Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

A COMPARISON OF TWO PARALLEL RANKING AND SELECTION PROCEDURES

Eric C. Ni
Shane G. Henderson

Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

Susan R. Hunter

School of Industrial Engineering
Purdue University
West Lafayette, IN 47907, USA

ABSTRACT

Traditional solutions to ranking and selection problems include two-stage procedures (e.g., the NSGS procedure of Nelson et al. 2001) and fully-sequential screening procedures (e.g., Kim and Nelson 2001 and Hong 2006). In a parallel computing environment, a naively-parallelized NSGS procedure may require more simulation replications than a sequential screening procedure such as that of Ni, Hunter, and Henderson (2013) (NHH), but requires less communication since there is no periodic screening. The parallel procedure NHH may require less simulation replications overall, but requires more communication to implement periodic screening. We numerically explore the trade-offs between these two procedures on a parallel computing platform. In particular, we discuss their statistical validity, efficiency, and implementation, including communication and load-balancing. Inspired by the comparison results, we propose a framework for hybrid procedures that may further reduce simulation cost or guarantee to select a good system when multiple systems are clustered near the best.

1 INTRODUCTION

The simulation optimization (SO) problem is a nonlinear optimization problem in which the objective function can only be observed with error through Monte Carlo simulation. Historically, algorithms developed to solve simulation optimization problems were created with serial computing platforms in mind — either explicitly, with algorithms incorporating information from function evaluations in a one-at-a-time manner (e.g., Kim and Nelson 2001) or implicitly, by failing to consider exploitation of parallel computing resources in algorithms that would otherwise easily be deployed on a parallel platform (e.g., exploitation of the need for multiple simulation replications in an "embarrassingly parallel" fashion). Given the current ubiquity of parallel computing platforms, there have been recent efforts by simulation researchers to develop simulation optimization algorithms that specifically exploit a parallel computing architecture — algorithms that we call "parallel" SO algorithms. In the context of parallel SO algorithms, we broadly assume that a single simulation replication is executed on a single core, which differs from work in parallel and distributed simulation, in which the goal is to coordinate the production of a single simulation replication across multiple cores (Fujimoto 2000).

Our focus, and the focus of much of the recent work on parallel SO algorithms, is on a class of SO problems also known as as ranking and selection (R&S), where the search space is a finite set of "systems." Algorithms that solve SO problems on finite sets can broadly be characterized as class $\mathcal{P}$ procedures, which provide a finite-time probabilistic guarantee on the solution quality, and class $\mathcal{L}$ procedures, which tend to provide guarantees on simulation efficiency (Pasupathy and Ghosh 2013). While parallel versions of class $\mathcal{L}$ procedures have been explored (Luo et al. 2000; Yoo, Cho, and Yücesan 2009), we further narrow our focus in this paper to parallel class $\mathcal{P}$ procedures that provide a finite-time guarantee. That is, we broadly focus on procedures that, upon completion, return the "best" system with probability greater than $1-\alpha$ when the best system is at least δ-better than the second-best for user-specified parameters α and δ.

On a serial computing platform with a modest number of systems, it is broadly known that fully-sequential implementations, such as the $\mathcal{KN}$ family of procedures (Kim and Nelson 2001), can save significant simulation replications over two-stage procedures. Thus it is natural to consider creating parallel versions of fully (or nearly-fully) sequential procedures. Indeed, recent work on parallel R&S procedures includes algorithms designed to sequentially eliminate systems from consideration via periodic screening on a parallel platform (Luo and Hong 2011; Luo et al. 2013; Ni, Hunter, and Henderson 2013). However, when moving from a serial platform to a parallel one, new challenges arise that call into question the relative efficiency of nearly-fully sequential parallel procedures over naively parallelized two-stage procedures for some problem types.

The primary challenge that arises on the parallel platform is the need for the algorithm to handle very large problems. With one hundred or more cores available, it seems reasonable that a parallel R&S algorithm should be able to handle problem sizes of one hundred to one thousand or more systems. However larger problems lead to increased computation required for screening, and a large problem deployed on a large number of cores may incur a large communication load to ensure timely statistical updates and screening of inferior systems. Further, as systems are eliminated, periodic load-balancing may be required to ensure full usage of available computing resources. Given the potential overhead required for communication, screening, and load-balancing that are inherent to designing parallel R&S procedures with sequential elimination, we ask:

> *On a stable parallel platform with a constant number of available cores, when is a sequential elimination procedure such as NHH (Ni, Hunter, and Henderson 2013) more efficient, in terms of total computational cost or total wall-clock time, than a naively parallelized version of a two-stage procedure designed for a large number of systems, such as NSGS (Nelson et al. 2001)?*

Note that a naively parallelized version of NSGS will require less communication, far less screening, and very little load-balancing since the required total number of observations is known in the second stage.

We explore this research question both analytically, by estimating the number of replications required for each procedure under different asymptotic regimes in the various problem parameters, and numerically, by running NHH and a naively-parallelized version of NSGS on a set of test problems. Our primary findings are as follows.

- The computational cost (in terms of number of replications) of parallelized NSGS is highly sensitive to δ, the indifference-zone parameter: cost increases at an approximate rate δ^{-2}.
- The computational cost of NHH is sensitive to the configuration of means and variances of the systems, being of order $\delta^{-1}\max\{\delta,\gamma\}^{-1}$, where γ represents the difference in objective value between the best and second-best systems. Accordingly, if γ is large, then NHH can be highly efficient.
- NHH seems to be much more efficient than parallelized NSGS when, as we expect in practice, the objective values of systems are "well spread out," in contrast to the (artificial) slippage configuration where all inferior systems are exactly δ worse than the best system.
- Parallelized NSGS is sensitive to the first-stage sample size n_1, whereas NHH is not.

