INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD • INDIA

Heuristics for the multi-product satiating newsboy problem

Avijit Khanra
Chetan Soman

W.P. No. 2014-01-02
January 2014

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380 015
INDIA

Heuristics for the multi-product satiating newsboy problem

Avijit Khanra*and Chetan Soman

Indian Institute of Management, Ahmedabad-380015, India

Abstract

Due to the preference of satiation (of the profit target) over maximization (of expected profit) in uncertain situations, the newsboy problem with the satiation objective is an important practical problem. In the multi-product setting, practically usable solution methods are available only for two-product and some restricted three-product problems. We develop heuristics to solve larger problems (more number of products). Two search-based heuristics are developed and tested with a large number of versatile test problems. These heuristics solve upto five-product problems in reasonable time with good accuracy.

1 Introduction

The newsboy problem is about balancing the penalties associated with under-stocking and over-stocking by deciding the stocking quantity in the face of uncertain demand. Traditionally, minimization of expected demand-supply mismatch cost (or maximization of expected profit) is considered as the objective (Silver et al., 1998, chap. 10). However, this objective does not fit into many real-life situations. A number of different objectives have been considered in the newsboy literature. For example, Ismail & Louderback (1979) considered the satiation objective (i.e., maximization of the probability of achieving a given profit target), Anvari (1987) maximized the firm value using the capital asset pricing model, Lin & Kroll (1997) considered the satiation constraint (i.e., the probability of profit exceeding a given target does not fall below a given level), Chen & Chuang (2000) considered the service-level constraint, Gotoh & Takano (2007) minimized the conditional value-at-risk (CVaR), Wu et al. (2009) maximized a mean-variance based objective function, and Oberlaender (2011) maximized exponential utility of profit. A comprehensive review of these different objectives (considered in the newsboy problem) can be found in Khouja (1999); Qin et al. (2011).

In this paper, our focus is on the newsboy problem with the satiation objective. When the decision maker is target oriented (and if the target is in terms of profit), maximization of the probability of achieving the profit target is a reasonable approximation for the decision maker's objective (Irwin & Allen, 1978). Target orientation is quite common in organizational set-ups (Abbas et al., 2009); a manager faced with a target is likely to focus on fulfilling the target.

*Corresponding author. *Tel:* +91 7405696960 *Email address:* avijitk@iimahd.ernet.in

At an individual level, too, target orientation is not uncommon; there it can be viewed as risk aversion (Anvari & Kusy, 1990).

Irwin & Allen (1978) were the first to considered the satiation objective in the newsboy problem. Ismail & Louderback (1979); H.-S. Lau (1980) studied the single-product satiating newsboy problem with normally distributed demand; H.-S. Lau (1980) found a closed-form solution for the optimum order quantity. Later, Sankarasubramanian & Kumaraswamy (1983) solved the single-product problem with uniform and exponential demand distributions. The case of zero stock-out cost turned out to be easier; the optimum order quantity was found to be independent of the demand distribution function (Norland, 1980; H.-S. Lau, 1980).

Researchers combined satiation objective with the other extensions of the classical newsboy model[1]. For example, A. H.-L. Lau & Lau (1988b) considered the price dependency of demand, Khouja (1995) considered the end-season price mark-down, Khouja & Robbins (2003) considered the advertisement sensitive demand, and Victor et al. (2011) considered competition among newsboys. This list in merely indicative; there are more such studies. Among these different extensions, our interest is on the multi-product problems.

Research into the multi-product satiating newsboy problem (MPSNP) started with the work of A. H.-L. Lau & Lau (1988a). They studied the two-product problem with zero stock-out costs and uniform demand; optimum solution was obtained for the case of identical products. Li et al. (1990) solved the same problem for the case of non-identical products; an efficient algorithm was developed for finding the optimum order quantities. Later, they considered the same problem with independent exponential product demands (Li et al., 1991).

After the works of A. H.-L. Lau & Lau (1988a); Li et al. (1990, 1991), there has been no progress in the field of MPSNP until recently. Khanra (2014) studied the general MPSNP (i.e., non-zero stock-out costs and no restrictions on demand) by considering it as a discrete optimization problem (unlike the previous studies, which used the continuous modelling). He identified a key property of the MPSNP, that led to the development of a computational method for finding the satiation probability (i.e., probability of meeting the profit target); satiation probability was maximized by complete enumeration of the search space. He also pointed out the difficulties associated with the continuous formulation of the MPSNP.

Time requirement for the enumeration-based optimization method of Khanra (2014) grows rapidly as the number of products increases. The method is not suitable for the MPSNP with three or more products. In this paper, we develop heuristics, that search in restricted space, to solve larger MPSNP (i.e., more number of products). Accuracy of the heuristics are tested by comparing with the optimum solution (obtained using the method of Khanra, 2014).

Remainder of this paper is organized as follows. For the ease of readers, key results of Khanra (2014) are revisited in Section 2. Our heuristics (named as demand rescaling heuristic and diagonal search heuristic) are presented in Section 3 and 4. Heuristic performances are discussed in Section 5. We conclude in Section 6.

[1]See Khouja (1999) for a list of different extensions of the classical newsboy model.

2 The discrete formulation

Following notations are used in this paper.

T	Profit target.
n	Number of products (positive integer).
m_i	Unit profit for the i^{th} product (positive).
c_i	Unit purchase cost less salvage value for the i^{th} product (positive).
s_i	Unit stock-out goodwill loss for the i^{th} product (positive).
X_i	Stochastic demand for the i^{th} product (integer-valued).
a_i	Lower limit of X_i (non-negative integer).
b_i	Upper limit of X_i (positive integer).
$p_i()$	Marginal probability mass function of X_i.
$P_i()$	Marginal cumulative distribution function of X_i.
X	Demand vector. $X = (X_i) = (X_1, X_2, \ldots, X_n)$.
Ω	Sample space of X. $\Omega = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_n$, where Ω_i is the sample space of X_i, i.e., $\Omega_i = \{a_i, a_i + 1, \ldots, b_i\}$ for $i = 1, 2, \ldots, n$.
$p()$	Probability mass function of X. For independent demand, $p(x) = \prod_{i=1}^{n} p_i(x_i)$.
Q_i	Order quantity of the i^{th} product (non-negative integer).
Q	Order quantity vector. $Q = (Q_i) = (Q_1, Q_2, \ldots, Q_n)$.
$P_T(Q)$	Satiation probability for ordering decision, Q.

Profit, Π in the MPSNP is stochastic as it depends on stochastic X; it also depends on Q. $\Pi(Q, X)$ is the sum of individual product profits, $\Pi_i(Q_i, X_i)$ for $i = 1, 2, \ldots, n$.

$$\begin{aligned}\Pi_i(Q_i, X_i) = &\text{ sales revenue } - \text{ over-stocking cost (if any) } - \text{ under-stocking cost (if any)} \\ &= m_i \min\{Q_i, X_i\} - c_i \max\{0, Q_i - X_i\} - s_i \max\{0, X_i - Q_i\}. \end{aligned} \tag{1}$$

Let us define the indicator function for $\Pi(Q, X) \geq T$, $I_{\Pi(Q,X)\geq T}$ as

$$I_{\Pi(Q,X)\geq T} = \begin{cases} 1 & \text{if } \Pi(Q, X) = \sum_{i=1}^{n} \Pi_i(Q_i, X_i) \geq T, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Then our objective in the MPSNP is to

$$\begin{aligned}\underset{Q\in\mathbb{N}_0^n}{\text{maximize}}\ P_T(Q) &= \sum_{x\in\Omega} I_{\Pi(Q,x)\geq T}\ p(x) \\ &= \sum_{x_1=a_1}^{b_1} \sum_{x_2=a_2}^{b_2} \cdots \sum_{x_n=a_n}^{b_n} I_{\Pi(Q,x)\geq T}\ p(x), \end{aligned} \tag{3}$$

where $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ and $\mathbb{N}_0^n$ is the corresponding n-dimensional space.

Following three properties of the MPSNP were established by Khanra (2014). With the help of these properties, he calculated the satiation probability and maximized it.

Property 1. *For any T, $P_T(Q)$ maximizing ordering decision, $Q^* \in \Omega$.*

Property 2. *Any $T \leq \underline{T} = \sum_{i=1}^{n} \max\{\Pi_i(\lfloor Q_{0i} \rfloor, b_i),\ \Pi_i(\lceil Q_{0i} \rceil, a_i)\}$ is achievable with certainty and any $T > \overline{T} = \sum_{i=1}^{n} m_i b_i$ is unachievable, where $Q_{0i} = \{(m_i + c_i)a_i + s_i b_i\}/(m_i + c_i + s_i)$ for $i = 1, 2, \ldots, n$.*

Property 3. *$I_{\Pi(Q,x) \geq T} = 1$ if and only if $x \in \text{co}D \cap \Omega$, where $\text{co}D$ is the convex hull of D. If $\sum_{i=1}^{n} m_i Q_i \geq T$, $D = \bigcup_{i=1}^{n} \{\underline{x}^{(i)}, \overline{x}^{(i)}\}$, else $D = \emptyset$, where*

$$\underline{x}^{(i)}(Q,T) = \left(Q_1, \ldots, Q_{i-1}, Q_i - \frac{\sum_{i=1}^{n} m_i Q_i - T}{m_i + c_i}, Q_{i+1}, \ldots, Q_n\right)$$
$$\overline{x}^{(i)}(Q,T) = \left(Q_1, \ldots, Q_{i-1}, Q_i + \frac{\sum_{i=1}^{n} m_i Q_i - T}{s_i}, Q_{i+1}, \ldots, Q_n\right)$$

for $i = 1, 2, \ldots, n$.

