{O}(H(l))$, where H is a polynomial.*

Assumption A5. *There exists a constant $1 > \Lambda \ge 0$ and a polynomial Λ-approximation algorithm for evaluating $\bar{p}_i$. Formally, given x, the polynomial approximation algorithm returns an integer $\tilde{p}_i(x)$ such that $\frac{|\tilde{p}_i(x) - \bar{p}_i(x)|}{\bar{p}_i(x)} \le \Lambda$.*

Assumption A6. *There exists a constant $1 > \Omega \ge 0$ and a polynomial Ω-approximation algorithm for evaluating $\bar{h}_i$. Formally, given x, the polynomial approximation algorithm returns an integer $\tilde{h}_i(x)$ such that $\frac{|\tilde{h}_i(x) - \bar{h}_i(x)|}{\bar{h}_i(x)} \le \Omega$.*

By using the framework from Van Hoesel and Wagelmans (2001), we give an FPTAS. Assumptions A1 and A2 are used by Van Hoesel and Wagelmans (2001), but, instead of Assumptions A3-A6 they require a polynomial evaluation of $\bar{p}_i$ and $\bar{h}_i$. Note also that in their setting, they allow backlogging while we do not. Assumptions A3 and A4 require that there exists a pseudo polynomial algorithm for evaluating $\bar{p}_i$ and $\bar{h}_i$, respectively. We point out that assumptions Assumptions A3-A6 are weaker than requiring that $\bar{h}_i, \bar{p}_i$ exhibit an FPTAS.

3.1.1 The Dynamic Program

Let B be an integer upper bound on Z^*. For every $i \in T$ and $b \in [B]$, let $F_i(b)$ be the maximum inventory at the end of time period i given the maximum budget b available for time periods 1 up to i. The budget is the sum of the production and holding cost. Van Hoesel and Wagelmans (2001) give the following recursion with the initial condition $F_0(b) = 0$ for every $b \ge 0$.

$$\begin{aligned} F_i(b) = \max_{0 \le a \le b} \{ \max \{ & \max_{\max\{0, d_i - F_{i-1}(a)\} \le x_i \le C_i} \{ F_{i-1}(a) + x_i - d_i | \bar{p}_i(x_i) + \bar{h}_i(F_{i-1}(a) + x_i - d_i) \\ & \le b - a \}, \max_{0 \le s_i \le F_{i-1}(a) - d_i} \{ s_i | \bar{h}_i(s_i) \le b - a \} \} \} \end{aligned} \tag{5}$$

Let a be a certain budget allocation to the first $i-1$ time periods. In order for the budget in the first i time periods to be less than or equal to b, the cost incurred in time period t must not exceed $b-a$.

Consider first the case when there exists a production quantity x_i such that $\bar{p}_i(x_i)+\bar{h}_i(F_{i-1}(a)+x_i-d_i) \leq b-a$. Then clearly the inventory level after i time periods equals to $F_{i-1}(a)+x_i-d_i$. If $d_i - F_{i-1}(a) > x_i$, then $F_{i-1}(a)+x_i-d_i < 0$ and thus we would incur a negative inventory level, which is not allowable by Assumptions A2. We conclude that $\max\{0, d_i - F_{i-1}(a)\} \leq x_i \leq C_i$. This establishes the first term in (5).

Let now

$$\bar{p}_i(x_i) + \bar{h}_i(F_{i-1}(a) + x_i - d_i) > b - a \text{ for every production quantity } x_i, 0 \leq x_i \leq C_i\,. \tag{6}$$

Let $\tilde{s}_{i-1}, \tilde{x}_i$ be the inventory after the first $i-1$ time periods and the corresponding production plan in time period i, respectively, under the condition of not exceeding budget b in the first i time periods and budget a in the first $i-1$ time periods, and such that $\tilde{s}_{i-1}+\tilde{x}_i-d_i$ is maximized. By definition $\tilde{s}_{i-1} \leq F_{i-1}(a)$. If $\tilde{s}_{i-1}+\tilde{x}_i > F_{i-1}(a)$, then consider an alternative plan with $\bar{s}_{i-1} = F_{i-1}(a)$ and $\bar{x}_i = \tilde{x}_i - F_{i-1}(a) + \tilde{s}_{i-1}$. Then $0 \leq \bar{x}_i \leq \tilde{x}_i \leq C_i$. We also have $\bar{p}_i(\bar{x}_i) + \bar{h}_i(F_{i-1}(a)+\bar{x}_i-d_i) \leq \bar{p}_i(\tilde{x}_i)+\bar{h}_i(F_{i-1}(a)+\tilde{x}_i-d_i) \leq b-a$. The first inequality follows from the non-decreasing property of production costs and the second one by the choice of $\tilde{s}_{i-1}, \tilde{x}_i$. This is a contradiction to (6). We conclude that $\tilde{s}_{i-1}+\tilde{x}_i \leq F_{i-1}(a)$.

Since $\tilde{s}_{i-1}+\tilde{x}_i \leq F_{i-1}(a)$, it is easy to see that there is a production plan in the first $i-1$ time periods that does not exceed budget a and with $s_{i-1} = \tilde{s}_{i-1}+\tilde{x}_i$. Since $a + \bar{h}_i(\tilde{s}_{i-1}-d_i) \leq a + \bar{h}_i(\tilde{s}_{i-1}+\tilde{x}_i-d_i) + \bar{p}_i(\tilde{x}_i) \leq b$, it follows that by not producing anything in time period i we do not exceed budget b. This establishes the second term in (5).

