INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD • INDIA

# An efficient heuristic for the multi-product satiating newsboy problem

Avijit Khanra

W.P. No. 2014-06-03
June 2014



**INDIAN INSTITUTE OF MANAGEMENT**
**AHMEDABAD-380 015**
**INDIA**

# An efficient heuristic for the multi-product satiating newsboy problem

Avijit Khanra*

*Indian Institute of Management, Ahmedabad-380015, India*

**Abstract**

Preference of satiation of a target performance over maximization of expected performance in uncertain situations is well-documented in the economics literature. However, the newsboy problem with satiation (of a profit target) objective has not received its due attention. In the multi-product setting, solution methods available in the literature are inefficient. We developed an efficient heuristic to solve the problem. The heuristic decomposes the multi-product problem into easily solvable single-product problems. We tested the heuristic with a large number of test instances. The heuristic can be adopted to solve the "target assignment problem". We demonstrated it with some numerical examples.

## 1 Introduction

The newsboy problem with satiation objective (i.e., maximization of the probability of achieving a given profit target) has quite a long history. Research into this important practical problem started with the work of Irwin & Allen (1978). They studied the single-product problem. The problem is non-convex unlike the classical newsboy problem. They identified necessary condition for optimality, which links the optimal decision with the demand distribution. The condition does not lead to a closed-form solution for the general case. Subsequent research on the single-product satiating newsboy problem (Ismail & Louderback, 1979; H.-S. Lau, 1980; Norland, 1980; Sankarasubramanian & Kumaraswamy, 1983) considered specific demand distributions and derived closed-form expressions for the optimal stocking quantity.

Research into the multi-product satiating newsboy problem (MPSNP) started with the work of A. H.-L. Lau & Lau (1988). They studied the two-product problem with zero stock-out costs and uniform demand. They derived closed-form solution for the case of identical products. Li et al. (1990) considered non-identical products in the same setting. They developed an efficient algorithm to compute the optimal stocking quantities. Li et al. (1991) did the same for the case of independent exponential product demands.

After Li et al. (1991), no paper has been published on the MPSNP. Recently, Khanra (2014), in his unpublished work, attempted the general MPSNP (i.e., no restriction on the number of

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

products, stock-out costs, and demand distribution) using a discrete formulation. Note that all previous papers on the satiating newsboy problem adapted continuous formulation. He identified key properties of the problem and showed that the continuous formulation of the problem is difficult to solve. His optimization algorithm, though solves the general MPSNP, is inefficient. In fact, beyond two-product instances, the method is not useful.

Khanra & Soman (2014) eased the issue of large computation time of the algorithm of Khanra (2014) to some extent. They developed search-based heuristics. In most cases, their heuristics identified the optimum. However, those heuristics are not efficient either. Beyond five-product instances, they are not useful. In this paper, we develop an efficient heuristic, which solves the MPSNP with any 'realistic number of products' in 'reasonable time'.

Remainder of this paper is organized as follows. In Section 2, we discuss the heuristic named as target splitting heuristic (TSH). There are two variants of TSH. We describe them along with their performances. TSH has an interesting implication for the "target assignment problem", which we discuss in Section 3. Finally, we conclude in Section 4.

## 2 Target splitting heuristic

We follow the discrete formulation of Khanra (2014). To avoid repetition, we do not discuss the details here. Readers are requested to go through it or its summary presented in Khanra & Soman (2014). For the ease of readers, the symbol list is provided in Appendix A.

