Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

A COMPARISON OF SAMPLE-PATH-BASED SIMULATION-OPTIMIZATION AND STOCHASTIC DECOMPOSITION FOR MULTI-LOCATION TRANSSHIPMENT PROBLEMS

Lei Zhao

Department of Industrial Engineering
Tsinghua University
Beijing, 100084, CHINA

Suvrajeet Sen

MORE Institute
Department of Systems & Industrial Engineering
University of Arizona
Tucson, AZ 85721, U.S.A.

ABSTRACT

Because of its applicability, as well as its generality, research in the area of simulation-optimization continues to attract significant attention. These methods, most of which rely on the statistically motivated search techniques, are at their best when very little is known about the structure of the function (e.g., function evaluations are treated as "black-box" function-calls). In some applications such as the one discussed in this paper, objective function values may be obtained through linear/network flow optimization models. In such cases, the objective function may be convex, and in such circumstances, very large instances can be solved using stochastic programming techniques. This paper presents a computational case for using such techniques, whenever applicable.

1 INTRODUCTION

The growth of computing technology has fueled continued growth in simulation modeling, and as a result, computationally intense activities such as simulation-optimization has grown significantly in the past decade (e.g., Alrefaei and Andradottir 1997, Pichitlamken and Nelson 2002). Ordinarily, simulation-optimization methodology is developed for applications in which the functions being optimized are performance measures that result from the output of a simulation model. As a result, it is customary to impose very few restrictions on the objective function being optimized. However, this generality comes at a price: the computational demands on algorithms which optimize very general performance measures are significant. Moreover, without any assumptions on the structure of the function, it becomes difficult to ascertain whether the output of a general-purpose optimizer is indeed optimum (up to a pre-specified tolerance). For some applications (e.g., multi-location transshipment problems), the models possess important properties (e.g., convexity). The question that we seek to answer is whether there are advantages to combining ideas from linear programming with those from simulation-optimization.

It so happens, that the area of stochastic programming (SP) has benefitted greatly from ideas that originally appeared in the simulation-optimization literature. For instance, stochastic quasi-gradient methods (SQG) for SP (Ermoliev 1983), importance sampling in SP (Dantzig and Glynn 1990), common random numbers in stochastic decomposition (SD) for SP (Higle and Sen 1991) and others originated in the simulation literature. Indeed the SQG algorithm, which extends the scope of applicability of stochastic approximation (SA) (Robbins and Monro 1951) to constrained non-differentiable optimization, is clearly familiar in the field of simulation-optimization. Since SQG is a generalization of SA, its application to SP is natural. Our seemingly obvious question (whether there is any benefit to using specialized schemes for SP) appears to have gone unanswered, even though the literature reports on the solution of SP problems using both specialized and non-specialized methods. In other words, specialized SP algorithms (such as SD) have not been compared with general-purpose methods to ascertain whether the former provide any clear advantages.

One of the prerequisites for this study was the existence of a class of SP instances that were studied using simulation-optimization techniques. We were able to locate such instances in a recent paper on multi-location transshipment models in Herer et al. (2006). For the sake of completeness, this model is summarized in the following section. The IPA algorithm used in the above paper, and its equivalence to SQG are also summarized in that section. Following this, we provide a brief description of the SD algorithm. Finally, the comparative computations are reported in Section 4, with our conclusions in Section 5.

2 MULTI-LOCATION TRANSSHIPMENT MODELS

This section is devoted mainly to ideas presented in the paper by Herer et al. (2006). The system described by

the authors arises when retailers and suppliers cooperate to minimize collective long-run average costs. In particular, consider one supplier, and N non-identical retailers who face uncertain customer demands. Each retailer reviews its own inventory periodically, and replenishes its stock by placing orders with the supplier. The following sequence of events describes the manner in which events unfold in any given period.

0. Assume that initial inventory level is known.
1. Replenishments arrive (based on orders placed in the previous period).
2. Backlogged orders are met, and then inventory level of each retailer is raised by the difference between replenishment quantity and backlogged orders.
3. The demand (a random variable) is observed at each retailer.
4. Once the demand is observed, the system redistributes its inventory by solving a reallocation problem in which inventory held at one retailer can be transshipped to another, at a cost.
5. Based on the inventory reallocation of step 4, each retailer meets as much demand as possible, and then the inventory, and backlogging quantities for each retailer are updated. In addition, orders are placed with the supplier.