Let now M be an integer with $0 < M \leq B$ and let $l \in \mathbb{Z}_+$. We define also $\bar{F}_i(l)$ to be the maximum inventory level s_i in the first i time periods such that the budget does not exceed lM and in each time period the production cost does not exceed a multiple of M and the holding cost does not exceed a multiple of M.

If follows from (5) that we can compute $\bar{F}_i$ based on the following recursion ($\bar{F}_0(l) = 0$ for every $l \geq 0$).

$$\begin{aligned}\bar{F}_i(l) = \max_{0\leq q\leq l}\{\max\{ &\max_{\substack{0\leq p\leq l-q\\0\leq w\leq l-q\\p+w\leq l-q}} \max_{\max\{0,d_i-\bar{F}_{i-1}(q)\}\leq x_i\leq C_i}\{\bar{F}_{i-1}(q)+x_i-d_i|\bar{p}_i(x_i)\leq pM,\\ &\bar{h}_i(\bar{F}_{i-1}(q)+x_i-d_i)\leq wM\}, \max_{0\leq s_i\leq \bar{F}_{i-1}(q)-d_i}\{s_i|\bar{h}_i(s_i)\leq (l-q)M\}\}\}\end{aligned}$$

Let $l = 0, 1, \ldots, L$ for an integer L. Then $\bar{F}_i(l)$ can be computed in time

$$\begin{aligned}&\mathcal{O}(L^4(P(L)+H(L))\cdot\sum_{i=1}^{t}\log C_i + L^2\cdot H(L)\cdot\sum_{i=1}^{t}\log(\sum_{\bar{i}=1}^{i}C_{\bar{i}}))\\ &\qquad = \mathcal{O}(tL^4(P(L)+H(L))\log C_{max},\end{aligned} \tag{7}$$

where $C_{max} = \max_i C_i$. To see this, note that

$$\max_{\max\{0,d_i-\bar{F}_{i-1}(q)\}\leq x_i\leq C_i}\{\bar{F}_{i-1}(q)+x_i-d_i|\bar{p}_i(x_i)\leq pM, \bar{h}_i(\bar{F}_{i-1}(q)+x_i-d_i)\leq wM\}$$

can be computed in $\mathcal{O}(\log C_i \cdot (P(L)+H(L)))$ time by bisection. Likewise, since $\bar{F}_{i-1}(q) \leq \sum_{\bar{i}=1}^{i} C_{\bar{i}}$, the optimization problem $\max_{0 \leq s_i \leq \bar{F}_{i-1}(q)-d_i} \{s_i | \bar{h}_i(s_i) \leq (l-q)M\}$ can be solved in $\mathcal{O}(\log(\sum_{\bar{i}=1}^{i} C_{\bar{i}}) \cdot H(L))$ time.

The following proposition extends the result from Van Hoesel and Wagelmans (2001).

Proposition 3. *Let*

$$l^* = \min_{l=0,1,\ldots,L} \{l | \bar{F}_t(l) \geq 0\}$$

where $L = \lfloor \frac{B}{M} \rfloor + t + 1$. *Then* $l^* M \leq Z^* + tM$.

Proof. Consider an optimal solution and let $prod_i, hold_i$ be the underlying production and holding cost in time period i. Note that $prod_i \leq V_i = (\left\lfloor \frac{prod_i}{M} \right\rfloor + 1)M$ and $hold_i \leq U_i = (\left\lfloor \frac{hold_i}{M} \right\rfloor + 1)M$. Thus allocating a production budget of V_i and a holding cost of U_i in each time period yields positive inventory at the final time period t. The total cost of the optimal solution is

$$\begin{aligned}
\sum_{i=1}^{t} (prod_i + hold_i) &= M \sum_{i=1}^{t} \frac{prod_i + hold_i}{M} \\
&\leq M \sum_{i=1}^{t} (\lfloor \frac{prod_i + hold_i}{M} \rfloor + 1) = M (\sum_{i=1}^{t} \lfloor \frac{prod_i + hold_i}{M} \rfloor + t) \,.
\end{aligned}$$

Setting $\tilde{l} = \sum_{i=1}^{t} \lfloor \frac{prod_i + hold_i}{M} \rfloor + t$, it is clear that $\bar{F}_t(\tilde{l}) \geq 0$. We also have

$$\tilde{l} \leq \sum_{i=1}^{t} \frac{prod_i + hold_i}{M} + t = \frac{Z^*}{M} + t \leq \frac{B}{M} + t \leq \left\lfloor \frac{B}{M} \right\rfloor + t + 1 \,. \tag{8}$$

From this we first conclude that $\tilde{l} \leq L$, which combined with the fact $\bar{F}_t(\tilde{l}) \geq 0$ yields $l^* \leq \tilde{l}$. From (8) we derive $l^* M \leq Z^* + tM$. □

3.1.2 A Polynomial Approximation Algorithm

Here we show how to find an upper bound on Z^* in polynomial time. We use the framework from Van Hoesel and Wagelmans (2001). Given x_i, from Assumption A5, we obtain $(1 - \Lambda)\bar{p}_i(x_i) \leq \tilde{p}_i(x_i) \leq (1 + \Lambda)\bar{p}_i(x_i)$, and given s_i, from Assumption A6 we have $(1 - \Omega)\bar{h}_i(s_i) \leq \tilde{h}_i(s_i) \leq (1 + \Omega)\bar{h}_i(s_i)$.

Let w be a number, which specifies the maximum production and holding cost in every time period. For every $i \in T$ let $x_i(w)$ be any production quantity with production cost not exceeding w. Similarly, let $s_i(w)$ be any inventory level in time period i not exceeding holding cost w. Given w, we can check if the specified $x_i(w)$ and $s_i(w)$ lead to a feasible solution, i.e., then do not imply any backlogging, by solving the following recursion:

$$M_i = \min\{M_{i-1} + x_i(w) - d_i, s_i(w)\}, \tag{9}$$

where $M_0 = 0$. If $M_i \geq 0$ for every $i \in T$, then we can find such a plan.