Let us consider the single-product problem. Let $m, c, s$ denote unit profit, unit purchase cost less salvage value of an unsold item, and goodwill loss due to unit unmet demand. Let $a, b$ denote the demand limits. $\Omega = \{a, a+1, \ldots, b\}$ denotes the demand space. Let $X$ denote the stochastic demand. Let $p()$ and $P()$ denote probability mass function (pmf) and cumulative distribution function (cdf) of $X$. If the target $T$ is achievable by order quantity $Q$, i.e., $mQ \geq T$, the set of terminal demand point is given by: $D = \{\underline{x}, \overline{x}\}$ where $\underline{x} = Q - (mQ - T)/(m + c)$ and $\overline{x} = Q + (mQ - T)/s$. The convex hull of $D$, $\text{co}D = [\underline{x}, \overline{x}]$. $\text{co}D \cap \Omega = \{\max\{a, \lceil \underline{x} \rceil\}, \max\{a, \lceil \underline{x} \rceil\} + 1, \ldots, \min\{b, \lfloor \overline{x} \rfloor\}\}$. Then the satiation probability (i.e., probability of achieving the given profit target), $P_T(Q) = Pr\{X \in \text{co}D \cap \Omega\} = P(\min\{b, \lfloor \overline{x} \rfloor\}) - P(\max\{a, \lceil \underline{x} \rceil\} - 1)$. If the target is unachievable by $Q$, $P_T(Q) = 0$. Clearly, $P_T(Q)$ can be calculated in $O(1)$ time. Then $P_T(Q)$ can be maximized by complete enumeration of $\Omega$ in $O(b - a)$ time. Algorithm 3 in Appendix B solves the single-product satiating newsboy problem optimally.

Let $n$ denote the number of products. If we split profit target of the MPSNP into $n$ product-specific targets, $T_1, T_2, \ldots, T_n$ such that $T = \sum_{i=1}^{n} T_i$ and $T_i \leq \overline{T}_i$ for every $i = 1, 2, \ldots, n$, where $\overline{T}_i = m_i b_i$ is the maximum achievable target for product-$i$, we can get a solution for the MPSNP by solving $n$ single-product problems optimally. This process would take $O(\sum_{i=1}^{n}(b_i - a_i)) = O(nr_{avg})$ time where $r_{avg}$ denotes the average demand range. If the target splitting procedure is efficient, this decomposition-based heuristic would take polynomial time (in $n$) unlike the search-based heuristics of Khanra & Soman (2014). Quality of the solution would depend on the target splitting rule. We have developed two rules.

## 2.1 Rule-I

Intuitively, targets should be assigned to products based on their 'profit making capabilities', which we measure by the maximum expected profits. In the single-product setting, expected profit maximizing order quantity $Q'$ and the maximum expected profit $\Pi^*$ are given by

$$Q' = \min\{Q \in \{a, a+1, \ldots, b\} : P(Q) \geq \xi\} \tag{1}$$

$$\Pi^* \approx (m+c)\sum_{x=a}^{Q'} xp(x) - s \sum_{x=Q'+1}^{b} xp(x) \tag{2}$$

where $\xi = (m+s)/(m+c+s)$ denotes the critical fractile. See Appendix C for the details. We split the profit target as

$$T_i = \min\left\{\overline{T}_i,\ \frac{\Pi_i^*}{\sum_{j=1}^{n}\Pi_j^*} \times T\right\} \quad \text{for } i = 1, 2, \ldots, n. \tag{3}$$

The above assignment ensures that individual profit targets are achievable. For sufficiently large $T$, one or more products get $\overline{T}_i$ as their individual targets. Such cases result in under-allocation, i.e., $\sum_{i=1}^{n} T_i < T$. Then the residual amount is distributed among the products with $T_i < \overline{T}_i$. The secondary allocation is executed like the primary allocation, but with revised proportions (see Procedure 1). This process is continued till the entire target is allocated. It terminates in finite time if the target is achievable, i.e., $T \leq \overline{T} = \sum_{i=1}^{n} \overline{T}_i$.

For sufficiently small $T$, a different situation arises. Then one or more products may get $T_i < \underline{T}_i$ as their individual targets where $\underline{T}_i$ denotes the maximum assured target for product-$i$. $\underline{T}_i = \max\{\Pi_i(\lfloor Q_{0i}\rfloor, b_i), \Pi_i(\lceil Q_{0i}\rceil, a_i)\}$ where $Q_{0i} = \{(m_i + c_i)a_i + s_i b_i\}/(m_i + c_i + s_i)$ and $\Pi_i(Q_i, x_i) = m_i \min\{Q_i, x_i\} - c_i \max\{0, Q_i - x_i\} - s_i \max\{0, x_i - Q_i\}$. For each $i$, $P_{T_i}(Q_i^*) = 1$ for every $T_i \leq \underline{T}_i$ where $Q_i^*$ denotes the optimal solution of the $i^{th}$ single-product problem. Thus, assigning a target lesser than the maximum assured target is never beneficial. Hence, we modify the target splitting rule from (3) to (4).

