Lot Sizing with Piecewise Concave Production Costs

Esra Koca

ekoca@bilkent.edu.tr

Hande Yaman*

hyaman@bilkent.edu.tr

M. Selim Aktürk

akturk@bilkent.edu.tr

Department of Industrial Engineering, Bilkent University

February 14, 2013

Abstract

We study the lot-sizing problem with piecewise concave production costs and concave holding costs. This problem is a generalization of the lot-sizing problem with quantity discounts, minimum order quantities, capacities, overloading, subcontracting or a combination of these. We develop a dynamic programming (DP) algorithm to solve this problem and answer an open question in the literature: we show that the problem is polynomially solvable when the breakpoints of the production cost function are time invariant and the number of breakpoints is fixed. For the special cases with capacities and subcontracting, the time complexity of our DP algorithm is as good as the complexity of algorithms available in the literature. We report the results of a computational experiment where the DP is able to solve instances that are hard for a mixed-integer programming (MIP) solver. We enhance the MIP formulation with valid inequalities based on mixing sets and use a cut-and-branch algorithm to compute better bounds. We propose a state space reduction based heuristic algorithm for large

*Corresponding author

instances and show that the solutions are of good quality by comparing them with the bounds obtained from the cut-and-branch.

Keywords: lot sizing; piecewise concave production cost; quantity discounts; subcontracting; dynamic programming.

1 Introduction

Lot-sizing problems arise in production, procurement and transportation systems under different cost and capacity settings. Given a planning horizon, demand, production (or procurement/shipment) and inventory holding costs, the aim of the lot-sizing problem is to propose a minimum cost production plan to satisfy the demand (see, e.g., the seminal works by Wagner and Whitin (1958) and Zangwill (1966) and the book by Pochet and Wolsey (2006)). In this paper, we study the lot-sizing problem where the inventory holding cost function is concave and the production cost function is a piecewise concave function. We call this problem the "lot-sizing problem with piecewise concave production costs" and abbreviate it with LS-PC.

A continuous piecewise concave function is the maximum of a finite sequence of continuous concave functions and therefore, it may not be concave. A piecewise concave function is more general as it can be discontinuous on the boundaries and its breakpoints (Zangwill, 1967). If p is a piecewise concave function with breakpoints at $b^0 < b^1 < \ldots < b^m$, then p is concave in each of the m intervals $[b^{j-1}, b^j]$ for $j = 1, \ldots, m$. Note that concavity of p in each of the intervals implies that it is lower semi-continuous.

Examples of piecewise concave production costs are depicted in Figures 1 and 2. In Figure 1, the first two functions represent common quantity discounts known as incremental discount and all units discount. Federgruen and Lee (1990) study the lot-sizing problem with these two types of discounts. They assume that the production cost function has two pieces and propose dynamic programming algorithms of complexity $O(n^3)$ and $O(n^2)$ for the problems with all units discount and incremental discount, respectively, where n is the number of periods. Chan et al. (2002) consider the modified all units discount depicted in Figure 1c. They prove that the lot-sizing problem with this cost structure is NP-hard when either the production cost functions vary from period to period or the number of breakpoints is not bounded by a constant. Li et al. (2012) study the lot sizing problem with all-units discount and resales under the assumptions that the breakpoints of the cost function are time-invariant, the number of breakpoints is fixed and there is no capacity constraint. They develop an $O(n^{m+3})$ time algorithm to solve this problem, where m is the number of breakpoints. Archetti et al. (2011) present polynomial time algorithms to solve special cases of the lot-sizing problem with modified all units discount and incremental discount when the cost functions are time-invariant.

Atamtürk and Hochbaum (2001) study the lot-sizing problem with subcontracting where the production and subcontracting costs are concave nondecreasing functions and the inventory holding cost is a linear function. The overall production cost function is depicted in Figure 1d: the first piece of the function corresponds to regular production and the second piece corresponds to subcontracting or overloading. The authors develop an $O(n^5)$ time dynamic programming algorithm for the case where the regular production capacities (the breakpoint of the cost function) are the same for all periods.

The production cost function given in Figure 1e models constraints on minimum production (order) quantities as studied by Hellion et al. (2012). In this setting, if there is a production at a given period, then the production amount should not be less than a minimum level b^1 and should not exceed the capacity b^2. The authors assume that the production and inventory holding cost functions are concave and propose a dynamic programming algorithm for this problem. The time complexity reported in Hellion et al. (2012) was corrected and reported as $O(n^6)$ (Hellion, 2013).

As seen above, piecewise concave functions can be used to represent discounts, subcontracting, capacity acquisition, overloading, as well as minimum quantity requirements and capacities. In addition, one can represent any combination of these using piecewise concave functions. In Figure 2a, we model a setting with discounts and overloading. The unit cost, c_0, up to the first breakpoint b^1 can be viewed as

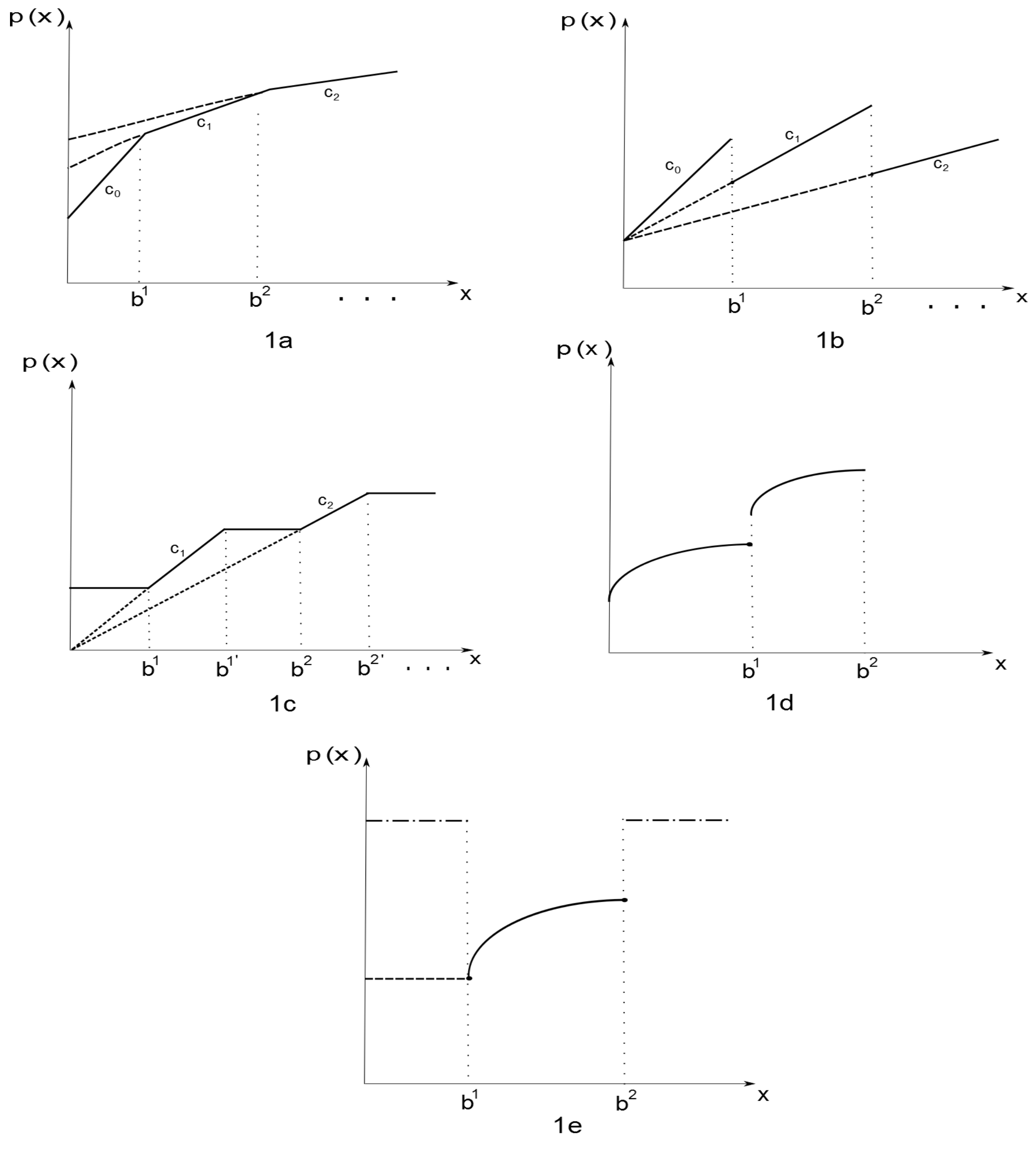

Figure 1: Some special cases of piecewise concave functions

the regular unit purchasing cost. Then a quantity discount applies and the unit cost becomes $c_1 < c_0$ up to the second breakpoint b^2, which is the capacity of the supplier. Afterwards, the supplier requires use of overtime (or subcontracting) in order to fulfill the additional orders and hence the unit cost is $c_2 > c_0$. Note that the resulting cost function is neither convex nor concave.

