Managing individual customer service constraints under stochastic demand

Jayashankar M. Swaminathan, Ramesh Srinivasan

Walter A. Haas School of Business, University of California, Berkeley, CA-94720-1900, USA
IBM Storage Systems Division, San Jose, CA-95193, USA

Received 1 November 1995; received in revised form 1 November 1998

Abstract

In recent years management of customer requirements has been the foremost concern of most manufacturers. In this paper, we consider a single-period problem in which the manufacturer faces stochastic demand and individual service level constraints from multiple customers. We present a formulation for the problem that utilizes the structure of the underlying allocation problem and provide an algorithm that generates the optimal procurement quantity under the optimal random allocation policy. For a special case when there are two identical customers and their demand is jointly uniform, we derive an analytical expression for the optimal procurement quantity.

Keywords: Inventory/production; Stochastic models; Stochastic programming

1. Introduction

Manufacturers often face difficulties while allocating products to customers with different service requirements. In this paper, we address the issue of optimal procurement and allocation of a single product for a manufacturer facing stochastic demand and Type-I service constraints (percentage of time periods in which demand is satisfied completely) from multiple customers. We consider a single-period model where the sequence of events in each period is as follows. (1) Manufacturer procures the product (perishable good); (2) Stochastic demand of different customers is realized; (3) Product is allocated to different customers so that individual service constraints are met in the long run; (4) Products left over at the end of the period are discarded. In such a scenario, the manufacturer has to make two important decisions – (1) how much inventory to keep (procurement quantity) and, (2) how to allocate the inventory under different demand realizations (allocation policy).

Our motivation for this work comes from the analysis of inventory management of fresh foods in the grocery industry. Perishable inventory problems have been studied extensively by a number of authors [5,6,10–13]. However, the above studies neglect customer service requirements and issues relating to allocation of products among different customers. Allocation of products is an important criterion in multi-product inventory models with substitution. Bitran and Leong [4] and Bitran and Dasu [3] consider
multi-product, multi-period substitution problem with random yields. Both papers approximate the stochastic program with a deterministic one and present solutions for the deterministic problem. Hsu and Bassok [9] consider a single-period, multi-product model with stochastic demand and random yields and derive an efficient greedy algorithm to solve the problem. Bassok et al. [2] use a similar approach for solving the inventory problem with substitution where they utilize the information about the underlying allocation policy to solve the optimal procurement problem. More recently, Rao et al. [14] consider a multi-product inventory system with stochastic demand and set up costs where downward conversion is utilized to effectively manage inventory. However, the above papers do not model customer service requirements explicitly.

The problem of allocation of a common component to multiple products is closely related to our problem. Sauer [14] and Gerchak and Henig [7] explore the qualitative impact of commonality on component stocks. Swaminathan and Tayur [15] consider inventory management under postponement using common sub-assemblies called vanilla boxes in a multi-product inventory system with stochastic demand. Baker et al. [1] consider a problem with two products assembled from a single common component and two unique components. They assume uniform distribution for the demand for the products and find the optimal inventory levels for the components under individual service constraints for the products. Gerchak et al. [8] extended this work further by considering a more general product structure of \( n \) products whose demands have general joint distribution, each product has a unique component price \( p_i \), and a component common (of price 1) to all products where one of each unique component and one common component is used to make end products. In this work, we extend the analysis to multiple customers with different service levels and a general joint demand density function. Typically, a manufacturer has three or four primary customers (or A customers in A–B–C classification) to whom she may promise a high service level and may want to satisfy their demands at all costs. These are the customers that we consider in this model. For the rest of the customers, the manufacturer may perform partial shipments with the remaining inventory.

The rest of the paper is as follows. In Section 2, we first present a formulation based on chance constraints. Since that problem is difficult to handle we reformulate the problem utilizing the underlying allocation mechanism to obtain the optimal procurement quantity. Subsequently, we discuss our solution methodology to find the optimal solution. Finally, we consider a special case of the problem with two identical customers whose demand is jointly uniform and derive a closed-form expression for the optimal procurement quantity. In Section 3, we provide results from our computational study and we conclude in Section 4.

