A Bilinear Reduction Based Algorithm for Solving Capacitated Multi-Item Dynamic Pricing Problems. *

Artyom G. Nahapetyan, Panos M. Pardalos †

Center for Applied Optimization

Industrial and Systems Engineering Department

University of Florida

Gainesville

September 26, 2006

Abstract

In a capacitated multi-item dynamic pricing problem one maximizes the profit by choosing a proper production level as well as pricing policy, where the latter depends on a satisfied demand. The objective function involves inventory, production and setup costs, and revenue functions. The products are required to satisfy joined production capacities. We consider a bilinear reduction of the linear mixed integer formulation of the problem and prove that the problem is equivalent to finding a global maximum of the bilinear problem. A heuristic algorithm is proposed, based on the reduction technique. Numerical experiments confirm the efficiency of the proposed technique.

Key Words: dynamic pricing, lot-sizing, supply chain, logistics.

*Research was partially supported by NSF and Air Force grants.

†Email: artyom@ufl.edu, pardalos@ufl.edu

1 Introduction

Supply chain problems with fixed costs and production planning problems involving lot-sizing have been active research topics during the last decades. Many research papers have addressed single-item problems with additional important features such as backlogging, constant and varying capacities, and different cost functions (see, e.g., Florian and Klein [9], Gilbert [11], Hoesel and Wagelmans [14], and Laparic et al. [23] and [24]). It is well known that uncapacitated problems can be reduced to a shortest path problem. Florian and M. Klein [9] have studied the capacitated single-item problems, where they characterized the optimal solution and proposed a simple dynamic programming algorithm for problems in which the capacities are the same in every period. The algorithm also can be viewed as a shortest path problem. The single-item problems with varying capacities are known to be NP-hard. A classification of different problems and a survey on existence of a polynomial algorithm for solving problems for different classes can be found in Wolsey [33], and Pochet and Wolsey [30]. Tight formulations for polynomially solvable problems are discussed in Miller and Wolsey [26], and Pochet and Wolsey [30].

Almost all practical problems involve multiple items, machines and/or levels, and polynomial results for those problems are limited. Using binary variables, one can construct a mixed integer linear programming (MIP) formulation of the problem. Different approaches to solve fixed charge transportation problems using the branch-and-bound method are discussed by Barr et al. [1], Cabot and Erenguc [6], Gray [12], Kennington and Unger [17], and Palekar et al. [29], where the authors employ penalty and decomposition techniques using the special structure of the feasible region. Marchand et al. [25] discuss cutting plain methods to solve general mixed integer problems. In addition, Cooper and Drebes [7], Diaby [8], Khang and Fujiwara [18], Khun and Baumol [22], and Hoesel Wagelmans [13] discuss approximate techniques based on linear problems. Kim and Pardalos [19] proposed a dynamic slope scaling procedure to find an approximate solution to the fixed charge network flow problem. In [20] the authors developed a variation of the procedure to solve concave piecewise linear network flow problems.

In this paper we discuss a capacitated multi-item dynamic pricing (CMDP) problem where one maximizes the profit by choosing a proper production level as well as pricing policy for each product. In the problem, the demand is a decision variable, and in order to satisfy a higher demand one needs to reduce the price of the product. On the other hand, reducing the price can decrease the revenue, which is the product of the demand and the price. In addition, the problem includes an inventory and production cost for each product, where the latter involves a setup cost. The objective of the problem is to find an optimal production strategy, which maximizes the profit subject to production capacities that are "shared" by the products. Different profit maximization or pricing variants of the single-item uncapacitated problem with deterministic demands are discussed by Gilbert [11], Loparic et al. [24], and Thomas [32]. A capacitated single-item problem with time invariant capacities is discussed by Geunes et al. [10]. The polynomial algorithms proposed by the authors are based on the corresponding results for the lot-sizing problems. In addition, there are some similarities between pricing problem discussed in this paper and a scheduling problem in a power generating system discussed by Bertsekas et al. [5]. To solve the scheduling problem, the authors applied the Lagrangian relaxation method.

Recently we have proposed a bilinear reduction technique, which can be used to find an approximate solution of concave piecewise linear and fixed charge network flow problems (see [27] and [28]). However, because of a different structure of the objective function, these methods cannot be directly applied to the CMDP problems. To solve the problem we consider another bilinear reduction of the problem with a disjoint feasible region and prove that solving the CMDP problem is equivalent to finding a global maximum of the bilinear problem. Although there are several cutting plain algorithms to find a global solution of general bilinear problems (see, e.g., Konno [21], Sherali and Shetty [31], and Horst and Tuy [16]), they employ a procedure, which converges to a local optimum of the problem, and overall performance of the procedures depends on the quality of found local solutions. The bilinear reduction of the CMDP problem typically has numerous local maxima; therefore, the algorithm may require many cuts. In this paper we propose a heuristic algorithm for finding a global solution of the bilinear reduction using approximate problems. In each iteration, the algorithm employs a well known iterative procedure to find a local maximum of the approximate problem (see, e.g., [15] and [16]) and decreases the approximation parameters. Numerical experiments on randomly generated problems confirm the efficiency of the algorithm.

For the remainder, Section 2 provides a linear mixed integer formulation of the problem and discusses a bilinear reduction of the problem. We prove that solving the CMDP problem is equivalent to finding a global maximum of the bilinear reduction. In Section 3 we propose a heuristic algorithm for solving the bilinear problem. Numerical experiments on the algorithm are provided in Section 4, and finally, Section 5 concludes the paper.

2 Problem Description

In this section we provide a nonlinear mixed integer formulation of the problem. Using some standard linearization techniques, the problem can be simplified. To solve the problem, we propose a bilinear reduction technique and prove some properties of the bilinear problem.

Let P and Δ represent the set of products and discrete times, respectively. In addition, let $f_{(p,j)}(d)$ denote the price of product p at time j as a function of the demand d, and $g_{(p,j)}(d) = f_{(p,j)}(d)d$, i.e., $g_{(p,j)}(d)$ represents the revenue obtained from selling d amount of product p at time j. In the problem, we assume that $f_{(p,j)}(d)$ and $g_{(p,j)}(d)$ are nonincreasing and concave functions, respectively (see Figures 1). If $f_{(p,j)}(d)$ is a concave function, then it is easy to show that concavity of $g_{(p,j)}(d)$ follows.

Let $x_{(p,i,j)}$ denote an amount of product p that is produced at time i to satisfy the demand at time j, and $y_{(p,i)}$ represent a binary variable, which equals one if product p is produced at time i and zero otherwise. Assume that inventory costs, $c^{in}_{(p,i,j)}$, production costs, $c^{pr}_{(p,i)}$, and setup costs, $c^{st}_{(p,i)}$, as well as production capacities, C_i, are given, where p, i, and j represent the product, producing time, and selling time, respectively. The following is the mathematical formulation of the CMDP problem.

