A New Algorithm and a New Heuristic for Serial Supply Systems.

Guillermo Gallego*
Department of Industrial Engineering and Operations Research
Columbia University†

Özalp Özer
Department of Management Science and Engineering
Stanford University

September 1, 2004

Abstract

We present a new dynamic programming formulation for the stochastic multi-stage serial inventory system based on the cost of sub-system with fewer stages. A heuristic based on judiciously selected common downstream holding costs requires solving one newsvendor problem per stage. A closed form approximate upper bound allows for accurate sensitivity analysis. [1]

Key words: stochastic inventory systems; multi-echelon; newsvendor bound

*Corresponding Author: 326 Mudd Building, Columbia University, New York, NY gmg2@columbia.edu.

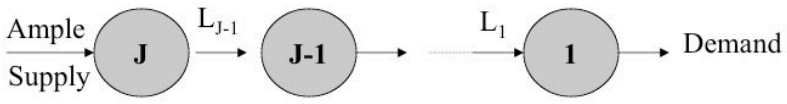
†Research Supported by NSF Grant DMI-02-18104

[1]Acknowledgments: We wish to thank Haengju Lee, Ozge Sahin and Jay Sethuraman for useful suggestions on earlier versions of this note.

1 Series Systems

The study of multi-stage serial inventory systems is central to the study of supply chain management both as a benchmark and as a building block for more complex supply networks. Existing policy evaluation and optimization algorithms are, unfortunately, difficult to understand and communicate. We provide a dynamic programming (DP) formulation based on the idea of optimally allocating a given echelon-inventory level between the upstream stage and the downstream series system. This formulation yields an algorithm that can be improved by incorporating gradient updates. We develop a simple, near-optimal, heuristic that follows from the DP formulation by judiciously selecting a common holding cost for the downstream stages. The heuristic calls for solving a single newsvendor problem per stage and is very accessible to students and practitioners. The need to develop accurate and accessible spreadsheet-based heuristics was recently identified by Shang and Song [13] who develop a heuristic based on solving two newsvendor problems per stage. We evaluate our heuristic and compare it to that of Shang and Song by testing it on the set of test problems in Gallego and Zipkin [8] and Shang and Song [13] and in additional experiments designed to test the performance when different stages have different lead times. Our numerical results indicate that our heuristic is near optimal with an average error that is lower than the Shang and Song heuristic. Finally, we provide an approximate distribution-free bound that accurately reflects the sensitivity of the optimal average cost to changes in system parameters.

Consider a series system consisting of J stages as illustrated in the figure. Stage $j < J$ procures from Stage $j + 1$ and Stage J replenishes from an outside supplier with ample stock. Customer demand occurs only at Stage 1 and follows a (compound) Poisson process, $\{D(t), t \geq 0\}$ with arrival rate λ and random demand size X with $E[X^2] < \infty$. It takes L_j units of time for a unit to arrive at Stage j once it is released by its predecessor.



Unsatisfied demand is backordered at each stage but only Stage 1 incurs a linear backorder penalty cost p, per unit, per unit time. We assume without loss of generality that each stage adds value as the item moves through the supply chain, so echelon holding costs h_j^e are positive. The local holding cost for stage j is $h_j = h^e[j, J] \equiv \sum_{i=j}^{J} h_i^e$, where sums over empty sets will be defined as zero. In particular, $h_{J+1} = 0$. The system is operated under continuous review, so an order is placed every time a demand occurs. As pointed out by Zipkin [14], this is justified for expensive and/or slow moving items.

The following random variables describe the state of Stage j in equilibrium:

$$\begin{aligned} D_j &= \text{leadtime demand}, \\ I_j &= \text{on-hand inventory}, \\ B_j &= \text{backorders}. \end{aligned}$$

The total long run average cost for *any* policy can be expressed as

$$E[\sum_{k=1}^{J} h_k I_k + pB_1 + \sum_{k=2}^{J} h_k D_{k-1}]. \tag{1}$$

Optimality of an echelon base stock policy $(s_J, \ldots, s_1)$ for this series system is well known (see Federgruen and Zipkin [4], Chen and Zheng [2] and Zipkin [14], and the original work by Clark and Scarf [3]). Under this policy, the manager continuously monitors the echelon inventory order position at each stage and places an order from Stage $j+1$ to bring it up to s_j whenever it is below this level. An echelon base stock policy $(\tilde{s}_J, \tilde{s}_{J-1}, \ldots, \tilde{s}_1)$ is equivalent to $(s_J, \ldots, s_1)$ where $s_j = \min(\tilde{s}_J, \ldots, \tilde{s}_j)$, so there exists an echelon base-stock policy satisfying the property $s_J \geq s_{J-1} \geq \ldots \geq s_1$. Let $s_o \equiv 0$ and let $s'_j \equiv s_j - s_{j-1}$, $j \geq 1$ denote the local base stock level at stages $j = 1, \ldots, J$. This local base stock policy is equivalent to the echelon base stock policy (see Axsater [1] Propositions 1 and 2 or Zipkin [14] pg 306). The following recursion gives the steady state distribution of the local on-hand inventory I_k and the backorder B_k at Stage k starting with $B_{J+1} = 0$.

$$I_k = (s'_k - D_k - B_{k+1})^+ \qquad B_k = (B_{k+1} + D_k - s'_k)^+ \quad k = J, \ldots, 1. \tag{2}$$

Then, the total long-run average cost can be expressed as

$$g_J(s_J, \ldots, s_1) = E[\sum_{k=1}^{J} h_k (s'_k - D_k - B_{k+1})^+ + pB_1 + \sum_{k=2}^{J} h_k D_{k-1}]. \tag{3}$$

Consider the sub-system consisting of Stages $\{j, \ldots, 1\}$ for some $j \in \{1, \ldots, J-1\}$ and assume that Stage j replenishes its inventory from an external supplier with ample supply. This sub-system is equivalent to the original J stages series system with $D_k \equiv 0, h_k \equiv 0$ for all $k > j$. Let $g_j(s_j, \ldots, s_1)$ be the long-run total average cost for an echelon base-stock policy $(s_j, \ldots, s_1)$. Then $g_j(s_j, \ldots, s_1)$ can be obtained via equations (2) and (3) by replacing J by j starting the computations with $B_{j+1} = 0$.

The next result provides a link between the cost of the different sub-systems.

Proposition 1

$$g_1(s) = E[h_1(s - D_1)^+ + p(D_1 - s)^+],$$

and for $j = 2, \ldots, J$

$$g_j(s_j, \ldots, s_1) = E[h_j(s'_j - D_j)^+ + g_{j-1}(\min(s_{j-1}, s_j - D_j), s_{j-2}, \ldots, s_1) + h_j D_{j-1}]. \tag{4}$$

Proof. Let $g_o(s) = p(-s)^+$, then for $j = 1$,

$$\begin{aligned} g_1(s) &= E[h_1(s - D_1)^+ + pB_1] \\ &= E[h_1(s - D_1)^+ + p(D_1 - s)^+] \\ &= E[h_1(s - D)1)^+ + g_o(\min(0, s_1 - D_1)). \end{aligned}$$

This shows that the result holds for $j = 1$. Suppose the result holds for some $j < J$. We now show that it holds for $j + 1$. Clearly from Equation (3),

$$g_{j+1}(s_{j+1}, s_j, \ldots, s_1) = Eh_{j+1}(s'_{j+1} - D_{j+1})^+ + E[\sum_{k=1}^{j} h_k(s'_k - D_k - B_{k+1})^+ + pB_1 + \sum_{k=2}^{j} h_k D_{k-1}] + h_{j+1}ED_j.$$

The term in square brackets looks exactly as the definition of $g_j(s_j, \ldots, s_1)$ except that $B_{j+1} = (D_{j+1} - s'_{j+1})^+$ instead of 0. Thus, the echelon base-stock level at Stage j is given by $s_j - B_{j+1} = \min(s_j, s_{j+1} - D_{j+1})$ instead of s_j and this justifies (4) for $j + 1$, completing the proof.