2. Model formulation

In this section, we introduce the problem and present a chance constrained mathematical formulation (P1) for the problem. Since the formulation cannot be solved without an allocation policy, we present another formulation (P2) for the problem which utilizes properties of the underlying allocation problem. We develop bounds for the optimal procurement quantity based on the formulation (P2) and present a solution procedure to calculate the optimal procurement quantity. Finally, we derive a closed-form expression for the optimal procurement quantity for the case when there are two identical customers and demand is jointly uniform.

We consider a single-period model where the manufacturer experiences stochastic demand and individual Type-I service constraints (percentage of time periods in which demand is satisfied completely) from multiple customers. The sequence of events in each period is as follows: (1) Manufacturer procures the product (perishable good); (2) Demands of different customers are realized; (3) Product is allocated to different customers so that individual service constraints are satisfied in the long run; (4) Product left over at the end of the period is discarded. We use the following notations in this paper:

\[ n \] number of customers for the manufacturer

\[ Q \] inventory of goods stored at the manufacturer

\[ d \] vector representing a realization of demand of \( n \) customers
vector representing allocation of inventory to customers

\[ f(x_1, \ldots, x_n) \]

joint probability density function for demand of \( n \) customers

\[ F_k \]

marginal cumulative density function for customer \((k)\)

\[ \beta_k \]

type-\( I \) service level requirements of customer \((k)\)

\[ \beta = (\beta_1, \beta_2, \ldots, \beta_n) \]

vector representing the service constraints of customers

\[ \phi(\beta, d, f(x_1, \ldots, x_n)) \]

allocation policy that is determined based on service level requirements, the actual realization and the joint density function for the demand

\[ q_1 \quad \text{allocation policy} \]

The manufacturer procures some quantity \( Q \) at the beginning of the period. The demand for the product comes from \( n \) different customers having a joint demand distribution given by \( f(x_1, \ldots, x_n) \). The manufacturer allocates \( q_k \) to each of the customer on realization of demands so that in the long run, the service-level \( \beta_k \) is achieved (refer Fig. 1). In our formulation, we consider Type-\( I \) service level measurement which measures the percentage of time periods demand was satisfied completely. The service levels \( \beta_k \) are used by the manufacturer as a surrogate for actual penalty costs. Left-over products are discarded and have no salvage value because we are considering a perishable product. The objective of the manufacturer is to order minimum amount of the product while satisfying service requirement of customers in the long run.

\[ \min_{\phi} Q, \]

s.t. \( P(q_k \geq d_k) \geq \beta_k \ \forall k \) \hspace{1cm} (1)

where

\[ q_k : q = \phi(\beta, d, f(x_1, \ldots, x_n)), \]

s.t. \( \sum_{k=1}^{n} q_k \leq Q, \) \hspace{1cm} (2)

\[ q_k \geq 0 \quad \forall k. \] \hspace{1cm} (3)

In the above formulation inequality (1) represents service level constraints of customers. Inequality (2) represents the inventory constraint on the allocation for any demand realization and (3) is a non-negativity constraint. The solution to P1 consists of the optimal allocation rule \( \phi(\beta, d, f(x_1, \ldots, x_n)) \) and the procurement quantity \( Q \). It is difficult to solve P1 because allocation \( \phi \) (which depends on the actual demand realization) needs to be considered at the time of procurement.

\[ \min_{\phi} Q, \]

s.t. \( P(q_k \geq d_k) \geq \beta_k \ \forall k \) \hspace{1cm} (1)

where

\[ q_k : q = \phi(\beta, d, f(x_1, \ldots, x_n)), \]

s.t. \( \sum_{k=1}^{n} q_k \leq Q, \) \hspace{1cm} (2)

\[ q_k \geq 0 \quad \forall k. \] \hspace{1cm} (3)

In this section, we provide a new formulation (P2) which exploits the inherent nature of allocation and solves the problem by partitioning the demand space into a finite set of mutually exclusive regions. We need to introduce some additional definitions and notations before we present P2.

\[ \text{Definition 2.1.} \quad \text{A condition} \ (i_1, \ldots, i_k) \ \text{represents full satisfaction of demand of a subset of customers} \ i_1, \ldots, i_k. \]

\[ \text{Definition 2.2.} \quad \text{A partitioning constraint is a constraint of the following type:} \]

\[ \sum_{i \in S_r} d_i \leq Q, \]
In order to make these partitionings more efficient we utilize the following three steps:

