Constraint-Based Local Search for Inventory Control Under Stochastic Demand and Lead Time

Constraint-based Local Search for Inventory Control under Stochastic Demand and Lead time

Roberto Rossi
Logistics, Decision and Information Sciences, Wageningen UR, the Netherlands, roberto.rossi@wur.nl
S. Armagan Tarim
Department of Management, Hacettepe University, Turkey, armtar@yahoo.com
Ramesh Bollapragada
Decision Sciences Department, College of Business, San Francisco State University, 1600 Holloway Avenue, San Francisco, California, USA, rameshb@sfsu.edu

In this paper, we address the general multi-period production/inventory problem with non-stationary stochastic demand and supplier lead time under service-level constraints. A replenishment cycle policy is modeled. We propose two hybrid algorithms that blend Constraint Programming and Local Search for computing near-optimal policy parameters. Both the algorithms rely on a coordinate descent Local Search strategy, what differs is the way this strategy interacts with the Constraint Programming solver. These two heuristics are, firstly, compared for small instances against an existing optimal solution method. Secondly, they are tested and compared with each other in terms of solution quality and run time on a set of larger instances that are intractable for the exact approach. Our numerical experiments show the effectiveness of our methods.

Key words: inventory control; demand uncertainty; supplier lead-time uncertainty; constraint-based local search; heuristics.
History: submitted in February 2010.

---

1. Introduction

Inventory theory provides methods for managing and controlling inventories under different constraints and environments. An interesting class of production/inventory control problems is the one that considers the single-location, single-product case under non-stationary stochastic demand and service level constraints. Such a problem has been widely studied because of its key role in practice [18, 35, 3].

Different inventory control policies can be adopted for the above mentioned problem. For a discussion of inventory control policies see [35]. One of the possible policies that can be

adopted is the replenishment cycle policy, (R, S). A detailed discussion on the characteristics of (R,S) can be found in [13]. In this policy an order is placed every R periods to raise the inventory level to the order-up-to-level S. This provides an effective means of dampening planning instability (deviations in planned orders, also known as *nervousness* [14, 21]) and coping with demand uncertainty. As pointed out by Silver et al. ([35], pp. 236–237), (R,S) is particularly appealing when items are ordered from the same supplier or require resource sharing. In these cases all items in a coordinated group can be given the same replenishment period. Periodic review also allows a reasonable prediction of the level of the workload on the staff involved, and is particularly suitable for advanced planning environments and risk management [37]. For these reasons, as stated by Silver et al. [35], (R, S) is a popular inventory policy.

Under the non-stationary demand assumption this policy takes the form (R^n,S^n), where R^n denotes the length of the nth replenishment cycle, and S^n the order-up-to-level value for the nth replenishment. It should be noted that this inventory control policy yields at most $2N$ policy parameters fixed at the beginning of an N-period planning horizon, therefore it is particularly easy to be implemented.

Due to its combinatorial nature, the computation of optimal (R^n,S^n) policy parameters, even in the absence of stochastic lead time, presents a difficult problem to solve to optimality. An early work in this area, by Bookbinder and Tan [10], proposes a two-step heuristic method. Tarim and Kingsman [40, 41] and Tempelmeier [44] propose a mathematical programming approach to compute policy parameters. Tarim and Smith [43] give a computationally efficient Constraint Programming formulation. An exact formulation of the policy and a solution method are presented in Rossi et al. [32].

All the above mentioned research assumes either zero or a fixed (deterministic) supplier lead time (i.e., replenishment lead time). However, the lead time uncertainty, in various industries an inherent part of the business environment, is having a detrimental effect on inventory systems. For this reason, there is a vast inventory control literature analyzing the impact of supplier lead time uncertainty on the ordering policy (Whybark and Williams [45], Speh and Wagenheim [36], Nevison and Burstein [25]). A comprehensive work on stochastic supplier lead time in continuous-time inventory systems is presented in Zipkin [46]. Kaplan [23] characterizes the optimal policy for a dynamic inventory problem, where the lead time is a discrete random variable with known distribution and the demands in successive periods are assumed to form a stationary stochastic process. Since tracking all the outstanding

orders through the use of Dynamic Programming requires a large multi-dimensional state vector, Kaplan assumes that orders do not cross in time and supplier lead time probabilities are independent of the size/number of outstanding orders (for details on order-crossover, see Hayya et al. [20]). The assumption that orders do not cross in time is valid for systems where supplier's production system has a single-server queue structure operating under a FIFO policy. Nevertheless, there are settings in which this assumption is not valid and orders do cross in time. This has been recently investigated in Hayya et al. [19], Riezebos [30], Bashyam and Fu [6]. In a recent work, Babaï et al. [4] analyze a dynamic re-order point control policy for a single-stage, single-item inventory system with non-stationary demand and lead time uncertainty. We argue that incorporating both a non-stationary stochastic demand and a stochastic supplier lead time in an optimization model that computes (R^n,S^n) policy parameters — without assuming that orders do not cross in time — is a relevant and novel contribution. To the best of our knowledge, the only existing work that addresses the computation of optimal (R^n,S^n) policy parameters under these assumptions is the one proposed in Rossi et al. [33]. Nevertheless, the approach proposed in [33] is only able to solve, in reasonable time, instances comprising a limited number of periods and a stochastic lead time that ranges over a small finite support.

In order to address this efficiency issue, in this paper we propose two heuristic techniques for computing (R^n,S^n) policy parameters under stochastic supplier lead time. We build on the work of Eppen and Martin [15], and by following a similar *scenario-based* approach (see also Birge and Louveaux [8]), we develop two *constraint-based local search* methods, based on a *coordinate descent* strategy, for finding near-optimal (R^n,S^n) policy parameters under non-stationary stochastic demand and supplier lead time (for a complete discussion on local search strategies in the literature refer to Focacci et al. [16], Nocedal and Wright [26]).

In the first part of this paper, we develop a technique that is analogous to a classical strategy of blending constraints with local search procedures (Backer et al. [5], Pesant and Gendreau [27]). In this approach, the local search engine is used to "guide" the search, while Constraint Programming is used to explore promising neighborhoods. In order to implement this strategy, we exploited Tarim and Smith's model [43] within a *coordinate descent* local search approach. In the second part of this work, we adopt an alternative strategy for integrating Constraint Programming and Local Search. In this strategy, Local Search techniques are introduced within a constructive global search algorithm (Cesta et al. [11]). In order to do so, we realized a deterministic equivalent modeling of the chance-

constraints (Charnes and Cooper [12]) enforcing the required service level, by employing a scenario based approach, and once more a *coordinate descent* heuristic for propagating these constraints. In this second strategy, *cost-based filtering* (Focacci et al. [17]) is employed to speed up the search. The results obtained with these two heuristic approaches are compared, for small instances, with the optimal solution produced by the approach presented in Rossi et al. [33].

Experimental results show that the approach proposed in Section 6 typically performs better than the one discussed in Section 5 in terms of solution quality. Nevertheless, the approach in Section 5 scales better and is faster than the approach in Section 6 in solving larger instances. Both the approaches run faster than the complete approach presented in Rossi et al. [33], which is able to solve only very small instances.

The paper is organized as follows. In Section 2, we introduce the problem and the assumptions adopted throughout the paper. In Section 3, we provide a Stochastic Programming formulation of the problem. In Section 4, we discuss a deterministic equivalent non-linear formulation of the Stochastic Programming model. In Section 5, we introduce a first heuristic solution method for the non-linear model discussed in Section 4. A second heuristic strategy is discussed in Section 6. Computational results are presented in Section 7. Summary and Conclusions are presented in Section 8.

2. Problem Definition

We discuss the general multi-period production/inventory control problem with non-stationary stochastic demand and lead time.

We consider a finite planning horizon of N periods and a demand d_t for each period $t \in \{1, \ldots, N\}$, which is a random variable with probability density function $g_t(d_t)$. The demand we consider is non-stationary, that is it can vary from period to period, and we also assume that demands in different periods are independent.

As in Eppen and Martin [15], an order placed in period $t \in \{1, ..., N\}$ is subject to a stochastic lead time l_t with probability mass function $f_t(\cdot)$. Note that $\{l_t\}$ are mutually independent and they are also independent of the respective order quantity. Since we consider a discrete stochastic lead time with probability mass function $f_t(\cdot)$ in each period $t = 1, \ldots, N$, this implies that an order placed in period t will be received exactly after k periods with probability $f_t(k)$. Since $f_t(k)$ is discrete, we assume that there is a maximum lead time L for

which $\sum_{k=0}^{L} f_t(k) = 1$, $i = 1, \ldots, N$. The probability of observing any lead time length $p > L$ will always be 0. Therefore, the possible lead time lengths are limited to $\Lambda = \{0, \ldots, L\}$, and the probability mass function is defined on the finite set Λ.

A fixed delivery cost a is incurred for each order at the time such an order is placed. A linear holding cost h is incurred for each unit of product carried in inventory from one period to the next, as well as those that are part of an outstanding order. This reflects the fact that we charge interests not only on the actual amount of items we have in stock, but also on outstanding orders. Doing so often makes sense, since companies may assess holding cost on their total invested capital and not simply on items in stock — this cost accounting strategy has been observed during our collaboration with Alcatel-Lucent manufacturing divisions. A further detailed justification for this cost accounting strategy can be found in Hunt [22].

Demands not met are assumed to be back-ordered, and satisfied as soon as the next replenishment order arrives. We assume that it is not possible to sell back excess items to the vendor at the end of a period and that negative orders are not allowed. If the actual inventory exceeds the order-up-to-level for that review, this excess stock is carried forward and not returned to the supply source. However, such occurrences are regarded as rare events and accordingly the cost of carrying this excess stock and its effect on the service levels of subsequent periods are ignored. This assumption is consistent with previous works in the literature (Bookbinder and Tan [10], Tarim and Kingsman [40], Tarim and Smith [43], Tempelmeier [44] and Tarim et al. [39]). Furthermore, a computational study in Rossi et al. [32] showed that such an assumption does not significantly impact the quality of the optimal solution obtained.

As a service level constraint, we require the probability that at the end of every period the net inventory will be non-negative to be at least a given value α. This value is assumed to be set by the management to a reasonably high threshold, therefore we will not consider values of α that are less than 0.5 — in real applications, α takes greater values, i.e. 0.95. Our aim is to minimize the expected total cost, which is composed of ordering costs and inventory holding costs, over the N-period planning horizon, satisfying the service level constraints.

The actual sequence of ordering and delivery is similar to the one described in Kaplan [23]. We adopt the same sequence of actions described in his paper, since it handles all the deliveries symmetrically, and allows for some delay in the arrival deliveries at the beginning of a period. The sequence is therefore as follows. At the beginning of a period, the inventory on-hand after the realization of demands from the previous periods is known. Since we are

assuming complete backlogging, this quantity may be negative. We also know the orders placed in previous periods, that have not been delivered yet. On the basis of this information, an ordering decision is made for the current period. All deliveries made during a period are assumed to arrive immediately after this ordering decision, and hence are on hand at the beginning of the period. A further discussion that states the convenience of this sequence of events can be found in Kaplan [23]. To summarize there are three successive events at the beginning of each period. First, the inventory on-hand and outstanding orders are determined. Second, an ordering decision is made on the basis of this information. Third, all supplier deliveries for the current period, including the most recent orders, are received.