The fact that we can solve ranking and selection problems with very large numbers of systems creates an interesting quandary. With very large numbers of systems, we might expect that for "most" selection problems, many systems might lie within δ of the best system. In this case, the correct-selection guarantee that many selection procedures are designed to deliver is no longer useful, because it offers no guarantee for such configurations. For these configurations, we prefer a *good selection* guarantee, e.g., Nelson and Matejcik (1995), in which a procedure guarantees, with high probability, to return a system within δ of the best, *irrespective of the configuration of the system means*. We introduce hybrid procedures, in the spirit

of Nelson et al. (2001), that provide such a good-selection guarantee. These hybrid procedures can be efficiently implemented on both sequential and parallel computing platforms, so may prove very important as parallel computing opens the door to a dramatic increase in the number of systems that can be handled through ranking and selection techniques.

2 PROBLEM SETTING AND PROCEDURES

In this section, we rigorously define the SO problem on finite sets and outline the procedures we analyze: the parallel procedure with sequential elimination, NHH, and a naively-parallelized version of NSGS.

2.1 Problem Setting

We consider problems of the type

$$\text{Find:} \quad \arg\max_{i \in \mathcal{S}} \mathrm{E}[f(i;\xi)],$$

where $\mathcal{S}$ is a finite set of systems, ξ is a random vector, and $\mathrm{E}[f(i;\xi)]$ is estimated by a sample mean. (One can consider ξ as the set of uniform random variables used to perform a simulation replication, and its distribution need not depend on i.) For notational simplicity, we henceforth define $\mu_i := \mathrm{E}[f(i;\xi)]$, such that we consider k systems having mean performance measures $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_{k-1} < \mu_k$, of which system k is the "best."

To solve such problems, consider a procedure that, after obtaining some amount of data from each system, returns to the user an estimated-best system. Let the correct selection (CS) event denote the event that the estimated-best system is equal to the true best system, System k. Then we broadly consider procedures that, upon termination, guarantee a CS event with at least probability $1-\alpha$ for some user-specified value $0 < \alpha < 1$ when the best system is at least $\delta > 0$ better than the second-best system. Here δ is the user-specified indifference-zone (IZ) parameter.

We also seek procedures that guarantee *good selection*. For $\delta > 0$, the good selection (GS) event is the event that the estimated-best system upon termination has a mean lying in the interval $[\mu_k - \delta, \mu_k]$. We would also like our procedure to guarantee that the probability of a GS event is at least $1-\alpha$.

In the remainder of the paper we let X_{ij} denote the (random) outcome of the jth simulation replication from system i and $\bar{X}_i(n) = \sum_{l=1}^{n} X_{il}/n$ for any system i and positive n. Furthermore, we let $S_i^2 = \sum_{l=1}^{n_1}(X_{il} - \bar{X}_i(n_1))^2/(n_1-1)$ where n_1 is the Stage 1 sample size defined in specific contexts.

2.2 Procedure NHH

Ni, Hunter, and Henderson (2013) proposed a R&S procedure (henceforth referred to as NHH) that utilizes parallel computing platforms in a master-worker pattern. The NHH parallel procedure is based on a sequential screening mechanism in Hong (2006) and hence inherits the latter's statistical guarantee on CS. The NHH procedure was partly motivated by Luo and Hong (2011).

The NHH procedure includes an (optional) initial stage, Stage 0, where workers run simulation replications in parallel to estimate completion times for each system, which are subsequently used to try to balance the workload. In Stage 1, a new sample is collected from each system in parallel for estimating variances. Prior to Stage 2, obviously inferior systems are screened. In Stage 2, the workers iteratively visit the remaining systems and run additional replications, exchange simulation statistics and independently perform screening over a subset of systems until all but one systems are eliminated. In more detail, the procedure is as follows.

Procedure NHH

1. Select overall confidence level $1-\alpha$, practically significant difference δ, Stage 0 sample size $n_0 \geq 2$, Stage 1 sample size $n_1 \geq 2$, and number of systems k. Set $a = \frac{n_1-1}{2\delta}\{[1-(1-\alpha)^{1/(k-1)}]^{-2/(n_1-1)} - 1\}$ and $\lambda = \delta/2$.

