Alcom-FT Technical Report Series
ALCOMFT-TR-02-78

A purchase lot sizing problem with quantity discounts

K.M.J. De Bontridder [1]
Universiteit Utrecht, Institute of Information and Computing Sciences,
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
koendb@cs.uu.nl

Abstract

In this paper we present a local search algorithm for a purchase lot sizing problem, which is an extended version of the classical lot sizing problem. The problem is to find a purchase plan for several items in such a way that the sum of the purchase costs and the inventory costs is minimized. We assume that we know how much of each item we need and the time at which we need it; we must decide in which period and from which supplier we buy each purchased item, where each supplier has its own prices and discounts. We present two mixed integer programming formulations and a local search algorithm. The local search algorithm consist of two stages. In the first stage we determine an initial solution on the basis of the linear programming relaxation of one of the mixed integer programming formulations. In the second stage we try to improve this solution by applying local search techniques. To determine the value of the neighbors we solve generalized nonlinear knapsack problems. We tested our approaches on randomly generated test instances. Our local search approach gives good results in a small amount of time.
Keywords: Metaheuristics, Purchasing, Integer programming

1 Introduction

Purchase lot sizing problems play an important role when suppliers specify price discounts based on the quantity ordered. In such an environment, a business can save money by grouping several orders into one large order. In order not to delay the production plan, the only option is to advance orders. However, such decisions increase the inventory of the nonrenewable. Therefore, we must find the optimal trade-off between the purchase costs and the inventory costs.

The purchase lot sizing problem is an extended version of the lot sizing problem; see Aggarwal and Park [1993] and Bahl et al. [1987]. The lot sizing problem is defined as follows. Given are a *finite time horizon*, which is divided into *periods*, and the demand for the nonrenewable in all periods. The problem is to determine

[1] Partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT). A part of the research was carried out at the Technische Universiteit Eindhoven with support of Baan.

for all periods the quantity (the lot size) to be produced, such that the costs of the production plan are minimized and the constraints are satisfied. In most of the articles that have appeared in the literature the costs consist of production costs and inventory costs. In these models the production costs in a period are not affected by the produced quantities in other periods.

In our purchase lot sizing problem we consider several nonrenewables, which are delivered by external suppliers. Note that these nonrenewables only can be purchased at given moments in time. Such a moment in time marks the beginning of a new period. So a nonrenewable purchased in a period is received at the beginning of that period. We assume that the length of the periods is a given constant and that the number of periods is finite. Our goal is to find a purchase plan in such a way that the sum of the purchase costs and the inventory costs is minimized. For each nonrenewable the demand in each period and a number of alternative suppliers are given. Each supplier has its own prices, discount functions, and other restrictions. The purchased quantity of a nonrenewable in a period from a supplier must be either zero, or between a given lower and upper bound. The true cost function for buying a quantity of a nonrenewable in a time period from a supplier is piecewise linear, strictly increasing, and concave. In addition the buying firm receives discounts based on the total quantity bought from one supplier over the entire time horizon. Discounts are always based on the purchased quantity: we assume that the discount function is piecewise linear, convex, and nondecreasing. The *purchase costs* are the true costs minus the discounts. In our model the *inventory costs* are proportional to the inventory size. Notice that in our model the purchase costs of a nonrenewable in a period are affected by the purchased quantities in other periods and the purchased quantities of other nonrenewables.

Numerous articles have been published on techniques for the management and control of inventory with quantity discounts. All of these articles use enhancements of the classical economic order quantity (EOQ) concept; for an example see Rubin and Benton [1993]. In these articles a standard assumption is a constant demand rate. The problem is to find a fixed order size and a supplier for each nonrenewable such that the total costs are minimized.

In our problem we do not assume a constant demand rate; we only assume that we know the amount of each nonrenewable we need and the time at which we need it. Therefore, we must determine for each nonrenewable the quantity to be bought from each supplier in each period.

Small instances can be solved using a mixed integer linear programming approach for which we give two different formulations. Another option is using a local search approach. Local search is based on the idea that a given solution may be improved by making small changes. Such a small change is called a move. The set of solutions that can be reached from a given solution by making one move is called its neighborhood. Roughly speaking a local search algorithm starts with an initial solution and then continually tries to find better solutions by selecting in each iteration a neighbor of the current solution. We refer to Aarts and Lenstra [1997] for an overview of local search. Our local search approach consists of two stages. In the first stage we determine an initial solution by using the solution of the linear programming relaxation of one of the mixed integer linear programming formulations. In the second stage we try to improve this solution by applying local search techniques. To determine the value of the neighbors we solve generalized nonlinear knapsack problems.

This paper is organized as follows. In Section 2 we give two mixed integer linear programming formulations for the problem. In Section 3 we present our neighborhood function. Our local search algorithm is described in Section 4. In Section 5 we give extensive computational results: we compare the results of our local search algorithm with the solutions of one of the mixed integer linear programming formulations. We conclude with some final remarks.

2 Two mixed integer linear programming formulations

The mixed integer linear programming formulations are used for two purposes. In our computational experiments we compare the results of our local search algorithm with the solutions of two mixed integer linear programming formulations. The solution of their linear programming relaxations are also used to construct initial solutions for our local search algorithm. The first formulation is rather elementary. In the second formulation we introduce a new set of decision variables, which allow us to define some useful valid inequalities. The number of variables in the second formulation is much larger than the number of variables in the first formulation.

2.1 Model

Let $\mathcal{T}$ denote the set of time periods and $\mathcal{I}$ the set of nonrenewables. For each nonrenewable $i \in \mathcal{I}$ we are given the demand $\overline{d}_i^t$ per time period $t \in \mathcal{T}$ and a set of suppliers $\mathcal{S}_i$.

We use two types of decision variables in our model, namely Y_{si}^t, which denotes the quantity of nonrenewable i purchased in period t from supplier s, and U_i^t, which denotes the inventory of nonrenewable i at the beginning of period t. For each quantity Y_{si}^t we are given a nonnegative lower bound $\underline{b}_{si}^t$, an upper bound $\overline{b}_{si}^t$, and a cost function α_{si}^t. The function α_{si}^t is piecewise linear, concave, and strictly increasing, and can be defined as the minimum of a number of linear functions;

each linear function can be seen as one *alternative*. If $\mathcal{G}^t_{si}$ denotes the set of possible alternatives, then we have

$$\alpha^t_{si}(Y^t_{si}) = \begin{cases} \min\limits_{j \in \mathcal{G}^t_{si}} \{\widetilde{g}^{tj}_{si} Y^t_{si} + \overline{g}^{tj}_{si}\} & \text{if } Y^t_{si} > 0, \\ 0 & \text{if } Y^t_{si} = 0, \end{cases}$$

where $\widetilde{g}^{tj}_{si}$ and $\overline{g}^{tj}_{si}$ represent the unit and the fixed purchase costs for alternative j, respectively. We request that $\min_{j \in \mathcal{G}^t_{si}} \{\overline{g}^{tj}_{si}\} \geq 0$.