- We partition the space of demand realizations in view of knowing which customers can have their demands fully satisfied. So, regions are defined in terms of conditions which represent the subset of customers whose demands are fully satisfied. It is to be noted that only one condition can be satisfied when the demand falls in the region.
- In order to make these partitions more efficient we utilize only non-dominated partitioning constraints. For example, in a region defined by \( d_1 + d_2 \leq Q \), though \( d_1 \leq Q \) and \( d_2 \leq Q \) are satisfied, they are not included in the definition because they are dominated by \( d_1 + d_2 \leq Q \).
- Further, we present a concise representation in which only the satisfied non-dominated partitioning constraints are listed. In reality, a region is defined by both a set of satisfied non-dominated partitioning constraints and a set of violated such constraints. For example, a region where \( d_1 \leq Q, d_2 \leq Q \) but \( d_1 + d_2 > Q \) would be defined by only \( d_1 \leq Q, d_2 \leq Q \) because the fact that \( d_1 + d_2 \leq Q \) is absent from the definition of the region implies that \( d_1 + d_2 > Q \).

Regions are generated using partitioning constraints in the following manner. Let \( K = \{ S_1, \ldots, S_p \} \) be the power set of \( \{ 1, 2, \ldots, n \} \) where \( S_i \) is the \( i \)th element of \( K \). Then each region \( (R) \) in the partition is defined by a set of conditions \( K_R \subset K \) such that \( S_i \not\subseteq S_j \) \( \forall S_i, S_j \in K_R \). In addition, \( K_R \) represents alternative conditions that could be satisfied individually but not concurrently in region \( (R) \).

**Remark.** A partitioning constraint generates a condition which represents satisfaction of demands of customer in \( S_p \). It is to be noted that partitioning constraints are restricted to be constraints which have \( \leq \) relationship with \( Q \). For example, \( d_1 + d_2 > Q \) is not a partitioning constraint.

**Definition 2.3.** A region is defined by a set of non-dominated conditions each of which is individually valid but is not valid concurrently with any other condition in that region. Our approach to this problem utilizes the following three steps:

- We partition the space of demand realizations in view of knowing which customers can have their demands fully satisfied. So, regions are defined in terms of conditions which represent the subset of customers whose demands are fully satisfied. It is to be noted that only one condition can be satisfied when the demand falls in the region.
- In order to make these partitions more efficient we utilize only non-dominated partitioning constraints. For example, in a region defined by \( d_1 + d_2 \leq Q \), though \( d_1 \leq Q \) and \( d_2 \leq Q \) are satisfied, they are not included in the definition because they are dominated by \( d_1 + d_2 \leq Q \).
- Further, we present a concise representation in which only the satisfied non-dominated partitioning constraints are listed. In reality, a region is defined by both a set of satisfied non-dominated partitioning constraints and a set of violated such constraints. For example, a region where \( d_1 \leq Q, d_2 \leq Q \) but \( d_1 + d_2 > Q \) would be defined by only \( d_1 \leq Q, d_2 \leq Q \) because the fact that \( d_1 + d_2 \leq Q \) is absent from the definition of the region implies that \( d_1 + d_2 > Q \).

Let us consider an example with two customers. Let \( P \) be the partition of the demand space that is formed by partitioning constraints defined above. In Fig. 2, we show the five regions that the partition \( P \) would contain when we consider just two customers. The set \( K=((1), (1,2), (1,2)) \) and \( S_1=\emptyset, S_2=(1), S_3=(2) \) and \( S_4=(1,2) \). Regions (shown in Fig. 2) are generated using the following conditions:

- Region-I: \( d_1 + d_2 \leq Q \) \( K_I = (S_4); \)
- Region-II: \( d_1 \leq Q \) \( K_{II} = (S_2); \)
- Region-III: \( d_2 \leq Q \) \( K_{III} = (S_3); \)
- Region-IV: no condition \( K_{IV} = (S_1); \)
- Region-V: \( d_1 \leq Q; d_2 \leq Q \) \( K_V = (S_2, S_3). \)

Recall that a condition represents full satisfaction of demand of a subset of customers. Since the service constraints are Type-I, there are no credits for partial satisfaction of demand. Hence, any optimal allocation needs to consider only alternative conditions that can be satisfied for a given demand realization. Here are some additional notations.