The key idea is to find the smallest such w by bisection and using $\tilde{p}_i(x_i)$, $\tilde{h}_i(s_i)$ to approximate the production and holding costs. An upper bound on w is

$$\max_{i \in T} \{\tilde{p}_i(C_i), \tilde{h}_i(\sum_{\bar{i}=i+1}^{t} d_{\bar{i}})\} \leq \max_{i \in T} \{(1 + \Lambda)\bar{p}_i(C_i), (1 + \Omega)\bar{h}_i(\sum_{\bar{i}=i+1}^{t} d_{\bar{i}})\} = U$$

and thus the bisection is performed on the interval $[0, U]$.

Let now w be fixed. In what follows we assume that all function arguments are integer values. Ideally we would like to solve

$$\begin{aligned} \tilde{x}_i^*(w) &= \max_{0 \leq x_i \leq C_i} \{x_i | \tilde{p}_i(x_i) \leq w\} \\ \tilde{s}_i^*(w) &= \max_{0 \leq s_i} \{s_i | \tilde{h}_i(s_i) \leq w\} \,. \end{aligned}$$

The difficult is that even though $\bar{h}_i, \bar{p}_i$ are monotone, their approximations $\tilde{h}_i, \tilde{p}_i$ might not be. Nevertheless, we obtain $x_i(w), s_i(w)$ by applying a variant of bisection on $\tilde{p}_i, \tilde{h}_i$, respectively. The modification requires forcing the sequence of obtained function values to be nondecreasing.

Let us consider the production cost case for time period i. We start by considering the initial interval $[0, C_i]$ and computing $\tilde{p}_i(C_i)$ (since $\bar{p}_i(0) = 0$ we define $\tilde{p}_i(0) = 0$). The algorithm generates a sequence of approximate function values p'_i with $p'_i(0) = 0, p'_i(C_i) = \tilde{p}_i(C_i)$. In a given iteration let the current interval be $[a, b]$ and $x = \lfloor (a + b)/2 \rfloor$. We first compute $\tilde{p}_i(x)$ and then define

$$p'_i(x) = \begin{cases} \tilde{p}_i(x) & p'_i(a) \leq \tilde{p}_i(x) \leq p'_i(b) \\ p'_i(a) & \tilde{p}_i(x) < p'_i(a) \\ p'_i(b) & \tilde{p}_i(x) > p'_i(b) \,. \end{cases}$$

Finally, if $p'_i(x) > w$, then the next interval is $[a, x]$, or $[x, b]$ if $p'_i(x) \leq w$. Since we consider only integer points, the procedure finishes in a finite number of steps with value $x_i(w)$.

In the same way we obtain $s_i(w)$ with respect to $\tilde{h}_i$. In this case the initial interval is $\left[0, \sum_{i \in T} C_i\right]$. This completely describes the approximation algorithm.

In order to analyze the algorithm, let us focus on the production cost case and the above variant of bisection. We first show that there exists a nondecreasing function $\hat{p}_i$ such that $x_i(w) = \max_{0 \leq x_i \leq C_i} \{x_i | \hat{p}_i(x_i) \leq w\}$ and $(1 - \Lambda)\bar{p}_i(x) \leq \hat{p}_i(x) \leq (1 + \Lambda)\bar{p}_i(x)$ for any $x \in [0, C_i]$. We show this in three steps.

Claim 1. *If $x, y, x < y$ were generated during the variant of bisection, then $p'(x) \leq p'(y)$, and for every generated x we have $(1 - \Lambda)\bar{p}_i(x) \leq p'_i(x) \leq (1 + \Lambda)\bar{p}_i(x)$.*

Proof. The nondecreasing property is easily shown by induction. If by the induction hypothesis, $p'_i(a) \leq p'_i(b)$, then by definition $p'_i(a) \leq p'_i(x) \leq p'_i(b)$.

The second property is also shown by induction. Let $(1 - \Lambda)\bar{p}_i(a) \leq p'_i(a) \leq (1 + \Lambda)\bar{p}_i(a)$ and $(1-\Lambda)\bar{p}_i(b) \leq p'_i(b) \leq (1+\Lambda)\bar{p}_i(b)$. If $p'_i(a) \leq \tilde{p}_i(x) \leq p'_i(b)$, then $(1-\Lambda)\bar{p}_i(x) \leq p'_i(x) \leq (1+\Lambda)\bar{p}_i(x)$ since $\tilde{p}_i$ is a Λ-approximation algorithm.

Let us now assume that $\tilde{p}_i(x) < p'_i(a)$. On the one hand we have $(1 - \Lambda)\bar{p}_i(x) \leq \tilde{p}_i(x) \leq p'_i(a)$, and on the other hand due to monotonicity of $\bar{p}_i$ and induction hypothesis for a we have $p'_i(a) \leq (1 + \Lambda)\bar{p}_i(a) \leq (1 + \Lambda)\bar{p}_i(x)$. Since in this case $p'_i(x) = p'_i(a)$, we obtain the desired result.