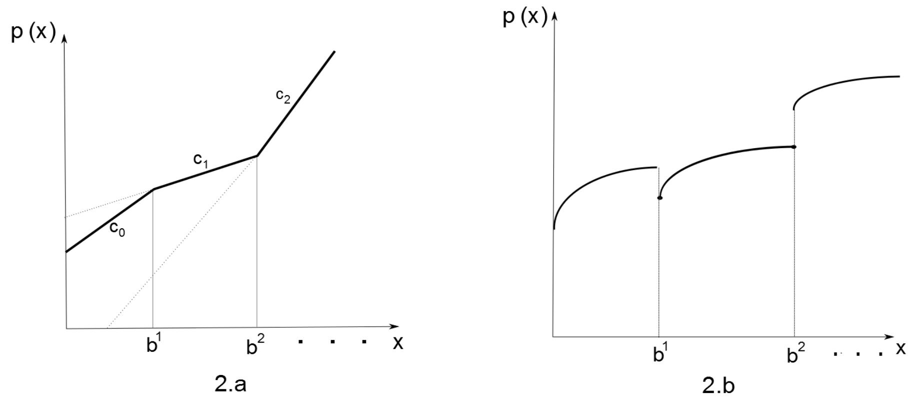

Figure 2: Examples of piecewise concave functions

Now consider the case where several suppliers give offers (possibly with discounts) for a product and the company purchases its products from at most one supplier in each period. Then the production cost is the minimum of the purchasing costs over all suppliers and is a piecewise concave function if the cost function of each supplier is concave. An example is given in Figure 2b in which each segment of the cost function represents a supplier. The second supplier offers the most attractive price but has a lower bound for procurement, b^1 units, and has a capacity of b^2 units. It is more beneficial to buy from the first supplier up to b^1 units and from the third supplier after b^2 units. Accordingly, decisions on the purchasing amounts in each period will also determine the supplier of each period. Therefore, this problem can be seen as a supplier selection and lot sizing problem.

As the lot-sizing problem with modified all units discount studied by Chan et al. (2002) is a special case of LS-PC, LS-PC is NP-hard unless the breakpoints are time-invariant and the number of breakpoints is bounded above by a constant.

Swoveland (1975) presents characteristics of an optimal solution when inventory holding and production cost functions are piecewise concave functions. He proposes a pseudo-polynomial dynamic programming algorithm to solve this problem. Shaw and Wagelmans (1998) present an algorithm for the capacitated lot-sizing problem with piecewise linear production costs (not necessarily convex or concave) and general inventory holding costs. Their algorithm is also pseudo-polynomial. VanHoesel and Wagelmans (1996) show that if the production cost function is piecewise concave and monotone and the number of pieces is polynomially bounded in the size of the problem, then there exists a fully polynomial approximation scheme.

The special cases of LS-PC with cost functions depicted in Figure 1 are polynomially solvable. However, to the best of our knowledge, there is no polynomial time algorithm to solve the problem with cost functions like those in Figure 2. Indeed, the complexity of the problem is open for the case where the number of breakpoints is fixed and the breakpoints are time-invariant. In this study, we show that in this case, the problem can be solved in polynomial time by proposing a dynamic programming (DP) algorithm. This algorithm generalizes the algorithm of Florian and Klein (1971) for the constant capacity lot-sizing problem, which corresponds to the special case with one breakpoint. For the special cases with regular production and subcontracting; and with minimum production quantities and constant capacities, our DP has the same time complexity as the one of Atamtürk and Hochbaum (2001) and Hellion (2013), respectively. We also conduct a computational study to see if the DP is useful in practice. We derive a mixed-integer programming (MIP) formulation and solve it with an off-the-shelf solver. Our results show that the DP outperforms the MIP approach for some instances even when we strengthen the formulation with valid inequalities. For larger instances, we propose a heuristic method based on state space reduction. Our computational experiments show that the heuristic provides good quality solutions in reasonable computation times when the solver and the exact DP fail.

The rest of the paper is organized as follows. In Section 2, we formally define the problem LS-PC and state some important properties of an optimal solution to the problem. In Section 3, we present a polynomial time dynamic programming algorithm for solving the problem when the number of breakpoints is fixed and the breakpoints are time-invariant and show that the complexity of the DP is as good as the complexity of algorithms available in the literature for some special cases of the problem. We then report our computational experiments in Section 4, and propose a state space reduction based heuristic algorithm for large instances in Section 5. Finally in Section 6 we present some concluding remarks.

2 Problem definition and properties of optimal solutions

In the lot-sizing problem, we would like to find a minimum cost production plan over a planning horizon of n periods. The demand d_t, the production cost function p_t and the inventory holding cost function h_t are given for each period t. Let x_t be the amount produced in period t and s_t be the stock on hand at the end of period t. Using these variables, the lot-sizing problem can be modeled as

$$\begin{aligned} \min \quad & \sum_{t=1}^{n} p_t(x_t) + \sum_{t=0}^{n} h_t(s_t) & & (1)\\ \text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t \qquad t = 1, \ldots, n, & & (2)\\ & s_0 = 0, & & (3)\\ & s, x \geq 0. & & (4) \end{aligned}$$

Constraints (2) are inventory balance constraints. The assumption on the initial inventory being zero is imposed by constraint (3) and is made without loss of generality. Constraints (4) are variable restrictions. The objective function (1) is the sum of production and inventory holding costs.

In LS-PC, the inventory holding cost function $h_t(.)$ is a concave function on $[0, \infty)$ and $p_t(.)$ is a piecewise concave function on $[0, \infty)$ with m_t finite breakpoints $b_t^1, \ldots, b_t^{m_t}$ such that $b_t^0 = 0$ and $b_t^{i-1} < b_t^i$ for $i = 1, \ldots, m_t$.

As typically done in the lot-sizing literature (see Pochet and Wolsey (2006)), we will use the concepts of regeneration intervals and fractional periods in analyzing the structure of optimal solutions. An interval $[j, l]$ with $1 \leq j \leq l \leq n$, $s_{j-1} = s_l = 0$ and $s_t > 0$ for $j \leq t < l$ is referred to as a *regeneration interval* and a period i

whose production level is not equal to any of the breakpoints of the production cost function, i.e., $x_i \in [b_i^0, \infty) \setminus \{b_i^0, \dots, b_i^{m_i}\}$ is referred to as a *fractional period.* We define $b_i^{m_i+1} = \infty$ for all i.

If the production cost function is not monotone (see Figures 1e and 2b), we may have positive ending inventory in all optimal solutions. Therefore, contrary to the case with the classical lot sizing problems, we cannot say that there exists an optimal solution that is composed of a series of successive regeneration intervals. However, for our problem, there exists an optimal solution that is composed of a series of regeneration intervals that cover the interval $[1, j-1]$ plus an interval $[j, n]$ for some $1 \le j \le n+1$. We know the following properties for these intervals.

Theorem 2.1 *[Swoveland (1975)] There exists an optimal solution to the problem LS-PC such that in each regeneration interval* $[j, l]$, *there exists at most one fractional period.*

Theorem 2.1 is a generalization of the "fractional period property" for the capacitated lot-sizing problem. Note that if $x_i > b_i^{m_i}$, then period i is a fractional period.

Theorem 2.2 *If the ending inventory is positive in all optimal solutions to LS-PC, then there exists an optimal solution that is composed of a series of regeneration intervals that cover the interval* $[1, j-1]$ *plus an interval* $[j, n]$ *for some* $1 \le j \le n+1$ *with no fractional period in the last interval* $[j, n]$.

Proof. Suppose that at all optimal solutions we have $s_n > 0$. Let (x, s) be an optimal solution with the largest j value such that $s_{j-1} = 0$ and $s_t > 0$ for $t = j, \dots, n$. Suppose that there exists a fractional period with $i \in [j, n]$ such that $b_i^k < x_i < b_i^{k+1}$ for some $k \in \{0, \dots, m_i\}$. Define $\alpha = \min\{\min_{t=i}^n s_t, x_i - b_i^k\}$ and $\beta = b_i^{k+1} - x_i$ if b_i^{k+1} is finite and $\beta = \alpha$ otherwise. Clearly, α and β are positive. Now consider the two solutions (x^1, s^1) and (x^2, s^2) that are the same as (x, s) except that $x_i^1 = x_i - \alpha$, $s_t^1 = s_t - \alpha$ for $t = i, \dots, n$, $x_i^2 = x_i + \beta$, and $s_t^2 = s_t + \beta$ for $t = i, \dots, n$. Both solutions are feasible. Optimality of (x, s) implies that $p_i(x_i - \alpha) + \sum_{t=i}^n h_t(s_t - \alpha) - p_i(x_i) - \sum_{t=i}^n h_t(s_t) \ge 0$ and $p_i(x_i + \beta) + \sum_{t=i}^n h_t(s_t + \beta) - p_i(x_i) - \sum_{t=i}^n h_t(s_t) \ge 0$. Since p_i is concave on $[b_i^k, b_i^{k+1}]$ and

h_t is concave on $[0, \infty)$ for each $t = i, \ldots, n$, we also have $\frac{\beta}{\alpha+\beta} p_i(x_i - \alpha) + \frac{\alpha}{\alpha+\beta} p_i(x_i + \beta) \leq p_i(x_i)$ and $\frac{\beta}{\alpha+\beta} h_t(s_t - \alpha) + \frac{\alpha}{\alpha+\beta} h_t(s_t + \beta) \leq h_t(s_t)$ for $t = i, \ldots, n$. Therefore, both (x^1, s^1) and (x^2, s^2) are also optimal. Either b_i^{k+1} is finite and (x^2, s^2) is an optimal solution where the fractional period i is eliminated. Or $k = m_i$ and as (x, s) is an optimal solution with the largest j value such that $s_{j-1} = 0$ (implying that $s_t^1 > 0$ for $t = i, \ldots, n$), (x^1, s^1) is an optimal solution in which i is not a fractional period anymore. $\square$

Due to Theorem 2.1, as it is done in the classical lot sizing problems, we can find the minimum cost solution for each regeneration interval $[j, l]$ by assuming that it consists at most one fractional period. However, it is not sufficient for finding a minimum cost solution for the problem since for the intervals $[j, n]$ we need to consider the case where it is not a regeneration interval. In this case, for the intervals $[j, n]$, due to Theorem 2.2, we can search for a minimum cost solution by assuming that it does not consist any fractional period. Consequently, we can find a minimum cost solution for each interval $[j, n]$ by picking the least cost solution among the cases that it is a regeneration interval or not. In the next section, we develop a dynamic programming algorithm for finding an optimal solution for LS-PC by using these results.