$$T_i = \max\left\{\underline{T}_i,\ \min\left\{\overline{T}_i,\ \frac{\Pi_i^*}{\sum_{j=1}^{n}\Pi_j^*} \times T\right\}\right\} \quad \text{for } i = 1, 2, \ldots, n. \tag{4}$$

For small $T$, the above rule may lead to over-allocation, i.e., $\sum_{i=1}^{n} T_i > T$. Then the excess amount is removed from the products with $T_i > \underline{T}_i$ just like the residue allocation scheme for large $T$. It terminates in finite time if the target is non-trivial, i.e., $T \geq \underline{T} = \sum_{i=1}^{n} \underline{T}_i$.

Based on the above discussions, Procedure 1 is implemented to performs the target splitting. It has three parts. First, the maximum expected profits are calculated (line number 1 to 4), which takes $O(nr_{avg})$ time. Next, the primary allocation is performed (line number 5 to 10), which takes $O(n)$ time. Finally, residual target (if any) is allocated or excess target (if any) is removed; both take $O(n^2)$ time. Thus, Procedure 1 takes $O(n^2 + nr_{avg})$ time. Together with the optimization in the resultant single-product problems (by Algorithm 3), TSH with Rule-I takes $O(n^2 + nr_{avg})$ time, which is polynomial (in $n$ and $r_{avg}$).

**Procedure 1** Target Splitting Rule-I

**Input:** $n$, $T$, parameters $\langle a_i, b_i, m_i, c_i, s_i$ for $i = 1, 2, \dots, n\rangle$, $p_i(\cdot)$ for $i = 1, 2, \dots, n$ $\langle$marginal pmf vectors$\rangle$ and $P_i(\cdot)$ for $i = 1, 2, \dots, n$ $\langle$marginal cdf vectors$\rangle$.

**Output:** $T_1, T_2, \dots, T_n$ $\langle$individual targets$\rangle$.

**for** $i = 1$ **to** $n$ **do**
  $\xi \leftarrow (m_i + s_i)/(m_i + c_i + s_i)$;
  $Q' \leftarrow \min\{Q \in \{a_i, a_i + 1, \dots, b_i\} : P_i(Q) \geq \xi\}$ $\langle$by binary search$\rangle$;
  $\Pi_i^* \leftarrow (m_i + c_i) \sum_{x=a_i}^{Q'} x p_i(x) - s_i \sum_{y=Q'+1}^{b_i} y p_i(y)$;
**for** $i = 1$ **to** $n$ **do**
  $Q_0 \leftarrow \{(m_i + c_i)a_i + s_i b_i\}/(m_i + c_i + s_i)$;
  $\Pi_a \leftarrow m_i a_i - c_i(\lceil Q_0 \rceil - a_i)$,  $\Pi_b \leftarrow m_i \lfloor Q_0 \rfloor - s_i(b_i - \lfloor Q_0 \rfloor)$;
  $\underline{T}_i \leftarrow \max\{\Pi_a, \Pi_b\}$,  $\overline{T}_i \leftarrow m_i b_i$;
**for** $i = 1$ **to** $n$ **do**
  $T_i \leftarrow \max\left\{\underline{T}_i, \min\left\{\overline{T}_i, \left(\Pi_i^* / \sum_{j=1}^n \Pi_j^*\right) \times T\right\}\right\}$ $\langle$primary allocation$\rangle$;
$re \leftarrow T - \sum_{i=1}^n T_i$ $\langle$residual/excess amount$\rangle$;
**while** $re > 0$ **do**
  **for** $i = 1$ **to** $n$ **do**
    **if** $T_i = \overline{T}_i$ **then**
      $\Pi_i \leftarrow 0$;
  **for** $i = 1$ **to** $n$ **do**
    $T_i \leftarrow \min\left\{\overline{T}_i, T_i + \left(\Pi_i^* / \sum_{j=1}^n \Pi_j^*\right) \times re\right\}$ $\langle$residue allocation$\rangle$;
  $re \leftarrow T - \sum_{i=1}^n T_i$;
**while** $re < 0$ **do**
  **for** $i = 1$ **to** $n$ **do**
    **if** $T_i = \underline{T}_i$ **then**
      $\Pi_i \leftarrow 0$;
  **for** $i = 1$ **to** $n$ **do**
    $T_i \leftarrow \max\left\{\underline{T}_i, T_i + \left(\Pi_i^* / \sum_{j=1}^n \Pi_j^*\right) \times re\right\}$ $\langle$excess removal$\rangle$;
  $re \leftarrow T - \sum_{i=1}^n T_i$;
**return** $T_1, T_2, \dots, T_n$.