Finally, let us assume that $\tilde{p}_i(x) > p'_i(b)$. Then from monotonicity of $\bar{p}_i$ and by the induction hypothesis for b we obtain $(1 - \Lambda)\bar{p}_i(x) \leq (1 - \Lambda)\bar{p}_i(b) \leq p'_i(b)$. We also have $p'_i(b) \leq \tilde{p}_i(x) \leq (1 + \Lambda)\bar{p}_i(x) \leq (1 + \Lambda)\bar{p}_i(b)$. Since $p'_i(x) = p'_i(b)$, this completes the proof. □

Claim 2. *Let $0 \leq n < m \leq C_i$ be two integers and we consider the interval $[n, m]$. We are also given two numbers $\alpha, \beta, \alpha \leq \beta$ such that $(1 - \Lambda)\bar{p}_i(n) \leq \alpha \leq (1 + \Lambda)\bar{p}_i(n)$ and $(1 - \Lambda)\bar{p}_i(m) \leq \beta \leq (1 + \Lambda)\bar{p}_i(m)$. Then there exists a nondecreasing function f_i such that $(1 - \Lambda)\bar{p}_i(x) \leq f_i(x) \leq (1 + \Lambda)\bar{p}_i(x)$ for every $x \in [n, m]$ and $f_i(n) = \alpha, f_i(m) = \beta$.*

Proof. We explicitly define

$$f_i(x) = \begin{cases} \max\{\alpha, (1-\Lambda)\bar{p}_i(x)\} & n \le x \le m-1 \\ \beta & x = m\,. \end{cases}$$

It is easy to see based on the statement conditions that $f_i(n) = \alpha, f_i(m) = \beta$. Since $\bar{p}_i$ is nondecreasing, it follows that f_i is nondecreasing on $[n, m-1]$. We also have $\alpha \le \beta$ and $(1-\Lambda)\bar{p}_i(m-1) \le \bar{p}_i(m-1) \le \bar{p}_i(m) \le \beta$, which implies

$$f_i(m-1) = \max\{\alpha, (1-\Lambda)\bar{p}_i(m-1)\} \le \beta = f_i(m)\,,$$

showing that f_i is nondecreasing.

It is clear that $f_i(x) \ge (1-\Lambda)\bar{p}_i(x)$ for every $x \in [n, m]$. Since $\bar{p}_i$ is nondecreasing, there exists $\bar{m}, n \le \bar{m} \le m$ such that

$$f_i(x) = \begin{cases} \alpha & n \le x \le \bar{m} \\ (1-\Lambda)\bar{p}_i(x) & \bar{m} \le x \le m-1 \\ \beta & x = m. \end{cases}$$

If $\bar{m} \le x \le m$, then it follows from definition that $f_i(x) \le (1+\Lambda)\bar{p}_i(x)$. If $n \le x \le \bar{m}$, then

$$f_i(x) = \alpha \le (1+\Lambda)\bar{p}_i(n) \le (1+\Lambda)\bar{p}_i(x)$$

where the last inequality follows from monotonicity of $\bar{p}_i$. This completes the proof. □

Claim 3. *There exists a nondecreasing function $\hat{p}_i^w$ such that $x_i(w) = \max_{0\le x_i \le C_i}\{x_i | \hat{p}_i^w(x_i) \le w\}$ and $(1-\Lambda)\bar{p}_i(x) \le \hat{p}_i^w(x) \le (1+\Lambda)\bar{p}_i(x)$ for any $x \in [0, C_i]$.*

Proof. Let $0 = x^1(w) \le x^2(w) \le \cdots \le x^{u-1}(w) \le x^u(w) = C_i$ be the ordered sequence of values generated by the variant of bisection. Note that the indices do not reflect the order encountered during the actual execution of the algorithm.

Based on Claim 1, we have $p_i'(x^1(w)) \le p_i'(x^2(w)) \le \cdots \le p_i'(x^{u-1}(w)) \le p_i'(x^u(w))$ and $(1-\Lambda)\bar{p}_i(x^j(w)) \le p_i'(x^j(w)) \le (1-\Lambda)\bar{p}_i(x^j(w))$ for every $j = 1, 2, \ldots, u$.

Now we can apply Claim 2 consecutively for $\left[x^1(w), x^2(w)\right], \left[x^2(w), x^3(w)\right], \ldots, \left[x^{u-1}(w), x^u(w)\right]$ to obtain a nondecreasing function $\hat{p}_i^w$ with the property $(1-\Lambda)\bar{p}_i(x) \le \hat{p}_i^w(x) \le (1+\Lambda)\bar{p}_i(x)$ for any $x \in [0, C_i]$.

From Claim 2 we also obtain $\hat{p}_i^w(x^j(w)) = p_i'(x^j(w))$ for every j. As a result, the bisection applied on $\hat{p}_i^w$ yields the same sequence of values with the final value $x_i(w)$. Since $\hat{p}_i^w$ is nondecreasing, the bisection finds an optimal value and thus $x_i(w) = \max_{0\le x_i\le C_i}\{x_i|\hat{p}_i^w(x_i) \le w\}$. □

The overall algorithm applies bisection with respect to w starting with $[0, U]$. For each fixed w we execute the recursion (9). In each step of the recursion, we apply the variant of bisection with respect to $\tilde{p}_i$ and $\tilde{h}_i$ to compute $x_i(w), s_i(w)$, respectively.

Even though we developed Claim 3 only for the production cost, it clearly holds also for the holding cost (nondecreasing property is the only property used). To justify bisection with respect to w we have the following claim.

Claim 4. *If $w_1 < w_2$, then $x_i(w_1) \le x_i(w_2)$.*

Proof. Let a, b be the last interval where the sequence of generated values is identical for the two different w values. If $p'_i(x) \leq w_1$ and $p'_i(x) > w_2$, then $p'_i(x) \leq w_1 < w_2 < p'_i(x)$, which is clearly a contradiction. We conclude that for w_1 the next interval is $[a, x]$ and for w_2 it is $[b, x]$. Then clearly $x_i(w_1) \in [a, x]$ and $x_i(w_2) \in [b, x]$. $\square$

Claim 4 implies that if there is no backlogging for w, then there is no backlogging for any $w_1 \geq w$, which justifies the bisection algorithm with respect to w.