3 Dynamic programming algorithm

In this section, we propose a dynamic programming algorithm for the special case where the breakpoints of the production cost function are time-invariant and the number of breakpoints is fixed, i.e., $b_t^i = b^i$ for all $t = 1, \ldots, n$ and $i = 0, \ldots, m$ where $m_t = m$ for all $t = 1, \ldots, n$ and $m (\geq 1)$ is fixed.

This algorithm is a generalization of the algorithm given by Florian and Klein (1971) for the constant capacity lot-sizing problem.

Let e_i be a unit vector of size m in which the i^{th} component is one and the other components are zero for $i = 1, \ldots, m$ and e_0 be a zero vector of size m.

3.1 Minimum cost for an interval $[j,l]$ with no fractional period

First, we compute the minimum cost for a regeneration interval $[j,l]$ with $1 \leq j \leq l \leq n-1$ and for an interval $[j,n]$ for $1 \leq j \leq n$ when there is no fractional period. To this end, we define the following function. Let $\tau \in \mathbb{Z}_+^m$ and $t \in \{j,\ldots,l\}$. If $l \leq n-1$, let $F_{jl}(t,\tau)$ be the minimum cost for periods j up to t during which τ_i times b^i, for $i=1,\ldots,m$, units are produced, no fractional production is done, given that $s_{j-1}=s_l=0$ and $s_u>0$ for $u \in \{j,\ldots,\min\{t,l-1\}\}$. If $l=n$, then we define the same function by dropping the requirement that $s_l=0$. For $j \leq t$, we let $d_{jt}=\sum_{i=j}^t d_i$.

Note that the amount of production between periods j and t is equal to $\sum_{i=1}^m \tau_i b^i$ and the number of periods in which production takes place is $\sum_{i=1}^m \tau_i$. If $t<l$ and $\sum_{i=1}^m \tau_i b^i \leq d_{jt}$, then we cannot have $s_t>0$. Also, if $t=l$ and $\sum_{i=1}^m \tau_i b^i \neq d_{jl}$, then $s_l=0$ is not possible. If $\sum_{i=1}^m \tau_i > t-j+1$, the production schedule is infeasible.

For $i=0,\ldots,m$, we let

$$F_{jl}(j,e_i)=\begin{cases} p_j\left(b^i\right)+h_j\left(b^i-d_j\right) & \text{if } d_j<b^i \text{ and } (j<l \text{ or } l=n),\\ p_j(b^i) & \text{if } d_j=b^i \text{ and } j=l,\\ \infty & \text{otherwise,}\end{cases}$$

and $F_{jl}(j,\tau)=\infty$ if $\sum_{i=1}^m \tau_i \geq 2$.

Let $t \in \{j+1,\ldots,l\}$, and $\tau \in \mathbb{Z}_+^m$. If we produce b^i units for some $i \in \{0,\ldots,m\}$ in period t, then the minimum cost for periods j to $t-1$ is $F_{jl}(t-1,\tau-e_i)$. Therefore, we compute $F_{jl}(t,\tau)$ as

$$F_{jl}(t,\tau)=\begin{cases} \infty & \text{if } \sum_{i=1}^m \tau_i > t-j+1 \text{ or}\\ & (\sum_{i=1}^m \tau_i b^i \leq d_{jt} \text{ and } t<l) \text{ or}\\ & (\sum_{i=1}^m \tau_i b^i \neq d_{jl} \text{ and } t=l \text{ and } l<n) \text{ or}\\ & (\sum_{i=1}^m \tau_i b^i < d_{jl} \text{ and } t=l=n),\\ \min_{i=0,\ldots,m:\tau\geq e_i}\left\{F_{jl}\left(t-1,\tau-e_i\right)+p_t\left(b^i\right)+h_t\left(\sum_{i=1}^m \tau_i b^i-d_{jt}\right)\right\} & \\ & \text{otherwise.}\end{cases}$$

We evaluate the recursion for increasing values of t and all possible values of τ. For given t and τ, $F_{jl}(t,\tau)$ can be computed in constant time since we assume that m is fixed. As $\tau_i \leq n$ for $i = 1, \ldots, m$, we have $O(n^m)$ possible τ vectors. As a result, the function F_{jl} can be evaluated in $O(n^{m+1})$ time for a given interval $[j, l]$.

3.2 Minimum cost for an interval $[j, l]$ with a fractional period

Next, we compute the minimum cost for a regeneration interval $[j, l]$ with $1 \leq j \leq n$ when the interval contains a fractional period. Note that for an interval $[j, n]$ that is part of an optimal solution, when the interval contains a fractional period, there exists an optimal solution with $s_n = 0$. Hence, we only consider regeneration intervals in this computation.

The minimum cost when a fractional period exists is computed for two separate cases:

Case a. The fractional production amount is less than b^m.

As we are interested in solutions with one fractional period, we know that there is no production greater than b^m.

Let $\tau \in \mathbb{Z}_+^m$, $\pi \in \mathbb{Z}_+^{m-1}$ and $t \in \{j, \ldots, l\}$. If τ_i times b^i, for $i = 1, \ldots, m$, units are produced in periods j up to $t-1$ and π_i times b^i, for $i = 1, \ldots, m-1$, and $\left\lfloor \frac{d_{jl} - \sum_{i=1}^m \tau_i b^i - \sum_{i=1}^{m-1} \pi_i b^i}{b^m} \right\rfloor$ times b^m units are produced in periods $t+1$ to l, then the production amount in period t is equal to

$$\rho_{jl}(\tau, \pi) = d_{jl} - \sum_{i=1}^{m} \tau_i b^i - \sum_{i=1}^{m-1} \pi_i b^i - \left\lfloor \frac{d_{jl} - \sum_{i=1}^m \tau_i b^i - \sum_{i=1}^{m-1} \pi_i b^i}{b^m} \right\rfloor b^m.$$

Now let $G_{jl}(t, \tau, \pi)$ be the minimum cost for periods j up to t during which τ_i times b^i units for $i = 1, \ldots, m$, are produced and one time a fractional production is done given that π_i times b^i, for $i = 1, \ldots, m-1$, and $\left\lfloor \frac{d_{jl} - \sum_{i=1}^m \tau_i b^i - \sum_{i=1}^{m-1} \pi_i b^i}{b^m} \right\rfloor$ times b^m units are produced after period t, $s_{j-1} = s_l = 0$ and $s_u > 0$ for $u \in \{j, \ldots, \min\{t, l-1\}\}$.

Let $\tau \in \mathbb{Z}_+^m$ and $\pi \in \mathbb{Z}_+^{m-1}$. If $\sum_{i=1}^m \tau_i \geq 1$ or $d_{jl} \leq \sum_{i=1}^{m-1} \pi_i b^i$ or $\sum_{i=1}^{m-1} \pi_i + \left\lfloor \frac{d_{jl} - \sum_{i=1}^{m-1} \pi_i b^i}{b^m} \right\rfloor > l - j$ or $\rho_{jl}(e_0, \pi) \in \{0, b^1, \ldots, b^m\} \cup (b^m, \infty)$, we set $G_{jl}(j, \tau, \pi) = \infty$.

