Calculating safety stocks for assembly systems with random component procurement lead times: A branch and bound algorithm

Mohamed-Aly Ould Louly, Alexandre Dolgui

A R T I C L E   I N F O

Article history:
Received 1 May 2005
Accepted 1 November 2007
Available online 17 June 2008

Keywords:
Supply chain management
Inventory
Uncertainty modelling
Combinatorial optimization
Branch and bound

A B S T R A C T

In this paper, a discrete single-level multi-component inventory control model for assembly systems with random component procurement lead times is considered. The economic order quantity (EOQ) policy is used for a type of finished product. The requirements of the components are constant and cyclic (periodic), and their values per period are deduced from the EOQ for the finished product. The paper focuses on the components safety stock calculation. The objective is to minimise the average holding cost of the components while keeping the desired service level for the finished product. For this, an upper bound, two lower bounds, two dominance properties and an efficient branch and bound algorithm are suggested. Several tests are executed and conclusions are drawn. The proposed model provides a substantial saving for assembly systems with a large number and unreliable delivery of components as in semi-conductor and automotive industries.

1. Introduction

One of the well-known and widely used inventory control models is the Economic Ordering Quantity (EOQ), or Wilson model (Zipkin, 2000). The EOQ is a continuous time model for a single-item inventory system with a known demand rate, fixed ordering and linear holding costs, no shortage and zero order lead time.

To use the EOQ model in the case of random demand and/or lead time, safety stocks are necessary. There is extensive literature concerning safety stock calculation for random demand of finished products (see e.g., Porteus, 1990; Lee and Nahmias, 1993; Nahmias, 1997; Alström, 2001). In the context of assembly systems, the problem of determining safety stocks for components has not been given much attention in the literature, especially for assembly systems with random component procurement lead times (Porteus, 1990; Lee and Nahmias, 1993). That is the motivation of this paper.

Here, we consider a single-level assembly system with several different types of components. These components are ordered from external suppliers. The lead times for the receipt of orders are independent random variables with known probability distributions. The EOQ policy for the inventory control of the finished product requires constant needs of components per period. The random component lead times generate surplus component stocks or shortages. The component safety stocks are needed to avoid the shortages while minimizing the surplus. With the former there are investment costs incurred and with the latter production bottlenecks. Therefore, the component safety stock calculation is a crucial issue.

In assembly systems, the difficulty of the safety stock calculation for components inventories resides in the interdependence between inventories of different components. Indeed, many types of components are needed to produce one finished product; therefore, the inventories of the different types of components become dependent. When there is a delay and stockout of only one component, automatically, there is no possibility to assemble the finished product. In addition, there will be overstocking of the other types of components, delivered but not used. In just in time (JIT) environment considered in this paper, i.e., there is no stockpiling of finished products, this leads to dissatisfaction of the customer and payment delays.

This paper develops a novel approach for components safety stock calculation for one level JIT assembly systems where component lead times are random. This approach is based on original lower bound and dominance properties and a branch and bound algorithm. Our objective is to determinate the optimal value of safety stock for each type of component minimizing average holding cost, while meeting a given service level. The service level is the percent of satisfied customers for the finished product.

The rest of the paper is organized as follows. Section 2 presents a review of the related literature. Section 3 deals with the optimization model proposed. Section 4 develops two lower bounds, two dominance properties and a new branch and bound algorithm. Computational tests are reported in Section 5, and finally, some concluding remarks are given in Section 6.
2. Literature review

The problem of component delivery for assembly systems is often formulated as a Mixed Integer Program (MIP) with branch and bound algorithms or heuristics approaches used to solve it (Clark and Armentano, 1995a,b). These MIP models assume deterministic component lead times. However, in practice, the lead times include setup, production, transport and waiting times. Several random factors exist for each of these activities, thus it is more realistic to use stochastic models.

The literature is rich in publications where stochastic models consider a random demand of finished product (Rao and Schneller, 1990; Khang and Fujiwara, 1993; Sox and Muckstadt, 1996). In this case, safety stocks can be determined, for example, using different generalizations of the Newsboy model (Porteus, 1990; Nandakumar and Morton, 1993; Petruzzi and Dada, 1999).

Stochastic lead time models are less developed and more complex. For single item problems, Kaplan (1970) suggests a finite horizon dynamic programming model whose optimal inventory policy turns out to depend on whether the ordering cost is fixed. Liberatore (1979) tries to extend directly the EOQ model so as to treat stochastic lead time, but no closed-form solution was given for the optimal order size. Bookbinder and Cakanyildirim (1999) deal with the situation where lead time is random and the demand rate is assumed to be constant and known. The author’s motivation is to help an inventory manager of a JIT system who could invest in decreasing the lead time in a stochastic–order sense. They use the \((Q, r)\) approach and compare their model with the classical \((Q, r)\) model (stochastic demand, fixed lead time) and the EOQ (deterministic demand).