- \( x_r \) probability of demand occurring in region \( (r) \)
- \( K_r \) set of conditions that define region \( (r) \)
- \( p_{rj} \) probability that \( j \)th condition is satisfied while in region \( (r) \)
- \( C_k \) set of regions where the demand for customer \( (k) \) is satisfied without any contention
\[ G_k \] set of regions where the demand for customer (k) is satisfied with a probability depending on the condition that is satisfied

\[ H_{rk} \] set of conditions in region (r) that contribute to customer (k)’s service level

**Theorem 2.1.** If the space of demand realization is partitioned according to partitioning constraints described above, then the original problem (P1) reduces to the following problem (P2):

\[
\begin{align*}
\text{(P2):} & \quad \min_{p_{ij}} \mathcal{Q} \\
\text{s.t.} & \quad \sum_{i \in C_k} \mathcal{z}_i + \sum_{i \in G_k} \sum_{j \in H_k} p_{ij} \cdot \mathcal{z}_i \geq \beta_k \quad \forall k, \\
& \quad \sum_{j \in K_i} p_{ij} = 1 \quad \forall i.
\end{align*}
\]

**Proof.** Let \( R_i \) denote ith region in the partition \( P \) and \( \mathcal{z}_i \) be the probability of the demand occurring in \( R_i \) obtained from the joint demand distribution function. Note that \( \mathcal{z}_i \) depends on both the set of partitioning constraints that are included in the definition of a region as well as other constraints that are valid but not included in the definition of the region. For example, in Fig. 2, \( \mathcal{z}_5 \) depends on \( d_1 \leq Q, d_2 \leq Q \) as well as on \( d_1 + d_2 > Q \). The allocation rule in each of the regions is as follows. If the demands in \( R_i \) can be satisfied without any contention (there is only one condition that can be satisfied in that region) then use the allocation that satisfies the demand. In case there is contention and \( K_i \) represents the set of allocations (conditions that can be satisfied) that are possible in \( R_i \) then assign a probability \( p_{ij} \) for choosing the jth allocation. Since the service constraints are Type-I and there are no credits for partial satisfaction of demand, the above allocation rule considers all alternative conditions in a region. As a result, the solution \( (p_{ij}) \) to the random allocation rule in each region is the best among all possible alternatives. Hence, a random allocation policy is optimal. It is to be noted that if it turns out that a deterministic allocation is optimal in a region then the solution would generate a probability of 1 for that allocation.

For each customer \( (k) \), associate two sets of regions, \( C_k \) and \( G_k \). Let \( C_k \) represent the set of regions where the demand for customer \( (k) \) is satisfied without any kind of contention \( (k \) belongs to the unique condition in that region) and \( G_k \) represent the set of regions where the demand for customer \( (k) \) is satisfied with a probability depending on the allocation that is used. Let \( H_{rk} \) represent set of allocations in region \( (i) \) that contribute to customer \( (k) \)’s service level. It is to be noted that \( C_k \) is a special case of \( G_k \) where only one allocation is possible. We have separated the two definitions in order to make our algorithm easier to understand. Then the probability of satisfying the demand of customer \( (k) \) is given by

\[
\sum_{i \in C_k} \mathcal{z}_i + \sum_{i \in G_k} \sum_{j \in H_k} p_{ij} \cdot \mathcal{z}_i.
\]

Thus, the solution to the original problem is the solution to P2. \( \square \)

### 2.3. Solution procedure

In this section we describe our solution procedure and develop bounds on the optimal procurement quantity. We finally provide an example to show that procuring inventory individually for customers may not be an efficient solution.

Our algorithm for finding the optimal procurement quantity consists of two parts (refer Fig. 3). The first part partitions the demand space into mutually exclusive regions using Algorithm-GR. Second part is an iterative procedure for finding the minimum value of \( Q \) which satisfies all the constraints in P2(\( Q \)) (feasibility problem once \( Q \) is fixed).