Under certain assumptions (stationarity and non-shortage inducing policies), Herer et al. (2006) show that for each retailer there exists an optimal order-up-to policy s_i, $i = 1,...,N$, where N denotes the number of retailers. This result forms the basis of their "LP-based" formulation which finds the optimal decisions for any period. We summarize this formulation here for the sake of completeness. Given a vector of order-up-to quantities $S = (s_1,\ldots,s_N)$ and a demand realization $D = (d_1,\ldots,d_N)$, the LP they formulate is based on the following notation.

Decision Variables

- $e_i =$ ending inventory held at retailer i.
- $f_i =$ stock at retailer i used to satisfy demand at retailer i.
- $q_i =$ inventory at retailer i increased through replenishment.
- $r_i =$ amount of shortage met after replenishment at retailer i.
- $t_{ij} =$ stock at retailer i used to meet demand at retailer j, using the transshipment option.

Because Herer et al. (2006) use a network to describe the relationships between the variables, they use a notation that is more convenient in the network setting. Since our intent is to simply present their formulation, we adopt a slightly abbreviated, although equivalent notation. Accordingly, constraints (3)-(6) of Herer et al. (2006) translate to the following:

$$f_i + \sum_{j \neq i} t_{ij} + e_i = s_i, \quad \forall i \tag{1a}$$

$$f_i + \sum_{j \neq i} t_{ji} + r_i = d_i, \quad \forall i \tag{1b}$$

$$\sum_i r_i + \sum_i q_i = \sum_i d_i \tag{1c}$$

$$e_i + q_i = s_i, \quad \forall i \tag{1d}$$

$$e_i, f_i, q_i, r_i, s_i, t_{ij} \geq 0, \quad \forall i, j.$$

The vector S will be treated as a first-stage (here-and-now) decision, whereas, the remaining variables are determined in the second-stage, once an outcome of the vector (of random variables) D has been observed. The data used to define the objective is represented as follows.

Objective Function Data

- $h_i =$ unit cost of holding inventory at retailer i.
- $c_{ij} =$ unit cost of transshipment from retailer i to j.
- $p_i =$ penalty cost for shortage at retailer i.

Given S and D, the minimum cost transshipment is obtained by optimizing the following objective, subject to (1a) - (1d):

$$h(S,D) = \min \sum_i h_i e_i + \sum_{i \neq j} c_{ij} t_{ij} + \sum_i p_i r_i.$$

Together, the objective function and the constraints will be referred to as problem (1). In order to obtain a policy which minimizes average long-run costs, the vector S should be chosen so as to solve the following

$$\min_{S \geq 0} E[h(S,\tilde{D})], \tag{2}$$

where $\tilde{D}$ denotes the vector of random demand variables.

Herer et al. (2006) suggest that it may be difficult to solve such an optimization problem using "analytical methods," and instead suggest that a sample-path optimization method such as IPA may be used to compute the optimal order-up-to levels S. We should note that while IPA provides valid gradient estimators for smooth objective functions, problems of the form (1) are known to be

non-smooth (see Sen 1993), even in cases where each demand random variable has a continuous distribution. Nevertheless, non-smoothness of the objective function does not render their approach untenable because the stochastic quasi-gradient (SQG) method accommodates a non-smooth objective function (see Ermoliev 1983 for a survey), and therefore provides the appropriate algorithmic framework. It is important to note that for two-stage stochastic programs such as (2), the IPA method presented in Herer et al. (2006) reduces to the SQG method. Accordingly, we present the SQG method below and then establish the equivalence between the SQG method and the IPA method as in Herer et al. (2006).

2.1 The Stochastic Quasi-Gradient (SQG) Method

Consider the optimization problem stated in (2). Because (1) is a linear program, the value function $h(S,D)$ is a piecewise linear convex function, and as a result, the objective function in (2) is also convex.