Ramasesh et al. (1991) study dual sourcing (simultaneous procurement from two suppliers) in the context of the \((s, Q)\) policy with constant demand and stochastic lead time and compare it with single sourcing. The authors consider one type of raw material and only the case where the distributions of lead times are uniform or exponential. Sculli and Wu (1981) study the same problem of one type of raw material ordered from two vendors whose supply lead times are normally distributed but with different parameters. Kelle and Silver (1990) study the reduction in the safety stock through order splitting when supply lead time follows a Weibull distribution. The following advantages of \(n\)-supplier systems compared with a single-supplier are demonstrated: for a given safety stock a higher service level can be achieved, and for a given service level a lower safety stock is needed. However, these studies do not address the problem of determining the optimal order quantity and order splitting strategy.

Another area of research concerns the inventory control of assembly systems. Kumar (1989) deals with one period model (a single assembly batch) of a multi-component assembly system where the component lead times are stochastic and the assembly date and quantity are fixed. The problem consists in determining the timing of each component order so that the total cost, composed of the component holding and the tardiness of the assembly costs is minimised.

An interesting single-period model of this type is developed in (Chu et al., 1993). Their model deals with a punctual fixed demand for one finished product. To assemble this product several types of components are needed. The lead times of the components are random variables. It is necessary to determine the order date for each type of component. The criterion considered is the mathematical expectation of the sum of the holding cost for the components and the backlogging cost for the finished product.

The previous single-period models do not take into account the dependence between stocks of successive periods for the periodic case where this inventory problem must be solved at each period, but the stocks of the previous periods can be used for the next and so on. The mathematical formulation of multi-period problems under lead time uncertainty is more difficult. Orders may cross, that is, they may not be received in the same sequence in which they were placed (He et al., 1998). In (Lee and Nahmias, 1993) it is assumed that orders do not crossover, and so a single-period problem is solved.

A multi-period model is proposed in (Gurnani et al., 1996) for assembly systems with two types of components. The lead time probability distributions are limited to two periods. The authors assume that components either arrive in the current period with a given probability \(\alpha\) or in the next period with probability \((1 – \alpha)\). This two period lead time model gives the optimal quantity of each component to order from each supplier.

An interesting model is suggested in Fujiwara and Sedarage (1997). The authors study a \((Q, r)\)-type model for a simple assembly system with stochastic component procurement lead times. Assembly is instantaneous and takes place intermittently in batches but cannot start until all the components are available. The authors use the following general assumptions: one finished product and several types of components, a constant and known demand rate, and infinite assembly capacity. They take into account the inventory holding cost for the components and the assembled product, the shortage cost for the assembled product and the setup cost. Fujiwara and Sedarage developed a continuous model for the following \((Q, r)\) ordering policy: when the component inventory depletes to a reorder level \(r_i\), for component \(i\), a batch of size \(Q\) of the component \(i\) is ordered. The value of \(Q\) is the same for all component types. The decisions to be made are the reorder points \(r_i\) and the lot size \(Q\). The obtained non linear optimization problem is decomposed into a family of sub-problems. Experimental results are presented for only two components, and the computational time is not furnished. This model is close to our problem, but it used several additional approximations. There is a continuous model instead of discrete one. In manufacturing systems, the number of components in stock and number of delivered lots are indeed discrete variables. This model gives a numerical solution instead of an analytical optimal solution. Finally, the shortage cost is used instead of the service level. However, shortage cost is notoriously difficult to estimate, while the service level is a commonly used characteristic of inventory systems.

Wilhelm and Som (1998) suggest a model for the inventory control for an assembly system with several types of components. Their model concentrates on a single finished product inventory, so the interdependence between inventory levels of different components is again neglected.

Dolgui et al. (1995) suggest a model for an assembly planning under constant demand and random component lead times for the lot for lot policy. Several types of products are considered. For assembly of each product several types of components are needed. The authors take into account the holding and backlogging costs. This model determines the number of components of each type to be ordered at the beginning of each period and products to be assembled. An approach based on the coupling of an integer linear programming model with a simulation and heuristics is developed. This problem is also studied in (Proth et al., 1997), but the components to be ordered and the products to be assembled are selected heuristically.

The model suggested in this paper is different from the above models for assembly systems in the following aspects: (i) we consider a multi-period model; (ii) there is no restriction of number of components; (iii) lead time density function for each component may differ from other components – any distribution known in advance; (iv) the problem is discrete instead of continuous; (v) we search an optimal instead of an approximate solution.
This work follows (Louly and Dolgui, 2002) where a generalized discrete Newsboy model was proposed for the same type of assembly systems as in (Chu et al., 1993) but providing some generalizations. This discrete Newsboy model is multi-period with random lead times and integer decision variables. The finished product demand is periodic and constant, i.e. the same for all periods. The criterion considered is the sum of the average holding cost for the components and the average backlogging cost for the finished product. This model gives the optimal values of the safety stocks when the component lead times are identically distributed (iid) random variables and the unit holding costs are the same for all types of components.

The aim of the present paper is to develop both a model and a method for the more general case than (Louly and Dolgui, 2002) when the backlogging cost is replaced by the service level for the finished product. The method proposed in this paper does not need the restrictive assumptions of iid random variables and identical holding costs, thus is broader in scope. In this paper, some original dominance rules, lower bounds and properties are proved which could be also used elsewhere.