#### 2.3.1. Partitioning Algorithm-GR

One of the main components of our algorithm is the generation of mutually exclusive regions where allocation rules can be efficiently determined. Algorithm-GR (refer Fig. 4) can handle any number of customers though the number of regions in the partition grows exponentially with the number of customers. We utilize a recursive procedure that generates these regions efficiently. The procedure employs an enumerative approach and propagates the constraints due to presence of a condition. For example, when a member of the power set (condition) is included in a region, all subsets of that member are automatically ruled out in that region. This is done because we utilize non-dominated conditions in our representation. Each region is represented by a vector of size \( 2^n - 1 \) where each bit in the vector corresponds.
• Set highQ and lowQ to upper and lower bounds of Q
• List-of-Regions := NIL, Region-Vector := (0,0...0).
• Algorithm-GR (2^n − 1, Region-Vector, List-of-Regions, Power-Set K)  (Part 1)
  • WHILE (highQ − lowQ < ϵ)  (Part 2)
    – midQ := (highQ + lowQ)/2
    – Generate-Probabilities(midQ, List-of-Regions)
    – Solve P2(midQ)
    – IF P2(midQ) has feasible solution
      • highQ := midQ
    ELSE
      • lowQ := midQ
  • RETURN highQ

Fig. 3. Solution methodology.

Algorithm-GR (index, current-vector, list-of-regions, power-set)

BEGIN
  • IF index = 0
    – ADD current-vector to list-of-regions.
  • ELSE
    – IF current-vector[index] = x
      • Algorithm-GR (index-1, current-vector, list-of-regions, power-set)
    – ELSE
      • vector1 := current-vector
      • vector2 := current-vector
      • vector1[index] := 0
      • Algorithm-GR (index-1, vector1, list-of-regions, power-set)
      • vector2[index] := 1
      • FORALL j < index
        • IF S_j ⊂ S_{index} THEN vector2[j]=x
      • Algorithm-GR (index-1, vector2, list-of-regions, power-set)

END

Fig. 4. Algorithm for partitioning demand space.
Table 1
Regions for a three customer problem

<table>
<thead>
<tr>
<th>Regions</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(1,2)</th>
<th>(1,3)</th>
<th>(2,3)</th>
<th>(1,2,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

to a condition (the condition corresponding to \(\phi\) (the null set) is not included). A value of 1 for a bit in the region vector indicates the presence of a condition and a value of 0 indicates absence of a condition. A value of \(x\) indicates that the choice is ruled out due to earlier choices of conditions. For example, in Region-2 (refer Table 1 which shows the regions generated for a problem with three customers) all other conditions are ruled out because they are subsets of \(\{1, 2, 3\}\). So this region is defined by the inequality \(d_1 + d_2 + d_3 \leq Q\).

The algorithm recursively generates regions using index as a pointer to the current bit being set in the region vector. Once the index reaches the first condition then definition of the region is complete and the region vector is added to list-of-regions already present.

2.3.2. Binary search

Once the regions have been defined, probabilities \(x_i\) are generated using a Monte Carlo simulation technique for each value of \(Q\), and a binary search is performed to find \(Q^*\). A binary search is valid for this problem because the feasibility of P2 is monotone in \(Q\) if \(P2\) is feasible for some \(Q\) then it is feasible for all \(Q' > Q\) and likewise if \(P2\) is infeasible for some \(Q\) then it is infeasible for all \(Q' < Q\). This is not extremely clear from P2 but is obvious for P1 because if any quantity \(Q\) can satisfy the service constraints then for any \(Q' > Q\) one could use the same allocation as before and as a result get a feasible solution. Using similar arguments it can be shown that if P1 is infeasible for some \(Q\) then it is infeasible for all \(Q' < Q\). Since P1 and P2 are equivalent, the results hold for P2 and thus binary search is valid. It might seem to the reader that this kind of search could have been done in the first place on the original problem P1, however, without an allocation rule it is not possible to have a solution for P1.