For other values, we compute

$$G_{jl}(j, e_0, \pi) = \begin{cases} p_j\left(\rho_{jl}\left(e_0, \pi\right)\right) + h_j\left(\rho_{jl}\left(e_0, \pi\right) - d_j\right) & \text{if } \rho_{jl}\left(e_0, \pi\right) > d_j \text{ and } j < l, \\ p_j\left(\rho_{jl}\left(e_0, \pi\right)\right) & \text{if } \rho_{jl}\left(e_0, \pi\right) = d_j \text{ and } j = l, \\ \infty & \text{otherwise.} \end{cases}$$

Now let $t \in \{j+1, \ldots, l\}$, $\tau \in \mathbb{Z}_+^m$ and $\pi \in \mathbb{Z}_+^{m-1}$. If $\sum_{i=1}^m \tau_i > t - j$ or $\sum_{i=1}^{m-1} \pi_i + \left\lfloor \frac{d_{jl} - \sum_{i=1}^m \tau_i b^i - \sum_{i=1}^{m-1} \pi_i b^i}{b^m} \right\rfloor > l - t$, then we set $G_{jl}\left(t, \tau, \pi\right) = \infty$. If $\sum_{i=1}^m \tau_i b^i + \rho_{jl}\left(\tau, \pi\right) \leq d_{jt}$ and $t < l$, then $s_t \leq 0$ and if $\sum_{i=1}^m \tau_i b^i + \rho_{jl}\left(\tau, \pi\right) \neq d_{jl}$ and $t = l$, then $s_l \neq 0$. If $d_{jl} < \sum_{i=1}^m \tau_i b^i + \sum_{i=1}^{m-1} \pi_i b^i$, then s_l cannot be zero. Moreover, we do not want to have $\rho_{jl}\left(\tau, \pi\right) \in \left\{0, b^1, \ldots, b^m\right\} \cup \left(b^m, \infty\right)$. Hence, we set $G_{jl}\left(t, \tau, \pi\right) = \infty$ in these cases. For the remaining values, we compute

$$G_{jl}\left(t, \tau, \pi\right) = h_t\left(\sum_{i=1}^m \tau_i b^i + \rho_{jl}\left(\tau, \pi\right) - d_{jt}\right) + \min\Big\{F_{jl}\left(t-1, \tau\right) + p_t\left(\rho_{jl}\left(\tau, \pi\right)\right), \\ \min_{i=0,\ldots,m:\tau \geq e_i} \left\{G_{jl}\left(t-1, \tau - e_i, \pi + \bar{e}_i\right) + p_t\left(b^i\right)\right\}\Big\},$$

where $\bar{e}_i$ is the restriction of e_i to the first $m-1$ entries. Here, we first add the inventory holding cost. If the fractional production takes place at period t, then the production cost is $p_t(\rho_{jl}(\tau, \pi))$ and the minimum cost for periods j to $t-1$ is $F_{jl}(t-1, \tau)$. If we produce b^i units in period t for some $i \in \{0, \ldots, m\}$, then the production cost is $p_t(b^i)$ and the minimum cost for periods j to $t-1$ is $G_{jl}\left(t-1, \tau - e_i, \pi + \bar{e}_i\right)$ since the fractional period is before period t.

For given t, τ and π, $G_{jl}\left(t, \tau, \pi\right)$ can be computed in constant time. Hence G_{jl} can be evaluated in $O(n^{2m})$ time.

Case b. The fractional production amount is greater than b^m.

Let $\tau \in \mathbb{Z}_+^m$, $\hat{\pi} \in \mathbb{Z}_+^m$, $t \in \{j, \ldots, l\}$ and $\hat{G}_{jl}\left(t, \tau, \hat{\pi}\right)$ be the minimum cost for periods j up to t during which τ_i times b^i units, for $i = 1, \ldots, m$, are produced and one time a fractional production $\hat{\rho}_{jl}(\tau, \hat{\pi}) = d_{jl} - \sum_{i=1}^m \tau_i b^i - \sum_{i=1}^m \hat{\pi}_i b^i > b^m$ is done given that $\hat{\pi}_i$ times b^i, for $i = 1, \ldots, m$, units are produced after period t, $s_{j-1} = s_l = 0$ and $s_u > 0$ for $u \in \{j, \ldots, \min\{t, l-1\}\}$. The function $\hat{G}_{jl}$ can be computed in a similar way to G_{jl}. As the dimension of the vector $\hat{\pi}$ is one more than the one of π, computing $\hat{G}_{jl}$ requires $O(n^{2m+1})$ time.

3.3 Time complexity

Overall, we can find the minimum cost for interval $[j, l]$ as

$$\mu_{jl} = \min_{\tau \in \{0,\ldots,n\}^m} \left\{ F_{jl}\left(l, \tau\right), G_{jl}\left(l, \tau, \bar{e}_0\right), \hat{G}_{jl}\left(l, \tau, e_0\right) \right\}.$$

Theorem 3.1 *The lot-sizing problem with piecewise concave production costs is polynomially solvable when the breakpoints of the production cost function are time-invariant and when the number of breakpoints is fixed.*

Proof. For an interval $[j, l]$ with $1 \leq j \leq l \leq n$, as evaluating the functions F_{jl}, G_{jl} and $\hat{G}_{jl}$ take $O(n^{m+1})$, $O(n^{2m})$ and $O(n^{2m+1})$ time, respectively, the minimum cost μ_{jl} can be computed in $O(n^{2m+1})$ time. Once these costs are computed, we can solve the problem by solving a shortest path problem as done for the classical lot-sizing problem. Let $G = (V, A)$ be a directed graph for $V = \{1, \ldots, n+1\}$ and $A = \{(j, l+1) : 1 \leq j \leq l \leq n\}$. The shortest path problem from node 1 to node $n+1$ in the graph G with cost μ_{jl} on arc $(j, l+1)$ with $d_{jl} > 0$ and cost 0 on arc $(j, l+1)$ with $d_{jl} = 0$, solves our problem. As μ_{jl} can be computed in $O(n^{2m+1})$ time and there are $O(n^2)$ intervals, we require $O(n^{2m+3})$ time to construct the graph. This dominates the time to compute a shortest path. Therefore, the overall complexity is $O(n^{2m+3})$ and is polynomial for fixed m. □

3.4 Special cases

Now we discuss some special cases. Suppose that the production amount in any period cannot exceed a given capacity C. This can be modeled by setting $b^m = C$ and $p_t(x) = \infty$ for $x \in (b^m, \infty)$ and $t = 1, \ldots, n$. In this case $\hat{G}_{jl} = \infty$ for all intervals $[j, l]$. Then the overall complexity of the algorithm decreases to $O(n^{2m+2})$. The constant capacity lot-sizing problem is the special case with $m = 1$. For this special case our algorithm runs in $O(n^4)$ time, and hence has the same time complexity as the one of Florian and Klein (1971).

Hellion et al. (2012) study the capacitated lot sizing problem with concave costs, minimum order quantities (L) and constant capacities (C). In order to model this special case, we let $p_t(x) = \infty$ if $x \in (0, L) \cup (C, \infty)$, so we assume that $m = 2$. In this case, again, $\hat{G}_{jl} = \infty$ for all intervals $[j, l]$. Therefore, our DP algorithm can solve this special case of the problem in $O(n^6)$ time, which is equal to the computational complexity of the algorithm of Hellion (2013).

Atamtürk and Hochbaum (2001) propose an $O(n^5)$ algorithm for the special case where the production cost function has two pieces; the first piece corresponds to regular work and the second piece represents subcontracting. As $m = 1$, our DP algorithm can also solve this problem in $O(n^5)$ time.

Finally, if we assume that backordering is allowed, we can redefine $h_t(s_t)$ as the cost of holding s_t units of inventory during period t if $s_t > 0$ and as the cost of backordering s_t units during period t if $s_t < 0$. We assume that $h_t(.)$ is a concave function on both $(-\infty, 0]$ and $[0, \infty)$, and consequently $h_t(.)$ is a piecewise concave function on $\mathbb{R}$. If we change the condition $s_t > 0$ to $s_t \neq 0$ in the definition of regeneration intervals, Theorem 2.1 and Theorem 2.2 still hold true in the case of backlogging. We can use the DP given in this section in order to solve the problem with some small modifications without changing the computational complexity.

In conclusion, for the special cases discussed above, our algorithm's performance is as good as the performance of algorithms in the literature.

4 Computational Results

In this section, we will examine the computational efficiency of our algorithm. Although our algorithm can solve the lot-sizing problem with any piecewise concave function, in order to compare the algorithm's performance with a mixed-integer programming (MIP) solver, we use piecewise linear production cost functions and linear holding costs in our computational study.