$CMDP:$

$$\max_{x,y,d} \sum_{p\in P}\sum_{j\in\Delta} g_{(p,j)}\left(\sum_{i\in\Delta|i\le j} x_{(p,i,j)}\right) - \sum_{p\in P}\sum_{i,j\in\Delta|i\le j}\left[c^{in}_{(p,i,j)} + c^{pr}_{(p,i)}\right] x_{(p,i,j)} - \sum_{p\in p}\sum_{i\in\Delta} c^{st}_{(p,i)} y_{(p,i)}$$

$$\sum_{p\in P}\sum_{j\in\Delta|i\le j} x_{(p,i,j)} \le C_i, \qquad \forall i\in\Delta,$$

$$\sum_{j\in\Delta|i\le j} x_{(p,i,j)} \le C_i y_{(p,i)}, \qquad \forall p\in P \text{ and } i\in\Delta,$$

$$x_{(p,i,j)} \ge 0, y_{(p,i)} \in \{0,1\}, \qquad \forall p\in P \text{ and } i,j\in\Delta.$$

The objective function of the problem maximizes the profit, which is the difference between the revenue and the costs. The latter includes the inventory as well as the production costs. The first constraint ensures the satisfaction of the capacity restrictions, and the second one makes sure that $y_{(p,i)}$ equals one if $\sum_{j\in\Delta|i\le j} x_{(p,i,j)} > 0$. In the above formulation, $\sum_{i\in\Delta|i\le j} x_{(p,i,j)}$ represents the demand of product p that is satisfied in the period j.

Although the above formulation belongs to the class of nonlinear mixed integer programs, using standard techniques one can approximate the revenue function by a piecewise linear one and linearize the objective function. Doing so, observe that from the concavity of the revenue function it follows that there exists a point, $\tilde{d}_{(p,j)}$, where the function reaches its maximum (see Figure 1). As a result, producing and selling more than $\tilde{d}_{(p,j)}$ is not profitable, and at optimality $\sum_{i\in\Delta|i\le j} x_{(p,i,j)} \le \tilde{d}_{(p,j)}$. To linearize the revenue function, divide $\left[0,\tilde{d}_{(p,j)}\right]$ into N intervals of equal length. Let $d^k_{(p,j)}$ denote the end points of the intervals, i.e., $d^k_{(p,j)} = k\tilde{d}_{(p,j)}/N$, $\forall k\in\{1,\ldots,N\}\bigcup\{0\} = K\bigcup\{0\}$, and $g^k_{(p,j)}$ represents the value of the revenue function at the point $d^k_{(p,j)}$, i.e., $g^k_{(p,j)} = g_{(p,j)}(d^k_{(p,j)}) = f_{(p,j)}(d^k_{(p,j)})d^k_{(p,j)}$. Using those parameters, construct the function

$$\tilde{g}_{(p,j)}(\lambda_{(p,j)}) = \sum_{k=0}^{N} g^k_{(p,j)}\lambda^k_{(p,j)} = \sum_{k=1}^{N} g^k_{(p,j)}\lambda^k_{(p,j)},$$

where it is required that

$$\sum_{i\in\Delta|i\le j} x_{(p,i,j)} = \sum_{k=0}^{N} d^k_{(p,j)}\lambda^k_{(p,j)} = \sum_{k=1}^{N} d^k_{(p,j)}\lambda^k_{(p,j)}, \qquad \forall p\in P \text{ and } j\in\Delta,$$

$$\sum_{k=0}^{N} \lambda_{(p,j)}^{k} = 1, \qquad \lambda_{(p,j)}^{k} \geq 0, \qquad \forall p \in P \text{ and } j \in \Delta,$$

and $\lambda_{(p,j)}^{k} \neq 0$ for at most two consecutive indices k. Observe that $g_{(p,j)}(d)$ is a concave nondecreasing function on the interval $\left[0, \tilde{d}_{(p,j)}\right]$; therefore, its piecewise linear approximation, i.e., $\tilde{g}_{(p,j)}(\lambda_{(p,j)})$, preserves the same property. From the latter, it follows that in the maximization problem the requirement on $\lambda_{(p,j)}^{k}$ being positive for at most two consecutive indices k can be removed from the formulation, and it is satisfied at optimality (for details see Chapter 11, Bazaraa et al. [2], Beale and Tomlin [3], and Beale and Forrest [4]). The following is the mathematical formulation of the approximation problem:

$$\max_{x,y,\lambda} \sum_{p\in P} \sum_{j\in\Delta} \sum_{k\in K} g_{(p,j)}^{k} \lambda_{(p,j)}^{k} - \sum_{p\in P} \sum_{i,j\in\Delta|i\leq j} \left[c_{(p,i,j)}^{in} + c_{(p,i)}^{pr}\right] x_{(p,i,j)} - \sum_{p\in p} \sum_{i\in\Delta} c_{(p,i)}^{st} y_{(p,i)},$$

$$\sum_{p\in P} \sum_{j\in\Delta|i\leq j} x_{(p,i,j)} \leq C_i, \qquad \forall i \in \Delta,$$

$$\sum_{j\in\Delta|i\leq j} x_{(p,i,j)} \leq C_i y_{(p,i)}, \qquad \forall p \in P \text{ and } i \in \Delta,$$

$$\sum_{i\in\Delta|i\leq j} x_{(p,i,j)} = \sum_{k\in K} d_{(p,j)}^{k} \lambda_{(p,j)}^{k}, \qquad \forall p \in P \text{ and } j \in \Delta,$$

$$\sum_{k=0}^{N} \lambda_{(p,j)}^{k} = 1, \qquad \forall p \in P \text{ and } j \in \Delta,$$

$$x_{(p,i,j)} \geq 0, \lambda_{(p,j)}^{k} \geq 0, y_{(p,i)} \in \{0,1\}, \qquad \forall p \in P, i,j \in \Delta \text{ and } k \in K \cup \{0\}.$$

Next, we simplify the formulation using nonnegative variables $x_{(p,i,j)}^{k}$, $k \in K$, instead of $x_{(p,i,j)}$, where $x_{(p,i,j)}^{k}$ represents the amount of product p that is produced at time i and sold at time j using the unit price $g_{(p,j)}^{k}/d_{(p,j)}^{k} = f_{(p,j)}^{k} = f_{(p,j)}(d_{(p,j)}^{k})$. Doing so, the third constraint in the above formulation can be replaced by the following one:

$$\sum_{i\in\Delta|i\leq j} x_{(p,i,j)}^{k} = d_{(p,j)}^{k} \lambda_{(p,j)}^{k}, \qquad \forall p \in P, j \in \Delta \text{ and } k \in K.$$