At each iteration, the SQG method uses an estimate of a subgradient of $E[h(S,\tilde{D})]$ by using Monte Carlo sampling. For a given policy S^k, the idea is to first draw M i.i.d. samples of the random vector $\tilde{D}$ (denoted $D^1,\ldots,D^M$, say), and use these realizations to solve linear programs defined in (1), with $S = S^k$. Upon evaluating $h(S^k,D^m)$, $m = 1,\ldots,M$, we obtain dual vectors corresponding to each LP. That is, for each pair (S^k,D^m), let $B_i^{k,m}$ and $E_i^{k,m}$ denote the optimal dual multipliers associated with constraints (1a) and (1d), respectively. The vector obtained by concatenating the multipliers for all retailers will be denoted $B^{k,m}$ and $E^{k,m}$. Then, an unbiased estimate of a subgradient of $E[h(S,\tilde{D})]$, denoted $\hat{\xi}^k$, is given by

$$\hat{\xi}^k = \frac{1}{M}\sum_{m=1}^{M}(B^{k,m}+E^{k,m}). \tag{3}$$

The SQG algorithm then generates a sequence of points according to the following iterative scheme:

$$S^{k+1} = P_+(S^k - \alpha_k\hat{\xi}^k), \tag{4}$$

where $P_+(\cdot)$ denotes the projection (or positive part) of the argument, and α_k are nonnegative scalars that satisfy

$$\alpha_k \to 0, \quad \sum_k \alpha_k \to \infty, \quad \text{and} \quad \sum_k \alpha_k^2 < \infty. \tag{5}$$

2.2 The Equivalence Between SQG and IPA for the Transshipment Problem

Let B_i, M_i, R, and E_i denote the dual multipliers of (1a) - (1d). The dual formulation of (1) is

$$\begin{aligned}
\text{maximize} \quad & \sum_i s_iB_i + \sum_i d_iM_i + \sum_i d_iR + \sum_i s_iE_i \\
\text{subject to} \quad & B_i + E_i \le h_i, \quad \forall i, && \text{(6a)}\\
& B_i + M_i \le 0, \quad \forall i, && \text{(6b)}\\
& B_i + M_j \le c_{ij}, \quad \forall i,j, i \ne j, && \text{(6c)}\\
& M_i + R \le p_i, \quad \forall i, && \text{(6d)}\\
& R + E_i \le 0, \quad \forall i. && \text{(6e)}
\end{aligned}$$

Let y_i denote the slack variable of (6a). The KKT complimentary slackness condition requires that

$$y_ie_i = 0, \quad \forall i. \tag{7}$$

As in Herer et al. (2006), let TC denote the total cost of (2) and dTC denote the "derivatives" of the total cost TC with respect to S. Mathematically, the function TC may not be differentiable, but since we are mainly interested in the connection between SQG and IPA, we follow the arguments provided by Herer et al. (2006). In the IPA method, the derivative $d\text{TC}_i$ is calculated as the difference between the holding cost at retailer i, h_i, and the reduced cost of e_i, which is the slackness of its dual constraint (6a), y_i. Thus we have

$$d\text{TC}_i = h_i - y_i = B_i + E_i, \quad \forall i. \tag{8}$$

Strictly speaking, the "derivative" above should be replaced by a subgradient.

When the inventory at retailer i is positive (i.e., $e_i > 0$), (6a) is binding due to (7). That is, $y_i = 0$. From (8), we have $d\text{TC}_i = h_i$. When e_i is zero, $d\text{TC}_i = B_i + E_i = h_i - y_i$.

By comparing the derivative calculation in (8) and the steps iii - vi (in Algorithm 1 of Herer et al. 2006) of the IPA method with (3) and (4) of the SQG method described above, we conclude that the IPA method in Herer et al. (2006) is equivalent to the SQG method we described earlier.

It can be shown that this simple algorithm (SQG) produces a sequence of iterates of S that converge in probability to an optimal solution of (2). This method traces its roots to the method of Robbins and Monro (1951), although several new advances such as adaptive step sizes, primal-dual extensions, etc. (see Sen and Sherali 1985) have been proposed. However, methods of this type suffer from a serious handicap: it is usually difficult to ascertain the quality of a solution during the course of the algorithm, and as a result, stopping criteria are often based on ad hoc rules.