Property 1 ensures computation in finite time. Property 2 identifies the target profit range for which the MPSNP has trivial solution[2]. *From now on, we assume that $T \in (\underline{T}, \overline{T})$ so that we work only with the non-trivial MPSNP.* Property 3 identifies the subset of Ω for which the target achievement indicator is positive. By Property 3, $P_T(Q)$ can be expressed as

$$P_T(Q) = \sum_{x \in \text{co}D \cap \Omega} p(x), \quad \text{where } D \text{ is defined in Property 3}^3. \tag{4}$$

Using (4), Khanra (2014) devised a computational method that finds $P_T(Q)$ for given T, Q, parameters (a_i, b_i, m_i, c_i, s_i for $i = 1, 2, \ldots, n$), and p (n-dimensional pmf array). An improved method was developed for the case of independent demand; then p_i for $i = 1, 2, \ldots, n-1$ (1-dimensional pmf arrays) and P_n (1-dimensional cdf array) were used instead of p.

The MPSNP was optimally solved by calculating $P_T(Q)\ \forall Q \in \Omega$ (as Ω contains the optimum solution by Property 1) and selecting the best. Since $\Omega = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_n$ is a product space, $|\Omega|$ grows exponentially with n. Naturally, this enumeration-based optimization fails (to provide the optimum solution in reasonable time[4]) for large n (actually, it failed at $n = 3$). We develop heuristics that search is restricted space so that larger problems can be solved.

[2]If $T \leq \underline{T}$ or $T \geq \overline{T}$, the MPSNP has trivial solution. If $T \leq \underline{T}$, the i^{th} element of an ordering decision that guarantees the target achievement is $\lfloor Q_{0i} \rfloor$ if $\Pi_i(\lfloor Q_{0i} \rfloor, b_i) \geq \Pi_i(\lceil Q_{0i} \rceil, a_i)$ and $\lceil Q_{0i} \rceil$ if $\Pi_i(\lfloor Q_{0i} \rfloor, b_i) \leq \Pi_i(\lceil Q_{0i} \rceil, a_i)$. Any $T > \overline{T}$ is unachievable and $T = \overline{T}$ is achieved only by $Q = (b_i)$. $\underline{T}$ and $\overline{T}$ are referred to as the maximum assured target and the maximum achievable target (Khanra, 2014).

[3]D is referred to as the set of terminal demand points. $\underline{x}^{(i)}$ and $\overline{x}^{(i)}$ are the terminal demand points for the i^{th} product (for $i = 1, 2, \ldots, n$). The name terminal demand is due to the fact that $I_{\Pi(Q,x) \geq T} = 0$ if $x_i \notin [\underline{x}_i^{(i)}, \overline{x}_i^{(i)}]$ for any $i = 1, 2, \ldots, n$ (Khanra, 2014).

[4]There can be different connotations of reasonable time. We set the limit at one hour, i.e., if a method, on an average, takes one hour or less for completion, we say that the method takes reasonable time.

3 Demand rescaling heuristic

Let us consider a two-product problem. Let $a_1 = 0, b_1 = 100, m_1 = 5, c_1 = 20, s_1 = 10$ and $a_2 = 500, b_2 = 1000, m_2 = 1, c_2 = 5, s_2 = 1$. In the enumeration-based optimization, $P_T(Q)$ is calculated $|\Omega| = (b_1 - a_1 + 1) \times (b_2 - a_2 + 1) = 50601$ times. By changing the quantity measurement units (denoted by u_1 and u_2), the same problem can be modelled as $a'_1 = 0, b'_1 = 10, m'_1 = 50, c'_1 = 200, s'_1 = 100$ and $a'_2 = 10, b'_2 = 20, m'_2 = 50, c'_2 = 250, s'_2 = 50$ (i.e., $u'_1 = 10 \times u_1$ and $u'_2 = 50 \times u_2$). Probability mass function needs to be modified accordingly. No change happens to the profit target. With these new parameter values, we need to calculate $P_T(Q)$ only 121 times to find the optimum (the original count is about 418 times this new count). However, we loose details in this process (i.e., Q can take fewer values and gap between two values is larger) and optimum solution of the rescaled problem is unlikely to be the optimum for the original problem. Demand rescaling heuristic (DRH) provides a way to keep the details and yet benefit from the rescaling technique.

Let $(Q'^*_1, Q'^*_2) = (4, 16)$ be the optimum solution of the rescaled problem. Then the optimum solution of the original problem, (Q^*_1, Q^*_2) is unlikely to be far from $(10 \times Q'^*_1, 50 \times Q'^*_2) = (40, 800)$, where 10 and 50 are the scale factors (let us denote them by k_1 and k_2). Let us construct a rectangle around $(k_1 Q'^*_1, k_2 Q'^*_2)$ of the form $R(r) = \{(Q_1, Q_2) : k_1(Q'^*_1 - r) \leq Q_1 \leq k_1(Q'_1 + r),\ k_2(Q'^*_2 - r) \leq Q_2 \leq k_2(Q'_2 + r), r > 0\}$. If we maximize $P_T(Q)$ for $Q \in \Omega \cap R(r)$, we can expect the resultant solution to be better than $(k_1 Q'^*_1, k_2 Q'^*_2)$. This is how DRH keeps the details. If $r = 1$, $\Omega \cap R(1) = \{(Q_1, Q_2) : (Q_1, Q_2) \in \mathbb{N}^2_0,\ 30 \leq Q_1 \leq 50,\ 750 \leq Q_2 \leq 850\}$. Then for the second-stage optimization, $P_T(Q)$ is calculated 2121 times. In the two stages, $P_T(Q)$ is calculated 2242 times (the original count is about 23 times this count).

The chances of $(Q^*_1, Q^*_2) \in \Omega \cap R(r)$ increases in r. However, computation time for the second-stage optimization increases in r too. We need to balance these factors by choosing r properly. Let $R'^* = \{(Q'_1, Q'_2) : Q'_1 \in \{\lfloor Q^*_1/k_1 \rfloor, \lceil Q^*_1/k_1 \rceil\}, Q'_2 \in \{\lfloor Q^*_2/k_2 \rfloor, \lceil Q^*_2/k_2 \rceil\}\}$. It is likely that $(Q'^*_1, Q'^*_2) \in R'^*$. Now, we do not know (Q^*_1, Q^*_2) and after the first-stage optimization, we know (Q'^*_1, Q'^*_2). With this information, if we assume that $(Q'^*_1, Q'^*_2) \in R'^*$, $R' = \{(Q'_1, Q'_2) : Q'_1 \in \{Q'^*_1 - 1, Q'^*_1, Q'^*_1 + 1\},\ Q'_2 \in \{Q'^*_2 - 1, Q'^*_2, Q'^*_2 + 1\}\}$ is the smallest set that envelopes R'^*. Projection of R' into Ω is a good choice for the second-stage optimization search space. Note that $\Omega \cap R(1)$ is this search space. Thus, we take $r = 1$.

We need to choose the scale factors (k) before we can implement DRH. Since accuracy of the DRH solution is unlikely to be affected by this choice, we choose k that minimizes the total computation time. There is one more question. Can we not apply this rescaling technique to the second-stage optimization problem? If we do so, computation time can be reduced further and the solution accuracy is unlikely to be affected seriously. Hence, we need to decide the number of rescaled problems to solve (denoted by z) before the final search (the second-stage optimization in our example) and the scale factors for each product in each rescaled problem (denoted by $k_i^{(j)}$

for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z$). We take

$$z = \left\lfloor \sum_{i=1}^{n} \ln\left(\frac{b_i - a_i}{2}\right) \right\rceil - 1, \quad \text{where } \lfloor y \rceil \text{ denotes the nearest integer to } y. \tag{5}$$

$$k_i^{(j)} = \left(\frac{b_i - a_i}{2}\right)^{1-j/(z+1)} \quad \text{for } i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, z. \tag{6}$$

The above choice approximately minimizes the total computation time in DRH (see Appendix A for the details).

In DRH, we solve z number of rescaled problems. After each rescaled problem is solved, we get a trust region (denoted by $\Omega \cap R(1)$ in our example); we believe that the optimum is in the trust region. Search space for the next rescaled problem is a scaled-down version of the current trust region. In the very beginning, the trust region is Ω. Thus, search space for the first rescaled problem is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor a_i/k_i^{(1)} \rceil \le Q_i \le \lfloor b_i/k_i^{(1)} \rceil \text{ for } i = 1, 2, \ldots, n\}$.