3. Optimization model

The inventory problem is illustrated in Fig. 1. The demand $D$ for the single product to be assembled in each period is constant. Several types of components are needed to assemble a product. The needs for components per period are also constant and deduced from the demand $D$ of finished product. The components are delivered by external suppliers. Demands for products are met at the end of each period and unfilled demands are backordered and have to be treated later. The assembly system capacity is supposed infinite, i.e., only the component inventory control problem is considered. There is no stockage of finished products. The unit holding cost per period for each type of component and a desired service level for the finished product are known. The lead times for orders made at different periods for the same type of component are independent and identically distributed discrete random variables. The distribution probability for the different types of components may be not identical. These distributions are known, and their upper values are finite. Our model is valid for all distributions.

All periods have the same duration. Without loss of generality, we suppose that one unit of each type of component is required to assemble a single product. The supply policy for components is lot for lot: one unit of each type of component is ordered at the beginning of each period. Therefore, the safety stocks, represented by the integer decision variables $x_i$, are equal to the initial inventory levels of the components.

In other words, to simplify the presentation of the model, we suppose that one finished product is assembled at each period, i.e., $D = 1$ (note: even when there are several units of a product assembled at each period, and many identical components are used for the product, no change is needed in the model). The inventory position, which is equal to the quantity on hand plus those ordered minus the backlogged demands, is constant. It is equal to the initial inventory level (safety stock).

Let us use the following notations:

1. $1 - e$ desired service level
2. $h_i$ unit holding cost of the component $i$ per period, $i = 1, 2, \ldots, n$
3. $L_i$ lead time of the components $i$, $i = 1, 2, \ldots, n$
4. $L_i^*$ effective lead time of the component $i$ ordered at the beginning of period $k$, $k = 1, 2, \ldots, \infty$
5. $n$ number of different types of components for the assembly system
6. $N_i^*$ number of component $i$ orders that are not delivered yet
7. $R_i^*$ product delivery delay due to lack of component $i$
8. $u_i$ upper value of $L_i$, $i = 1, 2, \ldots, n$
9. $x_i$ safety stock (initial inventory) of the components $i$, this is an integer decision variable and $i = 1, 2, \ldots, n$.

Note that the maximal value of the component $i$ lead time is equal to $u_i$ and that the orders are made at the beginning of each period. Therefore, at the end of a given period $k$ only the previous $u_i - 1$ orders for component $i$ may not be delivered yet. It is because the distribution of probability for $L_i$ has a finite upper value $u_i$ that the orders made earlier have been delivered.

Let $L_i^{k+1-1}, j = 1, 2, \ldots, u_i - 1$, be the lead times of the components $i$ ordered at the beginning of the periods $k, k - 1, \ldots, k - u_i + 2$. Using the standard function $1_A$ equal to 1 if $A$ is true and 0 otherwise, we know that order made in the period $k+1 - j$ is delivered after the end of the period $k$ when $1_{L_i^{k+1-1}}$ is equal to 1. Then, the random variable $N_i^*$ is the number of the component $i$ orders that are not yet delivered, i.e., waiting at the end of the period $k$:

$$N_i^* = \sum_{j=1}^{u_i-1} 1_{k+1-1,j}, \quad i = 1, 2, \ldots, n. \quad (1)$$

At the period $k$, there is a shortage if there is a component $i$ for which the sum of the initial inventory $x_i$ and the delivered quantity $(k - N_i^*)$ is smaller than the quantity to be assembled up to period $k$ which is equal to the cumulative demand for $k$ periods. Therefore, a shortage occurs when the value $k - x_i - (k - N_i^*) = (N_i^* - x_i)$ is larger than zero at the end of the period $k$.

Let $R_i^* = (N_i^* - x_i)$ be the delay due to lack of the components $i$ at the end of the period $k$. Then, the value $R_i^* = \max_{k=1,\ldots,\infty} R_i^*$ gives the maximal delay at the end of this period. Hence, there is a shortage when:

$$R_i^* > 0. \quad (2)$$

Therefore, the number of periods for which there are a necessary number of components is equal to $k - R_i^*$. The inventory $S_i^*$ of the component $i$ at the end of the period $k$ is equal to the initial inventory, plus the delivered quantities, minus those used:

$$S_i^* = x_i + (k - N_i^*) - (k - R_i^*) = (x_i - N_i^* + R_i^*). \quad (3)$$
Thus, the cost of the period \( k \) is obtained as follows:

\[
C_k(X, N^k) = \sum_{i=1}^{n} h_i(x_i - N^k_i) + HR^k,
\]  
(4)

where \( X = (x_1, x_2, \ldots, x_n) \), \( H = \sum_{i=1}^{n} h_i \) and \( N^k = (N^k_1, \ldots, N^k_n) \).

The cost of one period (4) is a random variable because \( N^k \) are random variables.

To describe the procurement process of each type of component, an ergodic Markovian chain has been proposed in (Dolgui and Louly, 2002). We proved that these Markov chains are homogeneous and the Markov chain for component \( i \) converges to a stationary distribution in less than \( u_i \) iterations, \( i = 1, 2, \ldots, n \). Therefore, the steady state values of the service level and the cost can be used for optimization.