Each supplier s gives a discount based on a weighted sum of the purchased quantity

$$V_s = \sum_{i:s \in \mathcal{S}_i, t \in \mathcal{T}} \widetilde{w}^t_{si} Y^t_{si},$$

where $\widetilde{w}^t_{si}$ is the given weight for triple (s, i, t). The discount function β_s is piecewise linear, convex, and nondecreasing. A set of alternatives $\mathcal{K}_s$ is used to define the discount function given by

$$\beta_s(V_s) = \max_{a \in \mathcal{K}_s} \{\widetilde{k}^a_s V_s + \overline{k}^a_s\},$$

where $\widetilde{k}^a_s$ and $\overline{k}^a_s$ represent the unit and the fixed discounts for alternative a, respectively. We assume that $\max_{a \in \mathcal{K}_s} \{\overline{k}^a_s\} = 0$, and that $\widetilde{g}^{tj}_{si} - \widetilde{w}^t_{si} \widetilde{k}^a_s > 0$ for each possible combination. Furthermore, we assume that we can discard purchased nonrenewables without extra costs. As a consequence we have that increasing the purchased quantity of a nonrenewable never decreases the purchase costs.

In our model inventory costs are proportional to the inventory size: h_i denotes the cost of holding one unit of inventory of nonrenewable i for one time period. This results in the following model.

$$\begin{array}{lll}
\min & \sum\limits_{t \in \mathcal{T}} \sum\limits_{i \in \mathcal{I}} \sum\limits_{s \in \mathcal{S}_i} \alpha^t_{si}(Y^t_{si}) - \sum\limits_{s \in \mathcal{S}} \beta_s(V_s) \;\; + \;\; \sum\limits_{t \in \mathcal{T}} \sum\limits_{i \in \mathcal{I}} h_i U^t_i & \\
\text{s.t.} & U^t_i + \sum\limits_{s \in \mathcal{S}_i} Y^t_{si} \geq U^{t+1}_i + \overline{d}^t_i & \forall i, t \\
& V_s = \sum\limits_{i:s \in \mathcal{S}_i} \sum\limits_{t \in \mathcal{T}} \widetilde{w}^t_{si} Y^t_{si} & \forall s \\
& Y^t_{si} = 0 \quad \vee \quad Y^t_{si} \geq \underline{b}^t_{si} & \forall s, i, t \\
& Y^t_{si} \leq \overline{b}^t_{si} & \forall s, i, t \\
& U^t_i \geq 0 & \forall i, t \\
& U^0_i = 0 & \forall i
\end{array}$$

The first class of constraints are the traditional lot sizing constraints. There is an inequality sign, since we can discard nonrenewables without extra costs. The third type of constraint indicates that the purchased quantity must be either zero or larger than a given lower bound.

2.2 First formulation

We model the concave functions α_{si}^t by introducing for each alternative $j \in \mathcal{G}_{si}^t$ a binary variable Z_{si}^{tj}, which is equal to 1 if alternative j is chosen, and 0 otherwise. The variable Y_{si}^{tj} is used to indicate the amount of nonrenewable i bought from supplier s in alternative j in period t. Alternatively, j can be chosen only if Y_{si}^{tj} belongs to the domain $[\underline{b}_{si}^{tj}, \overline{b}_{si}^{tj}]$. If $\overline{b}_{si}^{tj}$ is infinite, then we replace $\overline{b}_{si}^{tj}$ by some large integer. Now we add the following constraints.

$$
\begin{aligned}
&\underline{b}_{si}^{tj} Z_{si}^{tj} \le Y_{si}^{tj} \le \overline{b}_{si}^{tj} Z_{si}^{tj} && \forall s, i, t, j \\
&\sum_{j \in \mathcal{G}_{si}^t} Z_{si}^{tj} \le 1 && \forall s, i, t \\
&Z_{si}^{tj} \in \{0, 1\} && \forall s, i, t, j \\
&Y_{si}^{tj} \ge 0 && \forall s, i, t, j
\end{aligned}
$$

The function α_{si}^t is now written as

$$\alpha_{si}^t(Y_{si}^{tj}, Z_{si}^{tj}) = \sum_{s \in \mathcal{S}_i} \sum_{j \in \mathcal{G}_{si}^t} (\widetilde{g}_{si}^{tj} Y_{si}^{tj} + \overline{g}_{si}^{tj} Z_{si}^{tj}).$$

To model the functions β_s we use the same techniques. Recall that the total purchased quantity is equal to

$$V_s = \sum_{i: s \in \mathcal{S}_i} \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{G}_{si}^t} \widetilde{w}_{si}^t Y_{si}^{tj}.$$

The decision variable V_s^a is equal to V_s if alternative a is chosen, and zero otherwise. The binary variable W_s^a is equal to one if $V_s^a > 0$, and zero otherwise. Now we add the following constraints.

$$
\begin{aligned}
&\underline{b}_s^a W_s^a \le V_s^a \le \overline{b}_s^a W_s^a && \forall s, a \\
&\sum_{a \in \mathcal{K}_s} W_s^a \le 1 && \forall s \\
&W_s^a \in \{0, 1\} && \forall s, a \\
&V_s^a \ge 0 && \forall s, a
\end{aligned}
$$

The function β_s can now be written as

$$\beta_s(V_s^a, W_s^a) = \sum_{a \in \mathcal{K}_s} (\widetilde{k}_s^a V_s^a + \overline{k}_s^a W_s^a).$$

Hence, we get the following formulation.

$$\begin{array}{lll}
\min & \displaystyle\sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}}\sum_{s\in\mathcal{S}_i}\sum_{j\in\mathcal{G}_{si}^t}(\widetilde{g}_{si}^{tj}Y_{si}^{tj}+\overline{g}_{si}^{tj}Z_{si}^{tj}) - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{K}_s}(\widetilde{k}_s^a V_s^a + \overline{k}_s^a W_s^a) + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} h_i U_i^t & \\
\text{s.t.} & U_i^t + \displaystyle\sum_{s\in\mathcal{S}_i}\sum_{j\in\mathcal{G}_{si}^t} Y_{si}^{tj} \geq U_i^{t+1} + \overline{d}_i^t & \forall i, t \\
& \underline{b}_{si}^{tj} Z_{si}^{tj} \leq Y_{si}^{tj} \leq \overline{b}_{si}^{tj} Z_{si}^{tj} & \forall s, i, t, j \\
& \displaystyle\sum_{j\in\mathcal{G}_{si}^t} Z_{si}^{tj} \leq 1 & \forall s, i, t \\
& Z_{si}^{tj} \in \{0, 1\} & \forall s, i, t, j \\
& Y_{si}^{tj} \geq 0 & \forall s, i, t, j \\
& \displaystyle\sum_{a\in\mathcal{K}_s} V_s^a = \sum_{i:s\in\mathcal{S}_i}\sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{G}_{si}^t} \widetilde{w}_{si}^t Y_{si}^{tj} & \forall s \\
& \underline{b}_s^a W_s^a \leq V_s^a \leq \overline{b}_s^a W_s^a & \forall s, a \\
& \displaystyle\sum_{a\in\mathcal{K}_s} W_s^a \leq 1 & \forall s \\
& W_s^a \in \{0, 1\} & \forall s, a \\
& V_s^a \geq 0 & \forall s, a \\
& U_i^t \geq 0 & \forall i, t \\
& U_i^0 = 0 & \forall i
\end{array}$$

We refer to this mixed integer linear programming formulation and its relaxation by MIP1 and LP1, respectively.