3. Stochastic Programming Formulation

A stochastic programming formulation for the problem discussed in the previous section is given below,

$$\begin{aligned}
&\min \;\; E\{TC\} = \\
&\int_{d_1} \dots \int_{d_N} \sum_{t=1}^{N} \left[a \cdot \delta_t + h \cdot \max\left(\sum_{k=1}^{t} (X_k - d_k), 0 \right) \right] \qquad (1) \\
&g_1(d_1) g_2(d_2) \dots g_N(d_N) \mathrm{d}(d_1) \mathrm{d}(d_2) \dots \mathrm{d}(d_N) \\
&\text{subject to,} \\
&I_t = I_0 + \sum_{\{k|k\geq 1, l_k \leq t-k\}} X_k - \sum_{k=1}^{t} d_k \qquad t = 1, \dots, N \qquad (2) \\
&\delta_t = \begin{cases} 1, & \text{if} \quad X_t > 0 \\ 0, & \text{otherwise} \end{cases} \qquad t = 1, \dots, N \qquad (3) \\
&\Pr\{I_t \geq 0\} \geq \alpha \qquad t = L+1, \dots, N \qquad (4) \\
&I_t \in \mathbb{R}, \quad X_t \geq 0, \qquad t = 1, \dots, N. \qquad (5)
\end{aligned}$$

where

$E\{.\}$: the expectation operator,
TC : total cost,
d_t : the demand in period t, a random variable with probability density function, $g_t(d_t)$,
a : the fixed ordering cost (incurred when an order is placed),
h : the proportional inventory holding cost,

l_t : the lead time length of the order placed in period t, a discrete random variable with probability mass function $f_t(\cdot)$.

δ_t : a {0,1} variable that takes the value of 1 if a replenishment occurs in period t and 0 otherwise,

I_t : the inventory level (stock on hand minus back-orders) at the end of period t,

I_0 : the initial inventory,

X_t : the size of the replenishment order placed in period t, $X_t \geq 0$, (received in period $t + L$).

In this model, the objective function (Eq. 1) minimizes the expected total cost, which is comprised of ordering costs and inventory holding costs. As discussed earlier, the latter are charged in each period on delivered and outstanding orders, for this reason the stochastic lead time does not play a direct role in the objective function. Eq. 2 represents the inventory balance constraint, which states that the inventory level at period t, I_t, is the sum of the initial inventory, I_0, and of all the subsequent order quantities that are delivered by period t, $\sum_{\{k|k\geq 1, l_k \leq t-k\}} X_k$, minus the cumulative demand up to period t, $\sum_{k=1}^{t} d_k$. Eq. 3 states that if a replenishment occurs in period t — i.e. the order quantity X_t is greater than 0 — then the corresponding indicator variable δ_t must take a value of 1. Eq. 4 enforces the required service level constraint in each period. That is, the probability inventory level at the end of each period is positive, must be greater or equal to the threshold α. Finally, the inventory levels, I_t, are real valued decision variables and the order quantities, X_t, must be positive. Note that, depending on the lead time probability mass functions, it may not be possible to provide the required service level for some initial periods. In general, reasoning in a worst case scenario, it is always possible to provide the required service level α starting from period $L+1$. For this reason, the service level constraints are only enforced over periods $L + 1, \ldots, N$.

4. Deterministic Equivalent Modeling

In order to solve the above model, it is necessary to reformulate the service level constraints in Eq. 4, in terms of deterministic equivalent expressions. To do so, we blend a scenario

based approach — since the lead time probability distribution is assumed to be discrete — with a strategy similar to Bookbinder and Tan's "static-dynamic" uncertainty [10].

To begin, we discuss how to obtain a deterministic equivalent formulation for the chance-constraints that enforce the required service level when the lead time in each period varies and assumes a given deterministic value. Subsequently, we will generalize the same reasoning to the case in which the lead time is stochastic and assumes a different distribution from period to period.

When a dynamic deterministic lead time $L_t \geq 0$ is considered in each period $t = 1, \ldots, N$, an order placed in period t will be received only at period $t + L_t$. Eq. 2 therefore becomes,

$$I_t = I_0 + \sum_{\{k|k\geq 1, L_k+k\leq t\}} X_k - \sum_{k=1}^{t} d_k \qquad t = 1, ..., N. \tag{6}$$

Let us denote the inventory position (the total amount of inventory on-hand plus outstanding orders minus backorders) at the end of period t as P_t. It directly follows that

$$P_t = I_t + \sum_{\{k|1\leq k\leq t, L_k+k>t\}} X_k, \tag{7}$$

where we assume $P_0 = I_0$. It is easy, then, to reformulate the model using the inventory position. Furthermore, consider the expectation operator $E\{\cdot\}$, and since the demands $\{d_t\}$ are assumed to be mutually independent, we may rewrite the objective function as

$$\min \quad E\{TC\} = \sum_{t=1}^{N} \left(h \cdot E\left\{\max(P_t, 0)\right\} + a \cdot \delta_t\right). \tag{8}$$

When a stock-out occurs, all demand is back-ordered and fulfilled as soon as an adequate supply arrives. Following Bookbinder and Tan [10], since we have assumed that the management will set the non-stockout probability to a reasonably high level — certainly greater than 0.5 — we can safely replace the term $E\{\max(P_t, 0)\}$ with the term $E\{P_t\}$.

The general stochastic programming formulation can then be modified to incorporate the "replenishment cycle policy". Consider a review schedule, which has m reviews over the N period planning horizon with orders placed at $T_1, T_2, \ldots, T_m$, where $T_i > T_{i-1}$, $T_m \leq N - L_{T_m}$. For convenience, T_1 is defined as the start of the planning horizon and $T_{m+1} = N + 1$ as the period immediately after the end of the planning horizon. Note that the review schedule may be generalized to consider the case where $T_1 > 1$, if the opening inventory I_0 is sufficient to cover the immediate needs at the start of the planning horizon. The associated inventory

reviews will take place at the beginning of periods T_i, $i = 1, \ldots, m$. In the replenishment cycle policy considered here, clearly the orders X_i are all equal to zero except at replenishment periods $T_1, T_2, \ldots, T_m$. The inventory level I_t carried from period t to period $t+1$ is the opening inventory plus any orders that have arrived up to and including period t less the total demand to date. Hence, the inventory balance equation becomes,

$$I_t = I_0 + \sum_{\{i|L_{T_i}+T_i \leq t\}} X_{T_i} - \sum_{k=1}^{t} d_k, \quad t = 1, \ldots, N. \tag{9}$$

Define $T_{p(t)}$ as the latest review before period t in the planning horizon, for which all the former orders, including the one placed in $T_{p(t)}$, are delivered within period t. Therefore,

$$p(t) = \max\left\{i|\forall j \in \{1, \ldots, i\}, T_j + L_{T_j} \leq t, \quad i = 1, \ldots, m\right\}. \tag{10}$$

The inventory level I_t at the end of period t (Eq. 9) can be expressed as

$$I_t = I_0 + \sum_{i=1}^{p(t)} X_{T_i} + \sum_{\{i|i>p(t), L_{T_i}+T_i \leq t\}} X_{T_i} - \sum_{k=1}^{t} d_k, \quad t = 1, \ldots, N. \tag{11}$$

We now want to reformulate the constraints of the chance-constrained model in terms of a new set of decision variables R_{T_i}, $i = 1, \ldots, m$.

Define,

$$P_t = R_{T_i} - \sum_{k=T_i}^{t} d_k, \quad T_i \leq t < T_{i+1}, \quad i = 1, \ldots, m \tag{12}$$

where R_{T_i} can be interpreted as the inventory position up to which inventory should be raised after placing an order at the ith review period T_i. We now express Eq. 11 using R_{T_i} as decision variables

$$I_t = R_{T_{p(t)}} + \sum_{\{i|i>p(t), L_{T_i}+T_i \leq t\}} \left(R_{T_i} - R_{T_{i-1}} + d_{T_{i-1}} + \ldots + d_{T_i-1}\right) - \sum_{k=T_{p(t)}}^{t} d_k, \quad t = 1, \ldots, N. \tag{13}$$

As mentioned earlier, α is the desired minimum probability that the net inventory level in any time period is non-negative. Depending on the values assigned to L_t, it may not be possible to provide the required service level for some initial periods. In general, we provide the required service level α starting from the period t, for which the value $t + L_t$ is minimum. Let M be this period. Note that, it will never be optimal to place any order in a period t

such that $t + L_t > N$, since such an order will not be received within the given planning horizon.

By substituting I_t with the right hand term in Eq. 13 we obtain

$$G_S\left(R_{T_{p(t)}} + \sum_{\{i|i>p(t),L_{T_i}+T_i\leq t\}} (R_{T_i} - R_{T_{i-1}})\right) \geq \alpha, \qquad t = M, \ldots, N. \tag{14}$$

where $S = \sum_{k=T_{p(t)}}^{t} d_k - \sum_{\{i|i>p(t),L_{T_i}+T_i\leq t\}}(d_{T_{i-1}} + \ldots + d_{T_i-1})$, and $G_S(.)$ is the cumulative distribution function of S. The service level constraints are now deterministic and are expressed only in terms of the order-up-to-positions.

It is now relatively easy to obtain a deterministic equivalent model in which lead times are stochastic, under the original assumption that the lead time in each period is a discrete random variable l_t.

We first reformulate the chance-constrained model under stochastic lead time using the inventory position,

$$\min \quad E\{TC\} = \int_{d_1} \ldots \int_{d_N} \sum_{t=1}^{N} (a\delta_t + h \cdot P_t) \; g_1(d_1)g_2(d_2)\ldots g_N(d_N)\mathrm{d}(d_1)\mathrm{d}(d_2)\ldots\mathrm{d}(d_N) \tag{15}$$

subject to,

$$\delta_t = \begin{cases} 1, & \text{if} \quad X_t > 0 \\ 0, & \text{otherwise} \end{cases} \qquad t = 1, ..., N \tag{16}$$

$$P_t = I_0 + \sum_{k=1}^{t}(X_k - d_k) \qquad t = 1, ..., N \tag{17}$$

$$\Pr\{P_t \geq \sum_{\{k|1\leq k\leq t, l_k > t-k\}} X_k\} \geq \alpha \qquad t = L+1, ..., N \tag{18}$$

$$P_t \geq 0, \quad X_t \geq 0, \qquad t = 1, ..., N. \tag{19}$$

Also in this case, we want to adopt a replenishment cycle policy and express the whole model in terms of the new set of variables R_{T_i}, so that order quantities are decided only after the demand in the former periods is realized.

Similar to the dynamic deterministic lead time case, we now express the service level constraint as a relation between the opening-inventory-positions, such that the overall service level provided at the end of each period is at least α. In order to express this service level constraint, we propose a scenario based approach over the discrete random variables $\{l_t\}$. In

a scenario based approach (see also Birge and Louveaux [8] and Tarim et al. [42]), a scenario tree is generated which incorporates all possible realizations of discrete random variables into the model explicitly, yielding a fully deterministic model under the non-anticipativity constraints.

In our problem, we can divide random variables into two sets: the random variables $\{l_t\}$ which represent lead times and the random variables $\{d_t\}$ which represent demands. We deal with each set in a separate fashion, by employing a scenario based approach for the $\{l_t\}$ and a deterministic equivalent modeling approach for the $\{d_t\}$. This is possible since, under a given scenario, discrete random variables are treated as constants. The problem is then reduced to the general multi-period production/inventory problem with dynamic deterministic lead time and stochastic demands, which has been previously analyzed.

Consider a review schedule, which has m reviews over the N period planning horizon with orders placed at $T_1, T_2, \ldots, T_m$. A scenario ω_t is a possible lead time realization for all the orders placed up to period t in a given review schedule. Let $(l_{T_i}|\omega_t)$ be the realized lead time in scenario ω_t for the order placed in period T_i. Finally, let Ω_t be the set of all the possible scenarios ω_t.