2. **(Stage 0, optional)** Master sends an approximately equal number of systems to each worker. Each system is simulated for n_0 replications and its average completion time is reported to the master.
3. **(Stage 1)** Master assigns systems to load-balanced simulation groups G_ℓ^1 for $\ell = 1,\ldots,m$ where m is the total number of workers (using information from Stage 0, if completed).
4. For $\ell = 1,2,\ldots,m$ in parallel on workers:
   (a) Sample X_{ij}, $j = 1,2,\ldots,n_1$ for all $i \in G_\ell^1$.
   (b) Compute Stage 1 sample means and variances $\bar{X}_i(n_1)$ and S_i^2 for $i \in G_\ell^1$.
   (c) Screen within group G_ℓ^1: system i is eliminated if there exists a system $j \in G_\ell^1 : j \neq i$ such that $(S_i^2/n_1 + S_j^2/n_1)^{-1}[\bar{X}_i(n_1) - \bar{X}_j(n_1)] < \min[0, -a + \lambda(S_i^2/n_1 + S_j^2/n_1)^{-1}]$.
   (d) Report survivors, together with their Stage 1 sample means and variances, to the master.
5. **(Stage 2)** Master assigns surviving systems $\mathcal{S}$ to approximately equal-sized screening groups G_ℓ^2 for $\ell = 1,\ldots,m$. Master determines a sampling rule $\{n_i(r) : i \in \mathcal{S}, r = 1,2,\ldots\}$ where each $n_i(r)$ represents the total number of replications to be collected for system i by iteration r. A recommended choice is $n_i(r) = n_1 + r\lceil \beta S_i \rceil$ where β is a constant and a large batch size $\lceil \beta S_i \rceil$ reduces communication.
6. For $\ell = 1,2,\ldots,m$ in parallel on workers (this step entails some communication with the master in steps (6b) through (6e), the details of which are omitted):
   (a) Set $r_\ell \leftarrow 1$. Repeat steps (6b) through (6f) until $|\mathcal{S}| = 1$:
   (b) If the r_ℓth iteration has completed for all systems $i \in G_\ell^2$ and $|G_\ell^2| > 1$ then go to step (6d), otherwise go to step (6c).
   (c) (Following the Master's instructions) Simulate the next system i in $\mathcal{S}$ (not necessarily G_ℓ^2) for $\lceil \beta S_i \rceil$ replications and go to step (6b).
   (d) Screen within group G_ℓ^2: system i is eliminated if there exists system $j \in G_\ell^2 : j \neq i$ such that $\tau_{ij}(r_\ell)[\bar{X}_i(n_i(r_\ell)) - \bar{X}_j(n_j(r_\ell))] < \min[0, -a + \lambda\tau_{ij}(r_\ell)]$ where $\tau_{ij}(r_\ell) = [S_i^2/n_i(r_\ell) + S_j^2/n_j(r_\ell)]^{-1}$.
   (e) Also use a subset of systems from other workers, e.g., those with the highest sample mean from each worker, to eliminate systems in G_ℓ^2.
   (f) Remove any eliminated system from G_ℓ^2 and $\mathcal{S}$. Let $r_\ell \leftarrow r_\ell + 1$ and go to step (6b).
7. Report the single surviving system in $\mathcal{S}$ as the best.

2.3 A Parallel Version of NSGS

We next outline a parallel version of NSGS (Nelson et al. 2001), which we call NSGS$_p$. In NSGS$_p$, we use a master-worker framework in which, after an optional Stage 0 to estimate simulation replication completion time, the master assigns systems to load-balanced groups and deploys the groups to the workers for simulation of Stage 1. Upon completion of the required replications, the workers calculate the first stage statistics and the systems are screened — first within groups by the workers, and then across groups by the master, where only the systems that survive worker-level screening are reported to the master. The master then computes the second-stage sample sizes for the surviving systems, and organizes the remaining required simulation replications into batches that are deployed across the workers for simulation of Stage 2. Upon completion of the required Stage 2 replications, the workers report the final sufficient statistics to the master, which compiles the results and returns the system with the largest sample mean to the user.

NSGS$_p$ preserves the CS guarantee because it simply farms out the simulation work to parallel processors while employing the same sampling and screening rules as the original NSGS. Compared to NHH, NSGS$_p$ is easier to implement on a parallel platform because the sample sizes are pre-computed and only one round of screening is required in each stage.

Procedure NSGS$_p$

1. Select overall confidence level $1-\alpha$, practically significant difference δ, first-stage sample size $n_1 \geq 2$, and number of systems k. Set $t = t_{(1-\alpha_0)^{1/(k-1)},n_1-1}$ and $h = h(1-\alpha_1, n_1, k)$, where h is Rinott's constant (see e.g. Bechhofer, Santner, and Goldsman 1995), and $\alpha_0 + \alpha_1 = \alpha$.

2. **(Stage 0, optional)** Conduct the sampling of Stage 0 as in Step 2 of NHH to estimate the simulation completion time for load-balancing.
3. **(Stage 1)** Master assigns systems to load-balanced groups G_ℓ for $\ell = 1,\ldots,m$ where m is the total number of workers (using information from Stage 0, if completed).
4. For $\ell = 1,2,\ldots,m$ in parallel on workers:
   (a) Sample X_{ij}, $j = 1,2,\ldots,n_1$ for all $i \in G_\ell$.
   (b) Compute first-stage sample means and variances $\bar{X}_i^{(1)} = \bar{X}_i(n_1)$ and S_i^2 for $i \in G_\ell$.
   (c) Screen within group G_ℓ: For all $i \neq j$, $i,j \in G_\ell$, let $W_{ij} = t(S_i^2/n_1 + S_j^2/n_1)^{1/2}$. System i is eliminated if there exists system $j \in G_\ell : j \neq i$ such that $\bar{X}_j^{(1)} - \bar{X}_i^{(1)} > \max\{W_{ij} - \delta, 0\}$.
   (d) Report survivors to the master.
5. Master completes all remaining pairwise screening tasks according to step (4c).
6. **(Stage 2)** For each system i surviving Stage 1 screening, the master computes the additional number of replications to be obtained in Stage 2

$$N_i^{(2)} = \max\{0, \lceil (hS_i/\delta)^2 \rceil - n_1\}. \tag{1}$$

7. Master load-balances the required remaining work into "batches" which are then completed by the workers in an efficient manner.
8. Master compiles all Stage 2 sample means $\bar{X}_i^{(2)} = \sum_{j=1}^{n_1+N_i^{(2)}} X_{ij}/(n_1 + N_i^{(2)})$ and selects the system with the largest sample mean as best once all sampling has been completed.