The holding cost \( C(X) \) and the service level \( SL(X) \) on the infinite horizon are:

\[
C(X) = \lim_{m \to \infty} \frac{1}{m} \sum_{k=1}^{m} C(X, N^k) = E \left[ \sum_{i=1}^{n} h_i(x_i - N_i) + HR \right],
\]
(5)

\[
SL(X) = \lim_{m \to \infty} \frac{1}{m} \sum_{k=1}^{m} \sum_{i=1}^{n} 1_{x_i > 0} = Pr(R = 0),
\]
(6)

where \( E(Z) \) is the expected value of the random variable \( Z \), \( N_i = \sum_{j=1}^{u_i} 1_{x_j > 0} \), \( i = 1, \ldots, n \) are the steady state numbers of deliveries in waiting.

\[
R = E(\max_{i=1,\ldots,n} (N_i - x_i) +) \quad \text{is the steady state delay at the end of period.}
\]
(7)

**Proposition 1.** The expressions (5) and (6) can be rewritten as follows:

\[
C(X) = \sum_{i=1}^{n} h_i(x_i - E(N_i)) + H \sum_{k=0}^{\infty} \left( 1 - \sum_{i=1}^{n} F_i(x_i + k) \right),
\]
(8)

\[
SL(X) = \prod_{i=1}^{n} F_i(x_i),
\]
(9)

where

\[
F_i(k) = Pr(N_i \leq k).
\]
(10)

**Proof.** Eqs. (5) and (6) give:

\[
C(X) = E \left[ \sum_{i=1}^{n} h_i(x_i - E(N_i)) \right] + HE(\max_{i=1,\ldots,n} (N_i - x_i) +)

\[= \sum_{i=1}^{n} h_i(x_i - E(N_i)) + HE(Z),
\]
(11)

\[
SL(X) = 1 - Pr(Z > 0),
\]

where \( Z = \max_{i=1,\ldots,n} Z_i \) and \( Z_i = (N_i - x_i) + \). The mathematical expectation of the variable \( Z \) can be calculated as follows:

\[
E(Z) = \sum_{i=0}^{\infty} i Pr(Z = i) = \sum_{k=0}^{\infty} \sum_{i=0}^{k-1} Pr(Z = i) = \sum_{k=0}^{\infty} \sum_{i=0}^{k-1} Pr(Z = i) = \sum_{k=0}^{\infty} \sum_{i=0}^{k} Pr(Z > k).
\]

Taking into account that \( Pr(Z > k) = 1 - Pr(Z \leq k) \), we obtain:

\[
E(Z) = \sum_{k=0}^{\infty} (1 - Pr(Z \leq k)) = \sum_{k=0}^{\infty} (1 - Pr(\max_{i=1,\ldots,n} Z_i \leq k)).
\]

Using \( Pr(\max_{i=1,\ldots,n} Z_i \leq k) = Pr(Z_1 \leq k, Z_2 \leq k, \ldots, Z_n \leq k) \), this equation becomes:

\[
E(Z) = \sum_{k=0}^{\infty} (1 - Pr(Z_1 \leq k, Z_2 \leq k, \ldots, Z_n \leq k)).
\]

Since \( Z_i \) are independent random variables, therefore:

\[
E(Z) = \sum_{k=0}^{\infty} \left( 1 - \prod_{i=1}^{n} Pr(Z_i \leq k) \right) = \sum_{k=0}^{\infty} \left( 1 - \prod_{i=1}^{n} Pr(N_i - x_i) + \right).
\]

Because

\[
Pr((N_i - x_i)^+ \leq k) = Pr(\max(N_i - x_i, 0) \leq k) = Pr(N_i - x_i \leq k) \quad 1_{0 \leq k^*}.
\]

\[
E(Z) = \sum_{k=0}^{\infty} \left( 1 - \prod_{i=1}^{n} Pr(N_i - x_i) \leq k) \right) 1_{0 \leq k}. \quad (12)
\]

Putting (12) in the relation (11) we obtain:

\[
C(X) = \sum_{i=1}^{n} h_i(x_i - E(N_i)) + H \sum_{k=0}^{\infty} \left( 1 - \sum_{i=1}^{n} F_i(x_i + k) \right).
\]

Finally, \( Pr(Z > 0) \) is defined by the first element of the sum \( (k = 0) \) in (12), thus:

\[
SL(X) = 1 - \left( 1 - \prod_{i=1}^{n} F_i(x_i) \right) = \prod_{i=1}^{n} F_i(x_i). \quad \Box
\]

Then the problem is to minimize the cost given in (9) while keeping the desired service level, i.e., minimize \( C(X) \) subject to:

\[
SL(X) \geq 1 - \varepsilon. \quad (13)
\]

The main difficulty is that the decision variables, \( x_i, i = 1, 2, \ldots, n \), are integers.