We tested three well known linearizations of piecewise linear functions: multiple choice, incremental and convex combination formulations (see, e.g., Croxton et al. (2003)). Our preliminary tests showed that the multiple choice linearization outperformed the other two linearizations. For the capacitated lot-sizing problem, this lin-

Table 1: Experimental factors when $m = 2$

Factors		# levels	**Experimental Settings** 1	2	3	4
Fixed Costs	$(f^1,\, f^2)$	3	(3000,6000)	(3000,4000)	(3000,7500)	
Variable Costs	$(c^1,\, c^2)$	4	(0,0)	(0.5,1)	(1,0.5)	(1,1)
Breakpoints	$(b^1,\, b^2)$	3	(800,1600)	(900,1800)	(1000,2000)	

earization is as follows.

$$\min \quad \sum_{t=1}^{n}\sum_{j=1}^{m}(f_t^j y_t^j + c_t^j x_t^j) + \sum_{t=1}^{n} h_t s_t \tag{5}$$

$$\text{s.t.} \quad s_{t-1} + \textstyle\sum_{j=1}^{m} x_t^j = d_t + s_t \qquad t = 1, \ldots, n, \tag{6}$$

$$MC \qquad b^{j-1} y_t^j \le x_t^j \le b^j y_t^j \qquad t = 1, \ldots, n, j = 1, \ldots, m, \tag{7}$$

$$\textstyle\sum_{j=1}^{m} y_t^j \le 1 \qquad t = 1, \ldots, n, \tag{8}$$

$$s_0 = 0, \tag{9}$$

$$s, x \ge 0, y \text{ binary.} \tag{10}$$

In this formulation, if the production amount is in the j^{th} piece of the cost function, then there is a fixed cost f_t^j and a variable cost c_t^j (see Figure 3). We assume that the production cost function is lower semicontinuous. The inventory holding cost function is a linear function and h_t is the cost of holding one unit of inventory during period t. The variable y_t^j is equal to one if the production amount in period t lies in the segment $[b^{j-1}, b^j]$. Constraints (8) ensure that at most one of the y_t^j variables is one in period t. Consequently, constraints (7) guarantee that x_t^j should be in the segment $[b^{j-1}, b^j]$ if $y_t^j = 1$, and at most one of the production variables x_t^j will be nonzero for t. Constraints (6) are inventory balance constraints and the objective function (5) is the sum of production and inventory holding costs. By constraints (9), we impose the requirement that the initial inventory is zero.

We implemented the formulation MC in Xpress 1.22 and the DP in Java (JDK 7) and run them on a 2.53 GHz Intel Core 2 Duo Machine with 4GB memory running Windows 7. We let the solver run for 1000 seconds.

In our computational study, we only consider the capacitated problem and ignore

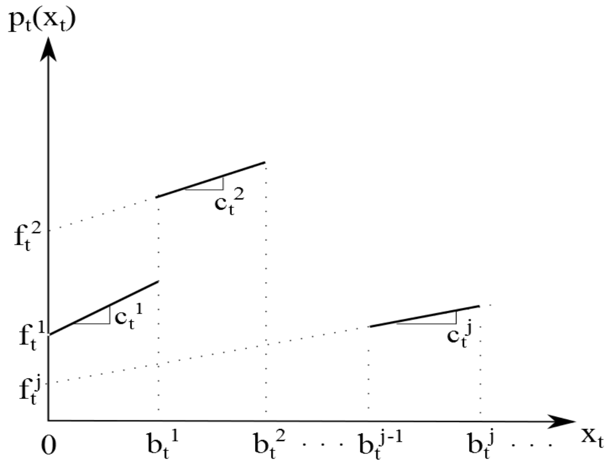

Figure 3: Production cost function for MC

the last piece of the cost function since we assume that it has a very large cost. We first analyze two cost segment instances, i.e., $m = 2$, and create randomly generated problems with different cost parameters, all time-invariant, as summarized in Table 1. Furthermore, for 40 and 50 period cases we assume that the demand has the same distribution and the holding cost is the same such that the inventory holding cost to be 0.05 and the demand to be an integer drawn from a uniform distribution, $U[400, 500]$. Consequently, for each case there are 36 randomly generated test problems.

For 20 periods and 3 cost segments instances, we consider different cost structures as summarized in Table 2. For example, increasing unit costs $(1.3, 1.5, 1.8)$ may represent a system with subcontracting, or decreasing unit costs $(1.8, 1.5, 1.3)$ may represent quantity discounts. Also, note that unit costs $(1.5, 1.3, 1.8)$ can be seen as a combination of these two systems (Figure 2a). We now generate 42 randomly generated problems for which we assume that the inventory holding cost to be 0.05 and the demand is an integer drawn from a uniform distribution, $U[500, 600]$.

As the linear programming relaxations have large gaps, valid inequalities could be used to compute better bounds. We use the valid inequalities recently developed by Sanjeevi and Kianfar (2012) for the multi-module lot-sizing problem. These inequalities are based on mixing set relaxations. We briefly describe these inequalities. Let $k < l$ be two periods and $S \subseteq \{k, \ldots, l\}$. For each $i \in S$, define $S_i = S \cap \{k, \ldots, i\}$ and

Table 2: Experimental factors when $m = 3$

Factors	# levels	**Experimental Settings** 1	2	3				
(f^1, f^2, f^3)	3	(3000,6000,9000)	(3000,5000,6500)	(3000,3500,5000)				
(b^1, b^2, b^3)	2	(500,1000,1500)	(600,1200,1800)					
		Experimental Settings 1	2	3	4	5	6	7
(c^1, c^2, c^3)	7	(0,0,0)	(1.3,1.5,1.8)	(1.3,1.8,1.5)	(1.5,1.3,1.8)	(1.5,1.8,1.3)	(1.8,1.3,1.5)	(1.8,1.5,1.3)

compute

$$n_i = \begin{cases} \min\{t : t \in S \setminus S_i\} & \text{if } S \setminus S_i \neq \emptyset, \\ l+1 & \text{if } S \setminus S_i = \emptyset. \end{cases}$$

Adding up the equations (6) from $t = k$ to $t = n_i - 1$, we obtain

$$s_{k-1} + \sum_{t=k}^{n_i-1} \sum_{j=1}^{m} x_t^j = d_{k,n_i-1} + s_{n_i-1}.$$

As $s_{n_i-1} \geq 0$ and $x_t^j \leq b^j y_t^j$, we have

$$s_{k-1} + \sum_{t \in \{k,\ldots,n_i-1\} \setminus S_i} \sum_{j=1}^{m} x_t^j + \sum_{j=1}^{m} \sum_{t \in S_i} b^j y_t^j \geq d_{k,n_i-1}.$$

Now, let $I \subseteq S$. Then $s_{k-1} + \sum_{t \in \{k,\ldots,n_{|I|}-1\} \setminus S} \sum_{j=1}^{m} x_t^j \geq s_{k-1} + \sum_{t \in \{k,\ldots,n_i-1\} \setminus S_i} \sum_{j=1}^{m} x_t^j$ for all $i \in I$. By letting $z_i^j = \sum_{t \in S_i} y_t^j$, one obtains the relaxation

$$s_{k-1} + \sum_{t \in \{k,\ldots,n_{|I|}-1\} \setminus S} \sum_{j=1}^{m} x_t^j + \sum_{j=1}^{m} b^j z_i^j \geq d_{k,n_i-1} \qquad i \in I.$$

Note that $s_{k-1} + \sum_{t \in \{k,\ldots,n_{|I|}-1\} \setminus S} \sum_{j=1}^{m} x_t^j \in \mathbb{R}_+$ and $z_i^j \in \mathbb{Z}_+$ for all $i \in I$ and $j = 1, \ldots, m$. Sanjeevi and Kianfar (2012) generate mixed n-step MIR inequalities based on this relaxation when the coefficients satisfy some conditions. Like Sanjeevi and Kianfar (2012), we consider all possible pairs k and l. We let $S = \{k, \ldots, l\}$, $S = \{t \in \{k, \ldots, l\} : \bar{y}_t^j > 0 \text{ for some } j \in \{1, \ldots, m\}\}$ and $S = \{t \in \{k, \ldots, l\} : \bar{y}_t^j < 1 \text{ for some } j \in \{1, \ldots, m\}\}$ where $(\bar{x}, \bar{y}, \bar{s})$ is the LP optimum. For these choices of S, we consider all possible two-element subsets I, i.e., $|I| = 2$, and add the resulting inequality if it is violated. We apply this cutting phase at the root node. Then we drop the inactive cuts and give the strengthened formulation to the solver.

In Tables 3-5, we report the results for the formulation MC, the formulation MC with valid inequalities (MC-CUTS) and our dynamic programming algorithm (DP). Columns BUB, LPGap, FGap correspond to the best upper bound obtained by the solver within the time limit, the percentage gap between the optimal value of the LP relaxation and the optimal value of the integer problem and the percentage gap between the best lower and upper bounds attained at the end of the time limit, respectively. Some instances are solved to optimality by MC or MC-CUTS; in this case we report the time spent to solve the formulation in parentheses in column (Time). Columns OPT and Time under DP correspond to the optimal value of the problem and the solution time of the dynamic programming algorithm.

We observe that none of the instances is solved to optimality using MC in 1000 seconds for 40 and 50 periods and 2 pieces instances and only 11 of the 42 instances of the 20 periods and 3 pieces instances are solved to optimality. As expected, the performance varies from one instance to another: the LPGap between 1 to 10% and the final gap between 0 to 5%. MC-CUTS can solve some instances in a second, whereas for others the final gap can be as large as 3-4%. Clearly, the DP has a stable solution time. Moreover, as shown in Table 5, the proposed DP can handle all of these different cost functions and solves the problems to optimality whereas the MC formulation in Xpress may end up with an optimality gap of 3% at the end of the time limit of 1000 seconds.