Under a given scenario ω_t, the service level constraint for a period t can be easily expressed using Eq. 14. It follows that the service level constraint is always a relation between at most $L+1$ decision variables R_{T_i} that represent the order-up-to-positions of the replenishment cycles covering the span $t-L, \ldots, t$. Let $p_\omega(t)$ be the value of $p(t)$ under a given scenario ω_t, when a review schedule Z is considered. In order to satisfy the service level constraints in our original model, we require that the overall service level under all possible scenarios, for each set of at most $L+1$ decision variables, is at least α. Equivalently, by using Eq. 14,

$$\sum_{\omega_t \in \Omega_t} \Pr\{\omega_t\} \cdot G_S \left(R_{T_{p_\omega(t)}} + \sum_{\{i|i>p_\omega(t),(l_{T_i}|\omega_t)\leq t-T_i\}} (R_{T_i} - R_{T_{i-1}}) \right) \geq \alpha, \qquad t = L+1, \ldots, N, \tag{20}$$

where $S = \sum_{k=T_{p_\omega(t)}}^{t} d_k - \sum_{\{i|i>p_\omega(t),(l_{T_i}|\omega_t)\leq t-T_i\}} (d_{T_{i-1}} + \ldots + d_{T_i-1})$. It should be noted that this equation is non-linear. In the remainder of the paper, we refer to Eq. 20 as "service level constraints" or "SL Constraints", as this equation is refered most commonly throughtout.

In our chance-constrained model, we can now replace the original service level constraints with the new formulation in Eq. 20. As a consequence, the service level constraints are now expressed only in terms of the order-up-to-levels. Therefore, the expectation operator can

be safely applied to the closing-inventory-levels, $\{P_t\}$, and to the stochastic demands, $\{d_t\}$, since these variables are only affecting the objective function in which we are minimizing an expected value. In what follows, the expected value of P_t and d_t are denoted by $\tilde{P}_t$ and $\tilde{d}_t$, respectively.

We can now express the whole model in terms of a new set of decision variables R_t, $t = 1, \ldots, N$. If there is no replenishment scheduled for period t, that is if $\delta_t = 0$, then R_t must be equal to the expected closing-inventory-position in period $t-1$, that is $R_t = \tilde{P}_{t-1}$. If there is a review T_i in period t, R_t is equal to the order-up-to-position R_{T_i} for this review. Therefore, the desired order-up-to-positions, $\{R_{T_i}\}$, as required for the solution to the problem, are those values of R_t, for which $\delta_t = 1$.

The complete model under the replenishment cycle policy is then:

$$\min \quad E\{TC\} = \sum_{t=1}^{N} \left(h \cdot \tilde{P}_t + a \cdot \delta_t \right) \tag{21}$$

subject to,

Eq. 20 (SL Constraints)

$$R_t > \tilde{P}_{t-1} \Rightarrow \delta_t = 1 \qquad t = 1, \ldots, N \tag{22}$$

$$R_t \geq \tilde{P}_{t-1} \qquad t = 1, \ldots, N \tag{23}$$

$$R_t = \tilde{P}_t + \tilde{d}_t \qquad t = 1, \ldots, N \tag{24}$$

$$R_t \geq 0, \quad \tilde{P}_t \geq 0, \quad \delta_t \in \{0, 1\} \qquad t = 1, \ldots, N, \tag{25}$$

where $\{T_1, \ldots, T_m\} = \{t \in \{1, \ldots, N\} | \delta_t = 1\}$.

It should be noted that the domain of each $\tilde{P}_t$ variable — as in the zero lead time case (see Tarim and Smith [43]) — is limited. In fact, since the period demand variance is additive, the uncertainty can only increase in the length of a replenishment cycle. Therefore the longer a cycle is, the higher the inventory levels that are required to achieve a certain service level. It directly follows that a single replenishment covering the whole planning horizon will provide upper bounds for the expected period closing-inventory-positions throughout the horizon.

4.1. An example

We assume an initial null inventory level and a normally distributed demand with a coefficient of variation $\sigma_t / \tilde{d}_t = 0.3$ for each period $t \in \{1, \ldots, 5\}$. The expected values for the demand in each period are: {36, 28, 42, 33, 30}. The other parameters are $a = 1$, $h = 1$, $\alpha =$

Policy cost: 356					
Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	125	124	129	87	55
δ_t	1	1	1	1	1
Shortage probability	–	–	5%	5%	5%

Table 1: A replenishment plan.

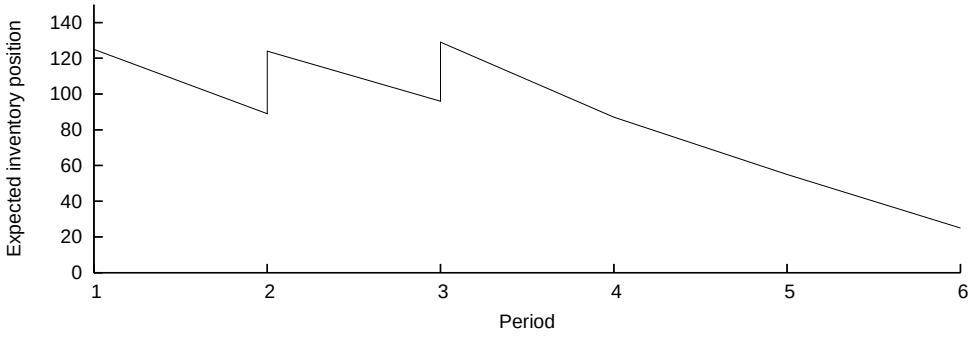

Figure 1: A graphical illustration of the replenishment plan in Table 1.

$0.95(z_{\alpha=0.95} = 1.645)$. We consider that every period i in the planning horizon has following lead time probability mass function $f_t(k) = \{0.3(0), 0.2(1), 0.5(2)\}$. This implies that we receive an order placed in period i after $t \in \{0, \ldots, 2\}$ periods with a given probability (0 periods: 30%; 1 period: 20%; 2 periods: 50%). It is obvious that in this case, we will always receive the order within 2 periods, after it is placed. In Table 1, we show the optimal solution. The optimal replenishment plan is also illustrated in Fig. 1. We now show, through Eq. 20 (SL Constraints), that the order-up-to-positions in this example satisfy every service level constraint in the model. We assume that for the first 2 periods, no service level constraint is enforced, since it is not possible to control the inventory in the first 2 periods. Therefore, we enforce the required service level on period 3, 4 and 5 (that is Eq. 20 or SL Constraints) for $t = 3, \ldots, N$. Let us verify that the given order-up-to-levels satisfy this condition for these three periods. Since we know the probability mass function $f_t(\cdot)$ for each period t in the planning horizon, we can compute the probability $Pr(\omega_t)$ for each scenario $\omega_t \in \Omega_t$. We thus have four of these scenarios for each period $t \in \{3, \ldots, N\}$, as we are placing an order in each period.

- S_1, $Pr\{S_1\} = 0.15$; in this scenario for period t, we receive all the former orders

- S_2, $Pr\{S_2\} = 0.35$; in this scenario for period t, we do not receive the last order placed in period t
- S_3, $Pr\{S_3\} = 0.35$; in this scenario for period t, we do not receive the last two orders placed in period t and $t-1$
- S_4, $Pr\{S_4\} = 0.15$; in this scenario for period t, we do not receive the order placed in period $t-1$, and we observe order-crossover.

In the described scenarios, every possible configuration is considered, without loss of generality. In fact, if some of the configurations are unrealistic (for instance, if we assume that order-crossover may not take place) we just need to set the probability of the respective scenario to zero. Now, it is possible to write SL Constraints (Eq. 20) for each period $t \in \{3, \ldots, N\}$. For period 3,

$$\begin{aligned} Pr\{S_1\} \cdot G\left(\frac{129-42}{0.3\sqrt{42^2}}\right) + Pr\{S_2\} \cdot G\left(\frac{124-(28+42)}{0.3\sqrt{28^2+42^2}}\right) + \\ Pr\{S_3\} \cdot G\left(\frac{125-(36+28+42)}{0.3\sqrt{36^2+28^2+42^2}}\right) + \\ Pr\{S_4\} \cdot G\left(\frac{125+(129-124)-(36+42)}{0.3\sqrt{36^2+42^2}}\right) = 94.60 \cong 95, \end{aligned} \tag{26}$$

where $G(\cdot)$ is the standard normal distribution function with zero mean and unit standard deviation. This implies that the combined effect of order delivery delays in our policy, under any possible scenario results in a stock-out probability of exactly 95% for period 3. A similar reasoning can be applied to verify that the given solution satisfies the required service level for period 4 and 5.

5. Heuristic Method I

In this section, we introduce a first heuristic method, named Heuristic I or, shortly, H1, for computing near optimal replenishment cycle policy parameters under non-stationary stochastic demand and lead time. The key intuition behind this heuristic strategy consists in noticing that when **all the replenishment decisions** have been fixed, the SL Constraints (Eq. 20) can be used to check if in a given period the required service level constraint is met. If the service level is not met, the gradient function is able to indicate which order-up-to-position should be increased in order to achieve the maximum service level improvement for

that period. This method employs one of the efficient techniques proposed in the literature such as the one in Tarim and Smith [43].

$$\min \; E\{TC\} = \sum_{t=1}^{N} \left(a\delta_t + h\tilde{I}_t \right) \tag{27}$$

subject to, for $t = 1 \ldots N$

$$\tilde{I}_t + \tilde{d}_t - \tilde{I}_{t-1} \geq 0 \tag{28}$$

$$\tilde{I}_t + \tilde{d}_t - \tilde{I}_{t-1} > 0 \Rightarrow \delta_t = 1 \tag{29}$$

$$\tilde{I}_t \geq b \left(\max_{j \in \{1,\ldots,t\}} j \cdot \delta_j, t \right) \tag{30}$$

$$\tilde{I}_t \in \mathbb{Z}^+ \cup \{0\}, \quad \delta_t \in \{0, 1\}, \tag{31}$$

where $b(i,j) = G^{-1}_{d_i+d_{i+1}+\ldots+d_j}(\alpha) - \sum_{k=i}^{j} \tilde{d}_k$, and $G^{-1}_{d_i+d_{i+1}+\ldots+d_j}(\cdot)$ denotes the inverse cumulative distribution function of $d_i + d_{i+1} + \ldots + d_j$.

In Tarim and Smith's model (Eqs. 27–31), each decision variable $\tilde{I}_t$, represents the expected inventory level at the end of period t. Each $\tilde{d}_t$ represents the expected value of the demand in a given period t, according to its probability density function $g_t(d_t)$. The binary decision variables δ_t state whether a replenishment is fixed for period t ($\delta_t = 1$) or not ($\delta_t = 0$). The objective function (27) minimizes the expected total cost over the given planning horizon. The two terms that contribute to the expected total cost are ordering costs and inventory holding costs. Constraint (28) enforces a no-buy-back condition, which means that received goods cannot be returned to the supplier. As a consequence of this, the expected inventory level at the end of period t must be no less than the expected inventory level at the end of period $t-1$ minus the expected demand in period t. Constraint (29) expresses the replenishment condition. We have a replenishment if the expected inventory level at the end of period t is greater than the expected inventory level at the end of period $t-1$ minus the expected demand in period t. This means that we receive some extra goods as a consequence of an order. Constraint (30) enforces the required service level α. This is done by specifying the minimum expected closing-inventory-level (or "buffer stock") required for each period t in order to assure that, at the end of each time period, the probability that the net inventory is non-negative is at least α. These buffer stocks, which are stored in matrix $b(\cdot,\cdot)$, are pre-computed following the approach suggested in Tarim and Kingsman [40].