The latter can be used to remove the variable $\lambda_{(p,j)}^{k}$ from the formulation. In particular, the fourth constraint is replaced by

$$\sum_{k\in K} \sum_{i\in\Delta|i\leq j} \frac{x_{(p,i,j)}^{k}}{d_{(p,j)}^{k}} \leq \sum_{k\in K} \sum_{i\in\Delta|i\leq j} \frac{x_{(p,i,j)}^{k}}{d_{(p,j)}^{k}} + \lambda_{(p,j)}^{0} = 1, \qquad \forall p \in P \text{ and } j \in \Delta,$$

and in the objective

$$g^k_{(p,j)}\lambda^k_{(p,j)} = f^k_{(p,j)}\left[\sum_{i\in\Delta|i\leq j} x^k_{(p,i,j)}\right] \qquad \forall p\in P, j\in\Delta \text{ and } k\in K.$$

The following is the resulting alternative formulation of the approximation problem, which we refer to as ACMDP:

$ACMDP:$

$$\max_{x,y}\sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}y_{(p,i)}\right]$$

$$\sum_{p\in P}\sum_{j\in\Delta|i\leq j}\sum_{k\in K} x^k_{(p,i,j)} \leq C_i, \qquad \forall i\in\Delta, \tag{1}$$

$$\sum_{j\in\Delta|i\leq j}\sum_{k\in K} x^k_{(p,i,j)} \leq C_i y_{(p,i)}, \qquad \forall p\in P \text{ and } i\in\Delta, \tag{2}$$

$$\sum_{k\in K}\sum_{i\in\Delta|i\leq j}\frac{x^k_{(p,i,j)}}{d^k_{(p,j)}} \leq 1, \qquad \forall p\in P \text{ and } j\in\Delta, \tag{3}$$

$$x^k_{(p,i,j)} \geq 0, y_{(p,i)}\in\{0,1\}, \qquad \forall p\in P, i,j\in\Delta \text{ and } k\in K, \tag{4}$$

where $q^k_{(p,i,j)} = f^k_{(p,j)} - c^{in}_{(p,i,j)} - c^{pr}_{(p,i)}$. Observe that at optimality $x^{*k}_{(p,i,j)} = 0$ for all indices such that $q^k_{(p,i,j)} \leq 0$, and those variables can be removed from the formulation. Therefore, without lost of generality, in the analysis below, we assume that $q^k_{(p,i,j)} > 0$.

Define $X = \{x|x \geq 0 \text{ and } x^k_{(p,i,j)} \text{ are feasible to (1) and (3)}\} \subset \mathbb{R}_+^{|P||K||\Delta|^2}$, and $Y = [0,1]^{|P||\Delta|}$. Consider the following bilinear program:

$ACMDP-B:$

$$\max_{x,y}\sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}\right]y_{(p,i)} = \varphi(x,y)$$

$$x\in X \text{ and } y\in Y.$$

Theorem 1 *Any local maximum of the ACMDP-B problem is feasible or can be transformed into a feasible solution of the ACMDP problem with the same objective function value.*

Proof: Let (x^*, y^*) denote a local maximum of the ACMDP-B problem. Observe that by fixing x to the value of the vector x^*, the ACMDP-B problem reduces to a linear one, and y^* is an optimal solution of the resulting problem. Assume that $\exists p \in P$ and $i \in \Delta$ such that $y^*_{(p,i)} \in (0,1)$, i.e., $y^*_{(p,i)}$ is a fractional number. If $\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} < 0$ or $\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} > 0$ then it is possible to improve the objective function value by assigning $y^*_{(p,i)} = 0$ or $y^*_{(p,i)} = 1$, respectively. The latter contradicts the optimality of (x^*, y^*). On the other hand, if $\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} = 0$ then by changing the value of the variable $y^*_{(p,i)}$ to zero the objective function value remains the same. Construct a vector $\hat{y}$, where $\hat{y}_{(p,i)} = \lfloor y^*_{(p,i)} \rfloor$. Note that $(x^*, \hat{y})$ is feasible to constraints (1), (3) and (4). If $(x^*, \hat{y})$ violates constraint (2) then $\exists p \in P$ and $i \in \Delta$ such that $\sum_{j\in\Delta|i\leq j}\sum_{k\in K} x^{*k}_{(p,i,j)} > 0$ and $\hat{y}_{(p,i)} = 0$. From the local optimality of $(x^*, \hat{y})$ it follows that $\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} \leq 0$, and it is not profitable to produce product p at time i. Furthermore, by assigning $x^{*k}_{(p,i,j)} = 0$, $\forall j \in \Delta$ and $k \in K$, the objective function value of the ACMDP-B problem remains the same. Let $\hat{x}$ denote the resulting vector, i.e.,

$$\hat{x}^k_{(p,i,j)} = \begin{cases} 0 & \text{if } \sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} \leq 0 \\ x^{*k}_{(p,i,j)} & \text{if } \sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^{*k}_{(p,i,j)} - c^{st}_{(p,i)} > 0 \end{cases} \tag{5}$$

The vector $(\hat{x}, \hat{y})$ is feasible to the ACMDP as well as the ACMDP-B problem and has the same objective function value as (x^*, y^*). ■

Theorem 2 *A global maximum of the ACMDP-B problem is a solution or can be transformed into a solution of the ACMDP problem.*

Proof: Observe that any feasible solution of the ACMDP is feasible to the ACMDP-B problem. Furthermore, if (x, y) is feasible to the ACMDP problem then

$$\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}y_{(p,i)} = \left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}\right] y_{(p,i)}, \quad \forall p \in P \text{ and } i \in \Delta.$$

From the latter it follows that the ACMDP-B problem can be obtained from ACMDP by replacing the objective function by

$$\max_{x,y} \sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}\right] y_{(p,i)},$$

Procedure 1 :

```
Step 1: Let $y^0$ denote an initial binary vector of $y_{(p,i)}$. $m \leftarrow 1$.
Step 2: Let $x^m = argmax\{LP(y^{m-1})\}$, and $y^m = argmax\{LP(x^m)\}$.
Step 3: If $y^m = y^{m-1}$ then stop. Otherwise, $m \leftarrow m+1$ and go to Step 2.
```

removing constraint (2) and relaxing the integrality of the variable $y_{(p,i)}$. In other words, the ACMDP-B problem is a relaxation of the ACMDP problem. From Theorem 1 it follows that a global solution of ACMDP-B is a solution (or can be transformed into a solution) of the ACMDP problem. ■

The above two theorems prove that solving the ACMDP problem is equivalent to finding a global maximum of the ACMDP-B problem. In particular, one can solve the ACMDP-B problem and if the solution is not feasible to the ACMDP problem, then use the method described in the proof of Theorem 1 to construct a feasible one with the same objective function value.

3 Solution Algorithm

In this section we discuss a heuristic algorithm for solving the ACMDP-B problem, which employs a well known iterative procedure for finding a local maximum of the ACMDP-B problem.