□

We now provide a recursion to find the optimal expected cost at each stage for an arbitrary echelon base stock level, as well as an algorithm to find optimal base-stock policies for each sub-system $\{j, \ldots, 1\}$ for $j = 1, \ldots, J$.

Let $c_1(s) = g_1(s)$ and for $j = 2, \ldots, J$ define

$$c_j(s) = \min_{x \in \{0,\ldots,s\}} c_j(x; s) \tag{5}$$

recursively, via

$$c_j(x; s) = E[h_j(x - D_j)^+ + c_{j-1}(\min(s - x, s - D_j)) + h_j D_{j-1}]. \tag{6}$$

Let $\mathcal{N} = \{0, 1, \ldots, \}$ be the set of non-negative integers and let

$$s_j^* \equiv \min\{s \in \mathcal{N} : \Delta c_j(s) > h_{j+1}\} \quad \text{for} \quad j = 1, \ldots, J, \tag{7}$$

We will show that $c_j(s)$ is the long run average cost of *optimally* managing the sub-system $\{j, \ldots, 1\}$ given *echelon* base stock level s and that optimal base stock levels are given by (7). Before we prove this result formally, we will provide an intuitive link between equations (6) and (5). Suppose that we have computed $c_j(\cdot)$ and consider the sub-system $\{j + 1, \ldots, 1\}$. Our goal is to compute $c_{j+1}(\cdot)$ from the knowledge of $c_j(\cdot)$. To link the two sub-systems, we decompose the echelon base stock level s of sub-system $\{j+1, \ldots, 1\}$ by allocating x units to Stage $j+1$ and $s-x$ units for sub-system $\{j, \ldots, 1\}$. Given this allocation, the net inventory at Stage $j + 1$ will be $(x - D_{j+1})^+$ which accrues at cost rate h_{j+1}. Since Stage $j + 1$ will face a shortage when $D_{j+1} - x > 0$, the effective echelon inventory for sub-system $\{j, \ldots, 1\}$ is $s - x - (D_{j+1} - x)^+ = \min(s - x, s - D_{j+1})$. Thus, a finite local base stock level at Stage $j+1$ imposes an externality to the sub-system $\{j, \ldots, 1\}$ whose expected cost is now $Ec_j(\min(s - x, s - D_{j+1}))$. As a result, when we allocate $x \leq s$ units of local base stock level to Stage $j + 1$, the cost of managing a series system with $j + 1$ stages is given by (6).

Theorem 1 *1) $c_j(s)$ is convex in s for all $j = 1, \ldots, J$ and given by (5) for $j = 2, \ldots, J$; 2) $x_j(s) = (s - s_{j-1}^*)^+$ minimizes $c_j(x; s)$ for $j = 2, \ldots, J$; 3) The echelon base stock policy $(s_j^*, \ldots, s_1^*)$ is optimal and $c_j(s_j^*)$ is the optimal expected cost for sub-system $\{j, \ldots, 1\}$, $j = 1, \ldots, J$.*

Proof. For $j = 1$, the optimal cost of managing the subsystem given echelon base stock level s is simply $c_1(s) = g_1(s)$ since there is nothing to optimize. Notice that $\Delta c_1(s) \equiv c_1(s+1) - c_1(s) = (h_1 + p)Pr(D_1 \leq s) - p$ is non-decreasing in s so $c_1(s)$ is convex. The largest optimal base stock level is given by $s_1^* = \min\{s \in \mathcal{N} : \Delta c_1(s) > 0\}$ which is consistent with (7) since $h_2 = 0$ for the sub-system consisting only of Stage 1. As a result $c_1(s_1^*)$ is the optimal expected cost for the sub-system consisting of Stage 1 only. This establishes Parts 1 and 3 for $j = 1$.

Consider now the sub-system $\{2, 1\}$ and notice that

$$\begin{aligned} c_2(s) &= \min_{s_1 \in \{0,\ldots,s\}} g_2(s, s_1) \\ &= \min_{s_1 \in \{0,\ldots,s\}} E[h_2(s - s_1 - D_2)^+ + g_1(\min(s_1, s - D_2)) + h_2 D_1] \\ &= \min_{x \in \{0,\ldots,s\}} E[h_2(x - D_2)^+ + c_1(\min(s - x, s - D_2)) + h_2 D_1] \\ &= \min_{x \in \{0,\ldots,s\}} c_2(x; s), \end{aligned}$$

where the second equality is from Equation (4) and the third is by substituting x for $s - s_1$. Therefore, $c_2(s)$ is given by (5). Now let $\Delta c_2(x; s) \equiv c_2(x+1; s) - c_2(x; s)$. Then,

$$\Delta c_2(x; s) = [h_2 - \Delta c_1(s - x - 1)]Pr(D_2 \leq x).$$

Notice that $\Delta c_2(x; s) = 0$ for all $x < 0$ on account of $D_2 \geq 0$. The convexity of c_1 implies that $\Delta c_1(s - x - 1)$ is decreasing in x. As a consequence, $\Delta c_2(x; s)$ has at most one sign change over the range $x \in \{0, \ldots, s\}$ and this would have to be from $-$ to $+$. From Equation (7), s_1^* is the smallest non-negative integer y such that $\Delta c_1(y) > h_2$. Notice that s_1^* is the largest minimizer of the newsvendor problem with holding cost $h_1 - h_2$, backorder cost $p + h_2$ and demand D_1. In particular, s_1^* is independent of the distribution of D_2. We have shown that $x = (s - s_1^*)^+$ is a minimizer of $c_2(\cdot; s)$ establishing Part 2 for $j = 2$. This result implies that allocating s_1^* units of echelon base stock level to Stage 1 is optimal when $s \geq s_1^*$. Therefore, we have

$$\begin{aligned} c_2(s) &= c_2((s - s_1^*)^+; s) = E[h_2((s - s_1^*)^+ - D_2)^+ + c_1(\min(s_1^*, s - D_2)) + h_2 D_1] \\ &= E[h_2(s - s_1^* - D_2)^+ + c_1(\min(s_1^*, s - D_2)) + h_2 D_1], \end{aligned}$$

where the last equation follows since $(x^+ - a)^+ = (x - a)^+$ when $a \geq 0$. To see that c_2 is convex, notice that

$$\Delta c_2(s) = h_2 Pr(D_2 \leq s - s_1^*) + \sum_{x=(s+1-s_1^*)^+}^{\infty} \Delta c_1(s - x) Pr(D_2 = x).$$

Let $\Delta^2 c_2(s) \equiv \Delta c_2(s+1) - \Delta c_2(s)$, then

$$\Delta^2 c_2(s) = \sum_{x=(s+2-s_1^*)^+}^{\infty} \Delta^2 c_1(s - x) Pr(D_2 = x) + [h_2 - \Delta c_1(s_1^* - 1)]Pr(D_2 = s + 1 - s_1^*).$$

Now, $\Delta^2 c_1(s - x) \geq 0$ on account of the convexity of c_1, so the first term is non-negative. The second term is also non-negative by the definition of s_1^*. This establishes Part 1 for $j = 2$. Recall

that $h_3 \equiv 0$ for sub-system $\{2,1\}$. Hence, the minimizer s_2^* of $c_2(s)$ is given by Equation (7) with $h_3 = 0$. With this final observation, we have shown that (s_2^*, s_1^*) is an optimal echelon base-stock policy for sub-system $\{2,1\}$ with optimal expected cost $c_2(s_2^*)$ establishing Part 3 for $j = 2$.