## 2.2 Rule-II

Instead of splitting $T$ in one go, we can break it into two parts: $\underline{T}$ and $(T - \underline{T})$. $\underline{T}$ can be allotted to individual products as $\underline{T}_1, \underline{T}_2, \dots, \underline{T}_n$ without loosing on the value of satiation probability as $P_{\underline{T}}(Q_i^*) = 1$. Next, $(T - \underline{T})$ can be allotted incrementally with prefixed step size of $\Delta T$. In every iteration, $\Delta T$ is allotted to the product with highest $P_{(T_i + \Delta T)}(Q_i^*)/P_{T_i}(Q_i^*)$ value.

Ideally, $\Delta T$ should be extremely small ($\mathrm{d}T$). It should be allotted to the product with the highest $\partial P_T(Q^o)/\partial T_i$ value, where $Q^o$ denotes the optimal solution of the MPSNP, as that would ensure least reduction in the objective function value. However, the form of $P_T(Q^o)$ is unknown. Note that $P_T(Q^o) \geq P_T(Q^*) \geq \prod_{i=1}^{n} P_{T_i}(Q_i^*)$ where $Q^* = (Q_1^*, Q_2^*, \ldots, Q_n^*)$. $\mathcal{M} := \prod_{i=1}^{n} P_{T_i}(Q_i^*)$ can be considered as an indicator for $P_T(Q^o)$. Then

$$\frac{\partial \mathcal{M}}{\partial T_i} = \frac{\mathcal{M}}{P_{T_i}(Q_i^*)} \frac{\mathrm{d}P_{T_i}(Q_i^*)}{\mathrm{d}T_i} \approx \frac{\mathcal{M}}{\Delta T} \left( \frac{P_{(T_i+\Delta T)}(Q_i^*)}{P_{T_i}(Q_i^*)} - 1 \right).$$

Clearly, highest $P_{(T_i+\Delta T)}(Q_i^*)/P_{T_i}(Q_i^*)$ is an indicator for highest $\partial P_T(Q^o)/\partial T_i$.

Based on the above discussions, Procedure 2 is implemented to performs the target splitting. Instead of fixing $\Delta T$, we fixed the number of steps $N$ in which $(T - \underline{T})$ is allotted. Procedure 2 has three parts. First, the initial allocation is performed (line number 1 to 4), which takes $O(n)$ time. Next, the incremental allocation scheme is initiated (line number 5 to 9), which takes $O(nr_{avg})$ time. Finally, the incremental allocation is performed, which takes $O(N(n + r_{max}))$ time where $r_{max}$ denotes the maximum demand range. For practical problems, $r_{max} > n$ and $N$ can be chosen to be greater than $n$. Note that a larger $N$ is likely to improve accuracy. Then Procedure 2 takes $O(Nr_{max})$ time. Together with the optimization in the resultant single-product problems (by Algorithm 3), TSH with Rule-II takes $O(Nr_{max})$ time.

**Procedure 2** Target Splitting Rule-II

**Input:** $n$, $T$, parameters $\langle a_i, b_i, m_i, c_i, s_i$ for $i = 1, 2, \ldots, n\rangle$, $P_i(\cdot)$ for $i = 1, 2, \ldots, n$ ⟨marginal cdf vectors⟩, and $N$ ⟨number of steps⟩.