It is easy to check that the running time of this procedure is $\mathcal{O}(t \cdot \log U \cdot \sum_{i=1}^{t}(\log C_i + \log \sum_{\bar{i}=1}^{i} C_{\bar{i}})) = \mathcal{O}(t^3 \log U \cdot \log C_{max})$, which is polynomial in the input size. Here we assume without loss of generality that the approximation algorithms from Assumptions A5 and A6 take 1 unit of time.

Let $\tilde{w}$ be the computed optimal w and let w^* be the smallest number such that in each time period the production budget of w^* and holding cost budget of w^* yields a solution without backlogging (these quantities are based on the true production and holding costs). Clearly the optimal solution of value Z^* does not exceed the production and holding cost budgets of Z^* in each time period. We conclude that $w^* \leq Z^*$.

Let

$$\begin{aligned}
\bar{x}_i^* &= \max_{0 \leq x_i \leq C_i} \{x_i | \bar{p}_i(x_i) \leq w^*\} \\
\bar{s}_i^* &= \max_{0 \leq s_i} \{s_i | \bar{h}_i(s_i) \leq w^*\}
\end{aligned}$$

and $\hat{w} = \max\{1 + \Omega, 1 + \Lambda\} \cdot w^*$. We have

$$\hat{p}_i^{\hat{w}}(\bar{x}_i^*) \leq (1 + \Lambda) \cdot \bar{p}_i(\bar{x}_i^*) \leq (1 + \Lambda) w^* \leq \max\{1 + \Omega, 1 + \Lambda\} \cdot w^* = \hat{w} \,.$$

Similarly

$$\hat{h}_i^{\hat{w}}(\bar{s}_i^*) \leq \max\{1 + \Omega, 1 + \Lambda\} \cdot w^* = \hat{w} \,.$$

Since $x_i(\hat{w}) = \max_{0 \leq x_i \leq C_i}\{x_i | \hat{p}_i^w(x_i) \leq \hat{w}\}$, it follows that $\bar{x}_i^* \leq x_i(\hat{w})$ for every i. Similarly we obtain $\bar{s}_i^* \leq s_i(\hat{w})$.

Clearly, by definition, $\bar{x}_i^*, \bar{s}_i^*$ yield no backlogging and thus the larger $x_i(\hat{w}), s_i(\hat{w})$ also do not require any backlogging. Since $\tilde{w}$ is the smallest w with respect to all $\hat{p}_i^w, \hat{h}_i^w$, we obtain that $\tilde{w} \leq \hat{w}$.

The approximation algorithm returns production quantities $x_i(\tilde{w})$ and inventory levels less than $s_i(\tilde{w})$. From Claim 3 we obtain

$$\bar{p}_i(x_i(\tilde{w})) \leq \frac{\hat{p}_i^{\tilde{w}}(x_i(\tilde{w}))}{1 - \Lambda} \leq \frac{\tilde{w}}{1 - \Lambda}$$

and likewise

$$\bar{h}_i(s_i(\tilde{w})) \leq \frac{\tilde{w}}{1 - \Omega} \,.$$

The cost of this approximate solution is less than or equal to

$$\begin{aligned}
\sum_{i \in T}(\bar{p}_i(x_i(\tilde{w})) + \bar{h}_i(s_i(\tilde{w}))) &\leq t\tilde{w}(\frac{1}{1 - \Lambda} + \frac{1}{1 - \Omega}) \leq t\hat{w}(\frac{1}{1 - \Lambda} + \frac{1}{1 - \Omega}) \\
&= t \max\{1 + \Omega, 1 + \Lambda\} \cdot w^*(\frac{1}{1 - \Lambda} + \frac{1}{1 - \Omega}) \\
&\leq t \max\{1 + \Omega, 1 + \Lambda\} \cdot (\frac{1}{1 - \Lambda} + \frac{1}{1 - \Omega}) Z^* \,.
\end{aligned}$$

Let $\Theta = \max\{1+\Omega, 1+\Lambda\} \cdot (\frac{1}{1-\Lambda} + \frac{1}{1-\Omega}) - 1$. The presented algorithm has the approximation ratio of $t\Theta$.

3.1.3 The Fully Polynomial Time Approximation Scheme

The scheme is given as follows, where $\epsilon > 0$ and we want to find a solution that is within a relative error of ϵ.

Step 1) Compute $\tilde{w}$ and denote by B the corresponding objective value.

Step 2) Let $M = \max\{\lfloor \frac{\epsilon B}{t^2\Theta} \rfloor, 1\}$.

Step 3) Compute $\bar{F}_t(l)$ for $l = 1, \ldots, \lfloor \frac{B}{M} \rfloor + t + 1$ and let $l^* = \min\{l : \bar{F}_t(l) \geq 0\}$.

Step 4) Output Ml^*.

We first argue that the produced value has the desired property. From Proposition 3 we obtain $l^*M \leq Z^* + tM$. Consider first the case of $M = \lfloor \frac{\epsilon B}{t^2\Theta} \rfloor$. Then $tM \leq \frac{\epsilon B}{t\Theta} \leq \epsilon Z^*$, where we have used $B \leq t\Theta Z^*$. If $M = 1$, then in Step 3 all possible values are considered and thus in this case $l^* = Z^*$.