2.3 Second formulation

The first formulation is rather elementary. We now introduce a new set of decision variables, which allow us to define some useful valid inequalities. To obtain these inequalities we use techniques of Aghezzaf and Wolsey [1994]. Instead of using the variable Y_{si}^{tj} we use variables of the form $Y_{si}^{tt'j}$, which indicate the amount of nonrenewable i bought from supplier s in alternative j in period t to satisfy the demand in period t'. We get the new formulation by replacing Y_{si}^{tj} by

$$\sum_{t' \in \mathcal{T}: t' \geq t} Y_{si}^{tt'j}.$$

As a consequence of $\widetilde{g}_{si}^{tj} - \widetilde{w}_{si}^{t} \widetilde{k}_{s}^{a} > 0$, $\max\{\underline{b}_{si}^{t}, \overline{d}_{i}^{t'}\}$ is a valid upper bound for $Y_{si}^{tt'j}$. Therefore, the following constraints are valid.

$$Y_{si}^{tt'j} \leq \max\{\underline{b}_{si}^{t}, \overline{d}_{i}^{t'}\} Z_{si}^{tj} \qquad \forall s, i, t, t', j.$$

By adding these constraints we get a stronger formulation, which can be solved for small instances by a mixed integer linear programming solver.

The new variables can also be used to define the quantity of inventory of nonrenewable i. Therefore, we replace $\sum_{t \in \mathcal{T}} U_i^t$ by

$$\sum_{s \in \mathcal{S}_i} \sum_{t \in \mathcal{T}} \sum_{t' \in \mathcal{T}: t' \geq t} \sum_{j \in \mathcal{G}_{si}^t} (t' - t + 1) Y_{si}^{tt'j},$$

and the first constraint by

$$\sum_{t \in \mathcal{T}: t \leq t'} \sum_{s \in \mathcal{S}_i} \sum_{j \in \mathcal{G}_{si}^t} Y_{si}^{tt'j} \geq \overline{d}_i^{t'} \qquad \forall i, t'.$$

We refer to the resulting mixed integer linear programming formulation and its relaxation by MIP2 and LP2, respectively.

3 Neighborhood

In our local search algorithm we consider one nonrenewable at a time. For each solution we must determine the quantity to be bought from each supplier in each period. To obtain a neighbor we shift a certain amount of demand from one period to another period or we change the assignment of the purchased quantity to the various suppliers in one period. To determine the cost of a neighbor we need to solve a nonlinear knapsack problem, which we discuss in Subsection 3.1. In Subsection 3.2 we describe the resulting neighborhood function.

3.1 Nonlinear knapsack problem

To determine the cost of a neighbor we must assign the purchased quantities of a nonrenewable in a period to suppliers. We determine these assignments by solving knapsack problems with piecewise linear, concave, and strictly increasing cost functions. To solve these generalized knapsack problems we use an enumerative algorithm that is based on the existence of simply structured optimal solutions. A similar approach for a generalized knapsack problem without lower bounds is given by Haberl [1999]. Our generalized knapsack problem is formally described as follows

$$
\begin{array}{rlr}
\min & \sum_{s \in \mathcal{S}} \alpha_s(Y_s) & \\
\text{s.t.} & \sum_{s \in \mathcal{S}} Y_s \geq d & \\
& Y_s = 0 \quad \vee \quad Y_s \geq \underline{b}_s & \forall s \in \mathcal{S} \\
& Y_s \leq \overline{b}_s & \forall s \in \mathcal{S}
\end{array}
$$

For any $s \in \mathcal{S}$ the function α_s is nonnegative, piecewise linear, concave, and strictly increasing for $Y_s > 0$ with $\alpha_s(0) = 0$. By α_s^+ and α_s^- we denote the one-sided derivatives of α_s: α_s^+ and α_s^- give an indication of the additional costs of increasing Y_s by one unit and decreasing Y_s by one unit, respectively. Since α_s is concave we have $\alpha_s^+(Y_s) \geq \alpha_s^-(Y_s)$ for all $Y_s \geq 0$. For each solution δ we denote by $\mathcal{S}^\delta$ the set $\{s \mid Y_s^\delta > 0\}$. Before we describe our solution approach, we prove that there is a minimally optimal solution with a simple structure, where an optimal solution δ is minimally optimal if there does not exist an optimal solution δ' with $|\mathcal{S}^{\delta'}| < |\mathcal{S}^\delta|$.

Theorem 1 *There exists a minimally optimal solution δ in which there is at most one $j \in \mathcal{S}^\delta$ such that $\underline{b}_j < Y_j^\delta < \overline{b}_j$.*

Proof. Suppose to the contrary that we have for each minimally optimal solution δ at least two variables Y_i^δ and Y_j^δ such that

$$\underline{b}_i < Y_i^\delta < \overline{b}_i \quad \text{and} \quad \underline{b}_j < Y_j^\delta < \overline{b}_j.$$

Without loss of generality we may assume that $\alpha_i^-(Y_i^\delta) \geq \alpha_j^-(Y_j^\delta)$. Since α_j is a concave function we have $\alpha_i^-(Y_i^\delta) \geq \alpha_j^-(Y_j^\delta) \geq \alpha_j^+(Y_j^\delta)$. Therefore, we can increase Y_j^δ and decrease Y_i^δ by $\min\{\overline{b}_j - Y_j^\delta, Y_i^\delta - \underline{b}_i\}$ without increasing the costs. Notice that $|\mathcal{S}^\delta|$ does not change. By repeating this argument we prove the theorem. □

Theorem 2 *If δ is a minimally optimal solution, then for each variable Y_j^δ with $Y_j^\delta > 0$ and $\alpha_j^-(Y_j^\delta) = \max\{\alpha_s^-(Y_s^\delta) \mid s \in \mathcal{S}^\delta\}$ we have that*

$$\sum_{s \in \mathcal{S}^\delta \setminus \{j\}} \overline{b}_s < d.$$

Proof. Suppose to the contrary that there is a variable Y_j^δ for which $Y_j^\delta > 0$, $\alpha_j^-(Y_j^\delta) = \max\{\alpha_s^-(Y_s^\delta) | s \in \mathcal{S}^\delta\}$, and

$$\sum_{s \in \mathcal{S}^\delta \setminus \{j\}} \overline{b}_s \geq d.$$

Since $\alpha_j^-(Y_j^\delta) \geq \alpha_s^-(Y_s^\delta) \geq \alpha_s^+(Y_s^\delta)$ for each $s \in \mathcal{S}^\delta$, we can set Y_j^δ equal to zero and get a feasible solution with equal or lower costs by increasing Y_s^δ for some $s \in \mathcal{S}^\delta$. Since $|\mathcal{S}^\delta|$ has been decreased, we have a contradiction. □

We represent a solution by a permutation of the integer numbers 1 through $|\mathcal{S}|$; given a permutation π we generate a solution δ to our knapsack problem by the following three steps.