3 ANALYSIS OF PROCEDURE PERFORMANCE

Many R&S procedures guarantee to select the best system upon termination, subject to a user-specified probability of selection error. Among procedures with the same level of statistical guarantee, an efficient procedure should terminate in as few simulation replications as possible. The most efficient procedure may vary from one R&S problem to another depending on the configuration (distribution of system means and variances) of the systems in consideration. In addition, user-specified parameters such as the indifference-zone parameter δ have a significant impact on the efficiency of R&S procedures.

To assess and predict the efficiency of the NHH and NSGS$_p$ procedures under various configurations, we provide approximations for their expected number of replications needed upon termination. To simplify our analysis, we assume that an inferior system can only be eliminated by the best system, System k, and that System k eliminates all other systems. Strictly speaking this assumption does not apply to NHH because screening is distributed across workers and so not every system will necessarily be compared with System k. However, NHH shares statistics from systems with high means across cores during screening in Step 6e, so that a near-optimal system will be compared to inferior systems with high probability. Therefore, the total number of replications required by the procedure can be approximated by summing the number of replications needed for System k to eliminate all others. Although in practice the best system is unknown and an inferior system may eliminate another before System k does, an inferior system i is most likely eliminated by System k because the difference between μ_k and μ_i is the largest.

In the remainder of this section we assume that System k has mean $\mu_k = 0$ and variance σ_k^2, and an inferior system i has mean $\mu_i < 0$ and variance σ_i^2. We let $\mu_{ki} := \mu_k - \mu_i > 0$. The same indifference zone parameter δ is employed by both procedures. We focus on the case where variances are unknown.

3.1 Expected Number of Replications under NHH

The NHH procedure uses a triangular continuation region $C = C(a,\lambda) = \{(t,x) : 0 \le t \le a/\lambda, |x| \le a - \lambda(t)\}$ and a test statistic $Z_{ij}(r) = [S_i^2/n_i(r) + S_j^2/n_j(r)]^{-1}[\bar{X}_i(n_i(r)) - \bar{X}_j(n_j(r))]$. System i is eliminated by System k when $([S_i^2/n_i(r) + S_k^2/n_k(r)]^{-1}, Z_{ki}(r))$ exits C for the first time. Using the result that $Z_{ki}(r)$ is equal in distribution to $B_{\mu_k-\mu_i}([\sigma_i^2/n_i(r) + \sigma_k^2/n_k(r)]^{-1})$ where $B_\Delta(\cdot)$ denotes a Brownian motion with drift Δ and

volatility 1, we approximate the expected number of replications from System $i \neq k$ by

$$N_i^{\mathrm{NHH}} \approx E[n_i(\inf\{r : B_{\mu_{ki}}([\sigma_i^2/n_i(r) + \sigma_k^2/n_k(r)]^{-1}) \notin C\})] \approx \sigma_i(\sigma_i + \sigma_k)E[\inf\{t : B_{\mu_{ki}}(t) \notin C\}] \quad (2)$$

assuming, as recommended in Hong (2006) and Ni, Hunter, and Henderson (2013), $n_i(r) \approx S_i r \approx \sigma_i r$. The last expectation is the expected time that a Brownian motion with drift μ_{ki} exits the triangular region C, and is given in Hall (1997) by

$$E[\inf\{t : B_{\mu_{ki}}(t) \notin C\}] = \sum_{\mu \in \{\mu_{ki}, -\mu_{ki}\}} \sum_{j=0}^{\infty} (-1)^j \frac{a(2j+1)}{\mu+\lambda} \cdot e^{2aj(\lambda j - \mu)} \cdot \left[\bar{\Phi}\left(\frac{(2j+1-(\mu+\lambda)/\lambda)a}{\sqrt{a/\lambda}} \right) - e^{2a(2j+1)(\mu+\lambda)} \bar{\Phi}\left(\frac{(2j+1+(\mu+\lambda)/\lambda)a}{\sqrt{a/\lambda}} \right) \right] \quad (3)$$

where $\bar{\Phi}(x)$ is the tail probability of the standard normal distribution and can be approximated by $x^{-1}e^{-x^2/2}/\sqrt{2\pi}$ for large x.

Equation (3) is complicated, so we approximate it by focusing on one parameter at a time. First, the terms in the infinite sum rapidly approach zero as j increases, so the $j = 0$ term dominates. Second, when System i is significantly worst than System k, μ_{ki} is large, and then (3) is of order $O(\mu_{ki}^{-1})$ and we expect NHH to eliminate System i in very few replications. Third, (3) does not involve σ_i^2 and by (2) the cost of eliminating system i should be approximately proportional to $\sigma_i^2 + \sigma_i\sigma_k$.

Moreover, since the indifference-zone parameters a and λ are typically chosen such that $a \propto \delta^{-1}$ and $\lambda \propto \delta$, we may analyze the expectation in (3) in terms of δ. After some simplification we see that as $\delta \downarrow 0$, (3) is dominated by the $\frac{a(2j+1)}{\mu_{ki}+\lambda}$ term which is $O(\delta^{-1})$ when $\mu_{ki} \geq \delta$ and $O(\delta^{-2})$ when $\mu_{ki} \ll \delta$.

In conclusion, the expected number of replications needed to eliminate system i is on the order of $O((\sigma_i^2 + \sigma_i\sigma_k)\mu_{ki}^{-1}\delta^{-1})$ for sufficiently large μ_{ki} and $O((\sigma_i^2 + \sigma_i\sigma_k)\delta^{-2})$ when $\mu_{ki} \ll \delta$. This result agrees with intuition: high variances require larger samples to achieve the same level of statistical confidence, a large difference in system means helps to eliminate inferior systems early, and a more tolerant level of practical significance requires lower precision, hence a smaller sample.