Testing for feasibility is nothing but the phase 1 of the simplex method. Our computational study indicates that the simplex method (in cplex 4.0 optimizer) is extremely quick for problems having 3–5 customers and takes between 1 ms and 0.1 s on ULTRA-1 machine. In addition, for the case where there are only two customers checking for feasibility is extremely simple because there is only one decision variable and there are just two constraints. This can be done very efficiently by comparing \(X_1 = (\beta_1 - x_1 - x_2)/x_3\) and \(X_2 = (x_1 + x_2 + x_3 - \beta_2)/x_3\). If \(X_2 \geq X_1\) and \(X_1 \leq 1\) and \(X_2 \geq 0\) then the problem is feasible else it is not. We make use of the above in our solution procedure. In addition, we make our procedure more efficient when there are more than two customers by using the feasible solution for the previous value of \(Q\) (when value of \(Q\) is reduced). If that solution satisfies all the new constraints the problem is feasible for the new \(Q\) else we test for feasibility by utilizing phase 1 of the simplex method.

We utilize a binary search because it is easy to implement and provides a worst case bound on the number of \(Q\) to be tested. However, the efficiency of the binary search depends on the initial lower and upper bounds and the search can be made more efficient by using better values for the lower and upper bounds of \(Q\).

2.3.3. Bounds on \(Q\)

**Lemma 1.** The optimal procurement \(Q^* \geq \max \{ F_{1^{-1}}(\beta_1), \ldots, F_{n^{-1}}(\beta_n) \} \) where \(\beta_1, \ldots, \beta_n\) represent the service level requirements of the \(n\) customers and
$F_1, \ldots, F_n$ represent marginal cumulative density function of the respective customer demand.

**Proof.** $Q^*$ is the optimal solution so it satisfies each of the constraints of type (1) in P2. Now, for any customer $k$,

$$\sum_{i \in C_k} \beta_i(Q^*) + \sum_{i \in G_k, j \in H_k} p_{ij} \cdot \beta_i(Q^*) \geq \beta_k,$$

where $\beta_i(Q)$ represents the probabilities $\beta_i$ for a given value of $Q$. The above inequality implies that

$$\sum_{i \in C_k} \beta_i(Q^*) + \sum_{i \in G_k, j \in H_k} \beta_i(Q^*) \geq \beta_k$$

because $p_{ij} \leq 1$. It is to be noted that conditions that are satisfied in $C_k$ and those in $H_k$ satisfy the demand for customer $k$ (i.e., $d_k \leq Q$) in addition to satisfying demand for other customers. Thus, $F_k(Q^*) \geq \beta_k$. Since this has to be true for every customer $k$, we get $Q^* \geq \max \{F_1^{-1}(\beta_1), \ldots, F_n^{-1}(\beta_n)\}$.

The above result is intuitive because irrespective of the type of distribution and the correlation between the demand of different customers, the optimal procurement quantity has to be greater than the quantity that would be stocked if the manufacturer was trying to satisfy the demand for the most demanding customer.

**Lemma 2.** The optimal procurement quantity $Q^* \leq \sum_{i=1}^{n} F_k^{-1}(\beta_k)$ where $F_k$ is the marginal cumulative density function of customer ($k$)'s demand.

**Proof.** Suppose we assume that at the time of procurement each unit of inventory is already marked for a particular customer. Then, each customer can be treated independently. For each customer $i$ we require that $F_i(Q_i) \geq \beta_i$ where $Q_i$ is the inventory that is marked for customer $i$. In that case the optimal procurement quantity $Q^*$ is simply equal to $\sum_{k=1}^{n} F_k^{-1}(\beta_k)$. Since more demand can be satisfied with the same inventory when the inventory is not marked for customers as opposed to the case where they are already marked, the optimal procurement $Q^*$ will be lower. Hence, $Q^* \leq \sum_{k=1}^{n} F_k^{-1}(\beta_k)$. □

The above upper bound is equivalent to solving the problem for $n$ customers independently. Though this seems to be an attractive heuristic solution to the problem where customer demands are independent, in many instances, the solution obtained could be quite different from the optimal solution. To illustrate this point we consider an example with two customers.

**Example 1.** Consider two customers who have service level constraints $\beta_1 = 0.95$ and $\beta_2 = 0.90$. Let the customers have independent uniform demand between [0,100]. Find the optimal inventory at the manufacturer.

**Solution.** The optimal value of inventory comes out to be $Q^* = 145.31$ (utilizing our solution procedure) and the randomized allocation rule suggests that $\frac{1}{2}$ of the time, the quantity must be allocated to the first customer when demand falls in region (5). Compared to the upper bound ($Q^* = 95 + 90 = 185$), $Q^*$ is much lower. This indicates that even though the demands are independent, large efficiency could still be gained by solving P2 to optimality.