Let $Q^{(j)^*}$ for $j = 1, 2, \ldots, z-1$ be the optimum solution of the j^{th} rescaled problem. The corresponding trust region is $\{Q : Q \in \mathbb{N}_0^n,\ L_i^{(j)} \le Q_i \le R_i^{(j)} \text{ for } i = 1, 2, \ldots, n\}$, where $L_i^{(j)} = \max\{a_i, k_i^{(j)}(Q_i^{(j)^*} - 1)\}$ and $R_i^{(j)} = \min\{b_i, k_i^{(j)}(Q_i^{(j)^*} + 1)\}$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z-1$. Search space of the next rescaled problem is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor L_i^{(j)}/k_i^{(j+1)} \rceil \le Q_i \le \lfloor R_i^{(j)}/k_i^{(j+1)} \rceil \text{ for } i = 1, 2, \ldots, n\}$. Since $k_i^{(j+1)} < k_i^{(j)}$ for every $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z-1$, the trust region gets finer after each rescaled problem is solved.

The rescaled problems have their own cost parameters ($m_i^{(j)}$, $c_i^{(j)}$, $s_i^{(j)}$) and probability mass functions ($p^{(j)}$). Cost parameters are defined as: $m_i^{(j)} = k_i^{(j)} m_i$, $c_i^{(j)} = k_i^{(j)} c_i$, and $s_i^{(j)} = k_i^{(j)} s_i$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z$. Demand space of the j^{th} rescaled problem, $\Omega^{(j)} = \{x^{(j)} : x^{(j)} \in \mathbb{N}_0^n,\ \lfloor a_i/k_i^{(j)} \rceil \le x_i^{(j)} \le \lfloor b_i/k_i^{(j)} \rceil \text{ for } i = 1, 2, \ldots, n\}$ for $j = 1, 2, \ldots, z$. Note that the search space of a rescaled problem is smaller than its demand space (except for the first rescaled problem, where they are the same). Probability mass functions, $p^{(j)}(x^{(j)})\ \forall x^{(j)} \in \Omega^{(j)}$ for $j = 1, 2, \ldots, z$ are calculated as

$$p^{(j)}(x^{(j)}) = \sum_{x \in R(x^{(j)})} p(x) = \sum_{x_1=l_1}^{r_1} \sum_{x_2=l_2}^{r_2} \cdots \sum_{x_n=l_n}^{r_n} p(x_1, x_2, \ldots, x_n), \tag{7}$$

$$\text{where } R(x^{(j)}) = \left\{x : x \in \Omega,\ k_i^{(j)}(x_i^{(j)} - 1/2) \le x_i < k_i^{(j)}(x_i^{(j)} + 1/2) \text{ for } i = 1, 2, \ldots, n\right\},$$

$$l_i = \min\left\{x_i : x_i \in \Omega_i,\ x_i \ge k_i^{(j)}(x_i^{(j)} - 1/2)\right\} \text{ for } i = 1, 2, \ldots, n, \text{ and}$$

$$r_i = \max\left\{x_i : x_i \in \Omega_i,\ x_i < k_i^{(j)}(x_i^{(j)} + 1/2)\right\} \text{ for } i = 1, 2, \ldots, n.$$

The above definition yields a valid probability mass function (see Appendix B for a proof).

Let $Q^{(z)^*}$ be the optimum solution of the last (i.e., z^{th}) rescaled problem. Since there is no more rescaling, the final search space is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor L_i^{(z)} \rceil \le Q_i \le \lfloor R_i^{(z)} \rceil \text{ for } i = 1, 2, \ldots, n\}$, where $L_i^{(z)} = \max\{a_i, k_i^{(z)}(Q_i^{(z)^*} - 1)\}$ and $R_i^{(z)} = \min\{b_i, k_i^{(z)}(Q_i^{(z)^*} + 1)\}$ for $i = 1, 2, \ldots, n$. Cost parameters and probability mass function of the final search is same as that of the original

problem. The profit target is always the same.

Based on the above description, Procedure 1 is proposed for the implementation of DRH. We assume that $z \geq 1$ so that the application of DRH is relevant. Comments in Procedure 1 (and the rest) are enclosed inside $\langle\rangle$ and the font colour is gray.

Procedure 1 Demand rescaling heuristic

Input: n, T, parameters $\langle a_i, b_i, m_i, c_i, s_i \text{ for } i = 1, 2, \ldots, n\rangle$, and $p()$ $\langle n$-dimensional pmf array$\rangle$

Output: Q_{drh} $\langle$DRH solution$\rangle$

1. $z \leftarrow \lfloor \sum_{i=1}^{n} \ln((b_i - a_i)/2) \rceil - 1$ $\langle$number of rescaled problems$\rangle$
2. $j \leftarrow 0$ $\langle$rescaled problems solved$\rangle$
3. **for** $i = 1$ **to** n **do**
4. $\quad L_i \leftarrow a_i$, $R_i \leftarrow b_i$ $\langle$the initial trust region$\rangle$
5. **end for**
6. **while** $j < z$ **do**
7. $\quad j \leftarrow j + 1$
8. $\quad$ **for** $i = 1$ **to** n **do**
9. $\quad\quad k_i \leftarrow ((b_i - a_i)/2)^{1-j/(z+1)}$ $\langle$scale factors for the j^{th} rescaled problem$\rangle$
10. $\quad$ **end for**
11. $\quad Q' \leftarrow Q'^{*}$ $\langle$Optimum solution of the j^{th} rescaled problem, obtained using Procedure 2$\rangle$
12. $\quad$ **for** $i = 1$ **to** n **do**
13. $\quad\quad L_i \leftarrow \max\{a_i, k_i(Q'_i - 1)\}$, $R_i \leftarrow \min\{b_i, k_i(Q'_i + 1)\}$ $\langle$refined trust region$\rangle$
14. $\quad$ **end for**
15. **end while**
16. **for** $i = 1$ **to** n **do**
17. $\quad L_i \leftarrow \lfloor L_i \rceil$, $R_i \leftarrow \lfloor R_i \rceil$ $\langle$the final search space$\rangle$
18. **end for**
19. $Q_{drh} \leftarrow Q^{*}$ such that $P_T(Q^{*}) = \max\{P_T(Q) : Q \in \mathbb{N}_0^n,\ L_i \leq Q_i \leq R_i \text{ for } i = 1, 2, \ldots, n\}$ $\langle$Optimum solution in the final search space, obtained by complete enumeration. $P_T(Q)$ is computed using satiation probability calculation (SPC) algorithm in Khanra (2014).$\rangle$
20. **return** Q_{drh}

$\langle$Note: For independent demand, $p()$ is replaced by $p_i()$ for $i = 1, 2, \ldots, n$ (1-dimensional pmf arrays). In line number 11, Procedure 4 is called instead of Procedure 2.$\rangle$

Procedure 1 utilizes Procedure 2 for optimization in the rescaled problems. Procedure 2 rescales the original parameters and probability mass function and then optimizes by complete enumeration of the refined search space. Optimization in the final search space (in Procedure 1) is also enumeration-based.

Procedure 4 in Appendix C is the "independent demand" version of Procedure 2. Resource requirements (both time and space) of Procedure 2 is significantly higher than that of its independent demand counterpart. Procedure 2 rescales and stores n-dimensional pmf array, whereas Procedure 4 rescales and stores n number of 1-dimensional pmf arrays; this process takes place every time a rescaled problem is solved.

Procedure 2 Optimization in rescaled problem

Input: n, T, parameters, $p()$, trust region $\langle L_i, R_i$ for $i = 1, 2, \ldots, n\rangle$, and scale factors $\langle k_i$ for $i = 1, 2, \ldots, n\rangle$

Output: Q'^* $\langle$rescaled problem solution$\rangle$

1. **for** $i = 1$ **to** n **do**
2. $\quad a'_i \leftarrow \lfloor a_i/k_i \rceil$, $b'_i \leftarrow \lfloor b_i/k_i \rceil$ $\langle$rescaled demand space$\rangle$
3. $\quad m'_i \leftarrow k_i m_i$, $c'_i \leftarrow k_i c_i$, $s'_i \leftarrow k_i s_i$ $\langle$rescaled cost parameters$\rangle$
4. $\quad L'_i \leftarrow \lfloor L_i/k_i \rceil$, $R'_i \leftarrow \lfloor R_i/k_i \rceil$ $\langle$refined search space$\rangle$
5. **end for**
6. $p'() \leftarrow 0$ $\langle$initializing rescaled pmf array of size $(b'_1 - a'_1 + 1) \times \cdots \times (b'_n - a'_n + 1)\rangle$
7. **for** $x_1 = a'_1$ **to** b'_1 **do**
8. $\quad l_1 \leftarrow \min\{y : y \in \{a_1, a_1 + 1, \ldots, b_1\},\ y \geq k_1(x_1 - 1/2)\}$ $\langle$by binary search$\rangle$
9. $\quad r_1 \leftarrow \max\{y : y \in \{a_1, a_1 + 1, \ldots, b_1\},\ y < k_1(x_1 + 1/2)\}$ $\langle$by binary search$\rangle$
10. $\quad$.
11. $\quad$.
12. $\quad$ **for** $x_n = a'_n$ **to** b'_n **do**
13. $\quad\quad l_n \leftarrow \min\{y : y \in \{a_n, a_n + 1, \ldots, b_n\},\ y \geq k_n(x_n - 1/2)\}$ $\langle$by binary search$\rangle$
14. $\quad\quad r_n \leftarrow \max\{y : y \in \{a_n, a_n + 1, \ldots, b_n\},\ y < k_n(x_n + 1/2)\}$ $\langle$by binary search$\rangle$
15. $\quad\quad p'(x_1, x_2, \ldots, x_n) \leftarrow \sum_{y_1=l_1}^{r_1} \sum_{y_2=l_2}^{r_2} \cdots \sum_{y_n=l_n}^{r_n} p(y_1, y_2, \ldots, y_n)$
16. $\quad$ **end for**
17. $\quad$.
18. $\quad$.
19. **end for**
20. $Q'^* \leftarrow Q^*$ such that $P_T(Q^*) = \max\{P_T(Q) : Q \in \mathbb{N}_0^n,\ L'_i \leq Q_i \leq R'_i$ for $i = 1, 2, \ldots, n\}$ $\langle$Optimum solution in the refined search space, obtained by complete enumeration. $P_T(Q)$ is computed using SPC Algorithm with rescaled parameters and rescaled pmf array.$\rangle$
21. **return** Q'^*