To establish that the running time is pseudo polynomial it suffices to argue that $L = \lfloor \frac{B}{M} \rfloor + t + 1$ is pseudo polynomial (see (7)). If $M = 1$, then $\frac{\epsilon B}{t^2\Theta} \leq 1$, which implies that $\frac{B}{M} = B \leq \frac{t^2\Theta}{\epsilon}$. On the other hand, if $M = \lfloor \frac{\epsilon B}{t^2\Theta} \rfloor$, due to $\lfloor x \rfloor \geq x/2$ for any $x \geq 1$, we obtain $\frac{B}{M} \leq \frac{B}{2\frac{\epsilon B}{t^2\Theta}} = \frac{t^2\Theta}{2\epsilon}$. This shows that in both cases B/M is upper bounded by $\frac{t^2\Theta}{\epsilon}$ and thus the algorithm has the desired running time.

3.2 A Fully Polynomial Time Approximation Scheme for Lot-sizing with Supplier Selection

In this section we show how to use the FPTAS from the previous section for the lot-sizing problem studied in this work.

It is easy to see that as long as the input data are integral, the lot size is always going to be integral, i.e. integrality of x is automatic.

Since the holding cost is linear, Assumptions A2, A4, and A6 clearly hold. We need to fulfill Assumptions A1, A3, and A5.

For any $i = 1, \ldots, t$ and any lot size x_i let

$$\begin{aligned} r_i(x_i) = \min & \sum_{j\in N} c_{ji} w_{ji} + \sum_{j\in N} f s_{ji} z_{ji} \\ & \sum_{j\in N} w_{ji} = x_i \\ & w_{ji} \leq K_{ji} z_{ji} \qquad\qquad j \in N \\ & w \geq 0, z \text{ binary.} \end{aligned}$$

It is easy to see that $r_i(x_i)$ is an increasing function of x_i. The production cost is $\bar{p}_i(x_i) = r_i(x_i) + \sigma(x_i) f p_i + p_i x_i$, where $\sigma(x_i) = 0$ if $x_i = 0$ and 1 if $x_i > 0$. Assumption A1 clearly holds.

For ease of notation from now on we will omit subscript i. Without loss of generality we assume $0 < c_1 \leq c_2 \leq \cdots \leq c_n$. For any $k = 1, \ldots n$ and $l \in \mathbb{Z}_+$, let

$$\bar{\alpha}_k(l) = \max \sum_{j=1}^{k} w_j$$

$$\sum_{j=1}^{k} c_j w_j + \sum_{j=1}^{k} f s_j z_j \leq l \tag{10}$$

$$w_j \leq K_j z_j \qquad j = 1, \ldots, k$$

$$w \geq 0, z \text{ binary.}$$

The following proposition relates $\bar{\alpha}_n$ and $r(x)$. Its proof is elementary.

Proposition 4. *We have*

$$r(x) = \min\{l | \bar{\alpha}_n(l) = x\}. \tag{11}$$

Next we design a dynamic program for computing $\bar{\alpha}_n(lM)$ for any fixed number M and non-negative integer l. For ease of notation, we denote $\alpha_k(l) = \bar{\alpha}_k(lM)$. For any $k \in N, u \in N$, and $l \in \mathbb{Z}_+$, we define

$$r_k^u(l) = \max \sum_{j=1}^{k} (c_u K_j - c_j K_j - f s_j) z_j$$

$$\sum_{j=1}^{k} (c_j K_j + f s_j) z_j \leq lM - f s_u - 1$$

$$\sum_{j=1}^{k} (c_j K_j + f s_j) z_j \geq lM - f s_u - K_u c_u + 1$$

$$z \text{ binary.}$$

If the underlying feasibility set is empty, we define $r_k^u(l) = -\infty$. For each $k \in N$, let $p(k)$ be such an index that $(p(k) - 1)M \leq c_k K_k + f s_k < p(k)M$.

Theorem 1. *For $k = 2, \ldots, n$ and $l \in \mathbb{Z}_+$, we have*

$$\alpha_k(l) = \max\{\alpha_{k-1}(l), \alpha_{k-1}(l - p(k) + 1) + K_k,$$
$$\max_{1 \leq p \leq p(k)-1} \{\frac{pM - f s_k}{c_k} + \alpha_{k-1}(l - p)\}, \frac{lM - f s_k + r_{k-1}^k(l)}{c_k}\}.$$

Proof. Consider $\alpha_k(l)$. Let (w^*, z^*) be an optimal solution. In addition, let p be such that $(p - 1)M \leq c_k w_k^* + f s_k z_k^* < pM$ for an integer p, $1 \leq p \leq l + 1$. We consider several cases.

Case 1. $w_k^* = 0$. Note that in this case we can assume $z_k^* = 0$. Clearly then $\alpha_k(l) \leq \alpha_{k-1}(l)$.

Case 2. $w_k^* = K_k$. Now we have $z_k^* = 1$. We also have $c_k K_k + f s_k < pM$ and $c_k K_k + f s_k \geq (p-1)M$, which implies $p = p(k)$. Hence $\sum_{j=1}^{k-1} c_j w_j^* + \sum_{j=1}^{k-1} f s_j z_j^* \leq lM - (c_k K_k + f s_k) \leq (l - p(k) + 1)M$. Therefore $\alpha_k(l) \leq K_k + \alpha_{k-1}(l - p(k) + 1)$.

Case 3. $0 < w_k^* < K_k$. It implies $z_k^* = 1$. In this case either (10) is an equality or there is an optimal solution with $c_k w_k^* + fs_k = pM$. Otherwise, we can increase w_k^* and either we have an optimal solution satisfying Case 2, or (10) is at equality, or $c_k w_k^* + fs_k = pM$.