3 STOCHASTIC DECOMPOSITION

From the statement in (1a) - (1d), it is clear that the model presented in (2) is a two-stage stochastic linear program (SLP) in which the first stage seeks an optimal order-up-to quantity, and the second stage models the cost of distribution and inventory. Indeed, it is easy to map the multi-location transshipment problem into the following form (a two-stage SLP with recourse) by mapping the variables $S \rightarrow x$ and $D \rightarrow \omega$.

$$\min_{x \in \Re^n} \quad f(x) := c^\top x + E[h(x, \tilde{\omega})] \tag{9a}$$

$$\text{s.t.} \quad Ax \ \leq \ b, \tag{9b}$$

where, A is $m_1 \times n_1$, b is $m_1 \times 1$, and

$$h(x, \omega) = \min_{y \in \Re^n} \quad g^\top y \tag{10a}$$

$$\text{s.t.} \quad Wy = r(\omega) - Tx, \tag{10b}$$

$$y \geq 0 \tag{10c}$$

Although two-stage SLP problems ordinarily allow randomness to influence several data elements, we have restricted (10) to a version that only includes randomness in the right-hand-side r. In order to help the reader map (2) and (1a) - (1d) to the above problem, recall that $S \rightarrow x$ implies $A = -I$, and $b = 0$. Similarly, the vector consisting of variables (f, t, e, r, q) in (1a) - (1d) are denoted by the vector y above, with the constraints of (1a) - (1d) being collected in the matrix W. Similarly, the right-hand-side $r(\omega)$ consists of elements of the random varibales d_i for rows (1b) and (1c), and 0 for the other rows of $r(\omega)$. By the same token, the matrix T is obtained by assigning values such that the vector Tx yields the right-hand-side of (1a) and (1d), with 0 for (1b) and (1c).

The Stochastic Decomposition (SD) algorithm (Higle and Sen 1991, Higle and Sen 1999) was designed to solve optimization problems of the form specified in (9) - (10). In essence, it is an extension of Benders' decomposition (Benders 1962), which has come to be known as the L-shaped method in the SP literature (Van Slyke and Wets 1969). The primary construct in these methods amounts to creating a piecewise linear approximation of the cost-to-go function (which is also known as the recourse function in the SP literature). The main computational difference between SD and the earlier methods (Benders' decomposition, L-shaped method) is the computational effort necessary to create a new piece (or cut) of the approximation. In the earlier (traditional) approximations, one needs to evaluate the cost-to-go function exactly for any setting of the first-stage decision x^k (in iteration k). This evaluation requires the solution of as many instances of (10) as there are outcomes in the sample space of $\tilde{\omega}$. Not only is such an undertaking very computationally intensive, it also limits the applicability of such decomposition algorithms to problems with discrete random variables. Moreover, the need to solve all these LPs, restricts the ability of the traditional methods to instances in which the random variables are modeled using a few outcomes. SD overcomes these limitations by introducing approximations aimed at reducing the computations required to generate a new piece (of the piecewise linear approximation).

At each iteration (say k), SD introduces a new sampled outcome (ω^k) into the collection of previously sampled outcomes $\{\omega^1, \ldots, \omega^{k-1}\}$. At iteration k, the SD algorithm constructs a *lower bounding linear approximation* of the sample mean function

$$H_k(x) = \frac{1}{k} \sum_{t=1}^{k} h(x, \omega^t). \tag{11}$$

If we were to derive a Benders' cut for this sample-mean function we would need to generate a subgradient (as in SQG) for each outcome $h(x, \omega^t)$, $t = 1, \ldots, k$, at the point x^k, and then use the sample-averaging operation to obtain a linear approximation of (11). However, obtaining a subgradient for each outcome $h(x, \omega^t)$, $t = 1, \ldots, k$, (at the point x^k) requires the solution of k linear programs, and as the number of iterations increases, such a method would again require the solution of a large number of LPs (of form (10)) in each iteration. The SD algorithm suggests that asymptotic convergence can be achieved without solving all these LPs. Instead, the SD suggests that it is sufficient to solve only one second-stage LP (10) in any iteration, provided we use previously obtained data (on the optimal dual solutions for (10)) to define a lower bounding approximation of $h(x, \omega^t)$, for $t < k$. This process is described next.