3.2 Expected Number of Replications under NSGS$_p$

As stated in §2.3, the NSGS$_p$ procedure begins by taking a sample of n_1 replications from each system in the first stage, and uses the sample for variance estimation and a round of screening. System i is eliminated by system k in this stage if $(t((S_i^2 + S_k^2)/n_1)^{1/2} - \delta)^+ < \bar{X}_k^{(1)} - \bar{X}_i^{(1)}$ (henceforth denoted as event A). If system i is not eliminated in the first stage, then a second-stage sample of size $N_i^{(2)}$ is taken per Equation (1). Dropping other systems and considering only systems i and k, we can approximate the expected number of replications from system i needed by NSGS$_p$ by $N_i^{\mathrm{NSGS}_p} \approx n_1 + E[(1 - \mathbb{1}(A))N_i^{(2)}]$ where $\mathbb{1}(\cdot)$ is the indicator function. Replacing random quantities by their expectations, and using a deterministic approximation for the indicator-function term, we obtain the crude approximation

$$N_i^{\mathrm{NSGS}_p} \approx n_1 + \mathbb{1}\{(t[((\sigma_i^2 + \sigma_k^2)/n_1)^{1/2} - \delta]^+ - \mu_{ki} > 0\}(\lceil (h\sigma_i/\delta)^2 \rceil - n_1)^+. \quad (4)$$

Like NHH, (4) shows that higher σ_i^2, σ_k^2, δ^{-1} or μ_{ki}^{-1} may lead to higher elimination cost. In addition, the dependence on Stage 1 sample size n_1 is somewhat ambiguous: small n_1 reduces the probability of elimination $P[A]$, whereas big n_1 may be wasteful. An optimal choice of n_1 depends on the problem configuration and is unknown from the user's perspective. Furthermore, it can be easily checked that for sufficiently large $n_1 (\geq 20)$, the constants t and h are very insensitive to n_1 and k.

The dependence of N_i^{NSGS} on μ_{ki} differs somewhat from that of N_i^{NHH}. For sufficiently large μ_{ki}, $P[A]$ is very low and the NSGS$_p$ procedure requires a minimum of n_1 replications for elimination. On the other

hand, when μ_{ki} is small, the elimination cost is less sensitive to μ_{ki} as the Stage 2 sample size is bounded from above. Moreover, for sufficiently large σ_i^2, σ_k^2, and δ^{-1}, the NSGS$_p$ procedure almost always enters Stage 2, and (4) implies that the elimination cost is then directly proportional to σ_i^2 and δ^{-2}.

To summarize, it takes $O(\sigma_i^2\delta^{-2})$ replications for the NSGS$_p$ procedure to eliminate an inferior system i. Compared to our previous analysis on NHH, this result suggests that NSGS$_p$ is less sensitive to σ_k^2, but more sensitive to small δ, and may spend too much simulation effort on inferior systems when μ_{ki} is large.

4 NUMERICAL EXPERIMENTS

To complement the analytical arguments in the previous section, we present empirical evidence on the efficiency of the NHH and NSGS$_p$ procedures by running them on a tractable and scalable test problem.

4.1 The Test Problem

We consider a simulation optimization problem with the objective of finding the optimal resource allocation on a three-server flow line. On an abstract level, the problem can be stated as

$$\begin{aligned} \max_{x=(r_1,r_2,r_3,b_2,b_3)\in\mathbb{Z}_+^5} & E[g(x;\xi)] \\ \text{s.t. } & r_1+r_2+r_3 = R \\ & b_2+b_3 = B \end{aligned} \tag{5}$$

where the random vector ξ captures the uncertainty in each sample path and the stochastic function $g(x;\xi)$ returns the simulated average flow line throughput given allocation x. Parameters R and B represent available units of resources and jointly determine the (finite) number of feasible solutions.

A more detailed description and sample simulation code in both Matlab and C++ in for this problem can be found on SimOpt.org (Pasupathy and Henderson 2011). A number of prior studies including Pichitlamken, Nelson, and Hong (2006), Ni, Hunter, and Henderson (2013) and Luo et al. (2013) have used different R&S approaches to solve a small version of the problem with $R = B = 20$ and $k = 3249$ systems. Compared to those studies, the parallel R&S procedures we use enable us to solve instances of the problem that have many more systems using a moderate amount of computation time on parallel platforms.

The three instances we study are based on $R = B = 20$, $R = B = 50$, and $R = B = 128$. By increasing the amount of available resources, we make the problem harder through increasing the number of feasible solutions. Fortunately, as the problem assumes exponentially distributed service times, we can model the state of the flow-line system with a Markov chain and calculate the steady-state expectation in (5) analytically for every feasible resource allocation x. The distribution of system expectations provides insights on the "hardness" of different instances of the problem, and is summarized in Table 1. We also plot in Figure 1 the distribution of system mean (computed using the Markov chain) and variance (estimated using simulation) for two problem instances, where each point on the plot represents a feasible solution.

Table 1: Summary of three instances of the throughput maximization problem.