Note that, the components \( x_i \) of the optimum \( X \) must satisfy the following constraint:

\[
0 \leq x_i - u_i - 1, \quad i = 1, 2, \ldots, n. \quad (14)
\]

In fact, we can replace each component \( x_i \) exceeding \( u_i - 1 \) by this value. The obtained vector respects (14) and leads to a reduction in cost.

**4. Lower bounds and dominance properties**

In this section, we present two lower bounds and two dominance properties which will be used in the branch and bound algorithm.

**4.1. Partial increments of the criterion function**

We introduce the following partial increment functions:

\[
G_i^0(X) = C(x_1, \ldots, x_i + 1, \ldots, x_n) - C(x_1, \ldots, x_i, \ldots, x_n), \quad (15)
\]

\[
G_i^0(X) = C(x_1, \ldots, x_i - 1, \ldots, x_n) - C(x_1, \ldots, x_i, \ldots, x_n). \quad (16)
\]

These functions represent changes in the objective function due to increment or decrement of a decision variable.

An optimal solution \( X \) must satisfy the requirements (17) and (18), otherwise there is a neighbouring feasible solution better than \( X \).
Theorem 1. The function $G_i^+(X)$ and $G_i^-(X)$ have the following properties:

(i) $G_i^+(X)$ is an increasing function on $x_i$ and a decreasing function on $x_j$ for all $j \neq i$.

(ii) $G_i^-(X)$ is a decreasing function on $x_i$ and an increasing function on $x_j$ for all $j \neq i$.

Proof. The second property (ii) is immediately derived from the first (i). So, it is sufficient to prove only the former.

The function $G_i^+(X)$ can be rewritten as follows:

$$G_i^+(X) = h_i - H \sum_{k=0}^{n} \text{Pr}(N_i = x_i + k + 1) \prod_{j=1}^{k} F_j(x_j + k) - H \sum_{k=0}^{n} \text{Pr}(N_i = x_i + k + 2) \prod_{j=1}^{k} F_j(x_j + k).$$

But

$$\sum_{k=0}^{n} \text{Pr}(N_i = x_i + k + 1) \prod_{j=1}^{k} F_j(x_j + k) = \text{Pr}(N_i = x_i + 1) \prod_{j=1}^{k} F_j(x_j) + \text{Pr}(N_i = x_i + k + 1) \prod_{j=1}^{k} F_j(x_j + k) = \text{Pr}(N_i = x_i + 1) \prod_{j=1}^{k} F_j(x_j) + \text{Pr}(N_i = x_i + k + 1) \prod_{j=1}^{k} F_j(x_j + k).$$

Then

$$G_i^+(X_1, \ldots, x_i + 1, \ldots, x_n) - G_i^-(X_1, \ldots, x_i + 1, \ldots, x_n) = \text{Pr}(N_i = x_i + 1) \prod_{j=1}^{k} F_j(x_j) + \text{Pr}(N_i = x_i + k + 1) \prod_{j=1}^{k} F_j(x_j + k).$$

4.2. Dominance properties

We note by $[A, B]$ the space of feasible solutions satisfying $a_i \leq x_i \leq b_i, \quad i = 1, \ldots, n,$ where $A = (a_1, a_2, \ldots, a_n)$, and $B = (b_1, b_2, \ldots, b_n)$.

Theorem 2. If $G_i^+(A) < 0$, then each solution $x_i$ of $[A, B]$ with $x_i = a_i$ is dominated. If $G_i^-(B) < 0$ and the vector $(a_1, \ldots, a_{i-1}, b_i - 1, a_{i+1}, \ldots, a_n)$ satisfies the constraint (13), then each solution $x_i$ of $[A, B]$ with $x_i = b_i$ is dominated.

Proof. If $G_i^+(A) < 0$, then the vector $A$ is dominated because it does not satisfy (17). Furthermore, for each vector $V$ of $[A, B]$ such that $v_i = a_i$ we have $G_i^+(V) < G_i^+(A) < 0$ because the function $G_i^+(X)$ is decreasing on $x_i$ for all $j \neq i$, hence $V$ is also dominated.

At the same time, if $G_i^-(B) < 0$ where $(b_1, \ldots, b_{i-1}, 1, \ldots, b_n)$ satisfies (13), then the vector $B$ is dominated because it does not satisfy (18). Furthermore, for each vector $W$ of $[A, B]$ such that $v_i = b_i$, we have $G_i^-(W) < G_i^-(B) < 0$ because the function $G_i^-(X)$ is increasing on $x_i$ for all $j \neq i$, hence $W$ is also dominated.