It can be observed from Table 3 that for each setting of fixed and variable costs, increasing breakpoint levels increases the final gaps of MC. However, we cannot generalize this result since according to Table 4 for each combination of the fixed and variable costs, MC ends up with the largest gap when $(b^1, b^2) = (800, 1600)$, and with the minimum one when $(b^1, b^2) = (900, 1800)$. Note that, all of the instances with $(b^1, b^2) = (900, 1800)$ in Table 4 are solved to optimality by MC-CUTS. Moreover, according to the final gaps of MC given in Table 3, the instances with $(f^1, f^2) = (3000, 4000)$ seem like the hardest ones. However, interestingly, when the valid inequalities are added,

Table 3: Results for $n = 40$ and $m = 2$

instance			**MC**			**MC-CUTS**			**DP**	
(f^1, f^2)	(c^1, c^2)	(b^1, b^2)	BUB	LPGap	FGap	BUB	LPGap	FGap (Time)	OPT	Time
1	1	1	69644.6	2.61	1.60	69737.1	1.37	1.22	69620.3	159.68
		2	63556.1	5.05	3.96	63779.3	3.64	3.63	63474.5	161.31
		3	57745.4	5.91	4.35	57888.0	4.02	3.90	57651.9	152.53
	2	1	78676.3	2.31	1.29	78762.6	1.15	0.61	78660.8	162.50
		2	72588.6	4.42	3.30	73285.9	3.10	1.92	72515.0	158.43
		3	66835.4	5.11	3.72	66801.9	3.44	1.33	66692.4	158.39
	3	1	79678.2	3.51	1.29	79707.4	1.51	0.76	79642.0	161.67
		2	73610	5.71	3.22	73604.6	3.00	2.38	73504.9	150.65
		3	67908.3	6.79	3.47	67967.6	3.54	2.64	67890.7	146.49
	4	1	87701.3	2.07	1.24	87781.3	1.09	0.53	87701.3	162.23
		2	81676.2	3.93	3.13	81786.8	2.83	2.72	81555.5	154.98
		3	75872.8	4.50	3.37	75797.0	3.06	2.76	75732.9	147.92
2	1	1	48332.1	6.34	2.79	48260.3	1.29	(35)	48260.3	161.80
		2	44261.8	9.22	4.67	44280.0	2.43	0.73	44261.8	152.92
		3	40523.5	10.74	5.06	40519.3	2.97	0.81	40514.8	146.48
	2	1	65963	4.03	1.31	65941.3	1.19	(315)	65941.3	161.30
		2	61905.3	5.87	2.64	61924.1	2.35	1.41	61892.8	151.16
		3	58118.9	6.64	2.72	58142.0	2.84	1.58	58098.0	144.96
	3	1	57755	5.90	2.93	57642.0	0.78	(4)	57642.0	161.41
		2	53595.3	8.01	4.27	53504.9	1.17	(3)	53504.9	153.65
		3	49913.7	9.40	4.70	49890.7	1.48	(6)	49890.7	149.81
	4	1	66363.1	4.61	1.91	66341.3	0.94	(13)	66341.3	161.61
		2	62360.1	6.55	3.36	62346.6	1.73	0.43	62342.8	153.27
		3	58616.1	7.43	3.53	58609.3	2.06	0.53	58595.8	146.49
3	1	1	69620.3	2.61	1.45	69620.3	1.30	(1)	69620.3	160.77
		2	63491.7	5.05	3.61	63552.7	3.50	2.58	63474.5	150.25
		3	57730.1	5.91	4.21	57810.7	3.98	3.42	57651.9	147.68
	2	1	78660.8	2.31	1.22	78660.8	1.15	(1)	78660.8	158.88
		2	72566.5	4.42	3.29	72626.0	3.06	2.50	72515.0	150.50
		3	66721.5	5.11	3.57	66700.1	3.44	2.74	66692.4	145.53
	3	1	87701.3	2.07	1.12	87701.3	1.03	(1)	87701.3	160.51
		2	81618.3	3.93	2.93	82461.0	2.78	2.64	81555.5	151.99
		3	75816.5	4.50	3.24	76412.7	3.04	2.56	75732.9	144.93
	4	1	87722.5	2.07	1.18	87848.6	1.03	0.51	87701.3	161.59
		2	81572.7	3.93	2.78	81679.4	2.72	2.30	81555.5	150.58
		3	75831.3	4.50	3.25	75827.0	3.03	2.50	75732.9	145.36

Table 4: Results for $n = 50$ and $m = 2$

instance			MC			MC-CUTS			DP	
(f^1, f^2)	(c^1, c^2)	(b^1, b^2)	BUB	LPGap	FGap	BUB	LPGap	FGap (Time)	OPT	Time
1	1	1	87849.5	3.97	3.17	87831.3	2.77	2.69	87727.0	719.77
		2	76621.0	1.88	0.99	76313.5	0.24	(6)	76313.5	677.49
		3	70052.2	3.65	2.20	70300.4	1.64	1.74	69943.9	653.19
	2	1	99074.2	3.52	2.76	99044.9	2.40	1.26	98959.0	744.83
		2	87718.0	1.64	0.50	87545.5	0.19	(2)	87545.5	689.00
		3	81292.8	3.14	1.86	81254.1	1.40	0.95	81175.9	658.82
	3	1	100021.0	4.52	2.57	100401.1	2.73	2.39	99990.3	741.92
		2	89027.8	3.27	0.42	89026.0	0.96	(10)	89026.0	696.00
		3	82740.4	4.92	1.76	82695.7	2.21	0.42	82695.1	658.40
	1	1	110270.0	3.16	2.49	110245.6	2.21	2.04	110191.0	720.37
		2	99265.0	1.45	0.98	98777.5	0.19	(11)	98777.5	685.54
		3	92473.8	2.76	1.56	92574.75	1.24	1.06	92407.9	653.36
2	1	1	60591.9	7.23	4.01	60598.8	2.80	1.16	60537.6	737.28
		2	53386.0	6.43	1.88	53348.5	1.49	(5)	53348.5	688.72
		3	49067.8	8.38	3.02	49035.6	2.81	(114)	49035.6	660.97
	2	1	82665.6	4.82	2.46	82684.0	2.31	1.31	82601.6	735.74
		2	75490.0	3.95	0.76	75362.5	1.08	(8)	75362.5	682.21
		3	71027.8	5.08	1.37	70999.6	2.13	(735)	70999.6	648.44
	3	1	72014.6	6.39	3.61	71990.3	2.45	(523)	71990.3	744.03
		2	64885.8	5.72	1.94	64862.9	1.26	(7)	64862.9	671.57
		3	60748.9	7.47	3.05	60695.1	2.49	(12)	60695.1	642.07
	4	1	83054.9	5.27	2.92	83001.6	2.04	0.77	83001.6	737.10
		2	75850.0	4.52	1.29	75812.5	1.05	(8)	75812.5	674.39
		3	71511.3	5.74	1.98	71499.6	1.88	(84)	71499.6	650.80
3	1	1	87801.8	3.97	3.04	87781.6	2.68	0.59	87727.0	727.36
		2	76440.4	1.88	0.48	76313.5	0.22	(1)	76313.5	676.49
		3	70020.4	3.65	2.06	70044.4	1.63	1.07	69943.8	642.60
	2	1	98989.1	3.52	2.62	98959.0	2.37	(24)	98959.0	732.44
		2	87725.5	1.64	0.52	87545.5	0.19	(2)	87545.5	680.33
		3	81269.5	3.14	1.76	81391.1	1.40	1.07	81175.9	636.95
	3	1	110247.0	3.16	2.41	111020.9	2.16	1.99	110191.0	720.73
		2	98905.0	1.45	0.40	98777.5	0.17	(2)	98777.5	681.56
		3	92543.1	2.76	1.66	92529.1	1.23	0.84	92407.9	639.65
	4	1	110341.0	3.16	2.46	111865.6	2.13	2.02	110191.0	725.70
		2	98867.5	1.45	0.35	98777.5	0.17	(2)	98777.5	677.62
		3	92483.4	2.76	1.55	92588.8	1.23	0.94	92407.9	645.63