Assume now that all three statements are true for sub-system $\{j,\ldots,1\}$ for some $j < J$. We now add Stage $j+1$ with local holding cost h_{j+1}. Then,

$$\begin{aligned}
c_{j+1}(s) &= \min_{0\le s_1\le\ldots\le s_j\in\{0,\ldots,s\}} g_{j+1}(s, s_j, \ldots, s_1) \\
&= \min_{0\le s_1\le\ldots\le s_j\in\{0,\ldots,s\}} E[h_{j+1}(s - s_j - D_{j+1})^+ + g_j(\min(s_j, s - D_{j+1}), s_{j-1}, \ldots, s_1) + h_{j+1}D_j] \\
&= \min_{x\in\{0,\ldots,s\}} E[h_{j+1}(x - D_{j+1})^+ + \min_{s_1,\ldots,s_{j-1}} g_j(\min(s - x, s - D_{j+1}), s_{j-1}, \ldots, s_1) + h_{j+1}D_j] \\
&= \min_{x\in\{0,\ldots,s\}} E[h_{j+1}(x - D_{j+1})^+ + c_j(\min(s - x, s - D_{j+1})) + h_{j+1}D_j] \\
&= \min_{x\in\{0,\ldots,s\}} c_{j+1}(x; s),
\end{aligned}$$

so $c_{j+1}(s)$ is given by (5). Notice that $c_{j+1}(x+1;s) - c_{j+1}(x,s)$ is non-zero only when $D_{j+1} \le x$ and is equal, in this case, to $h_{j+1} - c_j(s-x-1) + c_j(s-x) = h_{j+1} - \Delta c_j(s-x-1)$. Consequently,

$$\Delta c_{j+1}(x; s) = [h_{j+1} - \Delta c_j(s - x - 1)]Pr(D_{j+1} \le x).$$

Now, since c_j is convex it follows that $\Delta c_{j+1}(x;s)$ has at most one sign change and this must be from $-$ to $+$. Since the sign change occurs at $(s - s_j^*)$ when $s \ge s_j^*$, it follows that $x_{j+1}(s) = (s - s_j^*)^+$ minimizes $c_{j+1}(x;s)$ so $c_{j+1}(s) = c_{j+1}((s-s_j^*)^+; s)$ establishing Part 2 for $j+1$. This result implies that allocating s_j^* units of echelon base stock level to stage j is optimal when $s \ge s_j^*$. Therefore, we have

$$c_{j+1}(s) = E[h_{j+1}(s - s_j^* - D_{j+1})^+ + c_j(\min(s_j^*, s - D_{j+1})) + h_{j+1}D_j].$$

The convexity of c_{j+1} now follows the exact argument used to establish the convexity of c_2. Indeed,

$$\Delta^2 c_{j+1}(s) = \sum_{x=(s+2-s_j^*)^+}^{\infty} \Delta^2 c_j(s-x)Pr(D_{j+1} = x) + [h_{j+1} - \Delta c_j(s_j^* - 1)]Pr(D_{j+1} = s + 1 - s_j^*) \ge 0$$

because $\Delta^2 c_j \ge 0$ and by the definition of s_j^*. This proves Part 1 for $j+1$. Recall that $h_{j+2} = 0$ for sub-system $\{j+1,\ldots,1\}$, so it follows that the minimizer s_{j+1}^* of $c_{j+1}(s)$ is given by Equation (7) with $h_{j+2} = 0$, implying that $(s_{j+1}^*, \ldots, s_1^*)$ is an optimal echelon base stock policy for sub-system $\{j+1,\ldots,1\}$ and that $c_{j+1}(s_{j+1}^*)$ is the optimal expected cost for this sub-system. This establishes Part 3 for $j+1$ and concludes the induction argument for $j+1$ and hence the proof. □

Remark: Notice that the definition of s_j^* changes as we go from sub-system $\{j,\ldots,1\}$ to sub-system $\{j+1,\ldots,1\}$ because $h_{j+1} = 0$ for sub-system $\{j,\ldots,1\}$ but $h_{j+1} > 0$ for sub-system $\{j+1,\ldots,1\}$. However, the echelon base stock policy $(s_j^*,\ldots,s_1^*)$ does not change after we add Stage $j+1$ and in the course of the algorithm we need to find each s_j^* only once. Finally, notice that $c_{J+1}(s_J^*) = c_J(s_J^*)$ on account of $h_{J+1} = 0$ and $D_{J+1} \equiv 0$.

Optimal echelon base stock levels can also be found through solving the *traditional* recursive optimization for $j = 1, 2, \ldots, J$. This formulation is based on echelon cost accounting.

$$C_j(y) = E\{h_j^e(y - D_j) + C_{j-1}(\min[y - D_j, s_{j-1}^*])\} \quad (8)$$

$$s_j^* \equiv \max\{y : C_j(y) \leq C_j(x) \text{ for all } x \neq y\}, \quad (9)$$

where $C_0(y) = (p+h_1)[y]^-$, see Chen and Zheng [2] and Gallego and Zipkin [8]. The optimal system wide average cost is given by $C_J(s_J^*)$. We now verify that the new algorithm produces the same echelon base stock levels as the traditional algorithm and find an explicit relationship between the C_js and the c_js.

Proposition 2 *1)* $C_j(s) = c_j(s) - h_{j+1}E(s - D_j)$*; 2)* $C_J(s) = c_J(s)$*; 3)*$\Delta C_j(s) = \Delta c_j(s) - h_{j+1}$*, so* s_j^* *minimizes* $C_j(s)$*.*

Proof. The proof is based on an induction argument. For the case $j = 1$, we have $C_1(s) = E[h_1^e(s - D_1) + (p + h_1)(D_1 - s)^+] = E[h_1^e(s - D_1) - h_1(s - D_1) + h_1(s - D_1) + (p + h_1)(D_1 - s)^+] = c_1(s) + E[h_1^e(s - D_1) - h_1(s - D_1)] = c_1(s) - h_2E(s - D_1)$. Suppose the result holds for j, then $C_{j+1}(s) = E[h_{j+1}^e(s - D_{j+1}) + C_j(\min(s_j^*, s - D_{j+1})) = E[h_{j+1}^e(s - D_{j+1}) + c_j(\min(s_j^*, s - D_{j+1})) - h_{j+1}E(\min(s_j^*, s - D_{j+1}) - D_j)] \pm h_{j+1}E(s - s_j^* - D_{j+1})^+ = c_{j+1}(s) - h_{j+2}E(s - D_{j+1})$. Part 2 follows directly from the fact that $h_{J+1} \equiv 0$. Part 3 follows directly from Part 1 and the definition of s_j^*. □

Remark: Our result also allows us to interpret $C_j(s)$ as $c_j(s)$ plus the inventory in transit to Stage j minus an echelon inventory credit at rate h_{j+1}. In our opinion, the interpretation of $c_j(s)$ is easier to understand.

2 Algorithm with Gradient Updates

An algorithm strictly based on Equations (5), (6) and (7) requires computing $c_j(x; s)$ for each $x \in \{0, \ldots, s\}$ and a minimization over this set to compute $c_j(s)$. Evaluating $c_j(x; s)$ for each x takes computational work proportional to $\max(s - x, x)$. Therefore, evaluating $c_j(x; s)$ for all $x \in \{0, \ldots, s\}$ requires work proportional to s^2. The minimizing over $x \in \{0, \ldots, s\}$ does not add to the complexity because the order of complexity for searching the minimizer is $s \ln(s)$. In summary, computational work to evaluate $c_j(s)$ is proportional to s^2. Computing $c_j(0), \ldots, c_j(s_j^*)$ takes work proportional to $(s_j^*)^3$. Adding this work over all the stages gives us the computational complexity of the algorithm. The complexity for the traditional algorithm without the gradient updates is similar.

The next proposition establishes the link between $\Delta c_{j+1}(x)$ and $\Delta c_j(x)$ that requires considerably less work. Indeed, the work required to compute $\Delta c_j(s)$, via Equation (10), is proportional to s, so the work required to compute s_j^* is proportional to $(s_j^*)^2$. Adding the work over all the stages gives us the complexity of the algorithm with gradient updates.