4.3. Lower bounds

Theorem 3. The following values, $L_{B_1}$ and $L_{B_\infty}$ are lower bounds for the space $[A, B]$:

$$L_{B_1} = C(A) + \sum_{i=1}^{n} \min (b_i - a_i, 0),$$

$$L_{B_\infty} = C(B) + \sum_{i=1}^{n} \min (a_i - b_i, 0).$$

Proof. For each vector $X$ of $[A, B]$, we rewrite $\sum_{i=1}^{n} (b_i - x_i) - G_i^+(x_1, \ldots, x_{i-1}, b_i, \ldots, x_n) - CM(A)$ as

$$\sum_{i=1}^{n} (c(x_1, \ldots, x_i, a_{i+1}, \ldots, a_n) - C(x_1, \ldots, x_i, a_i, \ldots, a_n)).$$

Given that

$$C(x_1, \ldots, x_i, a_{i+1}, \ldots, a_n) - C(x_1, \ldots, x_i, a_i + j, a_{i+1}, \ldots, a_n),$$

we obtain

$$C(X) - C(A) = \sum_{i=1}^{n} \sum_{j=0}^{a_i - a_{i+1} - 1} G_i^+(x_1, \ldots, x_i - 1, a_i + j, a_{i+1}, \ldots, a_n).$$

As previously shown, the function $G_i^+$ is increasing on $x_i$, then

$$G_i^+(x_1, \ldots, x_i - 1, a_i + j, a_{i+1}, \ldots, a_n) \geq G_i^+(x_1, \ldots, x_i - 1, a_i, a_{i+1}, \ldots, a_n).$$

By using (27) while developing (26) we obtain:

$$C(X) - C(A) \geq \sum_{i=1}^{n} (x_i - a_i) |G_i^+(x_1, \ldots, x_i - 1, a_i, \ldots, a_n).$$

Given that $G_i^+$ is decreasing on $x_i$ for $j < i$, we obtain an expression without $(x_1, \ldots, x_n)$:

$$C(X) - C(A) \geq \sum_{i=1}^{n} (x_i - a_i) |G_i^+(b_1, \ldots, b_{i-1}, a_i, \ldots, a_n).$$
Note that the Eq. (29) means that \( LB_1 \) is a lower bound for the space \([A, B]\).

Furthermore, for each vector \( X \) of \([A, B]\), we rewrite \( C(X) - C(B) \) as:

\[
C(X) - C(B) = \sum_{i=1}^{n} C(x_i, \ldots, x_i, b_i, \ldots, b_n) - C(x_i, \ldots, x_i, b_i, \ldots, b_n).
\]

(30)

Given that

\[
C(x_1, \ldots, x_i, b_{i+1}, \ldots, b_n) - C(x_1, \ldots, x_i, b_i, \ldots, b_n)
= \sum_{j=1}^{b_i-x_i} G_i(x_1, \ldots, x_i-1, x_i + j, b_{i+1}, \ldots, b_n).
\]

(31)

we obtain

\[
C(X) - C(B) = \sum_{i=1}^{n} \sum_{j=1}^{b_i-x_i} G_i(x_1, \ldots, x_i-1, x_i + j, b_{i+1}, \ldots, b_n).
\]

(32)

As previously shown, the function \( G_i \) is decreasing on \( x_i \), then

\[
G_i(x_1, \ldots, x_i, x_i + j, b_{i+1}, \ldots, b_n) \geq G_i(x_1, \ldots, x_i, b_i, b_{i+1}, \ldots, b_n).
\]

(33)

By using (33) in (32) we obtain

\[
C(X) - C(B) \geq \sum_{i=1}^{n} (b_i - x_i) G_i(x_1, \ldots, x_i-1, b_i, \ldots, b_n).
\]

(34)

Given that \( G_i \) is increasing on \( x_j \) for \( j < i \), we obtain an expression without \( (x_i, \ldots, x_i) \):

\[
C(X) - C(B) \geq \sum_{i=1}^{n} (b_i - x_i) G_i(a_1, \ldots, a_{i-1}, b_i, \ldots, b_n)
\]

\[
\geq \sum_{i=1}^{n} (b_i - a_i) \min(G_i(a_1, \ldots, a_{i-1}, b_i, \ldots, b_n), 0).
\]

(35)

The Eq. (35) shows that \( LB_2 \) is a lower bound of objective function for the space \([A, B]\). □

5. Optimization algorithm

5.1. Basic idea of the branch and bound algorithm

In a pre-processing procedure, we try to eliminate solutions that do not respect the service level constraint. However, we do not eliminate all these unfeasible solutions in order to keep the simple structure of the tree node where a solution space is presented by two vectors \([A, B]\). An example of service level and cost values for an assembly system with two components is presented in Table 1 and Table 2. The required service level is equal to 0.6. Nevertheless, we keep the solution space with the values \( x_1 \in \{2, 3, 4\} \) and \( x_2 \in \{3, 4\} \) to stay rectangular, thus simpler to use. This solution space will be presented by its two extremities \([A, B]\), where \( A = (2, 3) \) and \( B = (4, 4) \). Note: this contains two unfeasible solutions: \((2, 3)\) and \((3, 3)\).

Based on the previous lower bounds and dominance properties, we developed a branch and bound algorithm that is able to find the optimal solutions for significant sized problems in a reasonable computational time.

Each node of the branch and bound tree represents a set of solutions \([A, B]\). Two descendants for each node are created. The descendants of a node are obtained by a division of the solution space \([A, B]\). The solution space of the node is split into two subspaces \([A, B]^i\] and \([A^i, B]\) as follows: we choose \( i \) such that \( i = \arg \max (b_i - a_i) \). The descendant \([A, B]^i\) is the subspace of vectors for whom the \( i \)-th component satisfies the condition \( a_i \leq x_i \leq \frac{b_i + a_i}{2} \) and the descendant \([A^i, B]\) is the subspace for whom the \( i \)-th component satisfies \( \frac{b_i + a_i}{2} + 1 \leq x_i \leq b_i \). This idea of node division is illustrated in Fig. 2 for two types of components \((n = 2)\).