Step 1. For $i = 1, \ldots, |\mathcal{S}|$ we set $Y_{\pi(i)}^\delta \longleftarrow 0$.

Step 2. While $\sum_{s \in \mathcal{S}} Y_s^\delta < d$ and $j \leq |\mathcal{S}|$

$$\begin{aligned} Y_{\pi(j)}^\delta &\longleftarrow \min\{\overline{b}_{\pi(j)}, \max\{\underline{b}_{\pi(j)}, d - \sum_{s \in \mathcal{S}} Y_s^\delta\}\}; \\ j &\longleftarrow j + 1. \end{aligned}$$

Step 3. While $\sum_{s \in \mathcal{S}} Y_s^\delta > d$ and $j \geq 1$

$$\begin{aligned} Y_{\pi(j)}^\delta &\longleftarrow \max\{\underline{b}_{\pi(j)}, Y_{\pi(j)}^\delta - \sum_{s \in \mathcal{S}} Y_s^\delta + d\}; \\ j &\longleftarrow j - 1. \end{aligned}$$

Theorem 3 *There exists a permutation π that represents an optimal solution.*

Proof. Let δ be an optimal solution that satisfies the properties of Theorem 1. To prove the theorem we construct a permutation π that results in solution δ. We distinguish two cases. If there is no variable Y_i^δ with $0 < \underline{b}_i = Y_i^\delta < \overline{b}_i$, then permutation π starts with the indices s for which $Y_s^\delta = \overline{b}_s$ in an arbitrary order. These indices are followed by the index s for which $\underline{b}_s < Y_s^\delta < \overline{b}_s$. We complete permutation π with the other indices in an arbitrary order. By z we denote the last index in permutation π for which $Y_z^\delta > 0$. We have $Y_s^\delta = \overline{b}_s$ for each $s \in \mathcal{S}^\delta \setminus \{z\}$. Combined by the optimality of δ and $\alpha_z(Y_z^\delta) > 0$, we get

$$\sum_{s \in \mathcal{S}^\delta \setminus \{z\}} \overline{b}_s < d. \tag{1}$$

Now it is easy to see that permutation π leads to solution δ.

Otherwise, if there is a variable Y_i^δ with $0 < \underline{b}_i = Y_i^\delta < \overline{b}_i$, we start permutation π with the indices s for which $\underline{b}_s < Y_s^\delta = \overline{b}_s$ in an arbitrary order. These indices are followed by the index s for which $\underline{b}_s < Y_s^\delta < \overline{b}_s$. Thereafter, we continue with the indices of the variables for which $0 < \underline{b}_s = Y_s^\delta$ in order of non-decreasing $\alpha_s^-(Y_s^\delta)$ value. We complete permutation π with the other indices in an arbitrary order. By z we denote the last index in permutation π for which $Y_z^\delta > 0$, and by i we denote the index of an arbitrary variable for which $0 < \underline{b}_i = Y_i^\delta < \overline{b}_i$. As a consequence of the optimality of δ and $Y_i^\delta < \overline{b}_i$, we have $\alpha_s^-(Y_s^\delta) \leq \alpha_i^+(Y_i^\delta)$ for each $Y_s^\delta > \underline{b}_s$. Therefore, we have $\alpha_s^-(Y_s^\delta) \leq \alpha_i^+(Y_z^\delta) \leq \alpha_i^-(Y_i^\delta) \leq \alpha_z^-(Y_z^\delta)$ for each $Y_s^\delta > \underline{b}_s$, and thus $\alpha_z^-(Y_z^\delta) = \max\{\alpha_s^-(Y_s^\delta) | s \in \mathcal{S}^\delta\}$. By applying Theorem 2 we get (1). Now it follows that permutation π leads to solution δ. □

So we can find an optimal solution by evaluating all possible permutations of the numbers 1 through $|\mathcal{S}|$. Our generalized knapsack problem is *NP*-hard, but it can be solved by dynamic programming. From a theoretical point of view this seems much better then evaluating $|\mathcal{S}|!$ permutations, but for our instances the dynamic programming algorithm is very inefficient. Also the number of permutations evaluated by our algorithm is in practice much smaller than $|\mathcal{S}|!$.

3.2 Resulting neighborhood

As mentioned earlier we consider one nonrenewable at a time. Let δ' and i denote the current solution and the nonrenewable under consideration, respectively. By $D_i(t)$ we denote the quantity that in the current solution is purchased in period t to satisfy the demand restrictions. Hence, for each $t \in \mathcal{T}$ the following restriction must hold.

$$\sum_{t' \in \mathcal{T}: t' \leq t} (D_i(t') - \overline{d}_i^{t'}) \geq 0.$$

It is possible that too much of a nonrenewable is purchased in order to satisfy the lower bounds. We use $\widetilde{Y}_{si}^t$ to denote the quantity of nonrenewable i purchased in period t from supplier s that is used to satisfy the demand restrictions. Therefore, we have

$$\sum_{s \in \mathcal{S}_i} \widetilde{Y}_{si}^t = D_i(t). \tag{2}$$

In our local search algorithm we record the values $\widetilde{Y}_{si}^t$. Given these values we determine Y_{si}^t as follows. If $\widetilde{Y}_{si}^t = 0$, then the purchased quantity $Y_{si}^t = 0$. Otherwise the purchased quantity is calculated by

$$Y_{si}^t = \max\{\widetilde{Y}_{si}^t, \underline{b}_{si}^t\}.$$

For each pair (i, t) with $D_i(t) > 0$ we have three types of neighbors. We refer to each neighbor by $(t \Rightarrow_i t', \omega)$: each neighbor can be interpreted as moving ω units of nonrenewable i from period t to period t'. To characterize the neighbors we store the values $\Delta^-(i, t)$ and $\Delta^+(i, t')$, which we define later; the values $\Delta^-(i, t)$ and $\Delta^+(i, t')$ give an indication of the additional costs of decreasing $D_i(t)$ by one unit and increasing $D_i(t')$ by one unit, respectively. We use the values $\Delta^-(i, t)$ and $\Delta^+(i, t')$ in our local search algorithm.