Proposition 3 *For $j = 1, \ldots, J$, we have*

$$\Delta c_{j+1}(s) = h_{j+1} Pr(D_{j+1} \leq (s - s_j^*)^+) + \sum_{k=0}^{\min(s, s_j^* - 1)} \Delta c_j(k) Pr(D_{j+1} = s - k) - pPr(D_{j+1} > s). \quad (10)$$

Proof. Since $c_{j+1}(s) = E[h_{j+1}((s - s_j^*)^+ - D_{j+1})^+ + h_{j+1} ED_j + c_j(\min(s_j^*, s - D_{j+1}))]$, after some algebra we obtain:

$$\Delta c_{j+1}(s) = h_{j+1} Pr(D_{j+1} \leq s - s_j^*) + \sum_{k=(s+1-s_j^*)^+}^{\infty} \Delta c_j(s - k) Pr(D_{j+1} = k).$$

If $s < s_j^*$, the first term is zero and

$$\begin{aligned} \Delta c_{j+1}(s) &= E\Delta c_j(s - D_{j+1}) = \sum_{k=0}^{\infty} \Delta c_j(s - k) Pr(D_{j+1} = k) \\ &= \sum_{k=0}^{s} \Delta c_j(s - k) Pr(D_{j+1} = k) - pPr(D_{j+1} > s) \\ &= \sum_{k=0}^{s} \Delta c_j(k) Pr(D_{j+1} = s - k) - pPr(D_{j+1} > s), \end{aligned}$$

where the last two equations follow from $\Delta c_j(s) = -p$ for $s < 0$. The last equation is equivalent to (10) for $s < s_j^*$. Next we show the result for $s \geq s_j^*$. In this case,

$$\Delta c_{j+1}(s) = h_{j+1} Pr(D_{j+1} \leq s - s_j^*) + \sum_{k=s+1-s_j^*}^{\infty} \Delta c_j(s - k) Pr(D_{j+1} = k).$$

By noticing that $\Delta c_j(s) = -p$ for $s < 0$, we can rewrite the difference as

$$\Delta c_{j+1}(s) = h_{j+1} Pr(D_{j+1} \leq s - s_j^*) + \sum_{k=0}^{s_j^* - 1} \Delta c_j(k) Pr(D_{j+1} = s - k) - pPr(D_{j+1} > s).$$

This is equivalent to Equation (10) for $s \geq s_j^*$, concluding the proof. □

Next we describe an algorithm to obtain an optimal echelon base stock policy and the optimal expected cost.

$s_1^* \leftarrow \min\{y \in \mathcal{N} : \Delta c_1(y) > h_2\}$,

$c_J(0) \leftarrow \sum_{i=1}^{J} h_{i+1} ED_i + pED[1, J]$

FOR $j = 1$ to $J - 1$ DO

$\Delta c_{j+1}(s) \leftarrow h_{j+1} Pr(D_{j+1} \leq (s - s_j^*)^+) + \sum_{k=0}^{\min(s, s_j^* - 1)} \Delta c_j(k) Pr(D_{j+1} = s - k) - pPr(D_{j+1} > s)$.

$s_{j+1}^* \leftarrow \min\{y \in \mathcal{N} : \Delta c_{j+1}(y) > h_{j+2}\}$.

END

PRINT $(s_J^*, \ldots, s_1^*)$ and $c_J(s_J^*) = c_J(0) + \sum_{y=0}^{s_J^* - 1} \Delta c_J(y)$.

3 Newsvendor Bounds and Heuristics

The DP formulation enables the design of a fast algorithm based on gradient updates. Yet, both the new and the traditional formulation are difficult to explain to non-mathematically oriented students and practitioners and do not allow for sensitivity analysis. We now provide a heuristic that can be implemented in a spreadsheet by solving one newsvendor problem per stage.

Consider the sub-system $\{j+1,\ldots,1\}$, for some $j \in \{1,\ldots,J-1\}$ at the time we are allocating s between Stage $j+1$ and sub-system $\{j,\ldots,1\}$. What complicates the problem of minimizing $c_{j+1}(x;s)$ is the fact that the holding costs h_k, $k=j+1,\ldots,1$ are increasing. To see how this simplifies when the downstream costs are equal consider what happens when $h_{j+1} < H_j = h_j = h_{j-1} = \ldots = h_1$. We will use the notation $c_{j+1}(x;s|H_j)$ to denote the cost of allocating x units to Stage $j+1$ and $s-x$ units to sub-system $\{j,\ldots,1\}$ where the downstream holding costs are H_j. When it costs the same to hold stock at stages $j,\ldots,1$ it is optimal to hold stock only at Stage 1 since this affords better protection against backorders. This observation implies that we can collapse stages $\{j,\ldots,1\}$ into a single stage with demand $D[1,j]$ and holding cost H_j. This reduces the problem to a two stage problem. Indeed,

$$c_{j+1}(x;s|H_j) = h_{j+1}(x-D_{j+1})^+ + h_{j+1}ED_j + H_j\sum_{k=1}^{j-1}ED_k + EG_j(\min(s-x,s-D_{j+1})|H_j), \quad (11)$$

where $G_j(s|H_j) = H_jE(s-D[1,j])^+ + pE(D[1,j]-s)^+$. Consequently, the first difference is given by

$$\Delta c_{j+1}(x;s|H_j) = [h_{j+1} - \Delta G_j(s-x-1|H_j)]Pr(D_{j+1}\leq x). \quad (12)$$

Notice also that $\Delta G_j(s|H_j) \equiv G_j(s+1|H_j) - G_j(s|H_j) = (p+H_j)Pr(D[1,j]\leq s) - p$ and consequently,

$$s_j^*(H_j) \equiv \min\{s\in\mathcal{N} : (p+H_j)Pr(D[1,j]\leq s) > p+h_{j+1}\}. \quad (13)$$

In other words, $s_j^*(H_j)$ is the largest minimizer of a newsvendor problem with demand $D[1,j]$, holding cost $H_j - h_{j+1}$, backorder penalty cost $p+h_{j+1}$ and that the solution is independent of the demand D_{j+1}. Substituting $x = (s-s_j^*(H_j))^+$, we obtain:

$$\begin{aligned} c_{j+1}(s|H_j) &= h_{j+1}(s-s_j^*(H_j)-D_{j+1})^+ + h_{j+1}ED_j + H_j\sum_{k=1}^{j-1}ED_k \\ &+ EG_j(\min(s_j^*(H), s-D_{j+1}|H_j). \end{aligned} \quad (14)$$

Notice that the heuristic is intimately related to the dynamic programming formulation; it follows the same steps, but that of finding s_j^* is greatly simplified by the assumption of equal holding costs for the downstream sub-system. We can improve on the cost estimate in Equation (14) by using the actual pipeline cost $\sum_{k=1}^{j} h_{k+1}ED_k$ instead of the approximation $h_{j+1}ED_j + H_j\sum_{k=1}^{j-1}ED_k$.

Consider now the general case where $h_j < h_{j-1} < \ldots < h_1$. Shang and Song (2003) *increase* the holding costs of stages $j,\ldots,2$ to $H_j = h_1$. For this sub-system, the cost of allocating x units to

$j+1$ and s units to j is given by $c_{j+1}(x;s|h_1)$ as defined in (11) and the optimal echelon base-stock level for Stage j is given by $s_j^*(h_1)$. Similarly, they *decrease* the holding cost of stages $j-1,\ldots,1$ to $H_j = h_j$. The resulting sub-system has expected cost $c_{j+1}(x;s|h_j)$ and optimal echelon base-stock level $s_j^*(h_j)$. Shang and Song thus solve two newsvendor problems per stage to obtain $s_j^*(h_1)$ and $s_j^*(h_j)$, $j=1,\ldots,J$ and show that $s_j^*(h_1) \le s_j^* \le s_j^*(h_j)$. They develop a heuristic for the original system by either truncating or rounding the average $(s_j^*(h_1)+s_j^*(h_j))/2$.