For the most recently generated outcome ω^k, we evaluate $h(x^k, \omega^k)$, which involves the solution of (10) with (x^k, ω^k) as inputs. Suppose that a dual optimum to this problem is denoted π_k^k. Then, the data collection process within the SD algorithm accumulates this optimal dual vector into a set of previously discovered optimal dual vectors denoted V_{k-1}. The updated set of optimal dual vectors at iteration k is denoted V_k (i.e., $V_k = V_{k-1} \cup \pi_k^k$). Thus each element of the set V_k is an optimal dual vector discovered during the evaluation of $h(x^t, \omega^t)$, $t = 1, \ldots, k$ in each previous iteration.

Next, a lower bounding function for the k^{th} sample mean function (11) is obtained by assigning some dual feasible solution for each previously observed outcome $\{\omega^t\}$, $t < k$. To see this, note that LP duality ensures that for $\pi \in V_k$,

$$\pi^\top [r(\omega^t) - Tx] \leq h(x, \omega^t), \ \forall x. \tag{12}$$

Thus, in iteration k, the dual vector in V_k that provides the best lower bounding approximation at $\{h(x^k,\omega^t)\}$, for $t<k$ is given by the dual vector for which

$$\pi_t^k \in \text{argmax}\{\pi^\top[r(\omega^t)-Tx] \mid \pi \in V_k\}. \tag{13}$$

When the above operation is undertaken with appropriate data structures (see Higle and Sen 1996), it is computationally faster than solving a linear program from scratch. In any event, it follows that

$$H_k(x) \geq \frac{1}{k}\sum_{t=1}^{k}(\pi_t^k)^\top[r(\omega^t)-Tx]. \tag{14}$$

Using the lower bound in (14), SD adopts a procedure similar to Benders' decomposition, in that the right-hand-side of (14) is added as a new piece (or cut) of the piecewise linear approximation of $E[h(x,\tilde{\omega})]$. However, we should recognize that since the number of outcomes grows with the number of iterations, the pieces (cuts) approximate sample mean linearizations use different sample sizes. Thus in iteration t one uses a piece that approximates $H_t(x)$, not $H_k(x)$, for $t<k$. As a result, the older pieces need to be re-adjusted so that they continue to provide lower bounds for $H_k(x)$. Without loss of generality, we can assume that $h(x,\omega)\geq 0$ almost surely. With this assumption, it is clear that $H_k(x)\geq \frac{t}{k}H_t(x)$. Hence, by multiplying each previously generated piece ($t=1,\ldots,k-1$) by the multiplier t/k, all previously generated pieces provide a lower bound for the sample mean approximation $H_k(x)$. In any event, the approximation for the first-stage objective function at iteration k is then given by

$$f_k(x) := c^\top x + \max_{t=1,\ldots,k}\left\{\frac{t}{k}\times\frac{1}{t}\sum_{j=1}^{t}(\pi_j^t)^\top[r(\omega^j)-Tx]\right\}.$$

Since these approximations are carried out in a recursive manner, it is best to consult Higle and Sen (1996) regarding the data structures, and updates to be used for efficient implementations.

The most basic version of SD uses the sequence $\{x^k\}$ such that

$$x^{k+1} \in \text{argmin}\{f_k(x) \mid x\in X\}$$

where $X=\{x \mid Ax\leq b\}$ denotes the first-stage feasible region. However, this sequence of candidates is not very stable, and we recommend the use of a regularized approximation with a sequence of incumbent solutions as defined in Higle and Sen (1994). Denoting an incumbent at iteration k as $\bar{x}^k$, we recommend the following:

$$x^{k+1} \in \text{argmin}\{f_k(x)+\frac{1}{2}\|x-\bar{x}^k\|^2 \mid x\in X\}.$$

One updates the incumbent by estimating whether the (sample mean) point estimate of the objective value at x^{k+1} is better than the point estimate of the objective value of the incumbent $\bar{x}^k$. If it is, then $\bar{x}^{k+1}=x^{k+1}$; else, $\bar{x}^{k+1}=\bar{x}^k$. This procedure is referred to as the Regularized Stochastic Decomposition (RSD) method which we use for our computational study.