Table 5: Results for $n = 20$ and $m = 3$

instance			MC			MC-CUTS			DP	
(f^1, f^2, f^3)	(c^1, c^2, c^3)	(b^1, b^2, b^3)	BUB	LPGap	FGap (Time)	BUB	LPGap	FGap (Time)	OPT	Time
1	1	1	69160.1	3.85	3.40	69160.2	3.81	3.42	69159.7	172.7
		2	57167	3.01	2.37	57164.1	2.99	2.40	57137	149.5
	2	1	84084.2	3.26	2.35	84084.2	3.12	2.37	84084.2	173.8
		2	71544.9	2.41	0.94	71544.9	2.39	0.86	71544.9	150.3
	3	1	84216.4	3.55	2.26	84216.4	3.18	2.27	84216.3	171.8
		2	71544.9	2.41	0.42	71544.9	2.39	0.46	71544.9	150.7
	4	1	83842.1	3.50	2.62	83866.6	3.37	2.65	83836.2	171.4
		2	71902.8	2.89	1.62	71929.0	2.75	1.65	71902.8	150.7
	5	1	84107.3	3.81	2.57	84132.8	3.02	2.49	84107.2	171.1
		2	72194.2	3.28	1.47	72194.2	2.66	1.46	72194.1	150.8
	6	1	83913.8	3.57	2.56	83908.3	3.44	2.53	83901.2	176.8
		2	72017.5	3.05	1.52	72017.5	2.90	1.56	72017.5	155.4
	7	1	84153.1	3.86	2.45	84152.0	3.01	2.38	84151.9	172.5
		2	72307.4	3.44	1.31	72307.4	2.54	1.15	72307.4	154.4
2	1	1	51062.9	5.95	2.64	51063.6	2.87	1.38	51062.9	174.5
		2	42680.2	6.23	(181.7)	42680.2	3.00	(33)	42680.2	153.5
	2	1	70712.2	3.87	1.85	70712.3	2.42	1.57	70712.2	172.4
		2	62329.6	3.78	0.84	62329.6	2.36	0.57	62329.6	151.3
	3	1	67942.4	4.84	1.93	67942.4	1.72	(993)	67942.4	173.1
		2	59184.7	4.29	(27.8)	59184.7	2.21	(6)	59184.7	151.6
	4	1	70527.8	3.60	1.95	70512.2	2.67	1.89	70512.2	173.8
		2	61893.9	3.11	1.13	61899.6	2.59	1.18	61893.9	151.2
	5	1	65874.8	5.21	1.91	65865.5	1.43	(42)	65865.5	171.9
		1	57194.1	4.83	(19.8)	57194.1	2.16	(3)	57194.1	152.7
	6	1	67488.1	4.20	1.72	67488.1	2.42	1.25	67487.3	172.6
		2	59282.2	4.45	(984.1)	59282.2	2.36	(350)	59282.1	152.2
	7	1	65653.1	4.90	1.37	65652	1.95	0.33	65651.9	172.7
		2	57307.4	5.02	(116.4)	57307.4	1.95	(3)	57307.4	155.4
3	1	1	39063.6	5.43	2.22	39062.9	3.75	2.09	39062.9	174.4
		2	32633.1	5.66	(718.2)	32633.1	3.89	(410)	32633.1	149.8
	2	1	57392.8	3.45	1.80	57394.5	3.26	1.70	57392.8	173.2
		2	50557.0	3.18	(382.2)	50557.0	2.97	(624)	50557.0	153.1
	3	1	55942.4	4.24	0.76	55942.5	2.09	0.53	55942.4	171.9
		2	49926.6	5.04	(126.8)	49926.6	2.05	(14)	49926.6	154.3
	4	1	55409.6	3.99	1.88	55409.6	3.80	1.70	55409.6	172.3
		2	48651.2	3.94	(311.2)	48651.2	3.73	(360)	48651.2	151.3
	5	1	53865.5	4.67	(515.5)	53865.5	1.75	(40)	53865.5	172.0
		2	48132.8	6.11	(144.5)	48132.8	1.84	(3)	48132.8	151.4
	6	1	55050.8	3.36	1.96	55064.1	3.17	1.99	55050.8	174.8
		2	48190.1	3.01	0.50	48181.7	2.79	0.41	48181.6	150.1
	7	1	53652.0	4.29	1.09	53652.0	2.38	0.78	53652.0	171.6
		2	47492.5	4.84	(95.4)	47492.5	2.34	(9)	47492.5	151.3

50% of these instances are solved to optimality and for the other ones with positive gaps the final gaps are relatively small compared to the other instances. A similar result can be obtained from Table 4.

Addition of valid inequalities to MC improves the LP gap in all of the instances. This improvement depends on the instance: LP gap may be decreased from 10.7% to 2.9% in one instance (in Table 3), but for another one the improvement may be quite small, like from 2.5% to 2% (in Table 4). Moreover, in Tables 3 and 4, (for the instances with positive final gap) none of the solutions found by MC-CUT at the end of time limit is optimal since the value of the best solution (BUB) is greater than the optimal value found by the DP.

For the instances with 3 pieces (Table 5), we can see that the difference between LP gaps of MC and MC-CUTS may be negligible as the improvement may be from 3.01% to 2.99% or from 2.41% to 2.39%. It can be observed from Table 5 that in 50% of the instances with positive final gaps MC finds optimal solutions but it cannot prove the optimality. Moreover, instances with smaller breakpoint levels and larger fixed costs seem harder due to the final gaps obtained by MC and MC-CUTS. However, we cannot generalize this result due to the result obtained for $m = 2$ instances; if we increase n we may come up with a different case.

According to the results given in Table 5, we cannot specify harder and easier instances with respect to the variable costs. For $(c^1, c^2, c^3) = (1.8, 1.5, 1.3)$, when $(f^1, f^2, f^3) = (3000, 5000, 6500)$ and $(b^1, b^2, b^3) = (600, 1200, 1800)$ MC-CUTS solves the problem in 3 seconds whereas when $(f^1, f^2, f^3) = (3000, 6000, 9000)$ it ends up with 1.15% optimality gap.

From Tables 3, 4 and 5, we can see that for a given breakpoint level, for instances with different fixed and variable cost settings the solution time of the DP is stable. For different breakpoint levels the difference between solution times is also very small (less than 100 seconds). On the other hand, it is hard to obtain a general result for the MIP approach. The performance of the MIP approach strongly depends on the data instance; by small changes in instances, we may come up with easier or harder instances.

Table 6: Summary of the results

(n,m)	MC						MC-CUTS						DP		
	LPGap			FGap			LPGap			FGap			Time		
	min	avg	max	min	avg	max	min	avg	max	min	avg	max	min	avg	max
(40,2)	2.07	5.08	10.74	1.12	2.94	5.06	0.78	2.31	4.02	0.00	1.43	3.90	144.9	154.2	162.5
(50,2)	1.45	4.04	8.38	0.35	1.91	4.01	0.17	1.60	2.81	0.00	0.68	2.69	637.0	687.8	744.8
(20,3)	2.41	4.08	6.23	0.00	1.27	3.40	1.43	2.70	3.89	0.00	1.10	3.42	149.5	162.4	176.8

A summary of the results is given in Table 6. In Table 6, columns named min, avg, max show the maximum, average and minimum values of the corresponding columns. As it can be observed from Table 6, when n or m increases, as expected, the solution time of DP gets larger. On the other hand, the DP solves all of the instances in less than 1000 seconds whereas Xpress may end up with positive optimality gaps even for the strengthened formulation.

We can conclude that for the small or medium sized instances, the DP outperforms the MIP approach. Furthermore, for solving larger instances of the problem we can easily modify the DP for getting good quality solutions in reasonable computation times as discussed below.

5 Heuristic for solving larger instances

The computational complexity of our dynamic programming algorithm strongly depends on the number of different τ, π and $\hat{\pi}$ vectors since we need to evaluate the functions F_{jl}, G_{jl} and $\hat{G}_{jl}$ for all possible τ, π and $\hat{\pi}$ vectors. As there are $O(n^m)$ possible τ and $\hat{\pi}$ and $O(n^{m-1})$ π vectors, for larger n and m it may not be a good choice to use the DP directly. Moreover, as Xpress could not solve some medium sized instances in our experiments, we expect its performance to get worse for larger instances.

In order to get a good solution for larger instances in a reasonable time, we develop a heuristic method based on our dynamic programming algorithm. We heuristically restrict the length of any regeneration interval (and also the final interval which may not be a regeneration interval) of a solution. Let ν $(1 \leq \nu \leq n)$ be a given upper bound on the length of any regeneration interval. We consider the interval $[j,l]$, $1 \leq$

Table 7: Experimental factors for the heuristic solution approach when $m = 3$

Factors	# levels	**Experimental Settings** 1	2	3
(f^1, f^2, f^3)	2	(3000,6000,9000)	(3000,5000,6000)	
(b^1, b^2, b^3)	2	(800,1600,2400)	(1000,2000,3000)	
(c^1, c^2, c^3)	3	(0,0,0)	(1,0.5, 0.7)	(1,0.5, 1)

$j \leq l \leq n$, and find the minimum cost μ_{jl} if $l - j + 1 \leq \nu$. Consequently, we reduce the number of intervals to be considered to $O(\nu n)$. Moreover, for a given interval $[j, l]$ the number of possible τ, π and $\hat{\pi}$ vectors become $O(\nu^m)$, $O(\nu^{m-1})$ and $O(\nu^m)$, respectively. Therefore, with this restriction we reduce the state space and consequently, the time complexity of the DP.

Note that when $\nu = n$, the restriction becomes redundant and the heuristic is the same as the exact DP. If $\nu = 1$, then the (trivial) solution is to produce in every period as much as the demand of that period. Moreover, if we know the maximum regeneration interval length in an optimal solution, say ν^*, then we can set $\nu = \nu^*$ and obtain an optimal solution to the problem with the heuristic. The performance of this heuristic depends on ν; we may obtain a better quality solution with larger ν but in longer computation time.

In order to test this solution method, we consider different ν values and compare the total cost of the solution obtained by this method with the lower bound obtained from MC-CUTS. We use larger instances that are created the same way as the instances used in the previous section. We have selected a representative set of instances to test the solution quality of the proposed heuristic. The experimental factors are listed in Table 7. For all of the instances, we assume that the inventory holding cost is 0.05 and the demand is an integer drawn from a uniform distribution, $U[400, 500]$ for all periods.