We propose a new heuristic that consists of solving a *single* newsvendor problem per stage based on the approximate holding cost rate

$$H_j = h_j^{GO} \equiv \sum_{k=1}^{j} \frac{L_k}{L[1,j]} h_k.$$

The idea is based on adding the holding cost as the product goes through the stages without delay and then dividing by the total lead time that it spends before reaching the end customer. In this way, we obtain an approximate holding cost rate for each stage and solve the system $c_{j+1}(x;s|h_j^{G0})$ obtaining $s_j^*(h_j^{G0})$ for $j=1,\ldots,J$. We explored other weighting strategies as discussed in the last paragraph of the numerical study.

Proposition 4 *For any given j and s we have:*

1. $c_j(s|h_j) \le c_j(s|h_j^{GO}) \le c_j(s|h_1)$,
2. $s_j^*(h_1) \le s_j^*(h_j^{GO}) \le s_j^*(h_j)$,
3. $G_j(s|h_j^{GO}) \le \sqrt{ph_j^{GO}}\sqrt{\lambda L[1,j]E[X^2]}$.

Proof. Notice that $h_j \le h_j^{GO} \le h_1$. Part 1 follows immediately from this inequality. We also have $\Delta G_j(s|h_j) \ge \Delta G_j(s|h_j^{GO}) \ge \Delta G_j(s|h_1)$ since $h_j \le h_j^{GO} \le h_1$. This implies Part 2. Finally Part 3 is the distribution-free bound in Gallego and Moon [5] and Scarf [11].

The last proposition imply that if the bounds in Part 1 of Proposition 4 are tight then $s_j^*(h_j^{GO})$ would be a close to optimal heuristic. We will illustrate the accuracy of this heuristic in the following section. If our heuristic is close-to-optimal, the cost of managing the series system can also be bounded by the distribution-free upper bound

$$c_J(s_J^*) = c_{J+1}(s_J^*) \simeq c_{J+1}(s_J^*(h_J^{G0})|h_J^{G0}) \le \sqrt{p(h_1L_1+\ldots+h_JL_J)\lambda EX^2} + \sum_{i=1}^{J-1} h_{i+1}ED_i. \qquad (15)$$

This formula contains all of the parameters of the problem. A simple sensitivity analysis reveals that (1) the system cost is proportional to $\sqrt{p}$, (2) downstream leadtimes have a larger impact on system performance than upstream leadtimes, (3) upstream echelon holding cost rates have a larger impact on the system performance than downstream echelon holding cost rates, (4) the system cost is proportional to $\sqrt{\lambda}$ and proportional to $\sqrt{E[X^2]}$, recall that X is the random demand size. Note that it is possible to approximate the optimal expected cost for each stage by using the normal

approximation. This approximation would result in a formula of the form $c_j(s_j^*) \simeq (p + H_j)\sigma\phi(z)$, where $z = \Phi^{-1}((p + h_{j+1})/(p + H_j))$ for $j = 1, \ldots, J$ and σ is the standard deviation of $D[1, j]$.

This type of parametric analysis enables a near characterization of system performance. Some system design issues may require investments in new processing plans or quicker but more expensive shipment methods. Marketing strategies could influence the demand as well as changing the backlogging costs. The closed form expression (15) enables a first cut estimate of the cost impact of any changes in the parameters. Our analysis suggests, for example, that management should focus on reducing the lead time at the upstream stages while reducing the holding cost at the downstream stages. If process re-sequencing is an option, the lowest value added processes with the longest processing times should be carried out sooner rather than later. In addition, the bound can provide an estimate of the decrease in cost that results from advanced demand information as explained in [7] and increases in customer lead times as explained in [9]. The model can also be used to study the benefits of outsourcing by truncating the supply chain using contractual costs instead. In a recent study, Lutze and Özer [10] use similar closed form solutions to design a mutually beneficial contract, referred to as a promised lead time contract, that trades of the value of advanced demand information to the seller and the cost of customer lead time to the buyer in a two level multi period supply chain.

4 Numerical Study

Here, we report the performance of our heuristic and of the distribution-free bound. We compare the exact solution to the newsvendor heuristics and report the percentage error $\epsilon^i\% = \frac{c_J(s_J^i) - c_J(s_J^*)}{c_J(s_J^*)}$ for $i = \{SS, GO\}$. Shang and Song (2003) use $s_j^{SS} \equiv \frac{s_j^*(h_j) + s_j^*(h_1)}{2}$ and truncate this average when $p \leq 39$ and round it otherwise. We use $s_j^{GO} \equiv s_j^*(h_j^{GO})$. By considering a larger set of experiments, we complement the numerical study in Shang and Song (2002). In particular, our numerical study includes unequal leadtimes. To manage the series system, we use an echelon base stock policy with echelon base stock levels s_j^{GO} for all j. The approximate cost is given by $G_J(s_J^{GO}) + \sum_{i=1}^{J} h_{i+1} E D_i$. Shang and Song (2003) approximate the optimal cost by $G_J(s_J^*(h_J) + \sum_{i=1}^{J} h_{i+1} E D_i$ instead of the average since the lower bounds become looser as the number of stages in the system increases. We study two sets of experiments: constant leadtime set and the randomized parameters set.

The first set of experiments is similar to that of Gallego and Zipkin [8] and Shang and Song [13]. The holding cost and the lead times are normalized so $h_1 = 1$ and $L[1, J] = 1$. We consider $J \in \{2, 4, 8, 16, 32, 64\}$; $\lambda \in \{16, 64\}$; and $p \in \{9, 39\}$ (corresponding to fill rates of $90\%, 97.5\%$). Within this group we consider *linear* holding-cost form ($h_j^e = 1/J$); *affine* holding cost form ($h^e[1, j] = \alpha + (1 - \alpha)j/J$ with $\alpha = 0.25$ and 0.75); *kink* holding cost form ($h_j^e = (1 - \alpha)/J$ for $j \geq J/2 + 1$ and $h_j^e = (1+\alpha)/J$ for $j < J/2+1$ with $\alpha = 0.25$ and 0.75) and *jump* holding cost form ($h_j^e = \alpha+(1-\alpha)/J$ for $j = N/2$ and $h_j^e = (1 - \alpha)/J$ for $j \neq J/2$ with $\alpha = 0.25$ and 0.75). Notice that Shang and Song (2002) consider only the case for $\lambda = 64$ and $p = 39$. We report the results in Table 1.

Out of 108 problem instances, in 24 cases the s^{GO} and in 20 cases the s^{SS} heuristic resulted in

the same solution as the recursive optimization. The s^{GO} (resp., s^{SS}) heuristic outperforms in 48 (resp., 44) cases and they tie in 17 cases. The average error for s^{GO} (resp., s^{SS}) heuristic is 0.195% (resp., 0.385%), while the maximum error is 3.68% and 1.24% for the GO and the SS heuristics respectively. The quality of the heuristics seems to deteriorate as the number of stages in the system exceeds 32. The SS heuristic seems to perform better for the jump holding cost case, while the GO heuristic tends to dominate in the other cases.

The second set of experiments allow for unequal leadtimes. It is here that we expect the GO heuristic to perform better. To cover a wider range of problem instances we generate the leadtimes and holding costs from uniform distributions. In particular, we use the following set of parameters:

$$h_j^e \in \{\text{Unif}(0,1), \text{Unif}(0,5), \text{Unif}(1,10)\},$$

$$L_j \in \{\text{Unif}(1,2), \text{Unif}(1,10), \text{Unif}(1,40)\},$$

$$J \in \{2,4,8,16,32\} \quad b \in \{1,9,39,49\} \quad \lambda \in \{1,3,6\}.$$

We consider 25 combinations, taken at random, from the above parameters. For each subgroup we generate 40 problem instances and calculate the worst case as well as the average performances. We present some of the problem instances in Table 2.