Observe that the problem belongs to the class of bilinear programs. By fixing vector x or y to a particular value, the problem can be reduced to a linear one. Let $LP(x)$ and $LP(y)$ denote the corresponding linear programs, i.e.,

$$LP(x): \quad \max_{y\in Y} \sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)}\right] y_{(p,i)}, \text{ and}$$

$$LP(y): \quad \max_{x\in X} \sum_{p\in P}\sum_{i\in\Delta}\sum_{j\in\Delta|i\le j}\sum_{k\in K}\left[q^k_{(p,i,j)}y_{(p,i)}\right] x^k_{(p,i,j)}.$$

Notice that the solution of the $LP(x)$ is easy to obtain. In particular,

$$y^*_{(p,i)} = \begin{cases} 0 & \text{if } \sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)} \le 0 \\ 1 & \text{if } \sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)} > 0 \end{cases}$$

is an optimal solution of the problem. The Procedure 1 describes a well known algorithm, which starts from an initial binary vector and converges to a local maximum of the ACMDP-B problem using a finite number of iterations (see [16] or [15]). However, the procedure has the following disadvantage.

Let (x^m, y^m) represent the solution obtained on iteration m, and assume that $\exists p \in P$ and $i \in \Delta$ such that $y^m_{(p,i)} = 0$. As a result, in the $LP(y^m)$ problem $q^k_{(p,i,j)} y^m_{(p,i)} = 0$, $\forall j \in \Delta$, $i \leq j$, and $k \in K$, and perturbations of the values of the corresponding variables $x^k_{(p,i,j)}$ do not change the objective function value. Furthermore, because the products "share" the capacity and other products can have a positive cost in the $LP(y^m)$ problem, it is likely that the value of the variable $x^{mk}_{(p,i,j)}$ decreases in the next iteration. From the latter it follows that $y^{m+1}_{(p,i)} = 0$. To summarize, if $y^m_{(p,i)} = 0$ then it is likely that *i)* $y^n_{(p,i)} = 0$, $\forall n > m$, and *ii)* the final solution is far from being a global one. To overcome those difficulties, next we propose an approximate problem, which avoids having zero costs in the objective of the $LP(y)$ problem.

Let $\varphi^1_{(p,i)}(x_{(p,i)}) = \sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} - c^{st}_{(p,i)}$ and

$$\varphi^2_{(p,i)}(x_{(p,i)}) = \frac{\varepsilon_{(p,i)}}{\varepsilon_{(p,i)} + c^{st}_{(p,i)}} \sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)},$$

where $\varepsilon_{(p,i)} > 0$ and $x_{(p,i)}$ is the vector of $x^k_{(p,i,j)}$. It is easy to show that both functions have the same value, $\varepsilon_{(p,i)}$, on the hyperplane $\sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} = \varepsilon_{(p,i)} + c^{st}_{(p,i)}$. Furthermore, $\varphi^1_{(p,i)}(x_{(p,i)}) > \varphi^2_{(p,i)}(x_{(p,i)})$ if $\sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} > \varepsilon_{(p,i)} + c^{st}_{(p,i)}$ and $\varphi^1_{(p,i)}(x_{(p,i)}) < \varphi^2_{(p,i)}(x_{(p,i)})$ if $\sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} < \varepsilon_{(p,i)} + c^{st}_{(p,i)}$. Define

$$\varphi^\varepsilon(x, y) = \sum_{p\in P} \sum_{i\in\Delta} \left[\varphi^1_{(p,i)}(x_{(p,i)}) y_{(p,i)} + \varphi^2_{(p,i)}(x_{(p,i)})(1 - y_{(p,i)})\right]$$

where ε denotes the vector of $\varepsilon_{(p,i)}$. The function $\varphi^\varepsilon(x, y)$ depends on the value of the vector ε, and $\varphi^\varepsilon(x, y) \geq \varphi(x, y)$, for all $\varepsilon > 0$ and (x, y) feasible to the ACMDP-B problem. Observe that if $\varepsilon \to 0$ then $\varphi^2_{(p,i)}(x_{(p,i)}) \to 0$, and $\varphi^\varepsilon(x, y) \to \varphi(x, y)$. In that sense, $\varphi^\varepsilon(x, y)$ is an ε-approximation of $\varphi(x, y)$, and it approximates the function from above. By replacing the objective function of the ACMDP-B problem by the function $\varphi^\varepsilon(x, y)$, we refer to the resulting problem as ε-approximation of the ACMDP-B problem and denote by ACMDP-B(ε).

Theorem 3 *There exists a sufficiently small $\varepsilon > 0$ such that a solution of the ACMDP-B problem is a solution of the ACMDP-B(ε) problem.*

Proof: Let (x^*, y^*) denote a vector solution of the ACMDP-B problem. As we have shown in the proof of Theorem 1, $(\hat{x}, \hat{y})$ is an alternative solution of the ACMDP-B problem, where $\hat{y}_{(p,i)} = \lfloor y^*_{(p,i)} \rfloor$ and $\hat{x}$ is computed according to the formula (5). Furthermore, from the Theorem 2 it follows that $(\hat{x}, \hat{y})$ is a solution of the ACMDP problem. Observe that according to formula (5) for all $p \in P$ and $i \in \Delta$ either $\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}\hat{x}^k_{(p,i,j)} > c^{st}_{(p,i)}$ or $\hat{x}^k_{(p,i,j)} = 0$, $\forall k \in K, j \in \Delta, i \le j$. Let

$$\hat{\varepsilon} = \min_{p\in P, i\in\Delta}\left\{\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}\hat{x}^k_{(p,i,j)} - c^{st}_{(p,i)} | \sum_{j\in\Delta|i\le j}\sum_{k\in K}\hat{x}^k_{(p,i,j)} > 0\right\}.$$

It is easy to show that $\phi^\varepsilon(\hat{x}, \hat{y}) = \phi(\hat{x}, \hat{y}) \le \phi(x, y) \le \phi^\varepsilon(x, y)$ for all (x, y) feasible to the problem, where ε denotes the vector of $\varepsilon_{(p,i)} = \hat{\varepsilon}$. Therefore, $(\hat{x}, \hat{y})$ is a solution of the ACMDP-B(ε). ■

Construct the corresponding $LP^\varepsilon(x)$ and $LP^\varepsilon(y)$ linear problems by fixing vectors x and y in the ACMDP-B(ε) problem to a particular value, i.e.,

$LP^\varepsilon(x)$: $\max\limits_{y\in Y} \sum_{p\in P}\sum_{i\in\Delta} \varphi^1_{(p,i)}(x_{(p,i)})y_{(p,i)} + \varphi^2_{(p,i)}(x_{(p,i)})(1 - y_{(p,i)})$, and

$LP^\varepsilon(y)$: $\max\limits_{x\in X} \sum_{p\in P}\sum_{i\in\Delta}\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}\left[y_{(p,i)} + \frac{\varepsilon_{(p,i)}}{\varepsilon_{(p,i)}+c^{st}_{(p,i)}}(1 - y_{(p,i)})\right] x^k_{(p,i,j)}$.