DRH saves time by reducing the search space. In enumerative optimization, $P_T(Q)$ is calculated $|\Omega| = \prod_{i=1}^{n}(b_i - a_i + 1)$ times. An under-estimation of this count for DRH is $2^n e \ln(\prod_{i=1}^{n}(b_i - a_i)/2)$ (see Appendix B). The ratio is about $y/(e \ln(y))$, where $y = \prod_{i=1}^{n}(b_i - a_i)/2$. This gives an idea of the time saving that DRH achieves.

4 Diagonal search heuristic

DRH saves time by reducing the search space. However, DRH search space continues to grow exponentially with the number of products (like the search space in enumerative optimization). Thus, we can expect it to fail at some point (i.e., some value of n). Based on the numerical results in the next section, it fails at $n = 5$ for independent demand and at $n = 3$ for dependent demand. To overcome this issue, diagonal search heuristic (DSH) restricts its search to 1-dimension and the search space does not grow with n.

Let us identify which factors influence the value of $P_T(Q)$. By (4), $P_T(Q) = Pr\{X \in \text{co}D \cap \Omega\}$, where co$D$ is the convex hull of D (defined in Property 3). The value of $P_T(Q)$ increases with the size of co$D \cap \Omega$ and the values of $p(x)$ for $x \in \text{co}D \cap \Omega$. Since Ω is fixed, size of co$D \cap \Omega$ depends on the size of co$D$ and its location w.r.t. Ω.

Size of co$D$ increases with the distances between points in D, which increases with profit cushion[5], $PC_T(Q) = \sum_{i=1}^{n} m_i Q_i - T$. D consists of n pair of points lying on n axes of $\mathbb{R}^n$ when Q is considered as the origin. Furthermore, one point of each pair lies on the positive-side of an axis, while the other point lies on the negative-side. Thus, in some sense, Q is at the centre of co$D$. Then the location of Q in Ω is a good indicator for the location of co$D$ w.r.t. Ω. For the same reason, the value of $p(Q)$ is a good indicator for the values of $p(x)$ for $x \in \text{co}D \cap \Omega$. Thus, we can say that $P_T(Q)$ is high if i) $p(Q)$ is high, ii) Q is "centrally located" in Ω, and iii) $PC_T(Q)$ is large. See the definition of D to verify the above arguments.

Let us consider the first factor. Let $p(Q_M) = \max\{p(x) : x \in \Omega\}$; Q_M is referred to as the mode. Then $p(Q)$ highest if $Q = Q_M$. Now, let us consider the second factor. Location of Q has to be such that co$D \cap \Omega$ is large. Following the discussion in the previous paragraph, Q_i lies between the i^{th} pair of points in D (denoted by $\underline{x}^{(i)}$ and $\overline{x}^{(i)}$ in Property 3) for every $i = 1, 2, \ldots, n$. $\underline{x}^{(i)}$ lies towards a_i and $\overline{x}^{(i)}$ lies towards b_i. Distances of $\underline{x}_i^{(i)}$ and $\overline{x}_i^{(i)}$ from Q_i are $PC_T(Q)/(m_i + c_i)$ and $PC_T(Q)/s_i$ respectively. If these distances are in the ratio, $(Q_i - a_i) : (b_i - Q_i)$, i.e., $(Q_i - a_i)(m_i + c_i) = (b_i - Q_i)s_i$, the overlap between $[\underline{x}_i^{(i)}, \overline{x}_i^{(i)}]$ and $[a_i, b_i]$ is highest. Due to the structure of D, co$D \cap \Omega$ is largest if such overlaps are highest for every $i = 1, 2, \ldots, n$. If Q satisfies these conditions, i.e., $Q_i = \{(m_i + c_i)a_i + s_i b_i\}/(m_i + c_i + s_i)$ for $i = 1, 2, \ldots, n$, we write $Q_C = \lfloor Q \rceil$. Q_C is "centrally located" in Ω.

$Q = Q_M$ is the best choice for maximizing $P_T(Q)$ if the first factor is considered alone, while $Q = Q_C$ is the best choice if the second factor is considered alone. It is unlikely that $Q_M = Q_C$; hence, we need to combine Q_M and Q_C. Let us refer to the resultant Q as the base, Q_B. Ideally, we should search along the line connecting Q_M and Q_C and choose Q with the highest $P_T(Q)$ value as Q_B. However, this can be time consuming, particularly for large n. We use a simpler way (less time-taking), we take $Q_B = \lfloor \lambda Q_M + (1 - \lambda) P_T(Q_C) \rceil$, where $\lambda = P_T(Q_M)/\{P_T(Q_M) + P_T(Q_C)\}$. If $P_T(Q_M) = P_T(Q_C) = 0$, $\lambda = 0.5$.

Now, we need to combine Q_B with the third factor to get a Q with better $P_T(Q)$ value. To accomplish this, we search along a direction, in which $PC_T(Q)$ increases. The direction with the highest growth rate of $PC_T(Q)$ is the line connecting Q_B and b. Search along this line can be implemented in multiple ways. We start at Q_B and move towards b with a step size of $sz = (b - Q_B)/al$, where $al = \lfloor (1/n) \sum_{i=1}^{n} (b_i - Q_{Bi}) \rceil$. We find $P_T(Q)$ at $Q = Q_B + \lfloor jsz \rceil$ for $j = 1, 2, \ldots, al$ as long as the search is giving higher $P_T(Q)$ value. Sometimes, due to odd location of Q_B, we may not see any improvement in the direction of b. In such cases, it is worth

[5] By Property 3, $D = \bigcup_{i=1}^{n} \{\underline{x}^{(i)}, \overline{x}^{(i)}\}$. The Euclidean distances between i) $\underline{x}^{(i)}$ and $\overline{x}^{(i)}$ is $PC_T(Q)\{(m_i + c_i)^{-1} + s_i^{-1}\}$ for $i = 1, 2, \ldots, n$, ii) $\underline{x}^{(i)}$ and $\underline{x}^{(j)}$ is $PC_T(Q)\{(m_i + c_i)^{-2} + (m_j + c_j)^{-2}\}^{1/2}$ for $i, j = 1, 2, \ldots, n$, $i \neq j$, iii) $\overline{x}^{(i)}$ and $\overline{x}^{(j)}$ is $PC_T(Q)\{s_i^{-2} + s_j^{-2}\}^{1/2}$ for $i, j = 1, 2, \ldots, n$, $i \neq j$, and iv) $\underline{x}^{(i)}$ and $\overline{x}^{(j)}$ is $PC_T(Q)\{(m_i + c_i)^{-2} + s_j^{-2}\}^{1/2}$ for $i, j = 1, 2, \ldots, n$, $i \neq j$. In every case, the distance is directly proportional to $PC_T(Q)$.

searching along the opposite direction (i.e., towards a). The search towards a, if required, is implemented in the same manner as the search towards b.

Based on the above description, Procedure 3 is proposed for the implementation of DSH.

Procedure 3 Diagonal search heuristic

Input: n, T, parameters $\langle a_i, b_i, m_i, c_i, s_i \text{ for } i = 1, 2, \ldots, n\rangle$, and $p()$ $\langle n$-dimensional pmf array$\rangle$
Output: Q_{dsh} $\langle$DSH solution$\rangle$