### 2.4. Two identical customers with joint uniform demand

In this section, we consider a special case of the problem when there are two customers whose demand distribution is jointly uniform. We derive the expression for the optimal procurement quantity and discuss the implications of changes in customer requirements and the demand distribution.

**Theorem 2.** The optimal procurement quantity $Q^* = 2D(1 - \sqrt{1 - \beta})$ when there are two identical customers with service level $\beta$, $0 < \beta < 1$ and their demand distribution is jointly uniform in the interval $((0, D), (0, D))$.

**Proof.** Since the demand distribution is jointly uniform, the joint density function is given by $f(x_1, x_2) = 1/D^2$ for $(0,0) \leq (x_1, x_2) \leq (D,D)$. In addition, the probabilities of the five regions (refer break Fig. 2) are given as follows:

$$x_1 = \frac{Q^2}{2D^2}, \quad x_2 = \frac{Q(D - Q)}{D^2}, \quad x_3 = \frac{Q(D - Q)}{D^2},$$

$$x_4 = \frac{(D - Q)^2}{D^2}, \quad x_5 = \frac{Q^2}{2D^2}.$$
For a given $Q$, the service constraints in P2 for the two customers are given by
\[
\frac{Q^2}{2D^2} + \frac{Q(D - Q)}{D^2} + p \frac{Q^2}{2D^2} \geq \beta
\]
and
\[
\frac{Q^2}{2D^2} + \frac{Q(D - Q)}{D^2} + (1 - p) \frac{Q^2}{2D^2} \geq \beta,
\]
where $p$ is the probability that demand of the first customer is satisfied in Region V (refer Fig. 2). On simplifying the above inequalities we get
\[
(1 - p)Q^2 - 2DQ + 2\beta D^2 \leq 0
\]
and
\[
pQ^2 - 2DQ + 2\beta D^2 \leq 0.
\]
The solutions to the above quadratic equations imply that
\[
D(1 - \sqrt{(1 - 2\beta(1 - p))})
\]
\[
\leq Q \leq D(1 + \sqrt{(1 - 2\beta(1 - p))})
\]
\[
\frac{1 - p}{p}
\]
\[
\leq Q \leq D(1 + \sqrt{(1 - 2\beta p)})
\]
Thus,
\[
Q^*(p) = \max \left\{ \frac{D(1 - \sqrt{(1 - 2\beta(1 - p))})}{1 - p}, \frac{D(1 - \sqrt{(1 - 2\beta p)})}{p} \right\}.
\]
The minimum value of $Q^*(p)$ is reached when $p=0.5$.

In that case $Q^* = 2D(1 - \sqrt{(1 - \beta))}$. \qed

**Lemma 3.** (i) The optimal quantity $Q^*$ is increasing and convex in $\beta$; (ii) The optimal quantity $Q^*$ is linear increasing in $D$.

**Proof.** (i) and (ii) can be easily verified by taking the first and second derivatives of $Q^* = 2D(1 - \sqrt{(1 - \beta))}$. \qed

The above result is interesting and intuitive in that the optimal procurement quantity is increasing in service level requirements and increases at greater rate at higher service levels (convexity). Typically, the service requirements for key customers is relatively high (> 0.90), as a result, small changes in the service level have a great impact on the optimal procurement quantity. The second result states that as the range in which demand could occur increases, the procurement quantities increases. In our computational study in the next section, we study some of these relationships under more general settings.

### 3. Computational results and insights

In this section, we provide results from our computational experience with the algorithm. We performed these experiments to validate our algorithm and also to get insights on the effect of covariance of demand, variance of demand, tightness of service constraints and similarity of service constraints on the optimal inventory level. We performed our experiments using cplex 3.1 for solving feasibility problems and used simulation to generate 5000 equally probable demand scenarios. We solved two and three customer problems in these experiments. These problems on an average took 23 s (for three customers) and 7 s (for two customers) on a SPARC-10 machine. The demand distribution was modeled as a multivariate normal distribution. Mean demand for customers was set to 80 and the high, low and medium variance correspond to a standard coefficient of variance (scv) of 0.250, 0.125 and 0.187, respectively.