**Output:** $T_1, T_2, \ldots, T_n$ ⟨individual targets⟩.

1. **for** $i = 1$ **to** $n$ **do**
2. $\quad Q_0 \leftarrow \{(m_i + c_i)a_i + s_i b_i\}/(m_i + c_i + s_i)$;
3. $\quad \Pi_a \leftarrow m_i a_i - c_i(\lceil Q_0 \rceil - a_i)$, $\Pi_b \leftarrow m_i \lfloor Q_0 \rfloor - s_i(b_i - \lfloor Q_0 \rfloor)$;
4. $\quad \underline{T}_i \leftarrow \max\{\Pi_a, \Pi_b\}$, $\overline{T}_i \leftarrow m_i b_i$, $T_i \leftarrow \underline{T}_i$;
5. $rsd \leftarrow (T - \sum_{i=1}^{n} \underline{T}_i)$, $\Delta T \leftarrow rsd/N$;
6. **for** $i = 1$ **to** $n$ **do**
7. $\quad tar \leftarrow \min\{\overline{T}_i, T_i + \Delta T\}$;
8. $\quad nv_i \leftarrow P_{tar}(Q_i^*)$ ⟨using Algorithm 3⟩;
9. $\quad cv_i \leftarrow 1$, $r_i \leftarrow nv_i/cv_i$;
10. **while** $rsd > 0$ **do**
11. $\quad j \leftarrow k$ s.t. $r_k = \max\{r_1, r_2, \ldots, r_n\}$ ⟨by linear search⟩;
12. $\quad T_j \leftarrow \min\{\overline{T}_j, T_j + \Delta T\}$, $rsd \leftarrow (T - \sum_{i=1}^{n} \underline{T}_i)$;
13. $\quad tar \leftarrow \min\{\overline{T}_j, T_j + \Delta T\}$, $cv_j \leftarrow nv_j$;
14. $\quad nv_j \leftarrow P_{tar}(Q_j^*)$ ⟨using Algorithm 3⟩;
15. $\quad$ **if** $T_j < \overline{T}_j$ **then**
16. $\quad\quad r_j \leftarrow nv_j/cv_j$;
17. $\quad$ **else**
18. $\quad\quad r_j \leftarrow 0$;
19. **return** $T_1, T_2, \ldots, T_n$.

## 2.3 Numerical results

We implemented the scheme of Khanra & Soman (2014) for evaluating heuristic performance. Heuristic accuracy is measure by deviation of the heuristic solution from the optimum solution ($\delta$) for scaled-down test instances. We solved full-scale independent demand (ID) test instances to understand time requirement of TSH. Full scale dependent demand (DD) instances are not solved as the heuristic does not differentiate between independent and dependent demand cases. We implemented the procedures in *GNU Octave 3.6.4 (GCC 4.6.2)*. Test instances were solved in *Intel Core i5 (3.30 GHz) processors* with *4 GB memory*.

Table 1 exhibits accuracy of TSH for various instance classes. An instance class is defined by the number of products (nP, $n = 2, 3, \dots$) and demand type (ID and DD). TSH is 'reasonably accurate'. Maximum deviation was less than 0.1 for Rule-I and slightly more than 0.1 for Rule-II. Average deviation was less than 0.02 for Rule-I and slightly more than 0.02 for Rule-II. Rule-I performed better than Rule-II for the independent demand case. Rule-II was marginally better than Rule-I for the dependent demand case.