We assume first that $c_k w_k^* + fs_k = pM$. Clearly $\alpha_k(l) \leq \frac{pM - fs_k}{c_k} + \alpha_{k-1}(l-p)$. From $w_k = \frac{pM-fs_k}{c_k} \leq K_k$, we obtain $p \leq p(k) - 1$.

Let us assume now that (10) is at equality. We claim that $w_j^* = K_j z_j^*$ for $j = 1, \ldots, k-1$. Suppose that there is an $m, 1 \leq m \leq k-1$ such that $w_m^* < K_m z_m^*$. Let $\varepsilon = \min\{\frac{c_k}{c_m} w_k^*, K_m z_m^* - w_m^*\} > 0$. Consider $\bar{w} = w^* + \varepsilon e_m - \varepsilon \frac{c_m}{c_k} e_k, \bar{z} = z^*$. Then $\sum_{j=1}^k \bar{w}_j = \sum_{j=1}^k w_j + \varepsilon - \varepsilon \frac{c_m}{c_k} \geq \sum_{j=1}^k w_j$, since $c_k \geq c_m$. We also have $\sum_{j=1}^k c_j \bar{w}_j = \sum_{j=1}^k c_j w_j + \varepsilon c_m - \varepsilon \frac{c_m}{c_k} \cdot c_k = \sum_{j=1}^k c_j w_j$. $(\bar{w}, \bar{z})$ is a feasible solution for $\alpha_k(l)$ with the objective value not less than the objective value of (w^*, z^*).

If $c_k > c_m$, we obtain a contradiction since the objective value of $(\bar{w}, \bar{z})$ is strictly larger than the objective value of (w^*, z^*). If $c_k = c_m$, then we either obtain an optimal solution with $w_k^* = 0$ or $w_m^* = K_m z_m^*$. In the former case, the condition of Case 1 holds. In the latter case, we repeat the procedure. This shows that we can assume $w_j^* = K_j z_j^*$ for $j = 1, \ldots, k-1$. In this case we have $\sum_{j=1}^{k-1} c_j K_j z_j^* + c_k w_k^* + \sum_{j=1}^{k-1} fs_j z_j^* + fs_k = lM$ and therefore $w_k^* = \frac{Ml - fs_k - \sum_{j=1}^{k-1}(c_j K_j + fs_j) z_j^*}{c_k}$. Then $\alpha_k(l) \leq \frac{Ml - fs_k + \sum_{j=1}^{k-1}(c_k K_j - c_j K_j - fs_j) z_j^*}{c_k}$. The condition $0 < w_k^* < K_k$ yields

$$\sum_{j=1}^{k-1} (c_j K_j + fs_j) z^* \leq Ml - fs_k - 1$$
$$\sum_{j=1}^{k-1} (c_j K_j + fs_j) z^* \geq Ml - fs_k - c_k K_k + 1\,.$$

We conclude that $\alpha_k(l) \leq (Ml - fs_k + r_{k-1}^k(l))/c_k$.

This shows that $\alpha_k(l)$ is less than or equal to the right hand side in the theorem. It is easy to see that any solution to the right hand side can be extended into a solution to $\alpha_k(l)$ with the appropriate value. $\square$

In order to compute $\alpha_k(l)$ from Theorem 1, we need to develop a recursive relationship for r_u^k. To this end, for every $k \in N, u \in N$, and $l \in \mathbb{Z}_+$ we need to define

$$\begin{aligned}
\mu_k^u(l) = \max & \sum_{j=1}^k (c_u K_j - c_j K_j - fs_j) z_j \\
& \sum_{j=1}^k (c_j K_j + fs_j) z_j \leq (l+1)M - fs_u - K_u c_u + 1 \\
& \sum_{j=1}^k (c_j K_j + fs_j) z_j \geq lM - fs_u - 1 \\
& z \text{ binary.}
\end{aligned}$$

If the underlying feasibility set is empty, we define $\mu_k^u(l) = -\infty$. It suffices to define μ_k^u only for those u with $K_u c_u \leq M + 1$.

Proposition 5. *For $k = 2, \ldots, n, u \in N$, and $l \in Z_+$ we have*

$$r_k^u(l) = \max\{r_{k-1}^u(l), c_u K_k - c_k K_k - f s_k + \max\{r_{k-1}^u(l - p(k) + 1), r_{k-1}^u(l - p(k)), \mu_{k-1}^u(l - p(k))\}\}$$

and

$$\mu_k^u(l) = \max\{\mu_{k-1}^u(l), c_u K_k - c_k K_k - f s_k + \max\{\mu_{k-1}^u(l - p(k) + 1), \mu_{k-1}^u(l - p(k)), r_{k-1}^u(l - p(k) + 1)\}\}.$$

Proof. Consider z^*, which is optimal to $r_k^u(l)$. If $z^* = 0$, then $r_k^u(l) = r_{k-1}^u(l)$. Let now $z_k^* = 1$. Then

$$(l - p(k))M - f s_u - K_u c_u + 1 \leq \sum_{j=1}^{k-1} (c_j K_j + f s_j) z_j^* \leq (l - p(k) + 1)M - f s_u - 1.$$

It is clear that

$$\begin{aligned}
&[(l - p(k))M - f s_u - K_u c_u + 1, (l - p(k) + 1)M - f s_u - 1] \\
&\quad = [(l - p(k))M - f s_u - K_u c_u + 1, (l - p(k))M - f s_u - 1] \\
&\quad \cup [(l - p(k))M - f s_u - 1, (l - p(k) + 1)M - f s_u - K_u c_u + 1] \\
&\quad \cup [(l - p(k) + 1)M - f s_u - K_u c_u + 1, (l - p(k) + 1)M - f s_u - 1].
\end{aligned}$$

The first and the third case yields $r_{k-1}^u(l - p(k)), r_{k-1}^u(l - p(k) + 1)$ respectively, and the second case yields $\mu_{k-1}^u(l - p(k))$. Similarly we can prove the second statement. $\square$

By definition

$$\alpha_0(l) = \mu_0^u(l) = r_0^u(l) = \begin{cases} 0 & \text{if } l \geq 0 \\ -\infty & \text{if } l < 0\,, \end{cases}$$

and all presented recursive formulas hold also for $k = 1$. We can now compute $\alpha_n(l)$ for $l = 1, \ldots, \tilde{l}$ as follows.