The buffer stocks mentioned in the previous paragraph refer to the case in which no lead time is considered and where every order is delivered immediately. These buffer stocks are

Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	54	42	63	49	45
δ_t	1	1	1	1	1
Shortage probability	–	–	78%	78%	77%

Table 2: No lead time solution.

typically lower than those required to provide the required service level α, when a stochastic delivery lead time is considered. Our strategy, in this first heuristic, consists in iteratively adjusting the buffer stocks in the matrix in order to increase the service level in each period till the required service level α is met in every period of the planning horizon. The local search procedure to compute policy parameters under stochastic lead time is shown in Algorithm 1. In this algorithm a replenishment plan is made of a number of replenishment cycles. A replenishment cycle $\mathcal{R}(i, j)$ is the set of periods that are located between two consecutive replenishment periods i and $j + 1$.

The method initially solves the model in Eqs. 27–31 (Algorithm 1, line 5), with buffer stocks set as in the no lead time case. This gives a replenishment plan, that is an assignment for decision variables δ_t and a (possibly) infeasible assignment for the respective order-up-to-positions. If this assignment is infeasible (Algorithm 1, line 6), we consider sequentially (Algorithm 1, line 10) each period in every replenishment cycle scheduled and increase the buffer stocks (Algorithm 1, line 15) of replenishment cycles affecting the service level in this period (Algorithm 1, line 11) until the required service level α is met (Algorithm 1, line 12). This can be checked by using SL Constraints (Eq. 20). Buffer stocks are increased according to a rule that increments at each step the buffer stock that produces the highest service level improvement for the period considered (Algorithm 1, line 14). This process is iterated by solving again the model in Eqs. 27–31 using this modified buffer stock matrix (Algorithm 1, line 17), until the model directly produces a feasible solution (Algorithm 1, line 18). We now provide a simple example to illustrate this procedure.

We present the same example proposed in Section 4.1. By disregarding the information on the stochastic lead time, we use the relevant data in the model shown in Eqs. 27–31. By solving this model, we obtain the solution shown in Table 2. Considering the stochastic lead time and by using SL Constraints (Eq. 20), it is possible to compute the service level provided in period 3, 4 and 5 for this solution (These service levels are also shown in Table

Algorithm 1: `Heuristic Method I`

input : $d_1, \ldots, d_N$; a; h; α
output: a replenishment plan

begin
  for each period i **in** $1, \ldots, N$ **do**
    for each period j **in** $i, \ldots, N$ **do**
      $b(i,j) \leftarrow G^{-1}_{d_i+d_{i+1}+\ldots+d_j}(\alpha) - \sum_{k=i}^{j} \tilde{d}_k$
  Solve the model in Eqs. 27–31 with input $d_1, \ldots, d_N$, a, h, α, and buffer matrix $b(\cdot,\cdot)$;
  By using SL Constraints (Eq. 20), check if the solution found provides the required service level α at the end of each period;
  while *the current solution does not provide service level* α **do**
    Let $\mathcal{R}$ be the set of consecutive replenishment cycles in the solution;
    for *each replenishment cycle* $\mathcal{R}(i,j)$ *in* $\mathcal{R}$ **do**
      for *each period* t *in* $\mathcal{R}(i,j)$ **do**
        Let $\mathcal{P}$ be the set of former cycles influencing the service level in period t according to Eq. 20;
        while *the service level in period* t *is less than* α **do**
          For each cycle $\mathcal{R}(m,n) \in \mathcal{P} \bigcup \{\mathcal{R}(i,j)\}$ obtain the respective minimum allowed order-up-to-position $\underline{R}_m = \tilde{I}_n + \sum_{i=m}^{n} \tilde{d}_i$;
          Let $\mathcal{R}(m,n) \in \mathcal{P} \bigcup \{\mathcal{R}(i,j)\}$ be the cycle for which a unit increment in $\underline{R}_m$ produces the highest service level improvement;
          $b(m,n) \leftarrow \tilde{I}_n + 1$;
          $\tilde{I}_n \leftarrow \tilde{I}_n + 1$;
    Solve the model in Eqs. 27–31 with input $d_1, \ldots, d_N$, a, h, α, and modified buffer matrix $b(\cdot,\cdot)$;
    By using SL Constraints (Eq. 20), check if the solution found provides the required service level α at the end of each period;
  return the current replenishment plan;
end

2). Clearly, these service levels are not higher than the required minimum service level α. Therefore, using SL Constraints we modify the buffer stock matrix $b(\cdot,\cdot)$ in such a way so as to increase the relevant buffers and thus obtain a feasible solution. For instance, we increase the buffer stock level $b(1,1)$ of replenishment cycle $\mathcal{R}(1,1)$ from 18 to 102, the buffer stock level $b(2,2)$ of replenishment cycle $\mathcal{R}(2,2)$ from 14 to 106, the buffer stock level $b(3,3)$ of replenishment cycle $\mathcal{R}(3,3)$ from 21 to 94, and the buffer stock level $b(4,4)$ of replenishment cycle $\mathcal{R}(4,4)$ from 16 to 50. This is based on the greedy rule discussed in Algorithm 1,

Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	138	102	74	83	50
δ_t	1	0	0	1	1
Shortage probability	–	–	5%	35%	48%

Table 3: Solution with increased buffers.

Policy cost: 397					
Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	138	134	136	94	61
δ_t	1	1	1	0	0
Shortage probability	–	–	2%	2%	5%

Table 4: Feasible solution.

and further based on the SL Constraints. We solve the model again using the modified buffer stock matrix $b(\cdot, \cdot)$. Since some of the buffers are increased, it is not anymore optimal to schedule a replenishment in each period, as noticed in the solution obtained using this new modified buffer stock matrix (Table 3). The service level provided in period 3 is now sufficiently high, but those provided in period 4 and 5 is not. We shall therefore iterate the process until we eventually converge to the feasible solution presented in Table 4. This heuristic is about 11% more costly than the optimal replenishment strategy.

6. Heuristic method II

The Heuristic Method I presented in the former section typically converges to a good solution in a few iterations, but often it may not produce solutions that are sufficiently close to the optimal. In order to produce higher quality solutions, we discuss here a different strategy that employs a Constraint Based Local Search approach. We name this second heuristic Heuristic II or H2.

6.1. Constraint Reasoning

Constraint Programming (see Apt [2]) is a declarative programming paradigm in which relations between decision variables are stated in the form of constraints. Informally speaking, constraints specify the properties of a solution to be found. The constraints used in con-

straint programming are of various kinds: logic constraints (i.e. "x or y is true", where x and y are boolean decision variables), linear constraints, and *global constraints* (Régin [29]). A global constraint captures a relation among a non-fixed number of variables. One of the most well known global constraints is the **alldiff** constraint (Régin [28]), that can be enforced on a certain set of decision variables in order to guarantee that no two variables are assigned the same value.

With each constraint, CP associates a *filtering algorithm* able to remove provably infeasible or suboptimal values from the domains of the decision variables that are constrained and, therefore, to enforce some degree of *consistency* (see Rossi et al. [31]). These filtering algorithms are repeatedly called until no more values are pruned. This process is called *constraint propagation*.

In addition to constraints and filtering algorithms, constraint solvers also feature some sort of *heuristic search engine* (e.g. a backtracking algorithm). During the search, the constraint solver exploits filtering algorithms in order to proactively prune part of the search space that cannot lead to a feasible or to an optimal solution.

6.2. Local Search

A *neighborhood structure* is a function $\mathcal{N} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ that assigns to every solution $s \in \mathcal{S}$, a set of neighbors $\mathcal{N}(s) \subseteq \mathcal{S}$. $\mathcal{N}(s)$ called the neighborhood of s. Without loss of generality, we here restrict the discussion to minimization problems. A *locally minimal solution* (or local minimum) with respect to a neighborhood structure $\mathcal{N}$ is a solution $\widehat{s}$ such that $\forall s \in \mathcal{N}(\widehat{s}) : f(\widehat{s}) \leq f(s)$. We call $\widehat{s}$ a strict local minimal solution if $\forall s \in \mathcal{N}(\widehat{s}) : f(\widehat{s}) < f(s)$. Local search (LS) algorithms for COPs start from some initial solution and iteratively try to replace the current solution by a better solution in an appropriately defined neighborhood of the current solution. In this process, it is extremely important to achieve a proper balance between *diversification* and *intensification* of the search. The term diversification generally refers to the exploration of the search space, whereas the term intensification refers to the exploitation of the accumulated search experience. Among the most popular local search strategies we recall the *Iterative Improvement*, or *Hill Climbing*, in which each move is only performed if the resulting solution is better than the current solution and the algorithm stops as soon as it finds a local minimum. *Tabu Search* is a more advanced strategy, in fact it is among the most cited and used. Tabu search explicitly uses the history of the search, both to escape from local minima and to implement an explorative strategy. *Iterated Local Search*

Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	54	42	63	–	–
δ_t	1	1	1	–	–
Shortage probability	–	–	78%	–	–

Table 5: Partial assignment.

and *Variable Neighborhood Search* constitute other examples of local search strategies. For a comprehensive survey on local search and metaheuristic strategies, the reader may refer to Blum and Roli [9]. In what follows, we will employ a strategy known as *coordinate descent*. Coordinate descent algorithms, sometimes called one-at-a-time, minimize (maximize) a given function by minimizing (maximizing) it over a single variable while holding all other variables constant. There are two approaches: cyclic algorithms that cycle through all of the variables; and greedy algorithms that choose the variable that reduces the cost by the largest amount in each iteration. In the coordinate descent strategy discussed in the next section, an algorithm belonging to this second "greedy" class will be employed.

6.3. The Approach

The key intuition behind the second heuristic strategy slightly differs from that of the first heuristic. In this heuristic, we emphasize that when **some replenishment decisions** have been fixed, SL Constraints (Eq. 20) can be used to check if in a given period, the required service level constraint is met. If the service level is not met, a gradient function indicates which order-up-to-position should be increased in order to achieve the maximum service level improvement in such a period. We shall now provide a simple example to clarify how this procedure works.

We consider again the example proposed in Section 4.1. In Table 5, we show a possible partial replenishment plan that schedules orders in period 1, 2, and 3; the remaining replenishment decisions are not yet fixed. Clearly, the order-up-to-level in each period $t = 1, \ldots, 3$ will be at least as high as those required to provide the service level α, when the lead time is 0. Therefore, a good starting configuration for the order-up-to-level is 54, 42, and 63 respectively for R_1, R_2, and R_3. As it is easy to observe using SL Constraints, it is now possible to compute the service level provided in period 3.

Policy cost: 366					
Period (t)	1	2	3	4	5
$\tilde{d}_t$	36	28	42	33	30
R_t	131	128	130	88	55
δ_t	1	1	1	1	1
Shortage probability	–	–	3%	3%	5%

Table 6: Full assignment.

$$Pr\{S_1\} \cdot G\left(\frac{63-42}{0.3\sqrt{42^2}}\right) + Pr\{S_2\} \cdot G\left(\frac{42-(28+42)}{0.3\sqrt{28^2+42^2}}\right) + Pr\{S_3\} \cdot G\left(\frac{54-(36+28+42)}{0.3\sqrt{36^2+28^2+42^2}}\right) + Pr\{S_4\} \cdot G\left(\frac{54+(63-42)-(36+42)}{0.3\sqrt{36^2+42^2}}\right) = 0.2192 \cong 0.22 \quad (32)$$

The service level provided (about 0.22) is not sufficient to satisfy SL Constraints (Eq. 20) in period 3. In order to decide which order-up-to-position to increase, we analyze the behavior of the service level at period 3, when R_1, R_2, and R_3 are increased respectively. If we increase R_1 by one unit, the service level at period 3 becomes 0.2229; if we increase R_2 by one unit, the service level at period 3 becomes 0.2175; finally, if we increase R_3 by one unit, the service level at period 3 becomes 0.2239 It follows that, an increase of one unit for R_3 achieves the maximum service level improvement. We proceed in a similar fashion by increasing at each step the order-up-to-position R_t that produces the maximum increase in the service level provided, until SL Constraints are satisfied for the period of interest. The reader may be easily convinced that, when we consider period 3, this process terminates after a few steps, when $R_1 = 131$, $R_2 = 95$, and $R_3 = 75$. We then proceed and repeat the same process, assuming that the ordering decisions are all fixed and that an order is scheduled in every period. In this case, we consider period 4 and period 5 sequentially. The final solution produced by this approach is shown in Table 6.