The first type of neighbor is created by changing the solution in period t. In this case $D_i(t)$ remains the same for all $t \in \mathcal{T}$. The new solution for pair (i, t) is optimal given the current solution for the other pairs. If $\widehat{Y}^t_{si}$ denotes the value of Y^t_{si} in the current solution δ', then the new solution is found by solving the following problem.

$$\begin{array}{lll} \min & \sum_{s \in \mathcal{S}_i} \alpha^t_{si}(Y^t_{si}) - \beta_s(V_s - \widehat{Y}^t_{si} + Y^t_{si}) & \\ \text{s.t.} & \sum_{s \in \mathcal{S}_i} Y^t_{si} \geq D_i(t) & \\ & Y^t_{si} = 0 \quad \vee \quad Y^t_{si} \geq \underline{b}^t_{si} & \forall s \in \mathcal{S}_i \\ & Y^t_{si} \leq \overline{b}^t_{si} & \forall s \in \mathcal{S}_i \end{array}$$

This is a generalized knapsack problem as discussed in Subsection 3.1. Solving it yields an optimal set of Y^t_{si} values, but as a consequence of the lower bounds we can have an optimal solution with $\sum_{s \in \mathcal{S}_i} Y^t_{si} > D_i(t)$. We need values of $\widetilde{Y}^t_{si}$ such that equality (2) holds. We know that there exists an $s' \in \mathcal{S}_i$ with $Y^t_{s'i} \geq \sum_{s \in \mathcal{S}_i} Y^t_{si} - D_i(t)$. To satisfy equality (2) we put

$$\widetilde{Y}^t_{s'i} = Y^t_{s'i} - (\sum_{s \in \mathcal{S}_i} Y^t_{si} - D_i(t))$$

for a supplier s' for which $Y^t_{s'i} \geq \sum_{s \in \mathcal{S}_i} Y^t_{si} - D_i(t)$, and $\widetilde{Y}^t_{si} = Y^t_{si}$ for each $s \in \mathcal{S}_i \setminus \{s'\}$. We refer to this neighbor by $(t \Rightarrow_i t, 1)$. In this case $\Delta^-(i, t)$ and $\Delta^+(i, t)$ are equal to $\max\{\alpha^{t-}_{si}(\widetilde{Y}^t_{si}) - \beta^-_s(V_s) \mid \widetilde{Y}^t_{si} > 0\}$ and $\min\{\alpha^{t-}_{si}(\widetilde{Y}^t_{si}) - \beta^-_s(V_s) \mid \widetilde{Y}^t_{si} < \overline{b}^t_{si}\}$, respectively. If there is a supplier $s \in \mathcal{S}_i$ with $\widetilde{Y}^t_{si} < \underline{b}^t_{si}$, then $\Delta^+(i, t) = 0$.

The second type of neighbor is created by shifting a part of the demand to an earlier period t': a shift from t to t' is only considered if $D_i(t') < \sum_{s \in \mathcal{S}_i} \overline{b}^{t'}_{si}$ and at least one of the following two conditions holds.

- There is no t'' with $t' < t'' < t$ and $D_i(t'') > 0$.
- Period t' is the last period before t for which there exists a supplier s such that $0 < \widetilde{Y}^{t'}_{si} < \overline{b}^{t'}_{si}$.

We demand that the shifted quantity was purchased in period t from the supplier s for which $\widetilde{Y}_{si}^{t} > 0$ and $\Delta^{-}(i,t) = \alpha_{si}^{t-}(\widetilde{Y}_{si}^{t}) - \beta_{s}^{-}(V_s)$ is maximal. Therefore, $\widetilde{Y}_{si}^{t}$ is an upper bound for the shifted quantity. We refer to this supplier s by $\xi^{-}(i,t)$. A second restriction is that the shifted quantity must be purchased in period t' from one supplier. Only one supplier s' is considered. To choose the supplier we use the following selection order:

1. If there exists a supplier s with $0 < \widetilde{Y}_{si}^{t'} < \underline{b}_{si}^{t'}$, then we choose an arbitrary supplier s' with $0 < \widetilde{Y}_{s'i}^{t'} < \underline{b}_{s'i}^{t'}$. In this case $\underline{b}_{s'i}^{t'} - \widetilde{Y}_{s'i}^{t'}$ is an upper bound for the shifted quantity, and $\Delta^{+}(i,t')$ is equal to 0.
2. Otherwise if there exists a supplier s with $\underline{b}_{si}^{t'} \leq \widetilde{Y}_{si}^{t'} < \overline{b}_{si}^{t'}$, then we choose the supplier s' with $\underline{b}_{s'i}^{t'} \leq \widetilde{Y}_{s'i}^{t'} < \overline{b}_{s'i}^{t'}$ for which $\alpha_{s'i}^{t'+}(Y_{s'i}^{t'}) - \beta_{s'}^{+}(V_{s'})$ is minimal. In this case $\overline{b}_{s'i}^{t'} - \widetilde{Y}_{s'i}^{t'}$ is an upper bound for the shifted quantity, and $\Delta^{+}(i,t')$ is equal to $\alpha_{s'i}^{t'+}(Y_{s'i}^{t'}) - \beta_{s'}^{+}(V_{s'})$.
3. Otherwise, we choose the supplier s' with $0 = \widetilde{Y}_{s'i}^{t'} < \overline{b}_{s'i}^{t'}$ for which $\alpha_{s'i}^{t'+}(Y_{s'i}^{t'}) - \beta_{s'}^{+}(V_{s'})$ is minimal. In this case $\overline{b}_{s'i}^{t'} - \widetilde{Y}_{s'i}^{t'}$ is an upper bound for the shifted quantity, and $\Delta^{+}(i,t')$ is equal to $\alpha_{s'i}^{t'+}(Y_{s'i}^{t'}) - \beta_{s'}^{+}(V_{s'})$.

The shifted quantity ω is equal to the minimum of the two upper bounds. We refer to this neighbor by $(t \Rightarrow_i t', \omega)$.

The third type of neighbor is created by shifting a part of the purchased quantity to a later period t'. In this case we must take the inventories into consideration: it must be possible to perform the shift without creating negative stocks. Therefore, a shift from t to t' is only considered if $D_i(t') < \sum_{s \in \mathcal{S}_i} \overline{b}_{si}^{t'}$, $\min_{a \in \{t,\ldots,t'-1\}} \{ \sum_{j \in \mathcal{T}: j \leq a} (D_i(j) - \overline{d}_i^{j}) \} > 0$, and at least one of the following two conditions holds:

- there is no t'' such that $t < t'' < t'$ and $D_i(t'') > 0$;
- period t' is the first period after t for which $0 < \widetilde{Y}_{si}^{t'} < \overline{b}_{si}^{t'}$.