Table 1: Accuracy of TSH

| | | Rule-I | | | Rule-II | | |
|---|---|---|---|---|---|---|---|
| Class | Num | $\delta_{min}$ | $\delta_{max}$ | $\delta_{avg}$ | $\delta_{min}$ | $\delta_{max}$ | $\delta_{avg}$ |
| 3PID | 50 | 0.00006 | 0.038 | 0.0088 | 0.00003 | 0.094 | 0.0127 |
| 4PID | 50 | 0.00001 | 0.078 | 0.0092 | 0.00000 | 0.103 | 0.0145 |
| 5PID | 25 | 0.00014 | 0.046 | 0.0105 | 0.00004 | 0.085 | 0.0176 |
| 6PID | 25 | 0.00000 | 0.077 | 0.0165 | 0.00000 | 0.102 | 0.0239 |
| 2PDD | 15 | 0.00000 | 0.009 | 0.0016 | 0.00001 | 0.015 | 0.0025 |
| 3PDD | 20 | 0.00028 | 0.060 | 0.0132 | 0.00000 | 0.054 | 0.0096 |
| 4PDD | 20 | 0.00001 | 0.058 | 0.0148 | 0.00000 | 0.049 | 0.0114 |
| 5PDD | 10 | 0.00022 | 0.044 | 0.0143 | 0.00000 | 0.019 | 0.0080 |
| 6PDD | 10 | 0.00000 | 0.082 | 0.0129 | 0.00000 | 0.044 | 0.0073 |

Table 2: Computation time for TSH

| | | Rule-I | | | Rule-II | | |
|---|---|---|---|---|---|---|---|
| Class | Num | *Min* | *Max* | *Avg* | *Min* | *Max* | *Avg* |
| 3PID | 50 | 0.0015 | 0.0016 | 0.0015 | 22.75 | 24.41 | 23.74 |
| 4PID | 50 | 0.0019 | 0.0021 | 0.0020 | 22.73 | 24.20 | 23.63 |
| 5PID | 25 | 0.0023 | 0.0025 | 0.0024 | 22.87 | 24.17 | 23.67 |
| 6PID | 25 | 0.0028 | 0.0030 | 0.0029 | 23.10 | 24.37 | 23.90 |
| 7PID | 50 | 0.0033 | 0.0036 | 0.0034 | 23.54 | 24.51 | 24.07 |
| 8PID | 50 | 0.0038 | 0.0041 | 0.0040 | 23.62 | 24.62 | 24.24 |
| 9PID | 50 | 0.0042 | 0.0047 | 0.0044 | 23.65 | 24.68 | 24.20 |

Table 2 shows computation time for TSH. In the case of Rule-II, $N = 10^5$ was taken.

Polynomial time complexity of the heuristic is evident from Table 2. Computation time for Rule-I is in milliseconds. Computation time for Rule-II is in seconds. Note that computation time for Rule-II can be controlled by adjusting the number of steps ($N$). An important observation regarding the computation time is its 'extremely slow' growth with the number of products. Any practical MPSNP can be solved by TSH in 'reasonable time'.

## 3 Implication for the "target assignment problem"

The target assignment problem is referred to as the problem of assigning divisional targets in multi-division organizations when each division handles one or more newsboy-like products. Here, we deal with the target assignment problem with satiation objective, i.e., the organization and its divisions are concerned with achieving their respective profit targets. Shi et al. (2010) studied this problem. They considered price-dependent demand. Their study assumes zero stock-out costs and single-product per division. In many real-life situations, these restrictions may not hold. In such situations, TSH for the MPSNP can be used to determine divisional targets. Demand needs to be exogenous for the application of THS.

The organization faces an MPSNP. We can employ the target splitting rules to determine product specific targets and then simply combine them to get the divisional targets. Table 3 demonstrates performance of this procedure for the three-product case.