We next describe the complete approach. Our technique exploits the model presented in Eq. 21 - 25. This model is implemented within Choco 1.2 (Laburthe et al. [24]), an open source Constraint Programming solver developed in Java. The variable selection heuristic branches first on decision variables δ_t. These variables are selected according to their natural order, that is $\{\delta_1, \ldots, \delta_N\}$. The value selection heuristic selects values in increasing order. SL Constraints (Eq. 20) cannot be directly implemented as such, and therefore are replaced, in our Constraint Programming model, by a global constraint able to dynamically compute

the required order-up-to-level for a given partial replenishment plan (that is, a partial assignment for decision variables δ_t). As discussed, the order-up-to-levels are computed using the gradient-based local search approach shown in the former example, which in practice follows a *coordinate descent* strategy. A pseudo-code describing the propagation logic of this constraint is presented below in Algorithm 2.

Algorithm 2: `Heuristic Method II - Propagation`

input : a partial assignment for decision variables δ_t, $t = 1, \ldots, N$,
the expected closing-inventory-positions $\tilde{P}_t$, $t = 1, \ldots, N$,
the service level α

1 **begin**
2 Let $\mathcal{R}$ be the set of consecutive replenishment cycles identified by the partial assignment for decision variables δ_t;
3 **for** *each replenishment cycle* $\mathcal{R}(i,j)$ *in* $\mathcal{R}$ **do**
4 **for** *each period* t *in* $\mathcal{R}(i,j)$ **do**
5 Let $\mathcal{P}$ be the set of former cycles influencing the service level in period t according to SL Constraint (Eq. 20);
6 **while** *the service level in period* t *is less than* α **do**
7 For each cycle $\mathcal{R}(m,n) \in \mathcal{R} \bigcup \{\mathcal{R}(i,j)\}$ obtain the respective minimum allowed order-up-to-position $\underline{R}_m$ as $Inf\{Dom(\tilde{P}_m)\} + \tilde{d}_m$;
8 Let $\mathcal{R}(m,n) \in \mathcal{R} \bigcup \{\mathcal{R}(i,j)\}$ be the cycle for which a unit increment in $\underline{R}_m$ produces the highest service level improvement;
9 $Inf\{Dom(\tilde{P}_m)\} \leftarrow Inf\{Dom(\tilde{P}_m)\} + 1$;
10 **end**

The propagation logic described in Algorithm 2 is triggered each and every time, during the search, a replenishment decision is fixed. Recall that our search strategy branches first on decision variables δ_t, according to their natural sequence, so that at each node of the search tree we have a set of consecutive replenishment decisions $\{\delta_1, \ldots, \delta_t\}$ that have been assigned. This implies, that at each node of the search tree, we will also have a set of consecutive replenishment cycles, $\mathcal{R} \equiv \{\mathcal{R}(1,i), \ldots, \mathcal{R}(j,t)\}$, that are uniquely identified by the current partial assignment (Algorithm 2, line 2). For each cycle, we consider each and every period (Algorithm 2, line 4). By using SL Constraints (Eq. 20) and more specifically condition 10, we can easily identify which former cycles (and order-up-to-positions) are affecting the service level in the period under consideration (Algorithm 2, line 5). We consider each of these order-up-to-positions, say R_m, and observe the behavior of the service level when the

minimum value allowed for it, $\underline{R}_m = Inf\{Dom(\tilde{P}_m)\} + \tilde{d}_m$, (Algorithm 2, line 7) is increased by one unit. That is, when the minimum value (Inf) in the domain (Dom) of $\tilde{P}_m$ plus $\tilde{d}_m$, is increased by one unit. Once the order-up-to-position R_m that produces the maximum service level improvement for the period considered is identified (Algorithm 2, line 8), we restrict the domain of the corresponding expected closing inventory position $\tilde{P}_m$ by removing the value $Inf\{Dom(\tilde{P}_m)\}$ (Algorithm 2, line 9). This process eventually produces (by subsequent greedy improvements), a set of order-up-to-positions that meet the required service level.

Also this second approach is clearly heuristic, since the order-up-to-levels are adjusted by using "local" moves that aim to locally maximize the improvement in the service level provided at a given period. It is also an example of a hybrid method that employs local search, at each node of the search tree, within the propagation logic of a global constraint. Nevertheless, this approach does perform better than the previous one in terms of quality of the solutions produced. For instance, with respect to the example presented in Section 4.1, this second heuristic approach produces a solution with cost 366, that is only 2.8% more costly than the optimal solution.

It is worth recalling that, as discussed in Section 1, the strategy described in this section introduces local search techniques within a constructive global search algorithm, i.e. the Constraint Programming solver. More specifically, a *coordinate descent* heuristic is employed in order to heuristically propagate the service level constraints within a Constraint Programming model. In contrast, the technique discussed in Section 5 exploits the local search engine to "guide" the search, by restructuring the buffer stock matrix, while Constraint Programming is used to explore promising neighborhood, i.e. to find the optimal solution with respect to a given buffer stock matrix.

7. Computational Results

In the illustrative example provided in the previous section, it was shown that Heuristic I and Heuristic II are 11% and 3% more costly than the optimal solution. In this section, we aim to further analyze both the effectiveness and the efficiency of these two heuristics. More specifically, we consider a large set of instances and we investigate, for each of the two heuristics, the optimality gap and the computational effort. The optimal solution used for comparison is obtained from the complete approach discussed in Rossi et al. [33]. In addition, we investigate how well the two heuristics scale, when the instances become intractable for

the complete approach.

We consider four patterns for the expected value of the demand in each period of the planning horizon. These patterns resemble the structure of the experiments proposed in Berry [7] and comprise a constant level, a life-cycle trend, a sinusoidal change, and a very erratic pattern (Fig. 2). The demand is normally distributed, in each period, about the forecast value. We consider two possible values for the coefficient of variation, $c_v \in \{0.2, 0.3\}$. c_v shows the effect of the size of random variation in demand about the mean. Recall that $\sigma_t = \tilde{d}_t c_v$, where σ_t is the standard deviation of the demand in period t and $\tilde{d}_t$ is the expected value of the demand in period t. The holding cost h is assumed to be fixed to 1 for all the instances, while the ordering cost a takes values in the set $\{100, 175, 250\}$. We consider two possible service levels $\alpha \in \{0.85, 0.95\}$. In our experiments, we consider five possible discrete probability density functions for the stochastic lead time. These are shown in Fig. 3. The lead time may therefore take one of the values shown in the x-axis with the probability indicated on the y-axis. It should be noted that it is not relevant in this work to compare the performance of our heuristics for a deterministic lead time. In fact, efficient complete solution methods have been proposed in the literature [39] for the case in which the lead time is absent. These methods, according to the discussion in [38], can be directly applied to solve instances in which the lead time is deterministic.

All the experiments were performed on an Intel® Pentium®4 3.66 Ghz with 2 Gb of RAM. Heuristic I has been implemented using ILOG OPL Studio 3.7 [1] and Solver 6.0 interfaced with a Java routine for updating the buffer stock matrix. Heuristic II has been implemented on the top of Choco 1.2 (Laburthe et al. [24]).

The CP model repeatedly solved by Heuristic I only comprises $2N$ variables and $3N$ constraints, as discussed in [43]. Several efficient approaches [43, 39, 34] exist for solving such a model in fraction of a second for planning horizons comprising hundreds of periods. Both the complete approach in Rossi et al. [33] and Heuristic II operate on CP models that comprise $3N$ variables and $3N + 1$ constraints. Therefore all these models are scalable with respect to lead time and planning horizon length.

In order to assess the quality of the solutions produced by the two heuristics, we first consider small instances over a 6-period planning horizon. We also limit the maximum length of the stochastic lead time, so that the resulting instances are tractable for the complete approach, which can therefore prove optimality in a reasonable time. In order to do so, we consider in Fig. 2 only the expected demand in periods $\{1, \ldots, 6\}$, and in Fig. 3 the lead

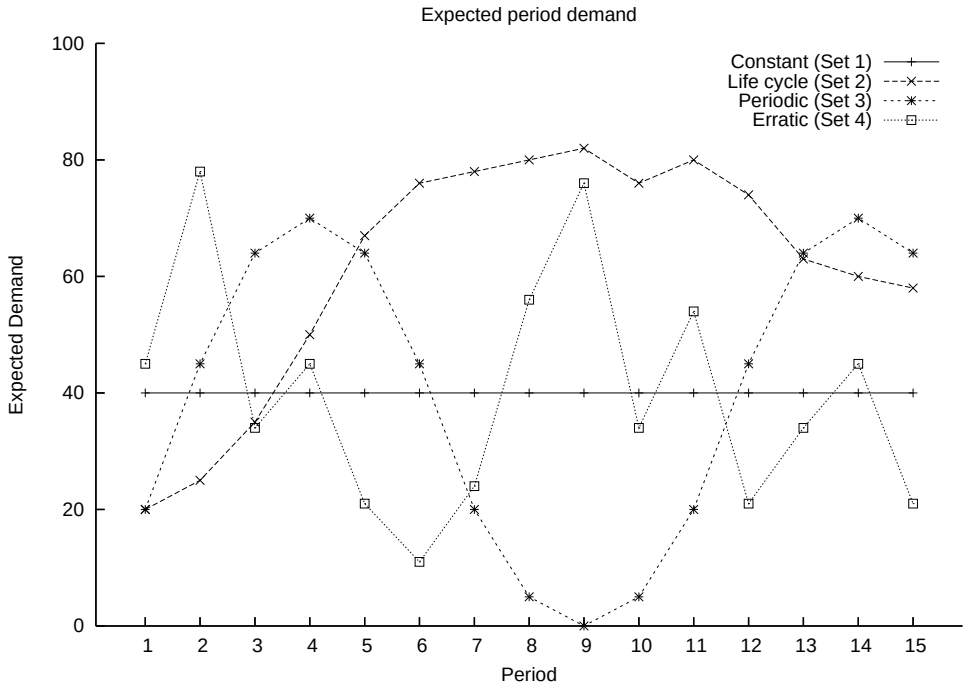

Figure 2: The mean demand patterns over time as in Berry [7].