The second restriction indicates that it is possible to perform a shift without creating a negative stock if at most $\min_{a \in \{t,\ldots,t'-1\}} \{ \sum_{j \in \mathcal{T}: j \leq a} (D_i(j) - \overline{d}_i^{j}) \}$ is shifted from t to t'. For each neighbor the shifted quantity ω is determined in almost the same way as the shifted quantity for the second type of neighbor. The only difference is the third upper bound, which takes care of the danger of creating negative stocks. The shifted quantity ω is equal to the minimum of the three upper bounds. We refer to this neighbor by $(t \Rightarrow_i t', \omega)$.

If $t \neq t'$, then neighbor $(t \Rightarrow_i t', \omega)$ is created as follows. First we decrease

$D_i(t)$ and increase $D_i(t')$ with ω. In period t we remove this quantity from supplier $\xi^-(i,t)$. Next, we solve the nonlinear knapsack problem to determine the suppliers of the purchased quantity in period t'.

4 Local search algorithm

Our local search algorithm repeatedly starts independent search processes in which we consider one nonrenewable. Our local search algorithm consists of the following three steps.

Step 1. Construct an initial solution.

Step 2. Apply a search process for all of the nonrenewables in any order.

Step 3. If Step 2 has been performed a given maximum number of times, or if the best solution has not been improved in the previous run of Step 2, then stop; otherwise, go back to Step 2.

The construction of an initial solution is described in Section 4.1. The search process in which we consider nonrenewable i is a kind of *variable depth search*; see Kernighan and Lin [1970]. In our search strategy we start from the current best solution and generate a sequence of subsequent neighbors until some stopping criterion is satisfied. Next, a new starting point is chosen for the next sequence. In Subsection 4.2 we describe how we create a sequence given a starting solution. In Subsection 4.3 we describe how we choose the starting point for the next sequence. We stop considering nonrenewable i when a maximum number of iterations has been reached, or when all possible starting points have been used.

4.1 Finding an initial solution

To find an initial solution we solve one of the linear programming relaxations LP1 or LP2, which have been described in Section 2. In LP1 Y_{si}^{tj} indicates the amount of nonrenewable i bought from supplier s in alternative j in period t. If we use LP1, then we get as initial solution

$$\widetilde{Y}_{si}^{t} = \sum_{j \in \mathcal{G}_{si}^{t}} Y_{si}^{tj}$$

for each triple (s,i,t). The second alternative is to use the linear programming relaxation LP2. In LP2 $Y_{si}^{tt'j}$ indicates the amount of nonrenewable i bought from supplier s in alternative j in period t to satisfy the demand in period t'. In this case we get the initial solution

$$\widetilde{Y}_{si}^{t} = \sum_{t' \in \mathcal{T}: t' \geq t} \sum_{j \in \mathcal{G}_{si}^{t}} Y_{si}^{tt'j}$$

for each triple (s, i, t). For each pair (i, t) we obtain the value of $D_i(t)$ by

$$D_i(t) = \sum_{s \in \mathcal{S}_i} \widetilde{Y}_{si}^{t}.$$

As a consequence of $\widetilde{g}_{si}^{tj} - \widetilde{w}_{si}^{t} \widetilde{k}_{s}^{a} > 0$, we have for each nonrenewable i that

$$\sum_{t \in \mathcal{T}} D_i(t) = \sum_{t \in \mathcal{T}} \overline{d}_i^{t}.$$

Given this assignment the inventory costs are equal to

$$\sum_{i \in \mathcal{I}} \sum_{t \in \mathcal{T}} \sum_{t' \in \mathcal{T}: t' \leq t} h_i (D_i(t') - \overline{d}_i^{t'}).$$

The initial costs are equal to the sum of the inventory costs plus the purchase costs.

4.2 Generating a sequence

Given are the nonrenewable under consideration i, a start point δ, the costs of the start solution $\psi(\delta)$, a nonempty neighborhood $\mathcal{N}_i(\delta)$, and the costs of the current best solution ψ^*. In each iteration we choose a new neighbor. The costs $\psi(\delta)$ consist of two components, namely the purchase costs $\psi_p(\delta)$ and the inventory costs $\psi_i(\delta)$. If δ' is neighbor $(t \Rightarrow_i t', \omega)$, then the inventory costs are calculated by

$$\psi_i(\delta') = \psi_i(\delta) - h_i \omega (t' - t).$$

To guide the search process we use a variable I, which indicates the number of moves that have been made to obtain the current solution. By I^* we denote the value of I at the last improvement. The variables I and I^* are both set equal to zero when we start considering a new nonrenewable.

We use a *first improvement* search strategy to choose a neighbor. This means that we take the first neighbor δ' for which $\psi(\delta') < \psi(\delta)$. If none of the neighbors satisfies this criterion, then we accept the neighbor δ' for which $\psi(\delta) - \psi(\delta')$ is minimal with probability

$$e^{((I - I^*)(\psi(\delta) - \psi(\delta'))/A)}.$$

Here A is a scaling parameter. If this neighbor is not accepted, then the sequence is terminated and a new starting point is chosen.

With a first improvement search strategy the order of evaluation can affect the search process. Therefore, we evaluate the neighbors in order of nonincreasing priority: the priority of neighbor $(t \Rightarrow_i t', \omega)$ is given by $e^{((\Delta^-(i,t)-\Delta^+(i,t'))/\omega)}$ multiplied by a random number. The randomization is useful to prevent long-term cycling, and the term $e^{((\Delta^-(i,t)-\Delta^+(i,t))/\omega)}$ is used to favor the more promising neighbors. A first improvement strategy has the positive side effect that it reduces the number of neighbors that have to be evaluated.

To prevent short-term cycling we use a kind of *tabu search*. In tabu search we temporarily forbid certain shifts: such a forbidden shift is called *tabu*. We have implemented the tabu search algorithm as follows. We introduce two mappings $\tau_1 : \mathcal{T} \to \mathbb{Z}$ and $\tau_2 : \mathcal{T} \to \mathbb{Z}$. When we consider a new nonrenewable both $\tau_1(t)$ and $\tau_2(t)$ are set equal to -1 for all $t \in \mathcal{T}$. A neighbor $(t \Rightarrow_i t', \omega)$ is tabu if

- $t' = t$, and $\tau_2(t) \geq I^* - 1$, or if
- $t' \neq t$, $\tau_1(t') = \tau_2(t)$ and $\tau_2(t) \geq I^*$.