As before, the solution of the $LP^\varepsilon(x)$ problem is easy to obtain by assigning

$$y^*_{(p,i)} = \begin{cases} 0 & \text{if } \sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)} \le \varepsilon_{(p,i)} \\ 1 & \text{if } \sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)}x^k_{(p,i,j)} - c^{st}_{(p,i)} > \varepsilon_{(p,i)} \end{cases} =$$

$$\begin{cases} 0 & \text{if } \varphi^1_{(p,i)}(x_{(p,i)}) \le \varphi^2_{(p,i)}(x_{(p,i)}) \\ 1 & \text{if } \varphi^1_{(p,i)}(x_{(p,i)}) > \varphi^2_{(p,i)}(x_{(p,i)}) \end{cases}.$$

The heuristic procedure (see Procedure 2) starts with a sufficiently large ε and finds a local maximum of the resulting ε-approximation problem. If the stopping criteria is not satisfied in Step 3 then it decreases the value of the vector ε to $\alpha\varepsilon$, where α is a constant from the open interval $(0, 1)$, and the process continues using a new ε-approximation problem. Observe that Procedure 2 uses vector y^m from the previous iteration as an initial

Procedure 2 :

```
Step 1: Let $\varepsilon_{(p,i)}$ be a sufficiently large number, and $y^0$ be such that $y^0_{(p,i)} = 1$, $\forall p \in P$ and $i \in \Delta$. $m \leftarrow 0$.
Step 2: Construct the $\varepsilon$-approximation problem ACMDP-B($\varepsilon$) and run Procedure 1 to find a local maximum of the problem, where $y^m$ is an initial binary vector. Let $(x^{m+1}, y^{m+1})$ denote the local maximum.
Step 3: If $\exists p \in P$ and $i \in \Delta$ such that $\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)} x^{(m+1)k}_{(p,i,j)} - c^{st}_{(p,i)} \le \varepsilon^m_{(p,i)}$ and $\sum_{j\in\Delta|i\le j}\sum_{k\in K} x^{(m+1)k}_{(p,i,j)} > 0$ then $\varepsilon \leftarrow \alpha\varepsilon$, $m \leftarrow m+1$ and go to Step 2. Otherwise, stop.
```

Procedure 3 :

```
Assign $x^k_{(p,i,j)} = 0$, $\forall p \in P$, $i, j \in \Delta$, and $k \in K$
for all $p \in P$ and $i \in \Delta$ do
  $\hat{C} = C_i$, $\hat{q}^k_{(p,i,j)} = q^k_{(p,i,j)}$, $q^{max} = \max\{\hat{q}^k_{(p,i,j)} d^k_{(p,j)} | k \in K, j \in \Delta, j \ge i\}$
  while $\hat{C} \ne 0$ and $q^{max} \ne 0$ do
    Let $j^{max}$ and $k^{max}$ are such that $q^{max} = \hat{q}^{k^{max}}_{(p,i,j^{max})} d^{k^{max}}_{(p,j)}$.
    Assign $x^{k^{max}}_{(p,i,j^{max})} = \min\{\hat{C}, d^{k^{max}}_{(p,j^{max})}\}$, $\hat{C} = \hat{C} - x^{k^{max}}_{(p,i,j^{max})}$, $\hat{q}^k_{(p,i,j^{max})} = 0$, $\forall k \in K$, and $q^{max} = \max\{\hat{q}^k_{(p,i,j)} d^k_{(p,j)} | k \in K, j \in \Delta, j \ge i\}$
  end while
  $\varepsilon_{(p,i)} = \sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} - c^{st}_{(p,i)}$
end for
```

vector. Let (x^*, y^*) denote the solution returned by the procedure. From the stopping criteria it follows that for all $p \in P$ and $i \in \Delta$ either $\sum_{j\in\Delta|i\le j}\sum_{k\in K} q^k_{(p,i,j)} x^{*k}_{(p,i,j)} > c^{st}_{(p,i)}$ and $y^*_{(p,i)} = 1$ or $x^{*k}_{(p,i,j)} = 0$, $\forall k \in K, j \in \Delta, i \le j$ and $y^*_{(p,i)} = 0$; therefore, x^* and y^* are solutions of $LP(y^*)$ and $LP(x^*)$ problems, respectively. Because (x^*, y^*) solves those two linear problems, and y^* is a unique solution of $LP(x^*)$, one concludes that it is a local maximum of the ACMDP-B problem.

The procedure depends on two parameters: the initial vector ε and the value of α. The value of $\varepsilon_{(p,i)}$ depends on parameters of the problem, and one can consider them equal to the maximum profit, which can be obtained by producing only product p at time i. Although such maximization problem is easy to solve using standard LP solvers, for large $|P|$ and $|\Delta|$ one finds computationally expensive solving the problem for all pairs $(p, i) \in P \times \Delta$. Instead, we propose an algorithm for finding the values of $\varepsilon_{(p,i)}$ (see Procedure 3). Observe that $x^k_{(p,i,j)} \leq d^k_{(p,j)}$, and the maximum additional profit that can be obtained using the variable $x^k_{(p,i,j)}$ is $q^k_{(p,i,j)} d^k_{(p,j)}$. Using this property, for all pairs (p, i) the procedure iteratively finds the maximum among $q^k_{(p,i,j)} d^k_{(p,j)}$ and assigns the demand (or the remanning of the capacity) to the corresponding variable $x^k_{(p,i,j)}$. The value of $\varepsilon_{(p,i)}$ is computed based on the formula $\varepsilon_{(p,i)} = \sum_{j\in\Delta|i\leq j} \sum_{k\in K} q^k_{(p,i,j)} x^k_{(p,i,j)} - c^{st}_{(p,i)}$. As for the parameter α, a larger value increases the computational time of the procedure, and it is likely to provide a better solution.

4 Numerical Experiments

In this section we discuss numerical experiments conducted on randomly generated problems. The problems are solved by Procedure 1 as well as Procedure 2 using different values for the parameter α. The latter procedure employs Procedure 3 to find an initial value for the vector ε. In addition, we solve the problems by the MIP solver of CPLEX using the ACMDP formulation. In the cases where the MIP solver is not able to solve large problems within posted CPU and memory limitations, we compare the solutions of the procedures with the best solutions found by CPLEX. The main purpose of the computations is the performance of the procedures for different capacities.

In the numerical experiments we consider problem sets with different numbers of products, $|P| = 5$, 10, or 20, and time horizons, $|\Delta| = 12$ or 52. For each problem set we randomly generate capacities for all $i \in \Delta$ using the formula $C_i = |P|U$, where U is a random number uniformly generated from interval $[10, 100]$, $[50, 150]$, $[100, 200]$, or $[150, 250]$. Note that all intervals allow generating capacities that are tight at optimality with respect to the revenue function discussed below. In addition, using term $|P|$ one generates capacities that depend on the number of products. The latter allows comparing of results across different numbers of products. As for the costs, we generate the production costs $c^{pr}_{(p,i)}$ and the inventory costs $c^{in}_{(p,i)}$ for each product p and period i according to the uniform distributions $U[20,40]$ and $U[4,8]$, respectively. Observe that on average the inventory cost is equal to 20% of the production cost. Using $c^{in}_{(p,i)}$, the inventory cost between i and j periods is computed based on the formula $c^{in}_{(p,i,j)} = \sum_{i<r\leq j} c^{in}_{(p,r)}$. Finally the setup cost $c^{st}_{(p,i)}$ is generated uniformly from interval $[600, 1000]$.