#### 3.1. Effect of variance and correlation of demands

We find the following results (summarized in Table 2). (1) An increase in variance of demand increases the optimal inventory level in all cases. This result is quite intuitive because variance increases the uncertainty in the demand process and as a result more inventory is required. (2) The amount of inventory required is less under negatively correlated demands as compared to independent and positively correlated demands. An intuitive explanation for this result is as follows. Under negatively correlated demand, the individual demands vary in opposite directions as a result when one demand is high, the other is low.
such a situation, it is possible to keep lower inventory and satisfy high service levels because in any period at most one customer’s demand will be high. (3) Inventory change due to increase in variance increases monotonically with \( \rho \) (coefficient of correlation). This can be explained by a similar argument as above in that under negatively correlated demand more variance takes demand from customers in opposite directions as a result has less impact on inventory requirements as compared to the case where demands are positively correlated. In some sense, the effect of variance adds up in a positively correlated demand setting.

### 3.2. Service constraints

We find the following results (summarized in Table 3). (1) An increase in service requirements increases the inventory level. This result is quite intuitive because an increase service requirements naturally requires one to hold more inventory. (2) Change in inventory due to increase in service requirements increases monotonically with \( \rho \). (3) The increase in inventory is convex in service requirements. We generally find that as the service constraints become more and more tight the inventory levels increase faster.

### 3.3. Similarity in customer service requirements

We find the following results (summarized in Table 4 and Table 5). (1) An increase in service requirements leads to increase in inventory level in most cases. However, in some cases more customers can be promised the maximum service level without any additional inventory. For example, compare the inventory for (95, 90, 90) and (95, 95, 90) in Table 4. We feel that this may be due to the particular demand distribution and scenarios that we chose because otherwise one would expect the inventory to increase with
Table 5
Optimal inventory for multi-variate normal demand with mean 80 and standard coefficient of variation \( \text{scv} = 0.25 \), service level \( \beta_i \) and coefficient of correlation \( \rho \). (two customers)

<table>
<thead>
<tr>
<th>Service level ( \beta_i )</th>
<th>( \rho = -0.4 )</th>
<th>( \rho = 0 )</th>
<th>( \rho = 0.4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.98, 0.98)</td>
<td>181.81</td>
<td>209.41</td>
<td>226.04</td>
</tr>
<tr>
<td>(0.95, 0.98)</td>
<td>178.67</td>
<td>201.20</td>
<td>215.85</td>
</tr>
<tr>
<td>(0.90, 0.98)</td>
<td>175.01</td>
<td>192.89</td>
<td>203.93</td>
</tr>
<tr>
<td>(0.95, 0.95)</td>
<td>176.32</td>
<td>195.49</td>
<td>208.21</td>
</tr>
<tr>
<td>(0.95, 0.90)</td>
<td>173.22</td>
<td>189.59</td>
<td>198.89</td>
</tr>
<tr>
<td>(0.90, 0.90)</td>
<td>170.74</td>
<td>183.70</td>
<td>192.18</td>
</tr>
</tbody>
</table>

increase in service requirements. (2) Correlation of demand plays an important role in the amount of inventory required. For example, a negatively correlated demand can satisfy 98% service for all customers with a lower inventory than 90% for positively correlated demand.

4. Conclusions

In this paper we analyze the problem of optimal procurement and allocation of a product while facing stochastic demand from multiple customers. We model individual service constraints and provide an algorithm for computing the optimal procurement quantity based on an optimal random allocation policy that we derive in this paper. We utilize the inherent structure of the problem in terms of the allocation rules to generate the optimal solution. For a special case when there are two identical customers and their demand is jointly uniform, we derive an analytical expression for the optimal procurement quantity. In our computational analysis we study the influence of variance and covariance of demand, similarity in customer service requirements and severity of service constraints on the optimal inventory level. Our algorithm grows exponentially with an increase in the number of customers and is not suited for managing a large customer base. Current research aims at improving the region generation mechanism so that the number of regions do not grow exponentially with the number of customers.

Acknowledgements

The authors wish to thank the anonymous referee and the associate editor for their insightful comments that have greatly improved the paper in terms of readability as well as its contribution.

References