Before closing this summary we note that as k changes, so does $H_k(x)$. However, all but one of the observations used in defining $H_k(x)$ are used in $H_{k-1}(x)$. Hence, as k becomes large, variance reduction is achieved by using common random numbers.

3.1 The Bootstrapped Stopping Rule for SD

One of the main advantages of adopting SD (or more accurately RSD) is that it allows for the possibility of using stopping rules based on observations made during the execution of the algorithm. We discuss this process which was originally outlined in Higle and Sen (1999).

The RSD stopping rule uses the empirical distribution associated with the sampled observations, $\{\omega^t\}_{t=1}^k$ in place of the original distribution. In particular, we use bootstrapping to re-sample primal and dual problems associated with the regularized master problem.

Suppose that in iteration K, we denote the primal incumbent solution by ($\bar{x}^K$). The solution of the primal approximation yields corresponding dual optimal solutions (θ^K, λ^K), where θ^K is the vector of multipliers associated with the pieces (or cuts) of $H_K(x)$, and λ^K denotes dual multipliers associated with the first-stage constraints $Ax\leq b$. It turns out that these dual multipliers are feasible to the QP dual problem of the master program used in the RSD method. Hence we test whether the primal-dual pair of solutions (x^K, θ^K, λ^k) yields a sufficiently small estimated duality gap.

Our statistical tests rely upon the bootstrap method (Efron 1979) which undertakes replications using the observed empirical distribution. In order to do so we test the quality of ($\bar{x}^K$, θ^K, λ^K) by re-sampling the pieces (or cuts) and then evaluating a duality gap which results from using the given solutions in the re-sampled primal and dual problems. If a large proportion of the re-sampled pairs of problems (primal-dual) indicate that the point ($\bar{x}^K$, θ^K, λ^K) is acceptable, then we may conclude that the given solution is sufficiently good. In the interest of brevity, we omit the details, and refer the reader to Higle and Sen (1999).

4 COMPUTATIONAL RESULTS

The computational results reported here are based on the data given in Herer et al. (2006). Both IPA/SQG and RSD algorithms were programmed for a general stochastic programming problem as stated in ((9)-(10)). While the

implementation reported in Herer et al. (2006) uses the special network structure of the transshipment problem (10), our implementations for both methods did not use any specialized algorithm for the solution of (10). One other difference between our implementation and that presented in Herer et al. (2006) is that we started out each method with a starting order-up-to-quantity at the same value as the expected demand at each retailer. However, the rest of the implementation of the IPA/SQG method used the same parameters as in Herer et al. (2006). Thus, the step sizes, the number of batches used in each iteration, and the number of iterations of IPA/SQG were the same as those in Herer et al. (2006). Finally, we should note that our LP solver used the ILOG CPLEX callable library, version 8.1, and all programs were compiled and run in a Unix environment provided on Sun 280R machines.

Table 1 reports our efforts at replicating the computations of Herer et al. (2006). As in the above paper, each iteration used a sample size $M = 50,000$, and the iterations carried out $K = 3000$ steps of the IPA/SQG method, with step sizes given by the sequence $\alpha_k = 1000/k$, $k = 1,\ldots,K$. We carried out 5 runs of the IPA/SQG method, and the recommended solutions (average values, standard deviations, and 95% half widths), together with CPU times (in seconds) are reported in Table 1.

The CPU time reported in Table 1 is excessive due to the choice of the sample size (50,000) used in Herer et al. (2006). In Table 2, we report the recommended solutions and CPU times (average of 20 runs) when the sample size M per iteration is reduced to 1,000. Clearly, there is no significant change in the quality of the solution, and the CPU time is reduced considerably. Since these choices of M are somewhat arbitrary, methods like IPA/SQG require some prior experience with instances if we are to avoid excessive computations.