Tables 8 and 9 summarize the results of this experiment for $m = 2$ and $m = 3$, respectively. Columns under MC-CUTS represent the results for the formulation MC with valid inequalities, and the columns under DP-HEUR represent the results of our heuristic method. For each instance, we consider different ν values in order to see the trade-off between the solution quality and the solution time and we sign the rows with the minimum optimality gap by bold. With MC-CUTS, we let Xpress run 1000 and 2000 seconds and we calculate the gap of the heuristic solution using the best lower bound.

As it can be seen from Tables 8 and 9, letting Xpress run for an additional 1000 seconds results in very little improvement in the final gaps. When the cost function has two pieces (Table 8), in all of the test instances, the heuristic finds better solutions than MC-CUTS in less than 50 seconds. Moreover, as it can be revealed from the table, when ν increases, the computation time increases (as expected) but the increase is not too fast. Therefore, the user can select a higher ν value and may obtain better solutions in reasonable computation times.

In Table 9, we report the results for the instances with 3 pieces. For 50 and 80 periods, the heuristic finds better solutions than MC-CUTS in very short computation times. For 100 periods, we again find better solutions by the heuristic algorithm but the computation time of the algorithm is about 2000 seconds. But note that for the second instance of 100 periods solution found for $\nu = 18$ (in less than 1000 seconds of time) is also a better solution than that of MC-CUTS. Moreover, we believe that by letting Xpress run for more than 2000 seconds we can only obtain slightly better optimality gaps. Thus, when $m = 3$ the heuristic algorithm still reports better solutions compared to the MIP approach in less computation times. Furthermore, according to Tables 8 and 9, similar to the exact DP, for given n, m and ν values, the computation time of the heuristic algorithm is stable.

6 Conclusion

In this paper, we studied the lot-sizing problem with piecewise concave production costs. A piecewise concave function can represent economies of scale, discounts, subcontracting, overloading, minimum order quantities and capacities. The computational complexity of this problem was an open question in the literature. We developed a dynamic programming algorithm and showed that the problem is polynomially solvable when the number of breakpoints of the production cost function is fixed and the

Table 8: Results of the heuristic for $m = 2$

n	instance			MC-CUTS				DP-HEUR			
	(f^1, f^2)	(c^1, c^2)	(b^1, b^2)	**1000 seconds**		**2000 seconds**					
				BUB	Gap	BUB	Gap	ν	BUB	Gap	Time
80	1	1	3	112847.2	2.20	112847.2	2.18	10	118080.8	6.53	0.7
								12	112492.6	1.89	1.5
								14	**112488.3**	**1.88**	**3.2**
								16	112488.3	1.88	6.2
								18	112488.3	1.88	11.4
								20	112488.3	1.88	20.2
								22	112488.3	1.88	31.9
	1	4	1	175598.6	0.44	175543.3	0.40	10	175376.2	0.31	0.7
								12	175351.3	0.30	1.6
								14	175338.6	0.29	3.3
								16	175338.6	0.29	6.6
								18	175338.6	0.29	12.6
								20	**175323.8**	**0.28**	**21.0**
								22	175323.8	0.28	34.5
100	1	1	2	154649.65	1.12	154649.65	1.11	10	157515.1	2.92	0.9
								12	157297.5	2.79	1.9
								14	157277.4	2.78	4.1
								16	157250.5	2.76	8.2
								18	157250.5	2.76	15.6
								20	154558.1	1.07	26.7
								22	**154498.1**	**1.03**	**44.1**
	1	4	1	218726.9	0.93	218646.95	0.89	10	220696.6	1.81	0.9
								12	220694.2	1.81	2.0
								14	220686.7	1.81	4.2
								16	220686.7	1.81	8.4
								18	220686.7	1.81	15.6
								20	**217918.7**	**0.56**	**27.2**
								22	217912.5	0.56	46.2

Table 9: Results of the heuristic for $m = 3$

n	**instance**			**MC-CUTS**				**DP-HEUR**			
	(f^1, f^2, f^3)	(c^1, c^2, c^3)	(b^1, b^2, b^3)	**1000 seconds**		**2000 seconds**					
				BUB	Gap	BUB	Gap	ν	BUB	Gap	Time
50	1	1	2	70121.7	2.26	70096	2.21	10	72742.1	5.79	8.6
								12	**69948.1**	**2.02**	**27.3**
								14	69943.9	2.02	73.8
								16	69943.9	2.02	175.5
								18	69943.9	2.02	379.6
								20	69943.9	2.02	763.1
								22	69943.9	2.02	1448.1
	2	3	1	83227.9	1.86	83183.4	1.77	10	83433.3	2.10	8.7
								12	83433.3	2.10	27.6
								14	83430.8	2.10	75.0
								16	**83041.3**	**1.64**	**179.9**
								18	83041.3	1.64	394.3
								20	83041.3	1.64	806.2
								22	83041.3	1.64	1507.7
80	2	2	2	105659.85	1.39	105560.35	1.27	10	110780.9	5.94	14.2
								12	108169.7	3.67	46.1
								14	**105432.6**	**1.17**	**124.8**
								16	105432.6	1.17	306.2
								18	105429.5	1.17	676.0
								20	105429.5	1.17	1396.7
								22	105429.5	1.17	2700.8
	1	1	2	113043.35	2.81	113043.35	2.80	10	118080.8	6.96	14.0
								12	**112492.6**	**2.33**	**48.6**
								14	112488.3	2.33	124.6
								16	112488.3	2.33	307.4
								18	112488.3	2.33	682.2
								20	112488.3	2.33	1402.4
								22	112488.3	2.33	2729.4
100	1	1	1	173555.3	1.45	173555.3	1.45	10	175485.6	2.53	19.3
								12	175483.2	2.53	59.6
								14	175475.7	2.53	165.7
								16	175475.7	2.53	414.0
								18	175475.7	2.53	918.9
								20	**172707.7**	**0.96**	**1915.9**
								22	172701.5	0.96	3759.9
	1	2	2	131323.4	0.97	131323.4	0.95	10	137834.9	5.65	17.6
								12	135443.9	3.98	58.1
								14	132401.3	1.78	160.4
								16	132401.3	1.78	394.0
								18	131275.1	0.93	883.4
								20	**131234.7**	**0.90**	**1827.2**
								22	131234.7	0.90	3575.9

breakpoints are time invariant. The algorithm performs well for small and medium sized instances and can easily be modified to be used as a heuristic for larger instances.

In our computational studies, in order to strengthen the formulation we used existing valid inequalities in the literature for the multi-module capacitated lot sizing problem. As a future research, we can develop stronger valid inequalities for the problem.

It may also be interesting to consider the problem when one of the pieces of the production cost function is convex (but not linear), which means that the function is not piecewise concave. A convex function can indicate increasing marginal costs, therefore convex part of this function may represent overloading or cost of extra usage of a resource.

Acknowledgments

This research is partially supported by TUBITAK grant number 112M220. The research of the second author is supported by the Turkish Academy of Sciences.

References

C. Archetti, L. Bertazzi, and M.G. Speranza. Polynomial cases of the economic lot sizing problem with quantity discounts. Technical report, Department of Quantitative Methods, University of Brescia, 2011.

A. Atamtürk and D. S. Hochbaum. Capacity acquisition, subcontracting, and lot sizing. *Management Sci.*, 47(8):1081–1100, 2001.

L.M.A. Chan, A. Muriel, Z-J. Shen, and D. Simchi-Levi. On the effectiveness of zero-inventory-ordering policies for economic lot sizing model with a class of piecewise linear cost structures. *Oper. Res.*, 50(6):1058–1067, 2002.

K. L. Croxton, B. Gendron, and T. L. Magnanti. A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Management Sci.*, 49(9):1268–1273, 2003.

A. Federgruen and C-Y. Lee. The dynamic lot size model with quantity discount. *Naval Res. Logist.*, 37:707–713, 1990.

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

B. Hellion. Personal communication, 2013.

B. Hellion, F. Mangione, and B. Penz. A polynomial time algorithm to solve the single-item capacitated lot sizing problem with minimum order quantities and concave costs. *Eur. J. of Oper. Res.*, 222(1):10–16, 2012.

C-L. Li, J. Ou, and V.N. Hsu. Dynamic lot sizing with all-units discount and resales. *Naval Res. Logist.*, 59(3-4):230 – 243, 2012.

Y. Pochet and L. A. Wolsey. *Production Planning by Mixed Integer Programming.* Springer, 2006.

S. Sanjeevi and K. Kianfar. Mixed n-step MIR inequalities: Facets for the n-mixing set. *Discrete Optim.*, 9(4):216 –235, 2012.

D.X. Shaw and A. P. M. Wagelmans. An algorithm for single-item capacitated economic lot-sizing problem with piecewise linear production costs and general holding costs. *Management Sci.*, 44:831–838, 1998.

C. Swoveland. A deterministic multi-period production planning model with piecewise concave production and holding-backorder costs. *Management Sci.*, 21:1007–1013, 1975.

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

H.M. Wagner and T.M. Whitin. Dynamic version of the economic lot size model. *Management Sci.*, 5(1):89 – 96, 1958.

I. W. Zangwill. A deterministic multi-period production scheduling model with backlogging. *Management Sci.*, 13:105–119, 1966.

I. W. Zangwill. The piecewise concave function. *Management Sci.*, 13(11):900–912, 1967.