1. $Q_M \leftarrow Q$ such that $p(Q) = \max\{p(x) : x \in \Omega\}$ $\langle$by array search$\rangle$
2. $Q_C \leftarrow (\lfloor\{(m_1 + c_1)a_1 + s_1 b_1\}/(m_1 + c_1 + s_1)\rceil, \ldots, \lfloor\{(m_n + c_n)a_n + s_n b_n\}/(m_n + c_n + s_n)\rceil)$
3. $p_M \leftarrow P_T(Q_M)$, $p_C \leftarrow P_T(Q_C)$ $\langle$using SPC Algorithm$\rangle$
4. **if** $p_M + p_C = 0$ **then**
5. $\quad Q_B \leftarrow (\lfloor(Q_{M1} + Q_{C1})/2\rceil, \ldots, \lfloor(Q_{Mn} + Q_{Cn})/2\rceil)$
6. **else**
7. $\quad Q_B \leftarrow (\lfloor(p_M Q_{M1} + p_C Q_{C1})/(p_M + p_C)\rceil, \ldots, \lfloor(p_M Q_{Mn} + p_C Q_{Cn})/(p_M + p_C)\rceil)$
8. **end if**
9. $Q_{dsh} \leftarrow Q_B$, $bm \leftarrow P_T(Q_B)$ $\langle$using SPC Algorithm$\rangle$
10. $al \leftarrow \lfloor(1/n)\sum_{i=1}^{n}(b_i - Q_{Bi})\rceil$, $sz \leftarrow ((b_1 - Q_{B1})/al, \ldots, (b_n - Q_{Bn})/al)$
11. $gn \leftarrow 0$ $\langle$gain indicator$\rangle$, $j \leftarrow 0$ $\langle$jumps taken$\rangle$
12. **while** $gn >= 0$ **and** $j < al$ **do**
13. $\quad Q \leftarrow Q_B + (\lfloor(j+1)sz_1\rceil, \ldots, \lfloor(j+1)sz_n\rceil)$
14. $\quad chk \leftarrow P_T(Q)$ $\langle$using SPC Algorithm$\rangle$, $gn \leftarrow chk - bm$
15. $\quad$ **if** $gn >= 0$ **then**
16. $\quad\quad Q_{dsh} \leftarrow Q$, $bm \leftarrow chk$, $j \leftarrow j + 1$
17. $\quad$ **end if**
18. **end while**
19. **if** $j = 0$ **then**
20. $\quad$ Search towards a $\langle$like the search towards $b\rangle$
21. **end if**
22. **return** Q_{dsh}

$\langle$Note: For independent demand, $p()$ is replaced by $p_i()$ for $i = 1, 2, \ldots, n$ (1-dimensional pmf arrays). The first line of the algorithm, where we find Q_M, is different. For $i = 1, 2, \ldots, n$, we set $Q_{Mi} = Q_i$ such that $p_i(Q_i) = \max\{p_i(x_i) : x_i \in \Omega_i\}$ by linear search.$\rangle$

In the beginning of Procedure 3, the mode is identified. For the general case, time requirement for this process is $O(\prod_{i=1}^{n}(b_i - a_i + 1))$, while it is $O(\sum_{i=1}^{n}(b_i - a_i))$ for the independent demand case. If the mode is known, which is the case when demand is modelled using standard distributions, we can skip this step.

DSH saves time by keeping the search space 1-dimensional. $P_T(Q)$ is calculated $3 + N_{dsh}$ times, where N_{dsh} denotes the number of search space points (first three points are Q_M, Q_C, and Q_B). In the worst case, DSH searches along the diagonal connecting a and b (for this, the name is diagonal search heuristic); thus, $N_{dsh} < \lfloor(1/n)\sum_{i=1}^{n}(b_i - a_i)\rceil$. Clearly, N_{dsh} is not large and does not increase with the number of products.

5 Heuristic performances

We test accuracy of heuristics by comparing the heuristic solutions with the optimum solution (OS) for different problem classes. Both independent demand (ID) and dependent demand[6] (DD) problems are solved. A problem class is defined by the number of products (nP, $n = 2, 3, \dots$) and demand type (ID and DD).

Khanra (2014) solved 100 2P-ID, 50 3P-ID, and 50 2P-DD problems optimally. Solutions of these problems can be used for heuristic testing. We do not test heuristics with 2P-ID problems as these problems can be solved optimally in quick time; the average computation time was less than 30 seconds. The average computation time for 2P-DD problems was not large (about 80 minutes), but some problems took fairly large amount of time. Heuristics are tested using 15 most time-taking 2P-DD problems. Average computation time for 3P-ID problems was about 9 hours. All 3P-ID problems are used for testing heuristics.

Khanra (2014) did not solve nP-ID, $n \geq 4$ and nP-DD, $n \geq 3$ problems optimally for their large time requirements. To enable heuristic testing in these problem classes, we scale-down demand limits of the test problems (i.e., $a'_i = a_i/k$ and $b'_i = b_i/k$ for $i = 1, 2, \dots, n$ are the new demand limits, where $k > 1$ is the scale factor). See Appendix D for the details. This reduces cardinality of the demand space, which decreases the time requirement for optimization. Test problem generation scheme of Khanra (2014) is retained. It generates test problems with varied profit target, cost parameters, demand ranges, and demand distributions.

Test problems in Khanra (2014) have three levels for the demand range ($b_i - a_i$); these are low, medium, and high. High and medium demand ranges are five and two times the low demand range. Demand scale-down does not change these ratios. Since optimization in the DD problems is more time consuming than their ID counterparts (as $P_T(Q)$ computation algorithm in Khanra, 2014, is slower for DD), to save time, (in addition to demand scale-down,) we restrict demand ranges of the nP-DD, $n \geq 3$ test problems to low and medium only.

In the scaled-down version of a test problem, the demand space is reduced, but other details of the test problem (i.e., profit target level, cost parameters, ratios of demand ranges, and demand distributions) remain unchanged. We do not see any reason to suspect that quality of the heuristic solutions is influenced by the demand scale-down in a systematic manner. Thus, we can expect the average accuracy of heuristics to remain unaffected by the scaled-down. Then we can extrapolate the accuracy test results to the full-scale problems.

In addition to the scaled-down problems, we solve the full-scale test problems using heuristics to understand their time requirements. We can estimate computation time for optimization in the full-scale problems by multiplying the computation time for optimization in the scaled-down problem with $(|\Omega|/|\Omega'|) \times (t/t')$, where Ω' is demand space of the scaled-down problem and t, t' are the average $P_T(Q)$ computation times for $Q \in \Omega$, and $Q \in \Omega'$. However, t is unknown, but

[6]Here, dependent demand does not mean endogenous demand. It refers to correlated product demands. The opposite is independent demand (i.e., uncorrelated product demands).

$t/t' > 1$ (as $P_T(Q)$ computation time in Khanra, 2014, increases with the demand ranges). We take $t/t' = 1$ and get an under-estimate.

We implemented the DRH and DSH procedures in *GNU Octave 3.6.4 (GCC 4.6.2)*. Test problems were solved in *Intel Core i5 (3.30 GHz) processors*. Probability mass function was not provided in the input; it was calculated as part of the procedures. This implementation scheme is similar to that of Khanra (2014).

Numerical results

Table 1 exhibits the accuracy of heuristics for various problem classes (scaled-down test problems are indicated by * mark). It also provides the average computation time (ACT) for optimization in the scaled-down problems. Large ACT values justify our decision to scale-down the demand limits. Heuristic accuracy is measure by deviation from the optimum, $\delta = P_T(Q^*) - P_T(Q_h)$, where Q^* is the OS and Q_h is the heuristic solution (note that $\delta \geq 0$); smaller δ is better. In total, 225 test problems were solved and DRH found the OS in 220 instances. Deviation from the OS was very small when DRH failed to find it. Accuracy of DSH is expected to be lower than that of DRH. The average deviation for DSH was found to be less than 0.05 in every problem class (note that $0 < P_T(Q) < 1$). The maximum deviation was quite high, particularly for the ID problems. However, deviation was high only for a few problems (it can be seen from the large difference between δ_{max} and δ_{avg}).

Table 1: Accuracy of heuristics

		OS	DRH			DSH	
Problem	Num	ACT	Optimal?	δ_{max}	δ_{min}	δ_{max}	δ_{avg}
3P-ID	50	32076.4	48	7.3×10^{-6}	0.00004	0.116	0.0249
4P-ID*	50	98299.4	50	0	0.00001	0.138	0.0265
5P-ID*	25	657124.2	25	0	0.00016	0.188	0.0375
6P-ID*	25	1133284.2	25	0	0.00000	0.174	0.0495
2P-DD	15	13525.8	14	7.9×10^{-9}	0.00000	0.030	0.0074
3P-DD*	20	354553.2	19	2.7×10^{-10}	0.00000	0.090	0.0168
4P-DD*	20	1413522.3	19	4.7×10^{-6}	0.00000	0.092	0.0211
5P-DD*	10	1823066.5	10	0	0.00008	0.091	0.0254
6P-DD*	10	1097913.2	10	0	0.00000	0.081	0.0157

Table 2 shows the computation time for heuristics. The average projected computation time for optimization in the full-scale problem is also shown. It is evident that DRH solves 3P-ID problems quickly; no problem took more than 4 seconds. 2P-DD and 4P-ID problems do not take much time either; the average computation times were about 10 and 13 minutes respectively (note that only time-taking 2P-DD problems were considered). 3P-DD problems take fairly large amount of time; the average was about 5 hours (note that nP-DD, $n \geq 3$ test problems did not have products with high demand range). DRH takes large amount of time for solving 5P-ID

problems; the average computation time was about 92 hours.

Computation time for DSH is expected to be lower than that of DRH. DSH solves 3P-ID and 2P-DD problems quickly; no 3P-ID and 2P-DD problems took more than 1 and 14 seconds respectively. 4P-ID and 3P-DD problems are solved quickly too; the maximum computation time in these problem classes were about 6 and 4 minutes respectively. 5P-ID and 4P-DD problems do not take large amount of time; the averages were about 32 and 36 minutes respectively. DSH takes large amount of time for solving 6P-ID and 5P-DD problems; the average computation times were about 90 and 39 hours respectively.