Table 3: Application of TSH for the target assignment problem

| | | | Config -I | | Config -II | | Config -III | |
|---|---|---|---|---|---|---|---|---|
| Sl | $T$ | $P_T(Q^*)$ | $T1, T23$ | $P_T(Q)$ | $T2, T31$ | $P_T(Q)$ | $T3, T12$ | $P_T(Q)$ |
| 1 | 5940 | 0.881 | $-5.7, 5945.7$ | 0.876 | $980.9, 4959.1$ | 0.868 | $4964.9, 975.1$ | 0.864 |
| 2 | 4530 | 0.200 | $-59.7, 4589.7$ | 0.190 | $1099.5, 3430.5$ | 0.193 | $3490.1, 1039.9$ | 0.192 |
| 3 | 2418 | 0.829 | $226.8, 2191.2$ | 0.823 | $910.1, 1507.9$ | 0.827 | $1281.1, 1136.9$ | 0.829 |
| 4 | 1150 | 0.452 | $-59.3, 1209.3$ | 0.424 | $46.5, 1103.5$ | 0.452 | $1162.8, -12.8$ | 0.449 |
| 5 | 3405 | 0.356 | $-58.6, 3463.6$ | 0.327 | $1031.8, 2373.2$ | 0.348 | $2431.8, 973.2$ | 0.352 |
| 6 | 4990 | 0.420 | $-57.3, 5047.3$ | 0.395 | $1419.5, 3570.5$ | 0.418 | $3627.8, 1362.2$ | 0.419 |
| 7 | 4750 | 0.808 | $23.1, 4726.9$ | 0.751 | $2266.0, 2484.0$ | 0.793 | $2460.9, 2289.1$ | 0.793 |
| 8 | 4250 | 0.551 | $23.9, 4226.1$ | 0.548 | $657.0, 3593.0$ | 0.544 | $3569.1, 680.9$ | 0.547 |
| 9 | 5680 | 0.803 | $951.4, 4728.6$ | 0.799 | $1016.3, 4663.7$ | 0.786 | $3712.3, 1967.7$ | 0.791 |
| 10 | 2850 | 0.416 | $254.5, 2595.5$ | 0.413 | $1504.7, 1345.3$ | 0.414 | $1090.8, 1759.2$ | 0.415 |

The organization has two divisions handling three products . One division handles single-product and the other division handles two products. This results in three configurations (i.e., P1+P23, P2+P31, P3+P12 where Pk indicates the $k^{th}$ product). Table 3 shows divisional targets for each configuration. Performance of the heuristic is measured by the deviation of the divisional decision (optimum for each division) from the theoretical maximum (centralized decision). We used Rule-I for target splitting. The procedure performs quite well. Maximum deviation was little more than 0.05, while average deviation was 0.01.

## 4 Conclusion

In this paper, we developed an efficient heuristic for the MPSNP. Prior to this, no such methods were available for the general case. Our heuristic decomposes the multi-product problem into easily solvable single-product problems. With this new heuristic, the MPSNP can be solved for any number of products. The heuristic is reasonably accurate. We also demonstrated the applicability of this heuristic for the target assignment problem.

We developed two decomposition rules. Better rules (in terms of accuracy) can be developed. We did not consider resource constraints. Like the splitting of profit target, the resource amount can be split too. The target assignment problem can also be studied in depth.

## Acknowledgements

The author is grateful to the Computer Center of IIM Ahmedabad for its support in computation.

### Appendix A

Following notations are used in this paper.

| | |
|---|---|
| $n$ | Number of products (positive integer). Subscript $i$ is used to represent parameters and variables specific to a product. Unless stated otherwise, $i = 1, 2, \ldots, n$. |
| $m_i$ | 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 $X_i$. |
| $Q_i$ | Order quantity of the $i^{th}$ product (non-negative integer). |
| $Q$ | Order quantity vector. $Q = (Q_i) = (Q_1, Q_2, \ldots, Q_n)$. |
| $T$ | Profit target. $\underline{T}$ and $\overline{T}$ denote its maximum assured and achievable values. We take $T \in (\underline{T}, \overline{T})$ so that the problem has non-trivial optimal solution. |
| $P_T(Q)$ | Satiation probability (i.e., the probability of achieving a given profit target) for ordering decision $Q$. |

### Appendix B

Algorithm 3 solves the single-product satiating newsboy problem optimally.