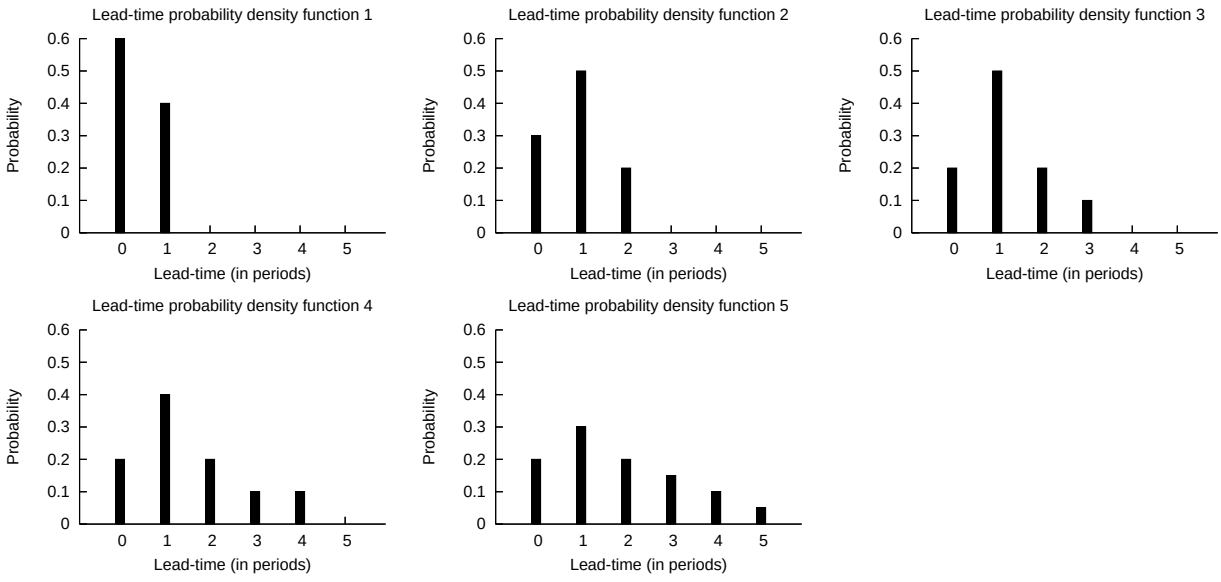

Figure 3: The lead time probability density functions.

	Cost Overhead			
Instances	H1	H2	H1 $\bigoplus$ H2	Exact - 10 secs
2.7%	0	0	0	0
25%	$<$ 0.26%	0	0	0
50%	$<$ 3.1%	$<$ 0.54%	$<$ 0.29%	0
80%	$<$ 7%	$<$ 1%	$<$ 0.75%	$<$ 8.5%
90%	$<$ 10%	$<$ 2%	$<$ 1%	$<$ 16%
100%	$<$ 22%	$<$ 8.5%	$<$ 4.27%	$<$ 45.7%

Table 7: Statistics on the cost overhead, over a 6-period planning horizon, incurred by Heuristic I (H1), Heuristic II (H2), a strategy that combines H1 and H2 (H1 $\bigoplus$ H2), and a strategy that limits to 10 seconds the run time of the exact approach (Exact - 10 secs). The cost overhead is expressed in percentage of the optimal policy cost.

time probability density functions (LT) 1, 2, and 3. By varying the model parameters (a, c_v, α, etc.) as discussed in the previous paragraphs, we obtain a total of 144 instances.

The results obtained by running the exact approach in Rossi et al. [33] over the test bed are shown in Table 9 and Table 10. The exact approach often required a significant amount of time in order to solve these instances to optimality. More precisely, in the worst case it ran for 105 hours before finding the optimal solution. On an average, the optimal solution was found in 2 hours. For half of the instances, the optimal solution was found in less than 45 seconds; approximately 80% of the instances required less than 30 minutes to be solved; and about 90% of the instances required less than 2 hours.

In Table 11, we compare the cost of the solutions found by the two heuristics to the optimal ones. For convenience, results are also summarized in Table 7. Heuristic I found the optimal solution only for 2.7% of the instances, while Heuristic II succeeded in finding the optimal solution for about 25% of the instances. For half of the instances, the cost overhead incurred by Heuristic I was below 3.1%, while that incurred by Heuristic II was significantly lower, being only 0.54%. In Table 7 we provide further statistics, namely, the 80% and the 90% percentile. In the worst case, Heuristic I produced a solution 22% costlier than the optimal one, while Heuristic II produced a solution that is 8.5% more costly. On an average, the solution found by Heuristic I was 3.8% expensive than the optimal one, while Heuristic II produced a solution that was 0.56% more expensive than the optimal. It should be noted that, since none of the two heuristics fully dominates the other in terms of solution quality, if they are used in conjunction (Table 7, column "H1 $\bigoplus$ H2") the overall performance improves. In fact, the average cost overhead decreases to 0.48%, the maximum cost overhead is halved

to 4.27%, and also the other quantiles, as shown in Table 7, significantly improve.

In Table 12, we present the run-times for the two heuristics. It is easy to observe that for all the instances Heuristic II was faster than Heuristic I. Nevertheless, both the heuristics did not require more than a few seconds to solve any of the instances. More precisely, the maximum run time observed is 8.5 seconds.

It might be argued that the exact approach in Rossi et al. [33] may converge quickly to good solutions and therefore be used as a heuristic approach by simply limiting the run time and collecting the best solution found within the given time limit. Since the maximum run time observed for our heuristic methods over the given test bed is 8.5 seconds, we allocated a run time of 10 seconds to the exact approach and observed the cost difference between the optimal solution produced by the exact approach without a time limit and the best solution that this approach could find in the given time limit of 10 seconds. In Table 13 we present cost differences, in percentage of optimal costs. For convenience, we also summarized the results in Table 7, column "Exact - 10 secs". The exact approach with a 10 seconds limit was able to reach the optimal solution for more than 50% of the instances. Nevertheless, the performance of this heuristic strategy quickly deteriorates when we consider the 30 most difficult instances. For 20% of the instances, the error exceeded 8.5%. For 10% of the instances, the error exceeded 16%. In the worst case, the error reached 45.7%. On an average, the solution found by this strategy was 4.8% more expensive than the optimal one. As shown, both Heuristic I and II produced better average and worst case results. In light of these results, it is clear that imposing a time limit to the exact approach in Rossi et al. [33] does not constitute a viable heuristic strategy. It should also be noted that longer planning horizons exacerbate the poor performance of this heuristic.

In order to assess the scalability of the two heuristics, we now consider the full demand patterns in Fig. 2, therefore we now run tests over a 15-period planning horizon. In addition, we also consider all the 5 possible discrete probability density functions shown in Fig. 3, therefore LT ranges now in $\{1, \ldots, 5\}$, and we vary the remaining model parameters (a, h, c_v and α) as discussed before. By doing so, we obtain a total of 240 instances. In Table 14 and Table 15 we compare, respectively, the run times and the cost of the solutions found by the two heuristics for this new set of instances. Since these instances are intractable for the exact approach, we will only compare the two heuristics among each other.

Over the 240 instances considered, Heuristic I produced the best solution only 25% of the times. In all the other cases, Heuristic II found a better solution (75% of the instances).

	Run time	
Instances	H1	H2
25%	0	0
50%	< 25 seconds	< 11 minutes
80%	< 40 seconds	< 30 minutes
90%	< 1 minute	< 35 minutes
100%	< 3 minutes	< 1 hour and 10 minutes

Table 8: Statistics on the run times incurred by Heuristic I (H1) and Heuristic II (H2), over a 15-period planning horizon.

In the worst case, Heuristic I (Heuristic II) produced a solution that was 14.6% (3.5%) more expensive than the one obtained using Heuristic II (Heuristic I). On an average, for the instances in which Heuristic I (Heuristic II) could not produce the best solution, the solution produced was 3.66% (0.80%) more expensive than that produced by Heuristic II (Heuristic I).

From this comparison, and from the previous discussion, it is possible to observe that Heuristic II typically performs better than Heuristic I in terms of quality of the solution produced. Nevertheless, in terms of run times (Table 8), the picture is different. In fact, Heuristic I maintains good performances over all the instances in the 15-period planning horizon test bed. More specifically, the average run time for Heuristic I was 30 seconds, in contrast to an average run time of about 16 minutes for Heuristic II. In the worst case, Heuristic I took less than 3 minutes to complete the search, while Heuristic II completed the search in 1 hour and 10 minutes. About 50% of the instances could be solved by Heuristic I in less than 25 seconds, in contrast to the 11 minutes required by Heuristic II. Furthermore, we observed that 80% of the instances required less than 40 seconds in order to be solved by using Heuristic I, in contrast to the 30 minutes required by Heuristic II. Finally, the 90% quantile was less than a minute for Heuristic I and about 35 minutes for Heuristic II.

8. Summary and Conclusions

We addressed the computation of near-optimal replenishment cycle policy parameters under non-stationary stochastic demand, stochastic supplier lead time and service-level constraints. Two hybrid heuristic algorithms that blend Constraint Programming and Local Search were proposed.

Firstly, we compared these two heuristics for small instances against an exact method.

In our experiments, Heuristic I was within 3.8% of the optimal, while Heuristic II was within 0.56% of the optimal. In addition, when used in conjunction, the two heuristics were within 0.48% of the optimal. In terms of computational time, the average run time for Heuristic I, Heuristic II and the Optimal solution is 2.96 seconds, 0.75 seconds and 2 hours respectively. The results proved that Heuristic II typically performs better than Heuristic I in terms of quality of the solutions produced, by achieving a very little cost overhead.

Secondly, the two heuristics were tested and compared with each other, in terms of both solution quality and run time over a set of larger instances that are intractable for the exact approach. In terms of solution quality, Heuristic II performed better and was on average 3% better than the performance of Heuristic I, while the average run times for Heuristic I and Heuristic II were 30 seconds and 16 minutes respectively. The results confirmed that Heuristic I typically produces lower quality solutions than Heuristic II, but also that Heuristic I runs much faster than Heuristic II.

We can conclude that, if run time is a critical aspect, Heuristic I may be a viable choice, while if the quality of the solution produced is a major concern, then Heuristic II (or a combination of the two heuristics) should be chosen, as it achieves high quality solutions and is orders of magnitude faster than the exact approach.

Acknowledgments

We thank the area editor for constraint programming and optimization, Dr. Michela Milano, the associate editor, the two anonymous referees, and editorial assistant Kelly Kophazi, whose feedback helped us to improve the contents and presentation of the paper. The author, Roberto Rossi, has received funding from the European Communitys Seventh Framework Programme (FP7) under grant agreement no 244994 (project VEG-i-TRADE). The author, S. Armagan Tarim, is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under the Support Programme-1001 and Hacettepe University BAB. On a personal front, the author, Ramesh Bollapragada, thanks his parents Rajarao Bollapragada and Mangatayaru Dulla Bollapragada, who were a great source of inspiration in his entire academic career.

References

[1] 2007. *OPL Studio 3.7 Users Manual*. ILOG, Inc., Incline Village, NV.

[2] Apt, K. 2003. *Principles of Constraint Programming*. Cambridge University Press, Cambridge, UK.

[3] Axsater, S. 2006. *Inventory Control (Second Edition)*. Springer's International Series.

[4] Babaï, M. Z., A. Syntetos, Y. Z. Dallery, K. Nikolopoulos. 2009. Dynamic re-order point inventory control with lead-time uncertainty: analysis and empirical investigation. *International Journal of Production Research* **47** 2461–2483.

[5] Backer, B. De, V. Furnon, P. Shaw, P. Kilby, P. Prosser. 2000. Solving vehicle routing problems using constraint programming and metaheuristics. *Journal of Heuristics* **6** 501–523.

[6] Bashyam, S., M.C. Fu. 1998. Optimization of (s,S) inventory systems with random lead times and a service level constraint. *Management Science* **44** 243–256.

[7] Berry, W. L. 1972. Lot sizing procedures for requirements planning systems: A framework for analysis. *Production and Inventory Management Journal* **13** 19–34.

[8] Birge, J. R., F. Louveaux. 1997. *Introduction to Stochastic Programming*. Springer Verlag, New York.

[9] Blum, C., A. Roli. 2003. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.* **35** 268–308.