If neighbor $(t \Rightarrow_i t', \omega)$ is accepted and $t \neq t'$, then we set $\tau_1(t)$ and $\tau_2(t')$ both equal to I. If neighbor $(t \Rightarrow_i t, \omega)$ is accepted, then we set $\tau_2(t)$ equal to I. If the best solution ψ^* is improved, then I^* is set equal to $I + 1$. Finally we increase I by one.

It is easy to prove that the size of the neighborhood $\mathcal{N}_i(\delta)$ is at most $5|\mathcal{T}|$. For instances with a large number of periods the number of evaluated neighbors and thus the computation time increases. Therefore, we have implemented a version of our local search algorithm that restricts the growth of the number of evaluated neighbors. In this version we increase the number of neighbors that are tabu. Neighbors that only change the solution in period t are considered only if $D_i(t)$ has been decreased in one of the previous steps; other neighbors $(t \Rightarrow_i t', \omega)$ are only considered if $D_i(t)$ or $D_i(t')$ has been changed in one of the previous steps. This is implemented by making neighbor $(t \Rightarrow_i t', \omega)$ also tabu if

- $I > I^*$, $t' = t$, and $\tau_1(t) < I^*$, or if
- $I > I^*$, $t' \neq t$, $\tau_1(t) < I^*$, $\tau_2(t) < I^*$, $\tau_1(t') < I^*$, and $\tau_2(t') < I^*$.

In Section 5 we study the effect of this restriction for instances with 52 periods.

4.3 Choosing a new starting point

As mentioned above, we terminate a search sequence if none of the neighbors is accepted. To choose the starting point for a new sequence we use the following method. If we reach a local minimum that improves the overall best solution and that has at least two non-tabu neighbors, then we store that solution δ, the mappings

τ_1 and τ_2, and the iteration values I and I^* in a list $\mathcal{L}$.

As starting point for the new sequence we use the local minimum with the lowest objective value in $\mathcal{L}$, and use its solution δ, its mappings τ_1 and τ_2, and the iteration values I and I^* as a restarting point for the algorithm. In the first step after the restart we select the best non-tabu neighbor that was not selected in the first step of another sequence. If all neighbors of a local minimum have been used, then we remove this local minimum from $\mathcal{L}$. The list $\mathcal{L}$ is emptied when we start considering a new nonrenewable.

5 Computational results

We have tested our algorithm on a randomly generated test sample, which are available on request. We compared the results of our algorithm with the solutions of two mixed integer linear programming formulations. To solve the mixed integer linear programming programs we used CPLEX; see CPLEX [2000]. With CPLEX we also solved the linear programming relaxations which we used to obtain the initial solutions. We used the following parameter settings. Step 2 of our local search algorithm is performed at most 10 times, and we consider a nonrenewable i for at most 100 iterations. The scaling parameter A is set equal to 10. We performed 10 independent runs for each instance. The experiments were performed on a Sun Ultra 10 333 MHz workstation.

5.1 Test instances

Our test sample is randomly generated, where the random variables are always drawn from a uniform distribution. For each pair (i, t) the demand $\overline{d}_i^t$ is a random integer from $U[1, 100]$. Each nonrenewable i is characterized by a global price $\overline{p}_i$, which is a random integer from $U[1, 1000]$. For each nonrenewable i the unit inventory costs h_i are equal to $\lceil B\overline{p}_i \rceil$, where B is an input parameter. The probability that supplier s delivers nonrenewable i is equal to $\frac{4}{|\mathcal{S}|}$. If the number of suppliers of nonrenewable i is smaller than 3, additional suppliers are chosen. In our test instances the true cost function, the lower bound, and the upper bound are independent of the period. To generate the true and the discount cost functions we characterize each supplier s by a randomly drawn number from $U[0.9, 1.2]$. In case of a small value the prices of supplier s without incorporating any discount are low, but the amount of discount is rather small. In case of a high value supplier s has high prices and large discounts. The number of alternatives $|\mathcal{G}_{si}^t|$ and $|\mathcal{K}_s|$ are random integers from $U[1, 5]$ and $U[2, 5]$, respectively. The lower bound $\underline{b}_{si}^t$ is strictly greater than zero with probability 0.5, and the upper bound $\overline{b}_{si}^t$ is not in-

	CPLEX		algorithm LS1			algorithm LS2		
name	opt	time	best	average	time	best	average	time
bp01	2722868.7	106.3	0.03	0.08	2.7	0.00	0.00	7.6
bp02	2799213.3	356.1	0.21	0.22	2.6	0.19	0.20	8.0
bp03	3075404.8	356.1	0.02	0.05	2.6	0.02	0.04	7.2
bp04	1459908.7	74.0	0.00	0.02	3.2	0.00	0.00	6.6
bp05	2119740.7	145.2	0.24	0.24	2.5	0.20	0.20	8.6
bp06	2266288.8	263.2	0.43	0.43	2.2	0.38	0.38	7.1
bp07	2199606.1	155.7	0.07	0.10	2.6	0.00	0.00	6.2
bp08	2410382.1	99.3	0.02	0.06	2.5	0.03	0.03	5.5
bp09	2271796.1	83.1	0.02	0.18	2.0	0.08	0.08	5.9
bp10	3293095.1	89.7	0.00	0.02	2.0	0.01	0.01	6.5
average	2461830.4	172.9	0.11	0.14	2.5	0.09	0.09	6.9

Table 1: Results per instance.

	CPLEX		algorithm LS1			algorithm LS2		
name	opt	time	best	average	time	best	average	time
bp11	2857638.0	47.5	0.12	0.14	2.3	0.00	0.00	6.1
bp12	2857991.4	108.5	0.99	1.04	2.1	0.16	0.47	7.2
bp13	3255613.0	72.8	0.00	0.13	2.8	0.00	0.00	6.4
bp14	1486949.8	53.0	0.00	0.00	1.7	0.00	0.00	5.7
bp15	2183421.6	85.7	0.27	0.29	2.0	0.00	0.03	7.3
bp16	2346366.1	65.0	0.14	0.61	1.9	0.09	0.57	6.0
bp17	2253123.7	82.8	0.06	0.33	2.2	0.00	0.00	6.2
bp18	2455873.0	53.4	0.01	0.12	1.9	0.26	0.26	4.7
bp19	2384743.1	46.7	0.07	0.13	1.7	0.12	0.12	5.0
bp20	3528175.8	48.0	0.12	0.12	2.4	0.03	0.03	5.5
average	2560989.6	66.3	0.18	0.29	2.1	0.07	0.15	6.0

Table 2: Results per instance.

finite with the same probability. The fixed purchase costs are strictly greater than 0 with probability 0.5. We generated 40 instances bp01-bp40. The first 20 instances bp01-bp20 have 5 suppliers, 10 nonrenewables, and 12 periods; the other instances bp21-bp40 consist of 5 suppliers, 10 nonrenewables, and 52 periods. For the instances bp01-bp10 and bp21-bp30 the input parameter $B = 0.05$. For the other instances $B = 0.1$.