In the experiments we restrict ourself by considering only linear price functions of the form $f_{(p,j)}(d) = f^{max}_{(p,j)} - (f^{max}_{(p,j)}/d^{max}_{(p,j)})d$. To avoid generating functions that at optimality result in unrealistically large profits, we introduce an index β, where

$$\beta = \frac{\sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K}\left(f^k_{(p,j)} - c^{in}_{(p,i,j)} - c^{pr}_{(p,i)}\right)x^{*k}_{(p,i,j)} - c^{st}_{(p,i)}y^*_{(p,i)}\right]}{\sum_{p\in P}\sum_{i\in\Delta}\left[\sum_{j\in\Delta|i\leq j}\sum_{k\in K}(c^{in}_{(p,i,j)} + c^{pr}_{(p,i)})x^{*k}_{(p,i,j)} + c^{st}_{(p,i)}y^*_{(p,i)}\right]}.$$

That is, the index measures the amount of the profit per unit of investment and it is computed based on the optimal or the best solution provided by CPLEX. By generating $f^{max}_{(p,j)}$ and $d^{max}_{(p,j)}$ according to the uniform distributions $U[70,90]$ and $U[500,1000]$, respectively, at optimality *i)* $\beta \in [0.7, 1.3]$, *ii)* all capacities considered above are tight, and *iii)* in most of the cases the satisfied demand is less than $\tilde{d}_{(p,j)}$ (see Figure 1). In addition, the proposed price function and distributions of the costs and capacities allow generating problems that have an optimal objective function value ranging from hundreds of thousands to several millions. Finally, in the construction of the piecewise linear approximation of the revenue function we use $N = 10$.

The model is constructed using the GAMS environment and solved by CPLEX 9.0 with a CPU restriction of 2000 sec. and a memory restriction of 2Gb, where the latter is the memory that is required to store the tree in the branch-and-bound algorithm. Computations are made on a Unix machine with dual Pentium 4 3.2Ghz processors and 6GB of memory. The results are tabulated in the Appendix.

In the experiments we solve 10 randomly generated problems for each problem set and capacity. Tables 1 and 2 compare the results provided by CPLEX with the solutions provided by both procedures. The relative error is computed using the formula

$$RE(\%) = \frac{Obj_{CPLEX} - Obj_{Proc.2(1)}}{Obj_{CPLEX}} 100.$$

In the Table 1, column A indicates the number of problems where the heuristic procedure finds a better solution than CPLEX. Note that CPLEX is able to provide an exact solution for all capacities from the problem set 5-12. In all other cases, the solver stops after reaching the CPU limit or the memory limit and returns the best found solution. Although the relative optimality gap of the final solutions of those problem sets varies from 2% to 5%, we believe that the solution is an optimal or close to an optimal one, and the large optimality gap is due to imperfect lower bounds. The fact that the heuristic procedures provide a slightly better solution in the case with $|\Delta| = 52$ than $|\Delta| = 12$ partially confirms our assumptions.

The relative errors in the Table 1 confirms the effectiveness of the heuristic procedure. In particular, in the majority of the problems the heuristic algorithm is able to provide a solution within 1% from the optimal one or the best one provided by CPLEX. Observe that the larger value of α provides a better solution and the number of problems where the heuristic procedure finds a better solution than CPLEX is increasing with the size of the problem. By comparing with the solutions provided by Procedure 1 (see Table 2) one notices that Procedure 2 outperforms the Procedure 1, and it is more stabile to the changes in the capacities. As for the CPU time (see Table 3), the heuristic procedures require less resources than CPLEX. In addition, unlike CPLEX the heuristic procedures do not require gigabytes of memory to store the tree.

5 Concluding Remarks

We have discussed a bilinear reduction scheme for the capacitated multi-item dynamic pricing problem, where solving the latter is equivalent to finding a global solution of the former. Based on theoretical results of the reduction problem, two procedures have been proposed to find a global maximum of the problem. The first one is a well known technique and has been intensively used to solve other bilinear problems. Because of the reasons discussed in Section 3, in the very beginning of the iterative process the procedure eliminates some products from the further consideration. The latter worsen the quality of the solution returned by the procedure. In the second procedure we construct approximate problems and gradually decrease parameters of the problems. As a result, during the iterative process the costs of the eliminated products remain positive and the procedure considers them again if need be. Although the second procedure requires more CPU time to stop than the first one, it provides a higher-quality solution.

The discussed technique can be easily applied to other pricing problems with a different pricing policy, e.g., a discounting policy, or a problem with backlogging. In addition, the technique can be applied to other production planning or supply chain management problems with additional restrictions on the feasible region.

6 Acknowledgements

The authors would like to thank Prof. L. A. Wolsey and Prof. J. P. Geunes for their suggestions on the generated test problems and useful discussion during the course of this research.

APPENDIX: Computational Results

Table 1: The Quality of the Solution: Procedure 2.