[10] Bookbinder, J. H., J. Y. Tan. 1988. Strategies for the probabilistic lot-sizing problem with service-level constraints. *Management Science* **34** 1096–1108.

[11] Cesta, A., G. Cortellessa, A. Oddi, N. Policella, A. Susi. 2001. A constraint-based architecture for flexible support to activity scheduling. *AI*IA 01: Proceedings of the 7th Congress of the Italian Association for Artificial Intelligence on Advances in Artificial Intelligence*. Springer-Verlag, London, UK, 369–381.

[12] Charnes, A., W. W. Cooper. 1959. Chance-constrainted programming. *Management Science* **6** 73–79.

[13] de Kok, A. G. 1991. Basics of inventory management: part 2 The (R,S)-model. Research memorandum, FEW 521. Department of Economics, Tilburg University, Tilburg, The Netherlands.

[14] de Kok, T., K. Inderfurth. 1997. Nervousness in inventory management: Comparison of basic control rules. *European Journal of Operational Research* **103** 55–82.

[15] Eppen, G. D., R. K. Martin. 1988. Determining safety stock in the presence of stochastic lead time and demand. *Management Science* **34** 1380–1390.

[16] Focacci, F., F. Laburthe, A. Lodi. 2002. Local Search and Constraint Programming. *In F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics, volume 57 of International Series in Operations Research and Management Science. Kluwer Academic Publishers, Norwell, MA* .

[17] Focacci, F., A. Lodi, M. Milano. 1999. Cost-based domain filtering. *Proceedings of the 5th International Conference on the Principles and Practice of Constraint Programming*. Springer Verlag, 189–203. Lecture Notes in Computer Science No. 1713.

[18] Hadley, G., T. M. Whitin. 1964. *Analysis of Inventory Systems*. Prentice Hall.

[19] Hayya, J. C., U. Bagchi, J. G. Kim, D. Sun. 2008. On static stochastic order crossover. *International Journal of Production Economics* **114** 404–413.

[20] Hayya, J. C., S. H. Xu, R. V. Ramasesh, X. X. He. 1995. Order crossover in inventory systems. *Stochastic Models* **11** 279–309.

[21] Heisig, Gerald. 2002. *Planning Stability in Material Requirements Planning Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[22] Hunt, J. A. 1965. Balancing accuracy and simplicity in determining reorder points. *Management Science* **12** B94–B103.

[23] Kaplan, R. S. 1970. A dynamic inventory model with stochastic lead times. *Management Science* **16** 491–507.

[24] Laburthe, F., the OCRE project team. 1994. Choco: Implementing a cp kernel. Tech. rep., Bouygues e-Lab, France.

[25] Nevison, C., M. Burstein. 1984. The dynamic lot-size model with stochastic lead-times. *Management Science* **30** 100–109.

[26] Nocedal, J., S. J. Wright. 1999. *Numerical Optimization*. Springer.

[27] Pesant, G., M. Gendreau. 1996. A view of local search in constraint programming. Eugene C. Freuder, ed., *Proceedings of the Second International Conference on Principles and Practice of Constraint Programming, Cambridge, Massachusetts, USA, August 19-22, 1996*, *Lecture Notes in Computer Science*, vol. 1118. Springer, 353–366.

[28] Régin, J.-C. 1994. A filtering algorithm for constraints of difference in csps. *Proceedings of the twelfth national conference on Artificial intelligence (vol. 1), Seattle, Washigton*. American Association for Artificial Intelligence, 362–367.

[29] Régin, J.-C. 2003. *Global Constraints and Filtering Algorithms*. in Constraints and Integer Programming Combined, Kluwer, M. Milano editor.

[30] Riezebos, J. 2006. Inventory order crossovers. *International Journal of Production Economics* **104** 666–675.

[31] Rossi, F., P. van Beek, T. Walsh. 2006. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA.

[32] Rossi, R., S. A. Tarim, B. Hnich, S. Prestwich. 2008. A global chance-constraint for stochastic inventory systems under service level constraints. *Constraints* **13** 490–517.

[33] Rossi, R., S. A. Tarim, B. Hnich, S. Prestwich. 2010. Computing the non-stationary replenishment cycle inventory policy under stochastic supplier lead-times. *International Journal of Production Economics* **127** 180–189.

[34] Rossi, R., S. A. Tarim, B. Hnich, S. Prestwich. 2010. A state space augmentation algorithm for the replenishment cycle inventory policy. *International Journal of Production Economics* doi:10.1016/j.ijpe.2010.04.017. Forthcoming.

[35] Silver, E. A., D. F. Pyke, R. Peterson. 1998. *Inventory Management and Production Planning and Scheduling*. John-Wiley and Sons, New York.

[36] Speh, T. W., G. Wagenheim. 1978. Demand and lead-time uncertainty: The impacts of physical distribution performance and management. *Journal of Business Logistics* **1** 95–113.

[37] Tang, C. S. 2006. Perpectives in supply chain risk management. *International Journal of Production Economics* **103** 451–488.

[38] Tarim, S. A. 1996. Dynamic lotsizing models for stochastic demand in single and multi-echelon inventory systems. Ph.D. thesis, Lancaster University.

[39] Tarim, S. A., B. Hnich, R. Rossi, S. Prestwich. 2009. Cost-based filtering techniques for stochastic inventory control under service level constraints. *Constraints* **14** 137–176.

[40] Tarim, S. A., B. G. Kingsman. 2004. The stochastic dynamic production/inventory lot-sizing problem with service-level constraints. *International Journal of Production Economics* **88** 105–119.

[41] Tarim, S. A., B. G. Kingsman. 2006. Modelling and Computing (R^n,S^n) Policies for Inventory Systems with Non-Stationary Stochastic Demand. *European Journal of Operational Research* **174** 581–599.

[42] Tarim, S. A., S. Manandhar, T. Walsh. 2006. Stochastic constraint programming: A scenario-based approach. *Constraints* **11** 53–80.

[43] Tarim, S. A., B. Smith. 2008. Constraint Programming for Computing Non-Stationary (R,S) Inventory Policies. *European Journal of Operational Research* **189** 1004–1021.

[44] Tempelmeier, H. 2007. On the stochastic uncapacitated dynamic single-item lotsizing problem with service level constraints. *European Journal of Operational Research* **181** 184–194.

[45] Whybark, D. C., J. G. Williams. 1976. Material requirements planning under uncertainty. *Decision Science* **7** 595–606.

[46] Zipkin, P. 1986. Stochastic leadtimes in continuous-time inventory models. *Naval Research Logistics Quarterly* **33** 763–774.

		$a = 100$				$a = 175$				$a = 250$			
		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$	
Set	LT	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$
1	1	620	682	640	728	770	833	797	893	920	983	947	1043
	2	677	764	686	826	834	914	836	976	970	1042	986	1126
	3	703	820	738	869	853	967	888	1020	970	1042	1030	1138
2	1	715	783	745	847	895	973	937	1051	1045	1123	1087	1201
	2	803	961	838	1028	1028	1131	1057	1215	1187	1281	1207	1365
	3	903	1025	953	1103	1057	1243	1105	1317	1207	1383	1255	1467
3	1	772	854	798	924	966	1052	1016	1148	1116	1202	1166	1298
	2	836	1032	886	1098	1061	1182	1100	1280	1214	1332	1250	1430
	3	889	1087	950	1162	1065	1278	1137	1362	1215	1378	1287	1504
4	1	530	590	560	650	680	740	710	800	808	886	860	950
	2	576	662	588	716	726	811	738	866	808	886	874	994
	3	545	678	600	734	695	811	750	884	808	886	874	994

Table 9: Exact approach, optimal policy costs, 6-period planning horizon.

		$a = 100$				$a = 175$				$a = 250$			
		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$	
Set	LT	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$
1	1	20	46	15	47	0.98	2.5	0.86	3.7	0.39	0.72	0.52	1.2
	2	210	582	142	793	4.4	16	3.2	24	0.66	1.9	0.80	4.5
	3	3188	15570	4258	18496	27	243	35	318	0.66	2.8	2.0	19
2	1	210	583	212	934	7.2	20	9.4	37	1.1	2.9	1.4	4.8
	2	1611	14409	1993	23781	150	521	148	935	16	43	13	76
	3	46082	88934	62789	132963	794	16092	1224	24845	49	537	82	943
3	1	571	1800	436	2413	28	69	39	153	3.3	8.3	4.3	16
	2	1866	16488	2486	20712	340	1176	379	2016	17	115	14	234
	3	70805	216835	104318	380934	2630	49724	6767	79378	110	1261	208	5472
4	1	3.6	6.1	3.1	9.5	0.42	1.2	0.69	1.9	0.19	0.53	0.44	1.1
	2	52	211	32	292	1.4	12	1.2	15	0.19	0.53	0.49	1.4
	3	90	2684	304	4139	2.0	40	7.7	90	0.20	0.53	0.47	1.4

Table 10: Exact approach, run times (secs), 6-period planning horizon.

		$a = 100$								$a = 175$								$a = 250$							
		$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$			
		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$	
Set	LT	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2
1	1	3.4	0.48	0.44	0	3.1	0.63	1.4	0.55	3.4	0.39	0.12	0.36	4.6	0.38	0.11	0.67	2.7	0.33	0.10	0.31	4	0.32	0	0.58
	2	5.0	0.44	0.52	0.79	5.4	0.87	0.85	0.97	3.2	0.48	0.44	0.66	4.4	0.72	0.82	0.82	4.2	0	0.19	0	4	0.61	0	0.71
	3	22	0	0.12	3.3	8.0	0.54	0.12	4.1	5.0	0	0.21	0	4.6	0.45	0.10	4.2	7.6	0	0.19	0	5	0	0.09	0
2	1	4.2	0.56	0.77	0.51	6.7	0.27	2.7	0.71	4.9	0.67	0.31	0.31	7.5	0.64	0.10	0.29	4.2	0.57	0.27	0.27	6	0.55	0.08	0.25
	2	14	0.50	1.2	0.94	14	1.1	1.6	0.58	5.6	0.39	0.27	0.18	6.0	0.85	0.25	0.49	4.1	0.17	0.23	0.16	5	0.75	0.22	0.44
	3	2.3	0	6.9	0.68	4.0	0.10	7.7	1.1	13	0	0.16	0.80	7.1	0	0	1.2	4.8	0	0.14	0	6	0	0.07	1.1
3	1	2.8	0.78	0.35	0.70	3.9	0.75	0.22	0.65	5.1	0.62	1.2	0.57	3.6	0.30	0.26	0.52	4.4	0.54	0.17	0.50	5	0.26	0.23	0.46
	2	15	2.0	0.29	0.78	8.9	2.6	2.3	0.82	6.4	1.0	0.25	0.68	4.7	0.91	0.47	0.31	5.4	0.66	0.23	0.60	4	0.80	0.28	0.28
	3	13	4.3	4.0	6.0	10	2.4	4.5	6.3	8.8	3.5	0.16	1.9	8.9	0	0.22	1.8	7.8	3.0	0.15	0	7	0	0.07	0
4	1	5.7	0.38	0.34	1.0	9.3	1.1	1.2	1.2	4.4	0.29	0.27	0.81	6.6	0.85	0.12	1.0	6.4	0	0.56	0	5	0.70	0.11	0.84
	2	5.2	0.69	0.76	1.2	9.4	1.0	0.42	1.1	4.1	0.55	0.49	0	6.1	0.81	0.35	0.92	10	0	0.45	0	7	0	0.10	0
	3	21	0	1.2	8.6	9.5	0	0.82	8.2	10	0	0.37	0	10	0	0.68	4.0	10	0	0.34	0	12	0	0.10	0

Table 11: Additional cost, in % of the cost of the optimal policy, incurred if Heuristic I (H1) or Heuristic II (H2) are used, 6-period planning horizon.