Table 2: Computation time for heuristics

		OS	DRH			DSH		
Prob	Num	Avg (proj)	Min	Max	Avg	Min	Max	Avg
3P-ID	50	3.2×10^4	0.6	4.0	1.6	0.0	0.3	0.1
4P-ID	50	5.8×10^7	48.9	6075.2	792.4	1.0	375.2	17.7
5P-ID	25	5.4×10^{10}	7914.4	1227065.8	330147.9	42.8	8367.3	1909.4
6P-ID	25	4.4×10^{13}	–	–	–	2993.1	1843202.3	322158.4
2P-DD	15	1.4×10^4	169.9	1301.6	590.1	0.5	13.5	3.3
3P-DD	20	2.8×10^6	5973.5	30078.3	18378.6	1.5	234.3	50.0
4P-DD	20	7.8×10^8	–	–	–	0.0	7846.3	2174.7
5P-DD	9	6.9×10^{11}	–	–	–	64.4	373412.8	139917.4

From the performance test, it is clear that DRH can be used for 3P-ID, 4P-ID, and 2P-DD problems. It is not suitable for nP-ID, $n \geq 5$ and nP-DD, $n \geq 3$ problems due to its time requirement. Accuracy is never a concern for DRH. On the other hand, DSH can be used for solving 3P-ID, 4P-ID, 5P-ID, 2P-DD, 3P-DD, and 4P-DD problems. However, if computation time of DRH is reasonable, it is a better choice than DSH for its accuracy. DSH is not suitable for nP-ID, $n \geq 6$ and nP-DD, $n \geq 5$ problems. If high demand range is included in nP-DD, $n \geq 3$ problems, DSH is likely to fail at 4P-DD problems.

6 Conclusion

Literature on the MPSNP is limited. Practically usable solution methods are available only for two-product and some three-product (with smaller demand ranges) problems. We have developed heuristics to solve larger MPSNP in reasonable time. Heuristics were tested using a large pool of versatile test problems.

Two heuristics are developed. The first heuristic, DRH reduces the n-dimensional search space and finds the best solution. It is very accurate; out of 225 test problems, it found the optimum in 220 instances. However, it is time consuming; beyond 4P-ID and 2P-DD problems, it is not suitable. The second heuristic, DSH restricts its search space to 1-dimension. DSH consumes much less time than DRH. Its efficiency comes at the cost of accuracy. It never found

the optimum; however, its deviation from the optimum was found to be reasonably small. The average deviation was less than 0.05. Even though it is less time consuming than DRH, beyond 5P-ID and 4P-DD problems, DSH is not suitable.

With our heuristics, the MPSNP can be solved upto five-product cases (four-product if the product demands are not independent). Clearly, there is a need for faster heuristics for solving the larger problems. One possibility is to utilize the approximation of Özler et al. (2009). They approximated distribution function of the total profit, $G_Q()$ (it depends on Q) by normal distribution for the independent demand case. Then Q with the lowest $G_Q(T)$ is the optimum solution as $P_T(Q) = 1 - G_Q(T)$. The approximation error gets smaller as the number of products increases. Note that exact evaluation of $G_Q()$ is difficult.

Acknowledgements

The authors are grateful to the Computer Centre of IIM Ahmedabad for its assistance in the computation.

Appendix A

We need to find $z \in \mathbb{N}$ and $k_i^{(j)} \in (1, \infty)$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z$ such that the total computation time (sum of computation times for solving the rescaled problems and the final search) is minimized. A good indicator for the total computation time is the number of times $P_T(Q)$ is calculated. Let this number be N. Let $N^{(j)}$ for $j = 1, 2, \ldots, z$ be the cardinality of the search space of the j^{th} rescaled problem and $N^{(f)}$ be the cardinality of the final search space. Then $N = \sum_{j=1}^{z} N^{(j)} + N^{(f)}$. We want to minimize N.

After each rescaled problem is solved, we get a trust region (denoted by $\Omega \cap R(1)$ in the description of DRH); we believe that the optimum is in the trust region. Search space for the next rescaled problem is a scaled-down version of this trust region. In the very beginning, the trust region is Ω. Thus, the search space of the first rescaled problem is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor a_i/k_i^{(1)} \rceil \leq Q_i \leq \lfloor b_i/k_i^{(1)} \rceil \text{ for } i = 1, 2, \ldots, n\}$. We approximate $N^{(1)}$ as

$$N^{(1)} = \prod_{i=1}^{n} \left(\lfloor b_i/k_i^{(1)} \rceil - \lfloor a_i/k_i^{(1)} \rceil + 1 \right) \approx \prod_{i=1}^{n} \left\{ (b_i - a_i)/k_i^{(1)} \right\}.$$

Let $Q^{(1)^*}$ be the optimum solution of the first rescaled problem. The corresponding trust region is $\{Q : Q \in \mathbb{N}_0^n,\ L_i^{(1)} \leq Q_i \leq R_i^{(1)} \text{ for } i = 1, 2, \ldots, n\}$, where $L_i^{(1)} = \max\{a_i, k_i^{(1)}(Q_i^{(1)^*} - 1)\}$ and $R_i^{(1)} = \min\{b_i, k_i^{(1)}(Q_i^{(1)^*} + 1)\}$ for $i = 1, 2, \ldots, n$. Note that $R_i^{(1)} - L_i^{(1)} \approx 2k_i^{(1)}$ for $i = 1, 2, \ldots, n$. Search space of the second rescaled problem is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor L_i^{(1)}/k_i^{(2)} \rceil \leq Q_i \leq$

$\lfloor R_i^{(1)}/k_i^{(2)} \rceil$ for $i = 1, 2, \ldots, n\}$. We approximate $N^{(2)}$ as

$$N^{(2)} = \prod_{i=1}^{n} \left\{ \lfloor R_i^{(1)}/k_i^{(2)} \rceil - \lfloor L_i^{(1)}/k_i^{(2)} \rceil + 1 \right\} \approx \prod_{i=1}^{n} \left\{ (R_i^{(1)} - L_i^{(1)})/k_i^{(2)} \right\} \approx \prod_{i=1}^{n} \left(2k_i^{(1)}/k_i^{(2)} \right).$$

Structure of the search spaces of the remaining rescaled problems are same as that of the second rescaled problem. Hence,

$$N^{(j)} \approx \prod_{i=1}^{n} \left(2k_i^{(j-1)}/k_i^{(j)} \right) = 2^n \prod_{i=1}^{n} \left(k_i^{(j-1)}/k_i^{(j)} \right) \quad \text{for } j = 1, 2, \ldots, z.$$

Let $Q^{(z)^*}$ be the optimum solution of the last rescaled problem. Since there is no more rescaling, the final search space is $\{Q : Q \in \mathbb{N}_0^n,\ \lfloor L_i^{(z)} \rceil \le Q_i \le \lfloor R_i^{(z)} \rceil \text{ for } i = 1, 2, \ldots, n\}$, where $L_i^{(z)} = \max\{a_i, k_i^{(z)}(Q_i^{(z)^*} - 1)\}$ and $R_i^{(z)} = \min\{b_i, k_i^{(z)}(Q_i^{(z)^*} + 1)\}$ for $i = 1, 2, \ldots, n$. Then

$$N^{(f)} = \prod_{i=1}^{n} \left\{ \lfloor R_i^{(z)} \rceil - \lfloor L_i^{(z)} \rceil + 1 \right\} \approx \prod_{i=1}^{n} \left(R_i^{(z)} - L_i^{(z)} \right) \approx 2^n \prod_{i=1}^{n} k_i^{(z)}.$$

Let us write $K^{(j)} = \prod_{i=1}^{n} k_i^{(j)}$ for $j = 1, 2, \ldots, z$. Note that $K^{(j)} \in (1, \infty)$ for every $j = 1, 2, \ldots, z$. Let us write $K_0 = \prod_{i=1}^{n}(b_i - a_i)/2$, i.e., $\prod_{i=1}^{n}(b_i - a_i) = 2^n K_0$. Then

$$N \approx 2^n \left(\frac{K_0}{K^{(1)}} + \frac{K^{(1)}}{K^{(2)}} + \frac{K^{(2)}}{K^{(3)}} + \cdots + \frac{K^{(z-1)}}{K^{(z)}} + K^{(z)} \right) = 2^n N_A \text{ (say)}.$$

Minimum N_A approximately minimizes N. Let us minimize N_A in $K^{(1)}, K^{(2)}, \ldots, K^{(z)}$ for fixed z. N_A is a smooth function. The first order necessary conditions give

$$\frac{\partial N_A}{\partial K^{(1)}} = -\frac{K_0}{\{K^{(1)}\}^2} + \frac{1}{K^{(2)}} = 0 \ \Rightarrow \{K^{(1)^*}\}^2 = K_0 K^{(2)^*}.$$

$$\frac{\partial N_A}{\partial K^{(j)}} = -\frac{K^{(j-1)}}{\{K^{(j)}\}^2} + \frac{1}{K^{(j+1)}} = 0 \ \Rightarrow \{K^{(j)^*}\}^2 = K^{(j-1)^*} K^{(j+1)^*} \quad \text{for } j = 2, 3, \ldots, z-1.$$

$$\frac{\partial N_A}{\partial K^{(z)}} = -\frac{K^{z-1}}{\{K^{(z)}\}^2} + 1 = 0 \ \Rightarrow \{K^{(z)^*}\}^2 = K^{(z-1)^*}.$$