Out of 1000 problem instances, in 188 cases the s^{GO} heuristic and in 133 cases the s^{SS} heuristic resulted in the exact solution. In 849 cases the error term for s^{GO} heuristic is smaller or equal to that of s^{SS} heuristic. The average error for the s^{GO} (resp., s^{SS}) heuristic is 0.23% (resp., 0.83%). We observe that as the variance of the leadtimes across stages increases the average error term for s^{GO} decreases (the average error for $L_j \sim \text{Unif}(1,10)$ is 0.14% whereas it is 0.39% for $L_j \sim \text{Unif}(1,2)$). Similarly the s^{GO} heuristic performs even better as the variance of echelon costs across stages in a series system increases. In the second half of Table 2 we present the ten worst cases that we encountered. The maximum worst case we observed was 3.62% for the s^{GO} heuristic and 4.36% for the s^{SS} heuristic.

In light of our numerical observations we suggest the s^{GO} heuristics for a series system with a small number of stages ($J \leq 32$) especially for systems with high variability in leadtimes and echelon holding costs across stages. More caution should be used for system with a large number of stages and for systems with jump holding costs.

We have also performed a numerical study comparing the actual cost to the distribution-free bound by performing simple linear regressions of the bound to the actual cost by fixing all but one of the parameters. The results of the regressions can be found in Tables 3 and 4 where we report the coefficients of determination R^2 for the different regressions. Notice that in all cases R^2 is close to 1. This observation suggests that the bound can safely be used to investigate the impact of process and design changes on the cost of managing a series system.

The simple newsvendor heuristic and the bound enable a manager to quantify the impact of re-sequencing a process. Consider, for example, a four stage series system where $h_1 = L[1,J] = 1$, $p = 1$ and $\lambda = 16$. We now compare two systems with different configurations of leadtimes. The first system has leadtimes $(0.1, 0.1, 0.1, 0.7)$ and the second has leadtimes $(0.7, 0.1, 0.1, 0.1)$. The

costs based on the distribution free bound (resp., recursive optimization) are 13.29 (resp., 12.77) for the first system and 5 (resp.,4.93) for the second system. The distribution free bound predicts a cost reduction of 62.4% while the actual cost reduction based on recursive optimization is 61.39%. This indicates that the distribution free bound enables a quick, yet accurate, what if analysis. In this case, we observe that postponing the shortest and the most expensive processes to a later stage can significantly reduce inventory related costs.

We explored using different weighting structures to approximate holding cost structure for the newsvendor heuristic. In particular, we used $\sum_{k=1}^{j} \frac{L_k^\alpha}{\sum_{i=1}^{j} L_i^\alpha} h_k$ for different $\alpha \in [0,1]$. We were unable to identify an α that consistently results in lower error terms than $\alpha = 1$. In addition, we also used the holding cost structure $H_j = \sum_{k=1}^{j-1} h_{k+1} \frac{L_k}{\sum_{i=1}^{j-1} L_i}$ for $j \geq 2$, and $H_1 = h_1$ that equates the pipeline holding cost under the approximation to the real pipeline cost. This holding cost structure did not perform better than the original proposed holding cost structure. To further investigate what other weighting strategies could work, we have also calculated the *implied* holding costs h_j^{im} for some problem instances. These holding cost when used in the newsvendor problem of Equation (13) yield the optimal echelon base stock levels s_j^* obtained through the exact algorithm. In other words we set $h_j^{\min} \equiv \min\{h \in \mathcal{R}_+ : s_j^*(h) = s_j^*\}$ and $h_j^{\max} \equiv \min\{h \in \mathcal{R}_+ : s_j^*(h) = s_j^* - 1\}$. Note that using an implied holding cost $h_j^{im} \in [h_j^{\min}, h_j^{\max})$ in Equation (13) yields the optimal echelon base stock level. In Table 5 we provide some examples. Due to the *robustness* of newsvendor cost the range for possible implied holding cost is large. The holding costs h_j^{GO} fall into this range for a large set of problem instances.

5 Conclusion

The results of this letter can also be applied to assembly systems by following the ideas in Rosling [12]. For distribution systems, the heuristic can be applied after using the decomposition principles in Gallego, Özer and Zipkin [6]. Our formulation can also be used to model whether or not it is beneficial to outsource an upstream portion of the supply chain by comparing the cost of the original supply chain and the cost of the truncated supply chain with the unit cost from the new source of supply.

References

[1] Axsater, S. 2003. Serial and Distribution Inventory Systems. Chapter 10 in Vol 11 Handbooks in ORMS, Supply Chain Manangement: Design, Coordination and Operation. Eds. A.G. de Kok and S.C. Graves. Elsevier, The Netherlands.

[2] Chen, F. and Y. Zheng. 1994. Lower Bounds for Multi-echelong Stochastic Inventory Systems. *Management Science* 40, 1426-1443.

[3] Clark, A. and H. Scarf. 1960. Optimal Policies for a Multi Echelon Inventory Problem. *Management Science* 6 475-490.

[4] Federgruen, A. and P. Zipkin. 1984. Computational Issues in an Infinite Horizon, Multiechelon Inventory Model. *Operations Research.* 32, 818-836.

[5] Gallego G. and I. Moon. 1993. The Distribution Free Newsboy Problem: Review and Extensions. *Journal of Operational Research Society.* 44, 825-834.

[6] Gallego G, Ö. Özer, and P. Zipkin. 1999. Bounds, Heuristics and Approximations for Distribution Systems. Working Paper.

[7] Gallego, G. and Ö. Özer. 2003. Optimal Replenishment Policies for Multi-Echelon Inventory Problems under Advance Demand Information. *M&SOM*, **5**, 157-175.

[8] Gallego G. and P. Zipkin. 1999. Stock Positioning and Performance Estimation in Serial Production-Transportation Systems. *M&SOM* 1, 77-87.

[9] Hariharan, R. and P. Zipkin. 1995. Customer Order Information, Lead Times, and Inventories. *Management Science.* 41, 1599-1607.

[10] Lutze, H. and Ö. Özer. 2004. Promised Leadtime Contracts and Renegotiation Incentives Under Asymmetric Information. Working Paper. Stanford University.

[11] Scarf, H. 1958. A Min-Max Solution of an Inventory Problem. *Ch. 12 in Studies in The Mathematical Theory of Inventory and Production*, Stanford Univ. Press.

[12] Rosling, K. 1989. Optimal Inventory Policies for Assembly Systems Under Random Demands. *Operations Research* 37, 565-579.

[13] Shang H. K. and J. Song. 2003. Newsvendor Bounds and Heuristics for Optimal Policies in Serial Supply Chains. *Management Science* 49, 618-638.

[14] Zipkin P. 2000. Foundations of Inventory Management. *The Irwin McGraw Hill Series.*