			CPLEX	Procedure 2 using Procedure 3 to find vector ε								
				$\alpha = 1/2$			$\alpha = 2/3$			$\alpha = 9/10$		
$\|P\|$-$\|\Delta\|$	C_i	β	Obj	Obj	RE%	A	Obj	RE%	A	Obj	RE%	A
5-12	[10,100]	1.02	132,803	131,206	1.19	-	131,312	1.11	-	131,612	0.88	-
	[50,150]	0.95	216,831	215,562	0.58	-	215,956	0.40	-	215,992	0.39	-
	[100,200]	0.86	283,494	282,418	0.38	-	282,752	0.26	-	282,913	0.20	-
	[150,250]	0.76	320,602	319,325	0.39	-	319,501	0.34	-	319,518	0.34	-
5-52	[10,100]	1.03	543,397	537,588	1.07	-	538,182	0.97	-	539,264	0.76	-
	[50,150]	0.96	910,778	906,211	0.50	-	907,304	0.38	1	909,156	0.18	3
	[100,200]	0.88	1,209,347	1,206,840	0.21	-	1,208,155	0.10	2	1,208,461	0.07	2
	[150,250]	0.78	1,378,794	1,375,431	0.25	-	1,376,059	0.20	-	1,377,227	0.11	1
10-12	[10,100]	1.13	246,410	243,290	1.29	-	244,268	0.87	-	244,754	0.70	-
	[50,150]	0.98	420,000	417,032	0.70	-	417,034	0.70	-	418,337	0.40	-
	[100,200]	0.98	563,297	562,049	0.22	1	562,286	0.18	-	562,665	0.11	2
	[150,250]	0.84	648,362	646,240	0.33	-	646,187	0.33	1	646,875	0.23	2
10-52	[10,100]	1.06	1,142,132	1,132,292	0.87	-	1,134,630	0.65	-	1,136,369	0.51	-
	[50,150]	0.98	1,878,849	1,872,771	0.32	-	1,873,833	0.27	-	1,877,128	0.09	1
	[100,200]	0.98	2,470,661	2,466,989	0.15	-	2,469,474	0.05	3	2,470,966	-0.01	7
	[150,250]	0.77	2,799,250	2,796,396	0.10	1	2,798,049	0.04	3	2,798,808	0.02	5
20-12	[10,100]	1.08	542,549	537,346	0.95	-	539,210	0.61	-	539,754	0.51	-
	[50,150]	1.00	881,231	875,963	0.59	-	877,176	0.46	-	878,836	0.27	-
	[100,200]	1.00	1,152,475	1,149,701	0.24	-	1,150,268	0.19	-	1,150,967	0.13	-
	[150,250]	0.79	1,305,170	1,301,587	0.28	-	1,302,134	0.24	1	1,302,963	0.17	3
20-52	[10,100]	1.08	2,303,214	2,285,508	0.77	-	2,286,716	0.71	-	2,292,545	0.46	-
	[50,150]	1.00	3,791,838	3,776,768	0.40	-	3,780,310	0.31	-	3,787,192	0.12	1
	[100,200]	1.00	4,993,056	4,983,599	0.19	-	4,987,746	0.11	-	4,992,368	0.01	3
	[150,250]	0.79	5,671,140	5,661,338	0.17	-	5,666,816	0.08	-	5,669,574	0.03	3

A - Number of problems where the heuristic procedure provides a better solution than CPLEX.

Table 2: The Quality of the Solution: Procedure 1.

			CPLEX	Procedure 1	
$\lvert P\rvert$-$\lvert\Delta\rvert$	C_i	β	Obj	Obj	RE%
5-12	[10,100]	1.02	132,803	123,747	6.84
	[50,150]	0.95	216,831	208,976	3.65
	[100,200]	0.86	283,494	278,342	1.82
	[150,250]	0.76	320,602	315,969	1.45
5-52	[10,100]	1.03	543,397	498,503	8.27
	[50,150]	0.96	910,778	873,749	4.08
	[100,200]	0.88	1,209,347	1,183,319	2.16
	[150,250]	0.78	1,378,794	1,357,718	1.53
10-12	[10,100]	1.13	246,410	226,538	8.11
	[50,150]	0.98	420,000	403,360	3.96
	[100,200]	0.98	563,297	553,278	1.78
	[150,250]	0.84	648,362	641,389	1.08
10-52	[10,100]	1.06	1,142,132	1,059,695	7.24
	[50,150]	0.98	1,878,849	1,811,766	3.58
	[100,200]	0.98	2,470,661	2,425,802	1.82
	[150,250]	0.77	2,799,250	2,765,560	1.20
20-12	[10,100]	1.08	542,549	504,771	6.97
	[50,150]	1.00	881,231	850,308	3.51
	[100,200]	1.00	1,152,475	1,133,555	1.64
	[150,250]	0.79	1,305,170	1,292,068	1.01
20-52	[10,100]	1.08	2,303,214	2,139,837	7.11
	[50,150]	1.00	3,791,838	3,658,202	3.53
	[100,200]	1.00	4,993,056	4,900,879	1.85
	[150,250]	0.79	5,671,140	5,601,932	1.22

Table 3: The CPU time of the procedures.

		CPLEX	Procedure 2									Procedure 1	
			$\alpha = 1/2$			$\alpha = 2/3$			$\alpha = 9/10$				
$\|P\|$-$\|\Delta\|$	C_i	CPU	CPU	A	B	CPU	A	B	CPU	A	B	CPU	A
5-12	[10,100]	335	0.52	641	4.7	0.78	432	6.7	1.78	188	21.1	0.19	1,811
	[50,150]	1,113	0.71	1,570	5.1	0.83	1,341	7.4	2.29	486	25.2	0.20	5,621
	[100,200]	1,580	0.60	2,656	5.3	0.84	1,886	7.6	2.55	617	25.1	0.26	6,148
	[150,250]	559	0.74	752	5.5	0.96	583	9.2	3.10	180	36.7	0.21	2,651
5-52	[10,100]	821	10.96	75	5.0	14.98	55	7.3	40.97	20	24.1	3.77	218
	[50,150]	348	12.73	27	5.5	18.42	19	8.3	52.06	7	27.4	4.82	72
	[100,200]	391	15.86	25	5.8	19.06	20	8.9	56.01	7	29.7	5.79	67
	[150,250]	596	12.71	47	6.5	28.51	21	19.7	60.43	10	39.7	3.23	185
10-12	[10,100]	820	1.19	688	5.3	1.64	500	8.0	4.27	192	25.1	0.39	2,096
	[50,150]	748	1.24	602	5.8	1.87	401	8.8	5.48	137	28.3	0.42	1,802
	[100,200]	802	1.35	594	5.8	2.05	392	8.5	5.77	139	26.6	0.44	1,814
	[150,250]	660	1.47	449	6.8	2.16	306	11.0	5.30	124	25.7	0.43	1,542
10-52	[10,100]	376	24.54	15	5.9	35.42	11	8.7	103.82	4	28.6	8.30	45
	[50,150]	324	25.69	13	6.0	39.51	8	9.4	115.94	3	30.8	10.19	32
	[100,200]	408	27.19	15	5.8	39.98	10	8.8	118.08	3	29.3	11.89	34
	[150,250]	327	26.34	12	6.0	37.67	9	8.8	107.77	3	28.7	7.16	46
20-12	[10,100]	857	2.51	341	6.0	3.69	232	8.8	10.82	79	28.8	0.85	1,005
	[50,150]	657	2.72	242	6.0	4.21	156	9.4	12.72	52	30.6	0.92	717
	[100,200]	633	3.07	206	6.0	4.00	158	8.6	12.05	53	29.0	0.93	684
	[150,250]	675	2.75	246	5.7	4.04	167	8.5	10.72	63	26.7	0.77	873
20-52	[10,100]	890	52.02	17	6.0	76.74	12	9.2	237.57	4	30.9	17.38	51
	[50,150]	750	55.27	14	6.1	86.82	9	9.7	256.99	3	31.4	28.64	26
	[100,200]	756	58.04	13	6.3	84.78	9	9.0	248.12	3	29.3	33.31	23
	[150,250]	785	57.70	14	5.7	77.98	10	8.7	236.52	3	28.2	14.11	56

A= CPU_{CPLEX}/CPU_{Proc}

B - average number of iteration in Procedure 2.

References

[1] Barr R, Glover F, Klingman D. A New Optimization Method for Large Scale Fixed Charge Transportation Problems. Operations Research 1981;29:448-463.