Putting $K^{(z)^*}$ into the expression of $K^{(z-1)^*}$, then putting $K^{(z-1)^*}$ into the expression of $K^{(z-2)^*}$, ..., and finally putting $K^{(2)^*}$ into the expression of $K^{(1)^*}$, we get

$$K^{(z-j+1)^*} = \left(K^{(z-j)^*} \right)^{j/(j+1)} \quad \text{for } j = 1, 2, \ldots, z-1.$$
$$K^{(1)^*} = K_0^{z/(z+1)}.$$

Note that K_0 is a constant. Now, putting back $K^{(1)^*}$ into $K^{(2)^*}$, then $K^{(2)^*}$ into $K^{(3)^*}$, ...,

and finally $K^{(z-1)^*}$ into $K^{(z)^*}$, we get the only stationary point as

$$K^{(j)^*} = K_0^{1-j/(z+1)} \quad \text{for } j = 1, 2, \ldots, z.$$

Now, we need to test the nature of $K^* = (K^{(1)^*}, K^{(2)^*}, \ldots, K^{(z)^*})$.

$$\begin{aligned}
\frac{\partial^2 N_A(K^*)}{\partial K^{(1)^2}} &= \frac{2K_0}{\{K^{(1)^*}\}^3} = \frac{2}{K_0^2} K_0^{3/(z+1)}. \\
\frac{\partial^2 N_A(K^*)}{\partial K^{(j)^2}} &= \frac{2K^{(j-1)^*}}{\{K^{(j)^*}\}^3} = \frac{2}{K_0^2} K_0^{(2j+1)/(z+1)} \quad \text{for } j = 2, 3, \ldots, z. \\
\frac{\partial^2 N_A(K^*)}{\partial K^{(j)} \partial K^{(j+1)}} &= -\frac{1}{\{K^{(j+1)^*}\}^2} = -\frac{1}{K_0^2} K_0^{(2j+2)/(z+1)} \quad \text{for } j = 1, 2, \ldots, z-1.
\end{aligned}$$

Second order partial derivatives of N_A of other forms (except the symmetric versions of the above) are zero. For simplicity, let us write $1/(z+1) = u$. Then

$$\frac{\partial^2 N_A(K^*)}{\partial K^{(i)} \partial K^{(j)}} = \begin{cases} 2K_0^{u-2} K_0^{u(i+j)} & \text{if } i = j \text{ for } i, j \in \{1, 2, \ldots, z\} \\ -K_0^{u-2} K_0^{u(i+j)} & \text{if } |i-j| = 1 \text{ for } i, j \in \{1, 2, \ldots, z\} \\ 0 & \text{if } |i-j| > 1 \text{ for } i, j \in \{1, 2, \ldots, z\}. \end{cases}$$

Quadratic form of the Hessian matrix can be written as

$$\begin{aligned}
x^T H(K^*) x &= \sum_{i=1}^{z} \sum_{j=1}^{z} \frac{\partial^2 N_A(K^*)}{\partial K^{(i)} \partial K^{(j)}} x_i x_j = \sum_{j=1}^{z} \frac{\partial^2 N_A(K^*)}{\partial \{K^{(j)}\}^2} x_j^2 + 2 \sum_{j=1}^{z-1} \frac{\partial^2 N_A(K^*)}{\partial K^{(j)} \partial K^{(j+1)}} x_j x_{j+1} \\
&= K_0^{u-2} \left[K_0^{2u} x_1^2 + \sum_{j=1}^{z-1} \left\{ K_0^{2ju} x_j^2 - 2K_0^{(2j+1)u} x_j x_{j+1} + K_0^{2(j+1)u} x_{j+1}^2 \right\} + K_0^{2zu} x_z^2 \right] \\
&= K_0^{u-2} \left[K_0^{2u} x_1^2 + \sum_{j=1}^{z-1} \left\{ K_0^{ju} x_j - K_0^{(j+1)u} x_{j+1} \right\}^2 + K_0^{2zu} x_z^2 \right] \geq 0.
\end{aligned}$$

Hence, K^* is a local minima of N_A for fixed z. Actually, it is the only critical point. Thus, K^* is the global minima. Let us write N_A at K^* as a function of z. At K^*, each term in the expression of N_A becomes $K_0^{1/(z+1)}$. Thus,

$$N_A^*(z) = (z+1) K_0^{1/(z+1)}.$$

Note that $z \in \mathbb{N}$. Let us relax this and assume that $z \in [1, \infty)$. Then

$$\frac{\mathrm{d}N_A^*(z)}{\mathrm{d}z} = K_0^{1/(z+1)} - \frac{\ln(K_0)}{z+1} K_0^{1/(z+1)}$$

$$\frac{\mathrm{d}^2 N_A^*(z)}{\mathrm{d}z^2} = -\frac{\ln(K_0)}{(z+1)^2} K_0^{1/(z+1)} - \left[-\frac{\ln(K_0)}{(z+1)^2} K_0^{1/(z+1)} - \frac{\{\ln(K_0)\}^2}{(z+1)^3} K_0^{1/(z+1)} \right]$$
$$= \frac{\{\ln(K_0)\}^2}{(z+1)^3} K_0^{1/(z+1)} > 0 \ \forall z \in [1, \infty) \text{ and any } K_0.$$

Hence, $N_A^*(z)$ is strictly convex in $z \in [1, \infty)$. The first order necessary condition gives

$$K_0^{1/(z^*+1)} \left\{ 1 - \frac{\ln(K_0)}{z^*+1} \right\} = 0 \ \Rightarrow z^* = \ln(K_0) - 1 \quad \text{as } K_0^{1/(z^*+1)} \neq 0.$$

If $z^* \in [1, \infty)$, i.e., $K_0 \geq e^2$, $N_A^*(z^*)$ is the minimum value of N_A. Typically, $K_0 = \prod_{i=1}^n (b_i - a_i)/2$ is a large quantity; hence, $K_0 \geq e^2$ normally holds. If $K_0 < e^2$ (theoretical possibility), the demand ranges are extremely small; then the problem can be easily solved optimally by complete enumeration. Since z is integer-valued, either $\lfloor z^* \rfloor$ or $\lceil z^* \rceil$ is the actual minima. We take $\lfloor z^* \rceil$ as the number of rescaled problems to be solved in DRH.

$$z = \left\lfloor \ln \left(\prod_{i=1}^n \left(\frac{b_i - a_i}{2} \right) \right) - 1 \right\rfloor = \left\lfloor \sum_{i=1}^n \ln \left(\frac{b_i - a_i}{2} \right) \right\rfloor - 1.$$

Once z is selected, optimum K^* (for given z) is

$$K^{(j)^*} = \prod_{i=1}^n k_i^{(j)^*} = K_0^{1-j/(z+1)} \quad \text{for } j = 1, 2, \ldots, z.$$

For each $j = 1, 2, \ldots, z$, there can be multiple choices for $k_i^{(j)^*}$ for $i = 1, 2, \ldots, n$. Since $K_0 = \prod_{i=1}^n (b_i - a_i)/2$, $k_i^{(j)^*} = \{(b_i - a_i)/2\}^{1-j/(z+1)}$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, z$ is one such choice. We take them as the scale factors in DRH.

$$k_i^{(j)} = \left(\frac{b_i - a_i}{2} \right)^{1-j/(z+1)} \quad \text{for } i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, z.$$

If we take $z^* = \ln(K_0) - 1$, $N_A^*(z^*) = \ln(K_0) K_0^{1/\ln(K_0)} = e \ln(K_0)$. Then

$$N^* \approx 2^n N_A^*(z^*) = 2^n e \ln \left(\prod_{i=1}^n \frac{b_i - a_i}{2} \right) = 2^n e \sum_{i=1}^n \ln \left(\frac{b_i - a_i}{2} \right).$$

N^* is an approximation for the number of times $P_T(Q)$ is calculated in DRH. Of course, this is an under-estimation as N_A is an under-estimate of N.

Appendix B

In (7), $p^{(j)}(x^{(j)})$, $x^{(j)} \in \Omega^{(j)}$ are calculated by selecting subsets of $\{p(x) : x \in \Omega\}$ and adding their elements. Since $p : \Omega \to [0, 1]$ is a valid probability mass function, it is sufficient to show that each $x \in \Omega$ belongs to exactly one of these subsets.

Let $x \in \Omega$ is in the subsets corresponding to $y^{(j)}, z^{(j)} \in \Omega^{(j)}$ and $y^{(j)} \neq z^{(j)}$. Then $y_i^{(j)} \neq z_i^{(j)}$ for one or more $i = 1, 2, \ldots, n$. Without loss of generality, let us assume that $y_1^{(j)} < z_1^{(j)}$. Since these are integers, $y_1^{(j)} + 1/2 \leq z_1^{(j)} - 1/2$. Since x is in both subsets, by the definition, $k_1^{(j)}(y_1^{(j)} - 1/2) \leq x_1 < k_1^{(j)}(y_1^{(j)} + 1/2)$ and $k_1^{(j)}(z_1^{(j)} - 1/2) \leq x_1 < k_1^{(j)}(z_1^{(j)} + 1/2)$. These conditions give $x_1 < x_1$, an impossibility. Hence, no $x \in \Omega$ belongs to two or more subsets.