The lower bounds are given by their analytical forms (24), (25). As initial Upper Bound, we use the first solution found by a depth first search algorithm which is calculated in polynomial time. After

### Table 1

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.246096</td>
<td>0.376582</td>
<td>0.468391</td>
<td>0.546069</td>
</tr>
<tr>
<td>3</td>
<td>0.278118</td>
<td>0.425582</td>
<td>0.529338</td>
<td>0.617123</td>
</tr>
<tr>
<td>4</td>
<td>0.450670</td>
<td>0.689624</td>
<td>0.857752</td>
<td>1.000001</td>
</tr>
</tbody>
</table>

### Table 2

<table>
<thead>
<tr>
<th>( x_2 )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.735521</td>
<td>2.606055</td>
<td>3.067274</td>
<td>3.834243</td>
</tr>
<tr>
<td>3</td>
<td>3.847330</td>
<td>3.473811</td>
<td>3.735800</td>
<td>4.472447</td>
</tr>
<tr>
<td>4</td>
<td>5.329675</td>
<td>4.681685</td>
<td>4.750557</td>
<td>5.323813</td>
</tr>
</tbody>
</table>
the division of a node, some cuts are used to reduce the corresponding solution spaces (the solution space of each descendant of initial node). The cuts are based on the dominance properties presented earlier in Section 4.2.

First, we present the algorithm for verification of the dominance properties and the cut rules, then the heuristic algorithm which gives the initial Upper Bound (UB) is reported and finally, the Branch and Bound algorithm is explained.

5.2. Reduction of solution space

First, let’s present the reduction procedures Reduce_A and Reduce_B which are based on the aforementioned dominance properties. They reduce the space [A, B] replacing A by a larger vector and B by a smaller vector. This eliminates the corresponding part of the solution space [A, B]. The reduction scheme is the same for these two procedures and they return “true” when the subset [A, B] is entirely dominated, i.e. the obtained cut completely eliminates the node [A, B].

Procedure Reduce_A:

Begin
  reduced ← false
  dominated ← false
  While (not reduced and not dominated)
    reduced ← true
    For i = 1 to n do
      While \( G_i^c(A) < 0 \) and dominated = false do
        reduced ← false
        if \( (a_i = b_i) \) dominated ← true
        else \( a_i ← a_i + 1 \)
      End while
    End for
  End while
End

Procedure Reduce_B:

Begin
  reduced ← false
  dominated ← false
  While (not reduced and not dominated)
    reduced ← true
    For i = 1 to n do
      While \( G_i^c(B) < 0, \) \( F_i(b_i - 1) \) \( \backslash \cdot F_i(a_i) \geq 1 - \epsilon \) and dominated = false do
        reduced ← 0
        if \( (a_i = b_i) \) dominated ← true
        else \( b_i ← b_i - 1 \)
      End while
    End for
  End while
End

Fig. 2. A fragment of the branch and bound tree.

5.3. Upper bound calculation

The upper bound calculation starts with the reduction of the initial solution space using the procedures Reduce_A and Reduce_B. Let [A, B] be the result of this reduction. This space is then divided in two subspaces. The new subspace [A, B^1] is reduced using Reduce_B. The procedure Reduce_A is not used here because the vector A is not modified, i.e. it was the original extremity for the space [A, B]. For the same reason, only Reduce_A is applied to the subspace [A^1, B]. Finally, the descendant with the best feasible solution is kept. Note that a vector is considered as feasible only when the constraint (13) is satisfied. The algorithm for Upper Bound calculation is as follows:

Begin of Upper_Bound([A, B])
  Apply Reduce_A and Reduce_B for [A, B]
  UB ← C(B)
  If the extremity A is a feasible solution then
    UB ← min(C(A), C(B))
  End if.
  While (A ≠ B) do
    divide [A, B] into two subspaces [A, B^1] and [A^1, B]
    apply Reduce_B to [A, B^1]
    apply Reduce_A to [A^1, B]
    UB ← the cost of the best feasible solution of: A, B^1, A^1, and B.
    If (this best solution is A or B^1) then
      [A, B] ← [A, B^1]
    Else
      [A, B] ← [A^1, B]
    End if
  End while
  Return UB
End

5.4. The branch and bound algorithm

The branch and bound algorithm is a variant of a width first search that consists in choosing the node that contains the maximum number of solutions. The lower bound (LB) is the maximum of the two bounds presented in Section 4.3.