Table 3 reports the solutions obtained by RSD when using a stopping tolerance of 0.0001 (i.e., the duality gap is 0.01% of the estimated objective value). As with the summary provided in Table 2, 20 runs were performed for the summary provided in Table 3. Since the method adaptively chooses the number of iterations necessary to attain the specified accuracy, Table 3 provides an additional row specifying the sample size. This reflects the number of iterations of RSD used to satisfy the stopping rules. As reported in this table, on average, the number of observations is around 1293 (compared with $50,000 \times 3,000$ or $1,000 \times 3,000$ observations in the SQG runs). Note that the smaller sample sizes may cause larger variability in the decision x; however, the important observation to make is that the standard deviation of the objective function at the solutions identified by RSD is only 0.2, and when compared with a mean objective value of 148.33, the coefficient of variation in the objective value estimates is negligible. The estimated objective values of all 20 RSD runs are shown in Figure 1.

Table 1: Solutions Using IPA/SQG ($M = 50,000$, $K = 3,000$)

Retailer	Average	Stdev	95% Half Width
1	106.65	0.04	0.04
2	215.98	0.05	0.05
3	160.05	0.04	0.03
4	185.95	0.06	0.05
5	193.47	0.03	0.02
6	180.06	0.02	0.02
7	185.91	0.05	0.04
CPU Time	41,994.1	260.2	228.1

Table 2: Solutions Using IPA/SQG ($M = 1,000$, $K = 3,000$)

Retailer	Average	Stdev	95% Half Width
1	106.30	0.10	0.04
2	216.39	0.19	0.08
3	159.70	0.09	0.04
4	186.44	0.20	0.09
5	193.09	0.08	0.04
6	179.69	0.12	0.05
7	186.50	0.08	0.03
CPU Time	839.6	3.5	1.5

Table 3: Solutions Using RSD (Stopping Tolerance = 0.01%)

Retailer	Average	Stdev	95% Half Width
1	106.46	0.73	0.32
2	216.06	3.32	1.45
3	159.50	1.88	0.82
4	186.05	2.15	0.94
5	194.08	2.47	1.08
6	179.14	1.57	0.69
7	186.02	2.31	1.01
CPU Time	117.0	74.4	32.6
Sample Size	1293	255	112
Obj. Value	148.3	0.2	0.1

The computational times (in seconds) of RSD runs are reported in the row "CPU Time" in Table 3. Each individual CPU time of RSD runs and SQG runs with sample size $M = 3,000$ are plotted in Figure 2. While the quality of solutions reported in Tables 2 and 3 is similar, the average time to obtain these solutions using RSD is less than one half the computational times as reported in Figure 2.