Table 1: Comparison of Optimal and Heuristic Policy

N	Form	$c(s_J^{SS})$	$c(s_J^{GO})$	$c(s_J^*)$	$\epsilon^{SS}\%$	$\epsilon^{GO}\%$	Form	$c(s_J^{SS})$	$c(s_J^{GO})$	$c(s_J^*)$	$\epsilon^{SS}\%$	$\epsilon^{GO}\%$
$(\lambda = 16, p = 39)$												
64	Kink	18.066	17.957	17.775	1.637	1.024	Kink	20.839	20.471	20.381	2.247	0.442
32	$\alpha = 0.25$	17.860	17.793	17.668	1.087	0.707	$\alpha = 0.75$	20.543	20.326	20.266	1.367	0.296
16		17.557	17.527	17.461	0.550	0.378		20.162	20.040	20.034	0.639	0.030
8		17.099	17.075	17.063	0.211	0.070		19.675	19.608	19.604	0.362	0.020
4		16.246	16.244	16.244	0.012	0.000		18.759	18.726	18.726	0.176	0.000
2		14.617	14.617	14.617	0.000	0.000		17.063	16.999	16.999	0.376	0.000
64	Affine	19.119	19.074	18.955	0.865	0.628	Affine	23.699	23.722	23.656	0.182	0.279
32	$\alpha = 0.25$	18.957	18.897	18.814	0.760	0.441	$\alpha = 0.75$	23.481	23.495	23.440	0.175	0.235
16		18.532	18.579	18.525	0.038	0.291		23.053	23.066	23.012	0.178	0.235
8		17.985	18.000	17.975	0.056	0.139		22.202	22.163	22.159	0.194	0.018
4		16.843	16.843	16.843	0.000	0.000		20.494	20.442	20.440	0.264	0.010
2		14.617	14.617	14.617	0.000	0.000		17.063	16.999	16.999	0.376	0.000
64	Jump	16.536	16.598	16.280	1.572	1.953	Linear	16.621	16.612	16.409	1.292	1.237
32	$\alpha = 0.25$	16.301	16.375	16.142	0.985	1.443		16.428	16.427	16.296	0.810	0.804
16		15.977	16.137	15.861	0.731	1.740		16.174	16.238	16.094	0.497	0.895
8		15.339	15.478	15.317	0.144	1.051		15.704	15.759	15.703	0.006	0.357
4		14.204	14.306	14.204	0.000	0.718		14.956	14.975	14.954	0.013	0.140
2		12.011	12.011	12.011	0.000	0.000		13.314	13.314	13.314	0.000	0.000
$(\lambda = 16, p = 9)$												
64	Kink	15.863	15.614	15.565	1.915	0.315	Kink	18.390	17.953	17.938	2.520	0.084
32	$\alpha = 0.25$	15.602	15.491	15.450	0.984	0.265	$\alpha = 0.75$	18.093	17.819	17.816	1.555	0.017
16		15.265	15.248	15.239	0.171	0.059		17.710	17.573	17.572	0.785	0.006
8		14.805	14.776	14.776	0.196	0.000		17.108	17.108	17.108	0.000	0.000
4		13.922	13.891	13.891	0.223	0.000		16.205	16.205	16.205	0.000	0.000
2		12.166	12.138	12.138	0.231	0.000		14.332	14.332	14.332	0.000	0.000
64	Affine	16.705	16.713	16.676	0.174	0.222	Affine	21.053	21.123	21.047	0.029	0.361
32	$\alpha = 0.25$	16.540	16.553	16.529	0.067	0.145	$\alpha = 0.75$	20.830	20.902	20.830	0.000	0.346
16		16.250	16.241	16.231	0.117	0.062		20.402	20.400	20.398	0.020	0.010
8		15.655	15.649	15.643	0.077	0.038		19.533	19.533	19.533	0.000	0.000
4		14.517	14.489	14.483	0.235	0.041		17.804	17.804	17.804	0.000	0.000
2		12.166	12.138	12.138	0.231	0.000		14.332	14.332	14.332	0.000	0.000
64	Jump	14.358	14.336	14.196	1.141	0.986	Linear	14.506	14.379	14.317	1.320	0.433
32	$\alpha = 0.25$	14.115	14.145	14.051	0.455	0.669		14.304	14.231	14.201	0.725	0.211
16		13.806	13.840	13.755	0.371	0.618		14.021	14.007	13.980	0.293	0.193
8		13.187	13.228	13.178	0.068	0.379		13.574	13.580	13.558	0.118	0.162
4		12.031	12.061	12.031	0.000	0.249		12.688	12.688	12.688	0.000	0.000
2		9.710	9.710	9.710	0.000	0.000		10.924	10.924	10.924	0.000	0.000
$(\lambda = 64, p = 39)$												
64	Kink	52.742	52.667	52.352	0.745	0.602	Kink	62.378	61.848	61.622	1.227	0.367
32	alpha=0.25	52.086	52.130	51.903	0.353	0.437	alpha=0.75	61.610	61.313	61.157	0.741	0.255
16		51.099	51.140	51.011	0.173	0.253		60.551	60.327	60.226	0.540	0.168
8		49.259	49.259	49.223	0.073	0.073		58.484	58.425	58.381	0.176	0.075
4		45.660	45.721	45.660	0.000	0.134		54.773	54.675	54.675	0.179	0.000
2		38.457	38.457	38.457	0.000	0.000		47.314	47.267	47.267	0.099	0.000
64	Affine	56.877	56.959	56.707	0.300	0.444	Affine	74.105	74.165	74.085	0.027	0.108
32	alpha=0.25	56.140	56.300	56.120	0.036	0.321	alpha=0.75	73.240	73.279	73.222	0.025	0.078
16		54.966	55.053	54.954	0.022	0.180		71.514	71.540	71.495	0.027	0.063
8		52.615	52.684	52.609	0.011	0.143		68.065	68.085	68.042	0.034	0.063
4		47.931	47.925	47.921	0.021	0.008		61.160	61.128	61.128	0.052	0.000
2		38.475	38.457	38.457	0.047	0.000		47.314	47.267	47.267	0.099	0.000
64	Jump	46.107	47.742	46.075	0.069	3.618	Linear	47.859	48.045	47.590	0.565	0.956
32	alpha=0.75	45.231	46.640	45.217	0.031	3.147		47.289	47.518	47.151	0.293	0.778
16		43.518	44.647	43.500	0.041	2.637		46.335	46.487	46.265	0.151	0.480
8		40.080	40.892	40.069	0.027	2.054		44.555	44.658	44.529	0.058	0.290
4		33.204	33.864	33.188	0.048	2.037		41.015	41.034	41.015	0.000	0.046
2		19.389	19.655	19.370	0.098	1.471		33.916	33.916	33.916	0.000	0.000

Note that this table includes all the experiments in Shang and Song (2002).

Table 2: Problem instances for $J = 4$.

(L_4, L_3, L_2, L_1)	$(h_4^e, h_3^e, h_2^e, h_1^e)$	s^{SS}	$c_4(s_4^{SS})$	s^{GO}	$c_4(s_4^{GO})$	$\mathbf{s}$	$c_4(s_4^*)$	$\epsilon^{SS}\%$	$\epsilon^{GO}\%$
$p = 49, \lambda = 1$									
(1.698,1.067,1.274,1.676)	(9.928,2.889,4.290,1.521)	(8,7,6,5)	113.143	(7,7,6,5)	110.650	(7,7,5,5)	110.606	2.29%	0.04%
(1.241,1.200,1.442,1.939)	(5.818,7.261,2.996,6.118)	(8,7,6,4)	119.258	(7,6,6,4)	117.140	(7,6,6,4)	117.140	1.81%	0%
(1.412,1.749,1.071,1.798)	(9.804,5.981,4.367,1.712)	(8,8,6,5)	135.000	(8,7,5,5)	134.123	(7,7,5,5)	132.365	2.04%	1.33%
(1.545,1.662,1.503,1.700)	(5.982,4.047,4.553,2.421)	(9,8,6,4)	105.408	(9,7,6,4)	105.094	(8,7,6,4)	103.954	1.40%	1.10%
(1.077,1.186,1.291,1.082)	(6.992,7.452,6.848,6.558)	(6,6,4,3)	112.003	(6,5,4,3)	110.826	(6,5,4,3)	110.826	1.06%	0%
(1.840,1.019,1.772,1.663)	(9.896,2.546,2.596,8.907)	(8,7,6,3)	120.961	(8,7,6,3)	120.961	(8,7,6,3)	120.961	0%	0%
(1.969,1.575,1.250,1.872)	(2.681,8.524,2.728,5.110)	(10,7,6,4)	101.835	(9,7,6,4)	100.681	(9,6,6,4)	99.852	1.99%	0.83%
(1.434,1.382,1.818,1.403)	(3.391,5.041,2.926,6.195)	(9,7,6,3)	87.307	(8,7,6,3)	86.457	(8,7,6,3)	86.457	0.98%	0%
(1.204,1.600,1.800,1.885)	(5.879,4.950,8.401,9.215)	(9,8,6,4)	141.850	(8,7,5,4)	136.538	(8,7,5,4)	136.530	3.89%	0%
(1.032,1.664,1.813,1.687)	(3.185,2.163,1.310,5.019)	(9,9,7,4)	65.950	(9,8,7,4)	65.937	(9,8,7,4)	65.937	0.02%	0%
Some of the observed worst cases for both heuristics									
$p = 49, \lambda = 1$									
(1.222,1.765,1.938,1.732)	(7.719,9.942,4.913,9.118)	(9,8,6,3)	172.428	(8,7,6,3)	165.232	(8,7,6,3)	165.232	4.355%	0%
(1.862,1.057,1.462,1.842)	(7.542,1.643,3.786,5.147)	(9,8,6,4)	102.465	(8,7,6,4)	99.765	(8,7,6,4)	99.765	2.707%	0%
(1.561,1.254,1.644,1.979)	(3.155,8.428,9.658,4.546)	(9,7,6,4)	135.289	(8,6,5,4)	131.429	(8,6,5,4)	131.429	2.937%	0%
(1.599,1.422,1.013,1.103)	(6.410,3.175,2.005,1.374)	(8,7,5,4)	75.141	(7,6,5,4)	72.836	(7,6,5,4)	72.836	3.165%	0%
(1.159,1.583,1.929,1.404)	(8.716,1.984,3.435,3.370)	(9,8,6,4)	112.422	(8,8,6,4)	108.227	(8,8,6,4)	108.227	3.876%	0%
(1.412,1.749,1.071,1.798)	(9.804,5.981,4.367,1.712)	(8,8,6,5)	135.06	(8,7,5,5)	134.123	(7,7,5,5)	132.365	2.04%	1.33%
$p = 1, \lambda = 1$									
(1.949,1.921,1.550,1.346)	(0.472,0.375,0.847,0.317)	(7,6,3,3)	7.328	(6,5,3,3)	7.126	(6,5,3,3)	7.126	2.823%	0%
(1.009,1.919,1.276,1.273)	(0.588,0.691,0.838,0.726)	(5,4,3,2)	8.312	(5,4,3,2)	8.312	(4,4,3,2)	8.138	2.133%	2.133%
(1.693,1.303,1.427,1.070)	(0.967,0.683,0.153,0.877)	(4,4,4,2)	8.177	(5,4,4,2)	8.352	(4,4,4,2)	8.177	0%	2.137%
(1.545,1.448,1.409,1.299)	(0.466,0.501,0.153,0.323)	(6,5,4,3)	5.767	(6,5,5,3)	5.766	(5,5,5,3)	5.662	1.866%	1.845%