Begin
  \( a_i ← l_i, i = 1, \ldots, n \), where \( l_i \) is the smallest integer satisfying
  \( F(x_i) \geq 1 - \epsilon \)
  \( b_i ← u_i - 1, i = 1, \ldots, n \).
Reduction [A, B]
Reduce_B [A, B]
current_min ← Upper_Bound( [A, B])
current_solution ← the solution given by Upper Bound

Procedure

// Initialize the set E of nodes which can be divided:
E ← [{A, B}]
If A ≠ B then
While (E is not empty) do
[A, B] ← the node of E that has the largest solution space
E ← E - [{A, B}]
Divide [A, B] into two subsets: [A, B'] and [A', B]
Apply Reduce_B to the set [A, B']
if (the reduced subspace [A, B'] is not dominated, A ≠ B', its LB < current_min, and, B' is a feasible solution) then
Add it to E: E ← E ∪ [{A, B'}]
End if
Reduce_A [A', B]
if (the reduced subspace [A', B] is not dominated, A' ≠ B and its LB < current_min) then
Add it to E: E ← E ∪ [{A', B}]
End if
// Update the Upper bound and the set E if necessary:
If ([A'] is a feasible solution and C(A') < current_min) then
current_min ← C(A')
current_solution ← A'
End if
If ([B'] is a feasible solution and C(B') < current_min) then
current_min ← C(B')
current_solution ← B'
End if
Delete the elements of E which have LB larger than current_min.
End while
End if
End

In our algorithm, the lower bounds are given by their analytical forms (24) and (25), and the node division is made when the cuts based on dominance properties cannot be used again to reduce the corresponding solution space.

6. Experimental results

Consider an example of an assembly system with 10 types of components (n = 10). The lead time for each type of component is a discrete random variable with values from 1 to 10 (i.e., u = 10), and their distribution probabilities are randomly generated. The required customer service level is fixed at 0.99. The unit holding costs h_i are generated in the interval [1, n]. Thus, for this example, the cardinal of the total number of solutions is equal to 10^{10}. Using the branch and bound algorithm, we obtained the optimal value of the objective function equal to 158.465 after 321 iterations. In Fig. 3, we present the change over time of the Lower Bound. The minimum values of the Lower Bounds for all current nodes for the iterations of the branch and bound algorithm are reported.

In Fig. 4 the change of the number of nodes during the iterations is presented. This shows that in the first stage, where the lower bound is not yet very precise, the number of nodes increases. In the second stage, it decreases and finally the set becomes empty after 321 iterations.

To test the performances of the branch and bound algorithm, 160 other examples were generated. They were grouped into 16 families. Each family gathers 10 examples with the same number of components n and the same value for the parameter u. The unit holding costs h_i are randomly generated in the interval [1, n]. The customer service level is fixed at 0.99.

We limited the calculation time for each example to 90 seconds. All these problems were optimally solved without exceeding this limit. Table 3 gives the average computing times in seconds.

7. Concluding remarks

In this paper, we considered a novel inventory control model for assembly systems with random component procurement lead times. The proposed model provides a substantial saving for assembly systems with a large number and unreliable delivery of components as in semi-conductor and automotive industries. The problem was how to calculate the component safety stock under a given service level constraint. Two original and powerful dominance properties were proved. In addition, two lower bounds of excellent quality were developed. These results lead to a better understanding of the behaviour of these inventory systems. They also facilitated the creation of an efficient branch and bound algorithm which calculates the optimal values of decision variables of the model. This algorithm was developed and tested. The experimental studies demonstrated its outstanding performance.
The results which extend the existing knowledge in inventory control are also promising for some industrial applications. In particular, for the assembly systems where the delivery of the finished product is controlled by the EOQ policy, the proposed algorithm gives the optimal safety stocks for the components. It can be also used for any assembly system with capacity inferior or equal to demand. In this case, this assembly system capacity defines the fixed demand for the components. The models with fixed demand are also used for assembly systems of intermediate modules which are then used in assembly of a finished product, for example, in production of electronic boards used in the fabrication of different types of televisions. The demand in televisions is not fixed, but for certain standard circuit boards used in different TVs, this demand can be considered as fixed. The models with fixed demand are also directly applicable for container transport, for example a container vessel with its capacity represents a fixed demand and containers to be transported can be considered as components, and so on.

The interest of this paper is not only in its direct application. The proposed model can be a satisfactory approximation for more general problems. The presented method can be useful, for example, for parameterization of MRP software tools. In industrial practice, often security coefficients are introduced to calculate the planned lead time for unreliable suppliers (planned lead time = contractual delay * security coefficient). These coefficients are empiric but anticipate the delays by creating a safety lead time. The more unreliable a supplier is the larger is its coefficient. The proposed method can be used to estimate these coefficients based on statistics for the procurement lead times for each supplier.

Our model and algorithms can be also easily generalized for a known demand with a fixed level for each interval or a dynamic demand with slow change over time, etc. For random demand, if a policy without nervousness is accepted (a relatively constant production plan), then a stock of finished product must be created. However, this does not prevent the use of safety stocks for components (calculated with our model) while the finished product stock is managed by other models. As demonstrated above this work has a broad application in many areas.

Acknowledgement

This work has been partially supported by European coordination action CODESNET and Princess Fatimah Alnajras’s Research Chair of Advanced Manufacturing Technology. This text has been checked by native English speaker Chris Yukna.

References