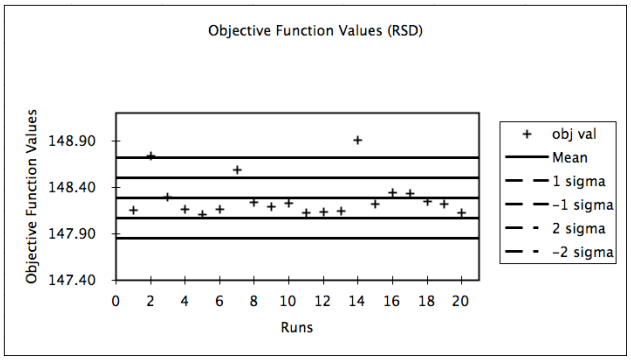

Figure 1: Objective Function Values: RSD

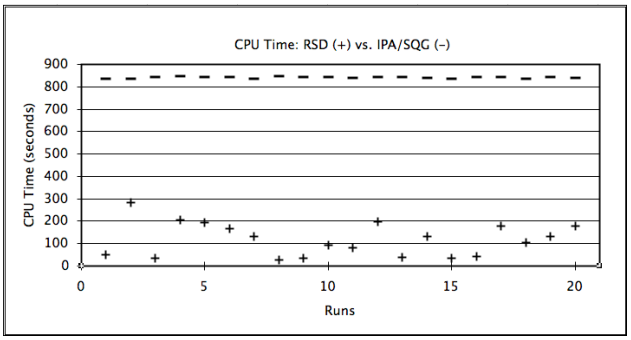

Figure 2: CPU Time: RSD Vs. IPA/SQG

5 CONCLUSIONS

In this paper, we have studied the performance of two alternative methods (IPA/SQG and RSD) for the solution of multi-location transshipment problems. Our computational study has shown that while the two methods provide solutions of similar quality, the amount of computational time required by RSD is significantly lower because it takes advantage of the special structure of the two-stage stochastic linear program. Computational times could be further reduced by using specialized network-flow solvers for the multi-location transshipment problem.

REFERENCES

Alrefaei, M., and S. Andradottir. 1997. Accelerating the convergence of the stochastic ruler method for discrete stochastic optimization. In *Proceedings of the Winter Simulation Conference*, 353–357.

Benders, J. F. 1962. Partitioning procedures for solving mixed variables programming problems. *Numerische Mathematik* 4:238–252.

Dantzig, G. B., and P. W. Glynn. 1990. Parallel processors for planning under uncertainty. *Annals of Operations Research* 22 (1-4): 1–21.

Efron, B. 1979. Another look at the jackknife. *Annals of Statistics* 7:1–26.

Ermoliev, Y. 1983. Stochastic quasigradient methods and their application to system optimization. *Stochastics* 9 (1-2): 1–36.

Herer, Y. T., M. Tzur, and E. Yucesan. 2006. The multilocation transshipment problem. *IIE Transactions* 38 (3): 185–200.

Higle, J. L., and S. Sen. 1991. Stochastic decomposition: an algorithm for two-stage linear programs with recourse. *Mathematics of Operations Research* 16:650–669.

Higle, J. L., and S. Sen. 1994. Finite master programs in stochastic decomposition. *Mathematical Programming* 67:143–168.

Higle, J. L., and S. Sen. 1996. *Stochastic decomposition: a statistical method for large scale stochastic linear programming*. Dordrecht, NL:Kluwer Academic Publisher.

Higle, J. L., and S. Sen. 1999. Statistical approximations for stochastic linear programming problems. *Annals of Operations Research* 85:173–192.

Pichitlamken, J., and B. L. Nelson. 2002. A combined procedure for optimization via simulations. In *Proceedings of the Winter Simulation Conference*, 292–300.

Robbins, H., and S. Monro. 1951. A stochastic approximation method. *The Annals of Mathematical Statistics* 22 (3): 400–407.

Sen, S. 1993. Subgradient decomposition and differentiability of the recourse function of a two stage stochastic linear program. *Operations Research Letters* 13:143–148.

Sen, S., and H. D. Sherali. 1985. On the convergence of cutting plane algorithms for a class of nonconvex mathematical programs. *Mathematical Programming* 31 (1): 42–56.

Van Slyke, R. M., and R. J.-B. Wets. 1969. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics* 17:638–663.

AUTHOR BIOGRAPHIES

LEI ZHAO is an assistant professor of Industrial Engineering at Tsinghua University, China, where he joined in January 2006. He received his Ph.D. in Systems and Industrial Engineering at the University of Arizona in December 2005. His research focuses on decision making under uncertainty in applications such as supply chain management/logistics and on emergency response planning.

SUVRAJEET SEN is Professor of Systems and Industrial Engineering at the University of Arizona, where he has been since 1982. Recently, he also served as a program director at NSF where he was responsible for the Operations Research and the Service Enterprise Engineering programs. Professor Sen's research is devoted to the theory and applications of large scale optimization algorithms, especially those arising in stochastic programming. He has authored or coauthored over seventy five papers, many of which have appeared in journals like Mathematical Programming, Mathematics of Operations Research, Management Science, and Operations Research. Professor Sen has served on the editorial board of several journals, including Operations Research as Area Editor for Optimization, and as Associate Editor in INFORMS Journal on Computing, Telecommunications Systems, as well as Operations Research. Professor Sen is the past-Chair of the INFORMS Telecommunications Section and founded the INFORMS Optimization Section. He is a Fellow of INFORMS.