Table 3: Performance of the Distribution Free Bound for factors N, b, and λ.

$p = 10$	$\lambda = 16$	$R^2 =$	$J = 4$	$\lambda = 16$	$R^2 =$	$J = 4$	$p = 10$	$R^2 =$
$h_i^e = 0.25$	$L_i = 0.25$	99.99%	$h_i^e = 0.25$	$L_i = 0.25$	96.63%	$h_i^e = 0.25$	$L_i = 0.25$	99.97%
J	DFB	$c(s_J^*)$	p	DFB	$c(s_4^*)$	λ	DFB	$c(s_4^*)$
2	6.48	4.03	1	9.16	9.05	2	4.29	3.43
3	10.75	7.87	2	10.47	10.17	4	6.50	5.14
4	16.00	12.87	4	12.33	11.34	6	8.37	6.62
5	22.25	18.95	6	13.75	12.02	8	10.07	7.98
6	29.49	26.10	8	14.94	12.48	10	11.66	9.27
7	37.73	34.32	10	16.00	12.87	12	13.16	10.50
8	46.97	43.59	12	16.95	13.15	14	14.60	11.69
10	68.45	65.25	14	17.83	13.38	16	16.00	12.87
12	93.93	90.98	16	18.65	13.59	20	18.68	15.11
14	123.40	120.77	18	19.42	13.77	30	24.94	20.47
16	156.87	154.60	20	20.14	13.94	32	26.14	21.50

Table 4: Performance of Distribution Free Bound wrt factors h_4, h_1, L_4, and L_1 When $J = 4$

$p = 10$	$\lambda = 16$	$R^2 =$	$p = 10$	$\lambda = 16$	$R^2 =$	$p = 10$	$\lambda = 16$	$R^2 =$	$p = 10$	$\lambda = 16$	$R^2 =$
$h_i^e = 0.25$	$L_i = 0.25$	99.99%	$h_i^e = 0.25$	$L_i = 0.25$	99.07%	$h_i^e = 0.25$	$L_i = 0.25$	99.93%	$h_i^e = 0.25$	$L_i = 0.25$	99.99%
h_4^e	DFB	$c(s_4^*)$	h_1^e	DFB	$c(s_4^*)$	L_4	DFB	$c(s_4^*)$	L_1	DFB	$c(s_4^*)$
1	29.83	26.07	1	17.40	15.12	1	17.40	13.81	1	29.83	24.59
2	46.49	42.20	2	19.04	17.35	2	19.04	14.68	2	46.49	39.18
3	62.24	57.61	3	20.49	19.25	3	20.49	15.36	3	62.23	53.23
4	77.46	72.43	4	21.81	20.55	4	21.81	15.95	4	77.45	67.00
5	92.33	86.88	5	23.03	21.83	5	23.03	16.49	5	92.33	80.60
6	106.94	101.04	6	24.16	23.10	6	24.17	16.96	6	106.93	94.06
7	121.35	115.17	7	25.23	24.02	7	25.24	17.41	7	121.35	107.4
8	135.61	128.75	8	26.25	24.74	8	26.25	17.82	8	135.61	120.66
9	149.73	142.34	9	27.22	25.45	9	27.21	18.22	9	149.72	133.85
10	163.74	155.93	10	28.14	26.16	10	28.14	18.59	10	163.74	146.98

Table 5: Implied Holding Costs When $J = 4$

(L_4, L_3, L_2, L_1)	(h_4, h_3, h_2, h_1)	$(h_4^{GO}, h_3^{GO}, h_2^{GO}, h_1^{GO})$	$(h_4^{min}, h_3^{min}, h_2^{min}, h_1^{min})$	$(h_4^{max}, h_3^{max}, h_2^{max}, h_1^{max})$
$p = 9, \lambda = 1$				
(7.279, 1.606, 3.467, 7.087)	(9.928, 12.817, 17.107, 18.628)	(14.619, 17.427, 18.128, 18.628)	(12.154, 15.991, 16.761, 17.909)	(17.495, 19.769, 19.586, 18.805)
(7.570, 6.158, 8.306, 3.782)	(2.134, 6.763, 16.100, 18.691)	(10.157, 13.486, 16.911, 18.691)	(8.372, 13.603, 15.611, 18.347)	(11.608, 18.867, 20.898, 21.495)
(3.265, 8.386, 5.579, 3.418)	(9.892, 19.141, 21.237, 24.797)	(19.181, 20.926, 22.589, 24.797)	(19.271, 19.270, 21.651, 22.701)	(27.310, 23.834, 23.768, 25.026)
(6.126, 3.384, 2.003, 3.404)	(7.601, 17.398, 22.789, 25.723)	(15.998, 21.850, 24.636, 25.723)	(12.598, 17.641, 23.521, 24.258)	(19.031, 24.803, 29.712, 26.604)
(2.327, 9.546, 2.274, 9.146)	(7.236, 10.963, 15.802, 17.435)	(13.604, 14.311, 17.110, 17.435)	(12.797, 13.299, 14.888, 17.208)	(17.858, 16.072, 17.567, 18.468)
$p = 39, \lambda = 1$				
(2.327, 9.546, 2.274, 9.146)	(3.468, 4.990, 7.129, 7.490)	(6.028, 6.313, 7.418, 7.490)	(5.728, 5.851, 6.839, 7.338)	(7.956, 6.896, 8.024, 7.543)
(1.210, 7.247, 3.146, 3.558)	(2.820, 5.927, 10.124, 11.574)	(7.875, 8.314, 10.894, 11.574)	(7.314, 8.531, 10.467, 11.560)	(11.030, 11.532, 14.224, 13.502)