5.2 Test results

We tested the following three versions of our local search algorithm.

LS1*:* The version that solves LP1 to obtain the initial solution.

		algorithm LS1			algorithm LS3		
name	lb	best	average	time	best	average	time
bp21	9588494.2	27.66	27.79	43.8	27.51	27.69	9.9
bp22	9659587.6	24.06	24.11	43.9	24.04	24.12	10.6
bp23	10412502.4	28.70	28.77	40.0	28.62	28.82	9.6
bp24	4872783.5	16.21	16.23	36.7	16.21	16.22	9.7
bp25	7336676.5	20.43	20.46	39.4	20.38	20.45	10.6
bp26	7977911.6	28.85	28.96	41.1	28.78	28.91	9.7
bp27	7775529.7	18.93	19.04	41.8	18.97	19.03	9.4
bp28	8672549.8	21.04	21.09	31.3	21.04	21.07	8.2
bp29	6634407.4	29.62	29.73	36.6	29.59	29.73	9.8
bp30	10686751.1	31.13	31.32	40.4	31.12	31.36	9.1
average	8361719.4	24.66	24.75	39.5	24.63	24.74	9.7

Table 3: Results per instance.

LS2*:* The version that solves LP2 to obtain the initial solution.

LS3*:* The version that solves LP1 to obtain the initial solution, and reduces the number of evaluated neighbors as described at the end of Subsection 4.2.

In Table 1 and Table 2 we give the results for the instances with 5 suppliers, 10 nonrenewables, and 12 periods. For these instances the optimal solution was obtained by solving MIP2. In the second and the third column we give the optimal solution (opt) and the computation time in cpu seconds (time), respectively. We performed ten independent runs of the algorithms LS1 and LS2 on these instances, and compared the best and the average solutions of both algorithms with the optimal solutions. For both algorithms the percentage of deviation of the best solution and the average solution from the optimal value are given. In the columns ‘time’ for each instance the average computation time per run is given in cpu seconds. Both versions give good results in a small amount of time. The second version always needs only a few seconds. The differences between the computation times of versions LS1 and LS2 are caused by the computation of the initial solutions.

In Table 3 and Table 4 we give the results for the instances with 5 suppliers, 10 nonrenewables, and 52 periods. For these instances CPLEX was unable to find the optimal solution. These instances are so large that CPLEX even was unable to solve LP2, which renders algorithm LS2 impracticable. Therefore, we only did ten independent runs of the algorithms LS1 and LS3 on these instances. We compared the best and the average solution of both algorithms with the solution of LP1, which is a lower bound for the optimal solution. This lower bound is given in the second column. For both versions the percentage of deviation from the solution of LP1 of the best solution and the average solution are given. In the

		algorithm LS1			algorithm LS3		
name	lb	best	average	time	best	average	time
bp31	9588494.2	34.93	34.94	30.7	34.90	34.93	8.1
bp32	9659587.6	27.87	28.12	37.5	27.89	28.06	10.0
bp33	10412502.4	33.93	33.98	24.9	33.89	33.92	7.7
bp34	4872783.5	18.15	18.15	15.7	18.15	18.15	5.3
bp35	7336676.5	24.22	24.25	42.5	24.21	24.23	9.1
bp36	7977911.6	34.39	34.43	36.5	34.37	34.39	8.3
bp37	7775529.7	22.28	22.31	30.1	22.29	22.30	8.0
bp38	8672549.8	23.47	23.49	18.8	23.47	23.48	5.9
bp39	6634407.4	37.71	37.77	33.6	37.68	37.79	8.2
bp40	10686751.1	39.67	39.75	32.6	39.56	39.67	8.3
average	8361719.4	29.66	29.72	30.3	29.64	29.69	7.9

Table 4: Results per instance.

columns 'time' for each instance the average computation times per run is given in cpu seconds. It is difficult to judge the quality of the solutions. To get an indication for the quality of the lower bounds we compared the solutions of LP1 with the results of algorithm LS1 for the small instances. For bp01-bp10 and bp11-bp20 the percentage deviation is 26.61% and 31.25%, respectively. So the results seem to be good. Also notice that the quality of the solutions of algorithm LS1 and LS3 is almost the same. So, evaluating fewer neighbors does not affect the quality of the solution, but reduces the computation time significantly.

6 Conclusion

In this paper we presented a local search algorithm for a purchase lot sizing problem with quantity discounts. For instances with 5 suppliers, 10 nonrenewables, and 12 periods we compared our approach with the solutions of a mixed integer linear programming problem solved with a standard approach. We obtained good results for instances with 5 suppliers, 10 nonrenewables, and 52 periods in a few seconds.

We made the assumption that the demand per period was given over time. In practice these demands are chosen in such a way that external deliveries do not cause delays in the production process. But there also will be some flexibility, i.e. some production orders may be moved backward without serious problems. In De Bontridder [2001] we show how we can use this flexibility to save costs.

Our model can be extended to handle discrete order quantities. A discrete order quantity means that the order quantity should be a multiple of a given number, which can be different for each supplier. Other extensions like capacity constraints, i.e. a maximum or minimum order quantity per set of time periods, per supplier, or per time period, can also be handled by our approach.

Acknowledgments

We thank Han Hoogeveen, Emile Aarts, Jan Karel Lenstra, and Marc Sol for useful discussions and comments on earlier drafts of this paper.

References

E.H.L. AARTS AND J.K. LENSTRA (EDITORS) [1997]. *Local Search in Combinatorial Optimization*, Wiley, Chichester.

A. AGGARWAL AND J.K. PARK [1993]. Improved algorithms for economic lot size problems, *Operations Research* 41, 549–571.

E.H. AGHEZZAF AND L.A. WOLSEY [1994]. Modelling piecewise linear concave costs in a tree partitioning problem, *Discrete Applied Mathematics* 50, 101–109.

H.C. BAHL, L.P. RITZMAN, AND J.N.D. GUPTA [1987]. Determining lot sizes and resource requirement: a review, *Operations Research* 35, 329–345.

CPLEX [2000]. *CPLEX 6.6 Documentation Supplement*, ILOG CPLEX Division, Incline Village, Nevada.

K.M.J. DE BONTRIDDER [2001]. *Integrating purchase and production planning: using local search in supply chain optimization*, Ph.D. thesis, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven.

J. HABERL [1999]. Fixed-charge continuous knapsack problems and pseudo-greedy solutions, *Mathematical Programming* 85, 617–642.

B.W. KERNIGHAN AND S. LIN [1970]. An efficient heuristic procedure for partitioning graphs, *Bell System Technical Journal* 49, 291–307.

P.A. RUBIN AND W.C. BENTON [1993]. Jointly constrained order quantities with all-units discounts, *Naval Research Logistics* 40, 255–278.