	Number of	Highest	pth percentile of system means			No. of systems in $[\mu_k-\delta,\mu_k]$		
Instance	systems k	mean μ_k	$p=75$	$p=50$	$p=25$	$\delta=0.01$	$\delta=0.1$	$\delta=1$
1	3249	5.78	3.52	2.00	1.00	6	21	256
2	57624	15.70	8.47	5.00	3.00	12	43	552
3	1016127	41.66	21.9	13.2	6.15	28	97	866

This example shows that system configurations for an actual SO problem can deviate quite dramatically from the typical slippage or equal-variance configurations assumed in traditional R&S literature. From the plots as well as Table 1, we see that the feasible regions contain many "inferior" systems that are far from the best, so conservative procedures based on worst-case assumptions may not perform well. We also

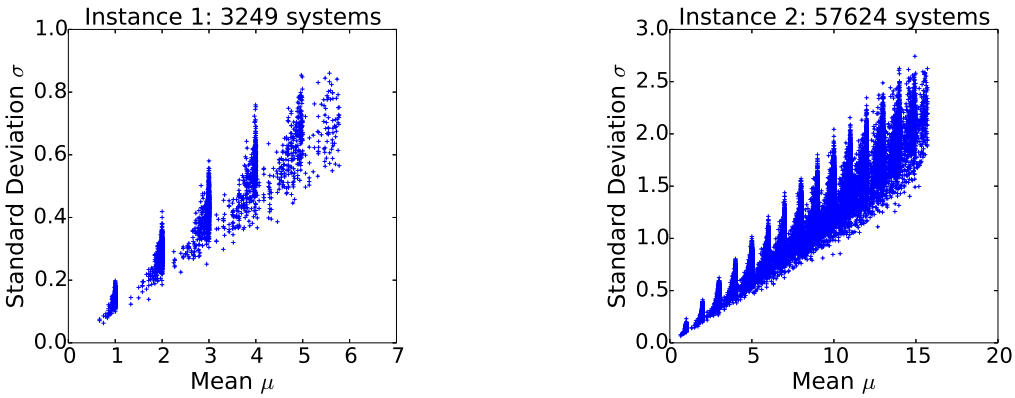

Figure 1: Mean-standard deviation profiles of three instances of the Throughput Maximization problem.

observe many "good" systems close to the best. As the problem gets bigger, we get more systems in the indifference zone $[\mu_k - \delta, \mu_k]$ for any user-specified δ, and thus it becomes harder to select the best system correctly. In addition, system mean and variance are shown to be positively correlated and the variance is highly nonuniform, so we expect procedures that allocate simulation replications according to variance estimates to outperform those that do not.

4.2 Performance of Parallel Procedures

Our numerical experiments are conducted on Extreme Science and Engineering Discovery Environment (XSEDE)'s Stampede HPC cluster. The Stampede cluster contains over 6,400 computer nodes, each equipped with two 8-core Intel Xeon E5 processors and 32 GB of memory and runs Linux Centos 6.3 OS (Stampede User Guide, Texas Advanced Computing Center 2014). Both NHH and NSGS$_p$ procedures are implemented in C++, using Message Passing Interface (MPI) for communication between cores. MPI is the de-facto standard of parallel programming supported by many high-performance platforms, and programs written in MPI tend to have low dependency on specific parallel computer architecture and can be migrated across computing platforms with relative ease. Our MPI implementation of NHH and NSGS$_p$ is available in source code on a Bitbucket repository (Ni 2014) and is being continuously developed.

We measure the empirical cost of R&S procedures by counting the total number of simulation replications collected upon procedure termination and measuring the wallclock time, while keeping the confidence level fixed at 95% in all test cases. We do not test the procedures' statistical validity in terms of the probability of making a correct selection, because to obtain an accurate estimate of the CS probability (which we expect to be close to one) requires running many metareplications on the same problem, which is too costly given that our computational budget is limited. Further, the CS guarantees have been proven in theory and verified for small problems (Nelson et al. 2001; Ni, Hunter, and Henderson 2013).

The results appear in Table 2. Columns 4 and 5 of Table 2 give sample averages over 20 metareplications which are accurate to approximately two significant figures, except for the problem with over one million systems. Column 6 is computed from Columns 4 and 5 and the number of cores employed.

The total computational cost of a parallel R&S procedure consists of two major components: the cost of obtaining simulation replications, which is affected by how efficiently the procedure collects and uses information and henceforth referred to as the "simulation cost", and the procedure "overhead" which includes screening, communication and waste due to imperfect load-balancing. The results in Table 2 show that the simulation cost, measured by the total number of simulation replications taken, is very sensitive to the choice of δ for NSGS$_p$. In both $k = 3249$ and $k = 57624$ cases, the number of total simulation replications used by NSGS$_p$ is increased by almost two orders of magnitude as δ is reduced from 0.1 to 0.01. The same change in δ increases the cost of NHH by less than one order of magnitude. This

Table 2: Summary of procedure costs on 3 instances of the throughput maximization problem with $\alpha = 0.05$, $n_0 = 20$, Stage 2 batch size constant $\beta = 500k/\sum_{i=1}^{k} S_i$.

Number of systems k	n_1	δ	Procedure	Total simulation replications ($\times 10^5$)	Total running time (s)	Per replication time (μs)
3249	60	0.01	NHH	39.0	23	375
(average of 20			NSGS$_p$	150.0	182	758
metareplications		0.1	NHH	11.0	7.0	396
on 64 cores)			NSGS$_p$	3.7	2.9	480
57624	80	0.01	NHH	920.0	536	365
(average of 20			NSGS$_p$	16,000.0	10,327	403
metareplications		0.1	NHH	300.0	179	371
on 64 cores)			NSGS$_p$	220.0	158	455
1016127	100	0.01	NHH	36,000.0	1,901	726
(one			NSGS$_p$	N/A (too costly)		
metareplication		0.1	NHH	7,200.0	324	572
on 1024 cores)			NSGS$_p$	23,000.0	1,724	564