		$a = 100$								$a = 175$								$a = 250$							
		$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$			
		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$	
Set	LT	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2
1	1	3.5	0.72	2.4	0.16	1.2	0.14	2.8	0.16	1.2	0.11	1.7	0.13	0.88	0.34	1.8	0.20	1.3	0.39	1.2	0.36	0.91	0.69	1.4	0.14
	2	3.9	0.72	3.5	0.33	3.0	0.56	5.2	0.41	1.8	0.19	2.4	0.27	2.3	0.45	1.5	0.27	1.6	0.30	1.8	0.70	2.0	1.0	1.8	0.67
	3	3.4	0.36	4.4	1.3	3.5	1.2	2.2	1.1	2.3	0.27	2.2	1.8	2.8	0.49	2.1	1.27	1.5	0.70	1.4	0.69	1.3	0.77	1.2	1.5
2	1	2.8	0.45	3.2	0.17	2.4	0.20	3.2	0.20	1.8	0.17	2.4	0.55	2.4	0.17	2.6	0.20	1.8	0.52	2.1	0.44	2.0	0.16	2.5	0.50
	2	3.8	0.89	2.9	0.66	4.7	0.77	3.2	1.5	3.5	0.70	3.4	0.95	3.6	0.38	3.4	1.1	2.2	0.83	2.2	0.94	2.3	0.39	2.6	1.0
	3	3.3	0.83	3.7	2.0	4.4	1.4	4.3	1.7	3.1	1.5	2.7	2.5	3.8	1.5	3.4	1.6	3.0	0.97	2.0	0.88	2.7	1.5	3.1	1.3
3	1	2.7	0.53	3.8	0.78	2.4	0.16	4.4	0.20	3.4	1.2	3.7	0.58	1.5	0.55	3.6	0.70	2.3	0.41	2.0	0.31	2.4	0.47	4.6	0.34
	2	4.3	0.80	6.4	1.7	3.2	0.88	5.7	1.0	3.8	1.1	4.3	0.67	2.9	0.78	3.9	1.2	4.1	0.69	2.5	0.91	3.0	0.61	4.1	0.94
	3	7.7	1.1	5.5	2.4	6.3	1.4	5.5	3.4	4.0	2.6	4.5	1.6	4.8	1.4	3.5	1.1	3.7	1.2	2.1	1.3	4.0	0.36	5.6	1.3
4	1	2.5	0.33	2.6	0.39	2.8	0.33	3.0	0.38	2.4	0.16	2.1	0.31	1.3	0.23	2.7	0.39	1.3	0.31	1.4	0.34	0.7	0.09	1.7	0.30
	2	3.2	0.59	4.0	0.74	3.9	0.70	3.0	0.58	2.7	0.16	2.9	0.50	1.7	0.72	3.9	0.70	2.0	0.50	2.1	0.55	2.1	0.36	2.3	0.17
	3	3.8	0.49	8.5	0.61	3.2	0.63	4.8	2.3	2.6	0.16	2.7	1.1	4.0	0.55	2.9	1.2	1.4	0.16	1.3	0.78	1.8	0.45	3.3	0.47

Table 12: Heuristic I (H1) and Heuristic II (H2) run times (secs), 6-period planning horizon.

		$a = 100$				$a = 175$				$a = 250$			
		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$		$c_v = 0.2$		$c_v = 0.3$	
Set	LT	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$	$\alpha = 0.85$	$\alpha = 0.95$
1	1	0	0	0	2.0	0	0	0	0	0	0	0	0
	2	1	2.3	0	0.85	0	0	0	0	0	0	0	0
	3	12	8.7	6.7	10	0.59	0	0	4.2	0	0	0	0
2	1	4.2	5.1	5.6	6.3	0	0	0	0	0	0	0	0
	2	42	28	45	31	1.3	15	22	17	0	2.4	0	1.6
	3	26	20	28	22	15	5.2	17	8.4	7.1	0	9.2	2.4
3	1	5.7	7	8.5	8	0	1.1	0	0	0	0	0	0
	2	12	17	7	17	2.8	3.9	0	11	0	3.4	0	1.6
	3	25	12	24	14	12	1.9	12	4.9	2.5	0	5.6	0
4	1	0	0	0	0	0	0	0	0	0	0	0	0
	2	0	0	0	2.9	0	0	0	0	0	0	0	0
	3	11	0	5	4.5	0	0	0	0	0	0	0	0

Table 13: Exact approach with a run time limited to 10 seconds. Additional cost, in % of the cost of the optimal policy.

		$a = 100$								$a = 175$								$a = 250$							
		$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$			
		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$	
Set	LT	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2
1	1	1.5	0	0.5	0	1.7	0	0	0.21	0.68	0	0	0.45	1.2	0	0	0.59	1.4	0	0	0.50	1.7	0	0	0.54
	2	2.0	0	0	0.18	1.1	0	0.04	0	1.3	0	0	0.84	1.5	0	0	0.91	1.5	0	0	0.37	2.2	0	0	0.85
	3	1.6	0	1.4	0	2.7	0	0	0.12	1.5	0	0	0	0.76	0	0	0.71	2.2	0	0	0.93	1.4	0	0	0.97
	4	9.8	0	3.4	0	11	0	2.6	0	11	0	1.0	0	5.0	0	2.3	0	2.6	0	1.3	0	1.6	0	0.33	0
	5	18	0	6.1	0	5.1	0	2.1	0	16	0	2.4	0	9.9	0	0	0.94	8.9	0	1.3	0	9.6	0	0.53	0
2	1	0.9	0	0	0.04	1.6	0	0	0	1.3	0	0	0.43	0.82	0	0	0.5	0.88	0	0	0.09	1.1	0	0	0.49
	2	6.2	0	0.5	0	4.0	0	0.9	0	3	0	0	0.48	2.4	0	0	0	3.0	0	0	0.55	0.77	0	0	0.58
	3	5.4	0	4.3	0	1.0	0	1.1	0	1.0	0	2.5	0	2.0	0	0.65	0	1.9	0	2.1	0	1.5	0	0.44	0
	4	2.9	0	5.7	0	1.7	0	2.5	0	3.5	0	2.9	0	3.2	0	3.5	0	2.5	0	2.6	0	3.0	0	2.88	0
	5	9.4	0	6.9	0	1.1	0	0	3.3	14	0	0	0.29	0.09	0	0	0.19	8.1	0	1.6	0	3.7	0	0	0.84
3	1	3.0	0	0	0.44	4.0	0	0	0	2.4	0	0	0	4.8	0	0.25	0	4.5	0	0	0.51	5.3	0	0	0.73
	2	8.6	0	0.5	0	5.5	0	0.42	0	6.0	0	0	0.54	2.4	0	0	0	3.6	0	0	0.48	4.3	0	0.32	0
	3	8.1	0	0	1.4	8.5	0	0	0	6.2	0	0.45	0	6.1	0	0	3.5	5.2	0	1.7	0	5.9	0	0	3.4
	4	15	0	3.3	0	11	0	0	0	6.0	0	2.4	0	9.6	0	0	0.32	5.3	0	2.2	0	9.5	0	0	0.27
	5	6.4	0	0.8	0	1.8	0	0	0	14	0	1.6	0	4.3	0	0	2.8	7.5	0	1.5	0	4.1	0	0	1.4
4	1	2.7	0	0.0	0.65	3.5	0	0	0.82	0.82	0	0	0.47	2.1	0	0	0.71	2.4	0	0	0.20	3.0	0	0	0.61
	2	2.2	0	1.9	0	3.7	0	0.55	0	2.6	0	0	0.70	3.1	0	0	0.86	2.3	0	0	0.44	2.7	0	0	0.81
	3	2.2	0	0.3	0	3.7	0	1.7	0	2.9	0	0	1.3	3.7	0	0	0.65	3.2	0	0	0.88	3.8	0	0	0.58
	4	14	0	7.0	0	8.0	0	3.9	0	5.0	0	6.2	0	1.5	0	5.3	0	5.5	0	3.0	0	3.1	0	2.85	0
	5	8.7	0	4.6	0	11	0	3.0	0	5.6	0	3.2	0	6.1	0	0.64	0	1.2	0	0	0.03	2.8	0	0	0.03

Table 14: Cost comparison between Heuristic I (H1) and Heuristic II (H2). A value of 0 is associated with the heuristic that has found the best solution, the other value denotes the cost difference — in percentage of the best solution — achieved by the other heuristic, 15-period planning horizon.

		$a = 100$								$a = 175$								$a = 250$							
		$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$				$c_v = 0.2$				$c_v = 0.3$			
		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$		$\alpha = 0.85$		$\alpha = 0.95$	
Set	LT	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2	H1	H2
1	1	24	161	17	224	18	195	15	218	12	150	16	196	10	127	18	194	21	69	21	102	14	60	21	104
	2	26	446	33	420	14	311	28	404	23	327	30	421	12	275	27	405	28	210	36	360	19	153	30	355
	3	26	602	33	763	24	568	43	741	29	588	36	762	19	560	36	742	40	437	49	730	28	397	46	702
	4	36	1096	53	1294	46	1088	51	1516	39	1098	43	1296	19	1089	41	1439	43	979	56	1291	38	935	46	1431
	5	48	1874	48	2178	37	1902	66	2391	30	1870	42	2190	31	1909	47	2391	54	1646	50	2191	50	1678	44	2402
2	1	11	295	16	328	14	281	15	327	17	299	15	331	10	282	16	330	12	255	23	311	12	231	21	310
	2	25	520	29	669	18	472	28	648	25	524	35	674	18	480	29	653	24	523	29	673	16	473	37	651
	3	28	959	35	1345	24	924	51	1219	23	961	35	1272	25	930	35	1220	26	961	40	1272	25	925	53	1217
	4	32	1837	99	2105	35	1826	27	2443	36	1836	59	2099	32	1825	38	2449	31	1843	71	2103	36	1825	84	2447
	5	37	3129	48	3470	46	3223	36	4206	38	3122	46	3479	38	3207	73	4221	30	3123	66	3475	33	3213	67	4267
3	1	21	196	21	218	18	189	21	220	13	120	21	172	12	122	19	194	12	43	18	80	6	50	17	110
	2	35	359	32	442	27	342	39	440	28	323	26	438	19	318	25	439	20	178	24	308	17	165	22	357
	3	33	629	33	793	36	605	38	811	25	587	34	792	25	576	26	809	22	360	30	689	22	366	24	761
	4	36	1219	42	1303	32	1202	29	1549	24	1207	30	1304	32	1190	31	1551	21	883	20	1283	27	853	28	1519
	5	30	1800	31	2076	30	1915	34	2422	26	1794	20	2070	28	1906	23	2421	19	1559	17	2044	22	1697	19	2418
4	1	11	200	12	219	9	182	12	217	12	98	22	141	9	75	17	137	15	39	17	64	4	29	16	61
	2	18	343	22	400	17	322	27	387	26	313	29	401	21	283	26	385	20	153	24	276	18	128	22	270
	3	13	547	27	722	21	530	39	706	19	515	29	721	26	493	30	705	21	290	29	616	21	278	27	605
	4	65	1018	153	1347	23	992	65	1353	30	1016	46	1345	32	1045	48	1351	33	784	35	1289	25	706	39	1288
	5	25	1709	40	2067	22	1720	52	2126	139	1702	46	2069	47	1717	37	2124	28	1465	41	2055	36	1471	53	2121

Table 15: Heuristic I (H1) and Heuristic II (H2) run times (secs), 15-period planning horizon.