[2] Bazaraa M, Sherali H, Shetty C. Nonlinear Programming: Theory and Algorithms, 2-nd Edition, New York: Wiley, 1993.

[3] Beale E, Tomlin J. Special Facilities in a General Mathematical Programming System for Nonconvex Problems Using Order Sets of Variables, in: J. Lawrence ed., Proceedings of the Fifth International Conference on Operational Research, London, 1970.

[4] Beale E, Forrest J. Global Optimization Using Special Order Sets. Mathematical Programming 1973;10:52-69.

[5] Bertsekas D, Lauer G, Sandell N, Posbergh T. Optimal Short-Term Scheduling of Large-Scale Power Systems. IEEE Transactions on Automatic Control 1983;28:1-11.

[6] Cabot A, Erenguc S. Some Branch-and-Bound Procedures for Fixed-Cost Transportation Problems. Naval Researcg Logistics Quarterly 1984;31:145-154.

[7] Cooper L, Drebes C. An Approximate Solution Method for the Fixed Charge Problem. Naval Research Logistics Quarterly 1967;14:101-113.

[8] Diaby M. Successive Linear Approximation Procedure for Generalized Fixed-Charge Transportation Problem. Journal of the Operations Research Society 1991;42:991-1001.

[9] Florian M, Klein M. Deterministic Production Planning with Concave Costs and Capacity Constraints. Management Science 1971;18:12-20.

[10] Geunes J, Romeijn E, Taaffe K. Requirements Planning with Pricing and Order Selection Flexibility. Operations Research 2006;54:394-401.

[11] Gilbert S. Coordination of Pricing and Multi-Period Production for Constant Priced Goods. European Journal of Operational Research 1999;114:330-337.

[12] Gray P. Exact Solution for the Fixed-Charge Transportation Problem. Operations Research 1971;19:1529-1538.

[13] van Hoesel C, Wagelmans A. Fully Polynomial Approximation Schemes for Single-Item Capacitated Economic Lot-Sizing Problems. Mathematics of Operations Research 2001;26:339-357.

[14] van Hoesel C, Wagelmans A. An $O(T^3)$ Algorithm for the Economic Lot-Sizing Problem with Constant Capacities. Management Science 1996;42:142-150.

[15] Horst R, Pardalos P, Thoai N. Introduction to global optimization, 2-nd Edition, Boston: Springer, 2000.

[16] Horst R, Tuy H. Global Optimization, 3-rd Edition, New York: Springer, 1996.

[17] Kennington J, Unger V. A New Branch-and-Bound Algorithm for the Fixed Charge Transportation Problem. Management Science 1976;22:1116-1126.

[18] Khang D, Fujiwara O. Approximate Solution of Capacitated Fixed-Charge Minimum Cost Network Flow Problems. Networks 1991;21:689-704.

[19] Kim D, Pardalos P. A Solution Approach to the Fixed Charged Network Flow Problems Using a Dynamic Slope Scaling Procedure. Operations Research Letters 1999;24:195-203.

[20] Kim D, Pardalos P. Dynamic Slope Scaling and Trust Interval Techniques for Solving Concave Piecewise Linear Network Flow Problems. Networks 2000;35:216-222.

[21] Konno H. A Cutting Plane Algorithm for Solving Bilinear Programs. Mathematical Programming 1976;11:14-27.

[22] Kuhn H, Baumol W. An Approximate Algorithm for the Fixed Charge Transportation Problem. Naval Research Logistics Quarterly 1962;9:1-15.

[23] Loparic M, Marchand H, Wolsey L. Dynamic Knapsack Sets and Capacitated Lot-Sizing. Mathematical Programming 2003;B95:53-69.

[24] Loparic M, Pochet Y, Wolsey L, The Uncapacitated Lot-Sizing Problem with Sales and Safety Stocks. Mathematical Programming 2001;A89:487-504.

[25] Marchand H, Martin A, Weismantel R, Wolsey L. Cutting Planes in Integer and Mixed Integer Programming. Discrete Applied Mathematics 2002;123:397-446.

[26] Miller A, Wolsey L. Tight Formulations for Some Simple Mixed Integer Programs and Convex Objective Integer Programs. Mathematical Programming 2003;B98:73-88.

[27] Nahapetyan A, Pardalos P. A Bilinear Relaxation Based Algorithm for Concave Piecewise Linear Network Flow Problems. To appear in the Journal of Industrial and Management Optimization.

[28] Nahapetyan A, Pardalos P. Adaptive Dynamic Cost Updating Procedure for Solving Fixed Charge Network Flow Problems. To appear in the Computational Optimization and Applications.

[29] Palekar U, Karwan M, Zionts S. A Branch-and-Bound Method for Fixed Charge Transportation Problem. Management Science 1990;36:1092-1105.

[30] Pochet Y, Wosley L. Algorithms and Reformulations for Lot-Sizing Problems, Combinatorial Optimization, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol 20, Providence, 1995.

[31] Sherali H, Shetty C. A Finitely convergent Algorithm for Bilinear Programming Problems Using Polar Cuts and Disjunctive Face Cuts, Mathematical Programming 1980;19:14-31.

[32] Thomas J. Price-Production Decisions with Deterministic Demand. Management Science 1970;16:747-750.

[33] Wolsey L. Solving Multi-Item Lot-Sizing Problems with an MIP Solver Using Classification and Reformulation. Management Science 2002;48:1587-1602.

Figure Captions

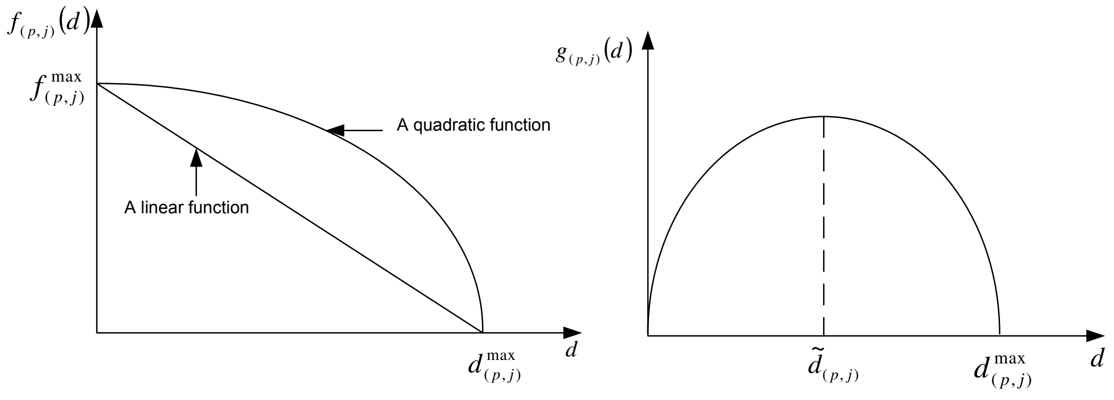

Figure 1: The price and the revenue functions.