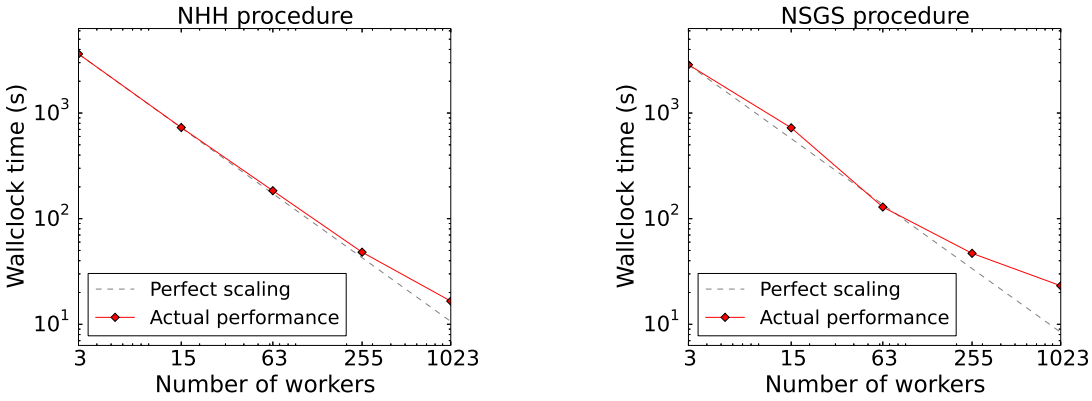

Figure 2: Scaling result on 57,624 systems with $\delta = 0.1$.

observation is consistent with our conclusion in Section 3 that NSGS is more sensitive to small δ. As a result, NHH outperforms NSGS$_p$ when $\delta = 0.01$, but underperforms when $\delta = 0.1$.

To assess procedure overhead, in Table 2 we also report the average time per replication, which is slightly higher for NSGS$_p$ than for NHH in most cases, implying that NSGS$_p$ incurs slightly more overhead. When k is increased from 3249 to 57624 while the number of cores remains at 64, the time-per-replication for NHH does not increase significantly, suggesting that it can handle larger problem instances without much additional overhead. However, when $\delta = 0.01$, the per-replication time for NSGS$_p$ changes dramatically, suggesting some inefficiency, perhaps in load-balancing (NHH balances workload in every iteration of Step 6, whereas NSGS$_p$ only does it once in Step 7). Eventually, as more cores are employed to solve the largest instance with 1016127 systems, we observe an increase in per-replication time for both procedures as parallel overhead increases.

In Figure 2 we further investigate procedure overhead with a scaling plot. We choose one particular problem instance with $\delta = 0.1$ and $k = 57624$ (because NHH and NSGS$_p$ perform similarly on this instance with 64 cores) and employ $m = 4, 16, 64, 256$ and 1024 parallel cores to solve the same instance. In an ideal situation with no parallel overhead, we expect the wallclock time of running the problem on m processors

to be inversely proportional to m, as illustrated by the straight dashed line in the log-log plot (we subtract 1 from m because one core is designated as the master). However, the plots exhibit gaps between the actual performance and the perfect scaling line due to parallel overhead, the amount of which is characterized by the size of the gap. As evidenced in Figure 2, the NHH procedure incurs less overhead relative to NSGS$_p$, but both parallel procedures deliver fairly strong scaling performance.

5 COMBINING SEQUENTIAL SCREENING AND INDIFFERENCE-ZONE SELECTION

In this section, we explore a class of hybrid R&S procedures which starts with sequential screening as the first stage, and at some point switches to a Rinott-like indifference-zone selection such as Stage 2 of NSGS$_p$. This approach allows us to combine different screening and switching rules to create hybrid procedures with desired properties, such as reduction in simulation cost (Nelson et al. 2001), or a probability guarantee of good selection when multiple systems fall in the indifference zone, or perhaps even both.

5.1 A Framework for Hybrid Procedures

A hybrid procedure begins by running sequential screening such as NHH as the first stage, simulating and screening various systems until some switching criterion is met or only one system remains. If multiple systems survive, then the procedure applies the Rinott (1978) indifference-zone method on the surviving systems in a second stage, taking $N_i^{(2)}$ additional samples from each system i as per (1). Finally, the procedure combines the first- and second-stage samples and selects the system with the highest sample mean as the best. In effect, the first stage acts like a subset-selection procedure (Kim and Nelson 2006).

A basic requirement for a hybrid procedure is that it should guarantee correct selection with high probability under the usual indifference-zone qualification. To achieve that, the first-stage screening rule should ensure that the best system survives the first-stage with high probability. Then, among the systems that survive the first stage, the Rinott procedure performed in the second stage should select the best, again with high probability. If the probabilities of making an error in both stages are carefully bounded, then the overall hybrid procedure guarantees CS with high probability as required (Nelson et al. 2001).

There are multiple ways to design the screening and switching rules employed in the first stage, each of which may result in different properties of the hybrid procedure. We discuss two approaches here. One switching rule attempts to directly reduce the total simulation cost by switching from sequential screening to Rinott when the latter is expected to terminate sooner. To achieve this, the procedure needs to compute the additional Rinott sample size $N_i^{(2)}$ in each iteration in the first stage, and to estimate the expected number of extra replications needed for sequential screening to terminate. If it is determined that, to complete screening the surviving systems, sequential screening will require a significantly larger sample than that determined by (1), we would switch to the second stage.