Step 1) For each $l \in [\tilde{l}], k \in N$, and $u \in N$ we compute $r_k^u(l), \mu_k^u(l)$ based on Proposition 5.

Step 2) For each $l \in [\tilde{l}], k \in N$, we compute $\alpha_k(l)$ based on Theorem 1.

The running time of this algorithm is $\mathcal{O}(n^2 \tilde{l})$.

We can assert if $\bar{p}_i(x_i) = r_i(x_i) + \delta(x_i) f p_i + p_i x_i \leq \tilde{l} M$ as follows. For every $l_1 = 0, 1, \ldots, \tilde{l}, l_2 = 0, 1, \ldots, \tilde{l}$ with $l_1 + l_2 \leq \tilde{l}$, we check if $\delta(x_i) f p_i + p_i x_i \leq l_1 M$ and if $x_i \leq \alpha_n(l_2)$. If we find such l_1, l_2, then $\bar{p}_i(x_i) \leq \tilde{l} M$, otherwise $\bar{p}_i(x_i) \geq \tilde{l} M$. From Proposition 4 and monotonicity of $\bar{\alpha}_n$ it follows that $r_i(x_i) \leq l_2 M$ if and only if $x_i \leq \alpha_n(l_2)$, which establishes the correctness of this procedure. The running time is $\mathcal{O}(n^4 \tilde{l})$. This shows that Assumption A3 holds.

It remains to argue that Assumption A5 holds. To this end let

$$
\begin{aligned}
Z(x) = \min \sum_{j \in N} c_j w_j + \sum_{j \in N} f s_j z_j & \\
\sum_{j \in N} w_j = x & \\
w_j \leq k_j z_j & \qquad j \in N \\
\sum_{j \in N_1 \setminus C} w_j + \sum_{j \in N_2 \setminus C} \min\{\lambda, K_j\} z_j \geq \lambda & \qquad C \subset N \text{ with } \lambda = x - \sum_{j \in C} K_j > 0, \\
& \qquad N = N_1 \cup N_2, N_1 \cap N_2 = \emptyset \\
w \geq 0, 0 \leq z \leq 1\,. &
\end{aligned}
$$

Carr *et al.* (2000) show that $\frac{z(x)}{r(x)} \leq 2$ and that $z(x)$ can be computed in polynomial time by the ellipsoid algorithm. The separation algorithm is polynomially solvable since it suffices to consider $C = \{j \in N | w_j^* \geq \frac{K_j}{2}\}$, where (w^*, z^*) is the current LP solution.

This gives a 2-approximation algorithm for Assumption A5.

References

Aggarwal, A. and Park, J. (1993). Improved algorithms for economic lot size problems. *Operations Research*, **41**, 549–571.

Aghezzaf, E. and Wolsey, L. (1994). Modelling piecewise linear concave costs in a tree partitioning problem. *Discrete Applied Mathematics*, **50**, 101–109.

Belvaux, G. and Wolsey, L. (2001). Modelling practical lot-sizing problems as mixed-integer programs. *Management Science*, **47**, 993–1007.

Bhatia, M. and Palekar, U. (2001). Lot sizing problems with strong set-up interactions - a variable redefinition approach. *IIE Transactions*, **33**, 357–370.

Carr, R., Fleischer, L., Leung, V., and Phillips, C. (2000). Strengthening integrality gaps for capacitated network design and covering problems. In D. Schmoys, editor, *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 106–115, San Francisco, CA. Society for Industrial and Applied Mathematics.

Chubanov, S., Kovalyov, M., and Pesch, E. (2006). An FPTAS for a single-item capacitated economic lot-sizing problem with monotone cost structure. *Mathematical Programming*, **106**, 453–466.

Federgruen, A. and Tzur, M. (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**, 909–925.

Florian, M. and Klein, M. (1971). Deterministic production planning with concave costs and capacity constraints. *Management Science*, **18**, 12–20.

Florian, M., Lenstra, J., and Kan, H. R. (1980). Deterministic production planning: algorithms and complexity. *Management Science*, **26**, 669–679.

Klose, A. (2008). Algorithms for solving the single-sink fixed-charge transportation problem. *Computers and Operations Research*, **35**, 2079–2092.

Padberg, M., Roy, T. V., and Wolsey, L. (1985). Valid linear inequalities for fixed charge problems. *Operations Research*, **33**, 842–861.

van Hoesel, C. and Wagelmans, A. (1996). An $O(T^3)$ algorithm for the economic lot-sizing problem with constant capacities. *Management Science*, **42**, 142–150.

Van Hoesel, C. and Wagelmans, A. (2001). Fully polynomial approximation schemes for single-item capacitated economic lot-sizing problems. *Mathematics of Operations Research*, **26**, 339–357.

Wagelmans, A., van Hoesel, A., and Kolen, A. (1992). Economic lot sizing: An $O(n \log n)$ algorithm that runs in linear time in the Wagner-Whitin case. *Operations Research*, **40**, 145–156.

Wagner, H. and Whitin, T. (1958). Dynamic version of the economic lot size model. *Management Science*, **5**, 89–96.