**Algorithm 3** Algorithm for the single-product satiating newsboy problem

**Input:** $T$, parameters $\langle m, c, s, a, b \rangle$, and $P(\cdot)$ $\langle$marginal cdf vector$\rangle$.

**Output:** $Q^*$ $\langle$the optimal solution$\rangle$.

1. $a_0 \leftarrow \max\{a, \lceil T/m \rceil\}$
2. $Q^* \leftarrow a_0, \quad bm \leftarrow 0;$
3. **for** $Q = a_0$ **to** $b$ **do**
4. $\quad \underline{x} \leftarrow \max\{a, Q - (mQ - T)/(m + c)\}, \quad \overline{x} \leftarrow \min\{b, Q + (mQ - T)/s\};$
5. $\quad chk \leftarrow P(\lfloor \overline{x} \rfloor) - P(\lceil \underline{x} \rceil - 1);$
6. $\quad$ **if** $chk > bm$ **then**
7. $\qquad Q^* \leftarrow Q, \quad bm \leftarrow chk;$
8. **return** $Q^*$.

## Appendix C

In the classical newsboy problem, stochastic profit $\Pi$ and its expected value are given by

$$\begin{aligned}
\Pi(Q, X) &= m \min\{Q, X\} - c \max\{0, Q - X\} - s \max\{0, X - Q\}. \\
E[\Pi(Q)] &= m \left[ \sum_{x=a}^{Q} xp(x) + \sum_{x=Q+1}^{Q} Qp(x) \right] - c \sum_{x=a}^{Q} (Q - x)p(x) - s \sum_{x=Q+1}^{b} (x - Q)p(x) \\
&= (m + c) \sum_{x=a}^{Q} xp(x) + Q\left[(m + s)\{1 - P(Q)\} - cP(Q)\right] - s \sum_{x=Q+1}^{b} xp(x) \\
&= (m + c) \sum_{x=a}^{Q} xp(x) - s \sum_{x=Q+1}^{b} xp(x) + Q(m + c + s)\{\xi - P(Q)\}
\end{aligned}$$

where $\xi = (m + s)/(m + c + s)$ denotes the critical fractile.

$$\begin{aligned}
E[\Pi(Q + 1)] - E[\Pi(Q)] &= (m + c)(Q + 1)p(Q + 1) + s(Q + 1)p(Q + 1) \\
&+ (m + c + s)\left[\{\xi - P(Q + 1)\} - Qp(Q + 1)\right] = (m + c + s)\{\xi - P(Q)\}.
\end{aligned}$$

Hence, $E[\Pi(Q + 1)] \geq E[\Pi(Q)]$ or $E[\Pi(Q)]$ is increasing if $\xi \geq P(Q)$ and $E[\Pi(Q + 1)] \leq E[\Pi(Q)]$ or $E[\Pi(Q)]$ is decreasing if $\xi \leq P(Q)$. Let us define $Q'_L$ and $Q'_R$ as

$$\begin{aligned}
Q'_L &= \max\left\{Q \in \{a, a + 1, \ldots, b\} : P(Q) \leq \xi\right\}, \\
Q'_R &= \min\left\{Q \in \{a, a + 1, \ldots, b\} : P(Q) \geq \xi\right\}.
\end{aligned}$$

Then $\xi \geq P(Q)$ if $Q \leq Q'_L$ and $\xi \leq P(Q)$ if $Q \geq Q'_R$. Note that $Q'_L \leq Q'_R$. Thus, $E[\Pi(Q)]$ is increasing till $Q'_L$ and decreasing after $Q'_R$. Hence, $Q \in \{Q'_L, Q'_L + 1, \ldots, Q'_R\}$ with the maximum value for $E[\Pi(Q)]$ is the expected profit maximizing order quantity $Q'$. If $p(x) > 0 \ \forall x \in \{a, a + 1, \ldots, b\}$, $Q'_R - Q'_L \leq 1$. We take $Q' = Q'_R$. This is unlikely to induce significant error. The maximum expected profit is calculated as

$$\Pi^* = E[\Pi(Q')] \approx (m+c)\sum_{x=a}^{Q'} xp(x) - s\sum_{x=Q'+1}^{b} xp(x).$$

## References

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.

Khanra, A., & Soman, C. (2014). *Heuristics for the multi-product satiating newsboy problem* (No. 2014-01-02). Indian Institute of Management Ahmedabad, India.

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

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.

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

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

Shi, C. V., Zhao, X., & Xia, Y. (2010). The setting of profit targets for target oriented divisions. *European Journal of Operational Research*, *206*(1), 86–92.