Now, let us assume that $x \in \Omega$ does not belong to any subset. Then $x_i \notin [k_i^{(j)}(y_i^{(j)} - 1/2), k_i^{(j)}(y_i^{(j)} + 1/2)) \; \forall y^{(j)} \in \Omega^{(j)}$ for at least one $i = 1, 2, \ldots, n$. Without loss of generality, let this $i = 1$. Then $x_1 \notin [k_1^{(j)}(y_1^{(j)} - 1/2), k_1^{(j)}(y_1^{(j)} + 1/2)) \; \forall y^{(j)} \in \Omega^{(j)} \Leftrightarrow x_1 \notin [k_1^{(j)}(\lfloor a_1/k_1^{(j)} \rceil - 1/2), k_1^{(j)}(\lfloor b_1/k_1^{(j)} \rceil + 1/2))$ as $\{y_1^{(j)} : y^{(j)} \in \Omega\} = \{\lfloor a_1/k_1^{(j)} \rceil, \lfloor a_1/k_1^{(j)} \rceil + 1, \ldots, \lfloor b_1/k_1^{(j)} \rceil\}$. There are two possibilities: i) $x_1 < k_1^{(j)}(\lfloor a_1/k_1^{(j)} \rceil - 1/2)$ and ii) $x_1 \geq k_1^{(j)}(\lfloor b_1/k_1^{(j)} \rceil + 1/2)$. Since $z - 1/2 < \lfloor z \rceil \leq z + 1/2$ for any $z \in \mathbb{R}$, the first possibility leads to $x_1 < a_1$ and the second possibility leads to $x_1 > b_1$. In both cases, $x_1 \notin [a_1, b_1] \Rightarrow x \notin \Omega$, a contradiction. Thus, each $x \in \Omega$ belongs to exactly one of the subsets.

Appendix C

Procedure 4 Procedure 2 for independent demand

Input: n, T, parameters, $p_i()$ for $i = 1, 2, \ldots, n$, trust region $\langle L_i, R_i \text{ for } i = 1, 2, \ldots, n \rangle$, and scale factors $\langle k_i \text{ for } i = 1, 2, \ldots, n \rangle$

Output: Q'^* ⟨rescaled problem solution⟩

1. **for** $i = 1$ **to** n **do**
2. $\quad a'_i \leftarrow \lfloor a_i/k_i \rceil$, $b'_i \leftarrow \lfloor b_i/k_i \rceil$ ⟨rescaled demand space⟩
3. $\quad m'_i \leftarrow k_i m_i$, $c'_i \leftarrow k_i c_i$, $s'_i \leftarrow k_i s_i$ ⟨rescaled cost parameters⟩
4. $\quad L'_i \leftarrow \lfloor L_i/k_i \rceil$, $R'_i \leftarrow \lfloor R_i/k_i \rceil$ ⟨refined search space⟩
5. **end for**
6. **for** $i = 1$ **to** n **do**
7. $\quad p'_i() \leftarrow 0$ ⟨initializing rescaled pmf array of size $(b'_i - a'_i + 1)$⟩
8. $\quad$ **for** $x = a'_i$ **to** b'_i **do**
9. $\quad\quad l \leftarrow \min\{y : y \in \{a_i, a_i + 1, \ldots, b_i\},\ y \geq k_i(x - 1/2)\}$ ⟨by binary search⟩
10. $\quad\quad r \leftarrow \max\{y : y \in \{a_i, a_i + 1, \ldots, b_i\},\ y < k_i(x + 1/2)\}$ ⟨by binary search⟩
11. $\quad\quad p'_i(x) \leftarrow \sum_{y=l}^{r} p_i(y)$
12. $\quad$ **end for**
13. **end for**
14. $Q'^* \leftarrow Q^*$ such that $P_T(Q^*) = \max\{P_T(Q) : Q \in \mathbb{N}_0^n,\ L'_i \leq Q_i \leq R'_i \text{ for } i = 1, 2, \ldots, n\}$ ⟨Optimum solution in the refined search space, obtained by complete enumeration. $P_T(Q)$ is computed using SPC Algorithm with rescaled parameters and rescaled pmf arrays.⟩
15. **return** Q'^*

Appendix D

The readers are requested to go through the test problem generation scheme of Khanra (2014) for understanding this appendix properly.

We scale-down demand limits of nP-ID, $n \geq 4$ and nP-DD, $n \geq 3$ test problems. Factors that depend on the demand limits, i.e., the mode and the standard deviation, whenever applicable, are scaled-down by the same factor. However, location of the demand mode (w.r.t. the demand limits) and ratios of different demand ranges do not change. Cost parameters, demand distribution, and correlation matrix for the DD problems remain unchanged. Since demand limits change, profit target changes, but the profit target level (low, medium, or high) remains unchanged. Table 3 shows the scale for different problem classes.

Table 3: Scale for demand scale-down

	Number of products				
Demand type	2P	3P	4P	5P	6P
Independent	–	1 : 1	1 : 5	1 : 10	1 : 20
Dependent	1 : 1	1 : 2	1 : 5	1 : 10	1 : 20

References

Abbas, A. E., Matheson, J. E., & Bordley, R. F. (2009). Effective utility functions induced by organizational target-based incentives. *Managerial and Decision Economics*, *30*(4), 235–251.

Anvari, M. (1987). Optimality criteria and risk in inventory models: The case of the newsboy problem. *Journal of the Operational Research Society*, *38*(7), 625–632.

Anvari, M., & Kusy, M. (1990). Risk in inventory models: Review and implementation. *Engineering Costs and Production Economics*, *19*, 267–272.

Chen, M., & Chuang, C. (2000). An extended newsboy problem with shortage-level constraints. *International Journal of Production Economics*, *67*(3), 269–277.

Gotoh, J.-Y., & Takano, Y. (2007). Newsvendor solutions via conditional value-at-risk minimization. *European Journal of Operational Research*, *179*(1), 80–96.

Irwin, W. K., & Allen, I. S. (1978). Inventory models and management objectives. *Sloan Management Review*, *19*(2), 53–59.

Ismail, B. E., & Louderback, J. G. (1979). Optimizing and satisficing in stochastic cost-volume-profit analysis. *Decision Sciences*, *10*(2), 205–217.

Khanra, A. (2014). *Multi-product newsboy problem with satiation objective* (No. 2014-01-01). Indian Institute of Management Ahmedabad, India.

Khouja, M. (1995). The newsboy problem under progressive multiple discounts. *European Journal of Operational Research*, *84*(2), 458–466.

Khouja, M. (1999). The single-period (news-vendor) problem: Literature review and suggestions for future research. *Omega*, *27*(5), 537–553.

Khouja, M., & Robbins, S. S. (2003). Linking advertising and quantity decisions in the single-period inventory model. *International Journal of Production Economics*, *86*(2), 93–105.

Lau, A. H.-L., & Lau, H.-S. (1988a). Maximizing the probability of achieving a target profit in a two-product newsboy problem. *Decision Sciences*, *19*(2), 392–408.

Lau, A. H.-L., & Lau, H.-S. (1988b). The newsboy problem with price-dependent demand distribution. *IIE Transactions*, *20*(2), 168–175.

Lau, H.-S. (1980). The newsboy problem under alternative optimization objectives. *Journal of the Operational Research Society*, *31*(6), 525–535.

Li, J., Lau, H.-S., & Lau, A. H.-L. (1990). Some analytical results for a two-product newsboy problem. *Decision Sciences*, *21*(4), 710–726.

Li, J., Lau, H.-S., & Lau, A. H.-L. (1991). A two-product newsboy problem with satisficing objective and independent exponential demands. *IIE Transactions*, *23*(1), 29–39.

Lin, C.-S., & Kroll, D. E. (1997). The single-item newsboy problem with dual performance measures and quantity discounts. *European Journal of Operational Research*, *100*(3), 562–565.

Norland, R. E. (1980). Refinements in the Ismail-Louderback's stochastic CVP model. *Decision Sciences*, *11*(3), 562–572.

Oberlaender, M. (2011). Dual sourcing of a newsvendor with exponential utility of profit. *International Journal of Production Economics*, *133*(1), 370–376.

Özler, A., Tan, B., & Karaesmen, F. (2009). Multi-product newsvendor problem with value-at-risk considerations. *International Journal of Production Economics*, *117*(2), 244–255.

Qin, Y., Wang, R., Vakharia, A. J., Chen, Y., & Seref, M. M. H. (2011). The newsvendor problem: Review and directions for future research. *European Journal of Operational Research*, *213*(2), 361–374.

Sankarasubramanian, E., & Kumaraswamy, S. (1983). Note on "Optimal ordering quantity to realize a pre-determined level of profit". *Management Science*, *29*(4), 512–514.

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory Management and Production Planning and Scheduling* (3rd ed.). New York: John Wiley & Sons.

Victor, C., Yang, S., Xia, Y., & Zhao, X. (2011). Inventory competition for newsvendors under the objective of profit satisficing. *European Journal of Operational Research*, *215*(2), 367–373.

Wu, J., Li, J., Wang, S., & Cheng, T. (2009). Mean-variance analysis of the newsvendor model with stockout cost. *Omega*, *37*(3), 724–730.