Another choice of switching rule deals with situations where there are many systems near the best. While some procedures (for instance, NSGS$_p$) are specifically designed to deliver a good-selection guarantee regardless of $\mu_k - \mu_{k-1}$ (Nelson et al. 2001), others (e.g., Kim and Nelson 2001, Hong 2006, and NHH) are designed to deliver a correct-selection guarantee when $\mu_k - \mu_{k-1} \geq \delta$, and for the latter procedures it is not clear whether they also satisfy a good-selection guarantee or not when $\mu_k - \mu_{k-1} < \delta$ (Hong and Nelson 2014). We can deliver a GS guarantee if we adopt a two-stage structure as outlined next.

5.2 Good-Selection Procedure (GSP)

Using the hybrid framework, we may choose the sequential selection parameters and specify a switching time such that the best system survives the first stage with high probability, regardless of the actual distribution of system means. Then, the Rinott procedure in the second stage will guarantee to select a "good" system among those that survive the first stage (Nelson and Matejcik 1995). One can therefore prove that overall the hybrid procedure has a high probability of making a good selection.

For illustration, we present here a good-selection procedure (GSP) assuming system variances $\{\sigma_i^2 : i \in \mathcal{S}\}$ are known, in which case there is no need to estimate variances in a separate stage, and the Rinott procedure in the second stage can be replaced by its known-variance equivalent; see, e.g., Kim and Nelson 2006 §2.2. We omit parallelism and load-balancing from our presentation, even though we continue to use them, because our focus is on guaranteeing good selection.

Procedure GSP

1. Choose the indifference zone parameter $\delta > 0$, Type-I error rates α_0, α_1, triangular continuation parameters $a > 0, \lambda > 0$. Choose a stopping time t_S such that $P_0(t_S, U) \leq 1 - (1-\alpha_0)^{1/(k-1)}$ where $P_\Delta(t,U)$ is the probability that a Brownian motion with drift Δ exits the continuation region along the upper boundary before t, and can be calculated following Hall (1997). Choose a sampling rule $n(s) = \{n_1(s), \ldots, n_k(s)\}$ where each $n_i(s)$ is an integer-valued, non-decreasing sequence in s representing the total number of replications to be collected for system i by iteration s. Let $r \leftarrow 0$.
2. **(Stage 1)** Let $r \leftarrow r+1$ and take the rth sample (i.e. $n_i(r) - n_i(r-1)$ additional replications from each surviving system i) following the sampling rule.
3. For each $i \neq j$ compute $t_{ij}(r) = \left[\sigma_i^2/n_i(r) + \sigma_j^2/n_j(r)\right]^{-1}$ and $Z_{ij}(t_{ij}(r)) = t_{ij}(r)\left[\bar{X}_i(n_i(r)) - \bar{X}_j(n_i(r))\right]$.
4. $\mathcal{S} \leftarrow \mathcal{S} \backslash \{i \in \mathcal{S} : Z_{ij}(t_{ij}(r)) < \min[0, -a + \lambda t_{ij}(r)]$ for some $j \in \mathcal{S}$ such that $j \neq i$ and $t_{ij}(r) < t_S]\}$.
5. If $|\mathcal{S}| = 1$ then select the system whose index is in $\mathcal{S}$.
6. If $t_{ij}(r) > t_S$ for all pairs $i, j \in \mathcal{S}$ then go to Step 7. Otherwise go to Step 2.
7. **(Stage 2)** For all systems $i \in \mathcal{S}$ compute $N_i = \max\left\{n_i(r), \lceil 2(h\sigma_i/\delta)^2 \rceil\right\}$ using significance level $1-\alpha_1$ for calculating h (see Kim and Nelson 2006, pp. 504-505 for details). Take $N_i - n_i(r)$ additional samples from each system i. Select the system with the highest sample mean $\bar{X}_i(N_i)$.

Theorem 1 states the "good-selection" guarantee. Due to space constraints we omit the proof here.

Theorem 1. *Let I be the random index of the system selected by GSP. Then $P(\mu_k - \mu_I \leq \delta) \geq 1 - \alpha_0 - \alpha_1$.*

The survival of System k in Stage 1 is guaranteed by the choice of switching time t_S, which can be calculated for any set of parameters a and λ. As the triangular continuation region no longer needs to be designed to bound the error probability, the choice of a and λ is completely flexible and the optimal construction of a continuation region is an interesting open problem.

ACKNOWLEDGMENTS

This work was partially supported by NSF grant CMMI-1200315, and used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575. The authors would like to thank Nicholas Samson for implementing NSGS$_p$.

REFERENCES

Bechhofer, R. E., T. J. Santner, and D. M. Goldsman. 1995. *Design and analysis of experiments for statistical selection, screening, and multiple comparisons*. Wiley New York.

Fujimoto, R. M. 2000. *Parallel and Distributed Simulation Systems*. New York: Wiley.

Hall, W. J. 1997. "The distribution of Brownian motion on linear stopping boundaries". *Sequential Analysis* 16 (4): 345–352.

Hong, L. J. 2006. "Fully sequential indifference-zone selection procedures with variance-dependent sampling". *Naval Research Logistics* 53 (5): 464–476.

Hong, L. J., and B. L. Nelson. 2014. "Personal communication".

Kim, S.-H., and B. L. Nelson. 2001. "A fully sequential procedure for indifference-zone selection in simulation". *ACM Transactions on Modeling and Computer Simulation* 11 (3): 251–273.

Kim, S.-H., and B. L. Nelson. 2006. "Selecting the best system". In *Simulation*, edited by S. G. Henderson and B. L. Nelson, Volume 13 of *Handbooks in Operations Research and Management Science*, 501–534. North-Holland Publishing, Amsterdam.

Luo, J., J. L. Hong, B. L. Nelson, and Y. Wu. 2013. "Fully Sequential Procedures for Large-Scale Ranking-and-Selection Problems in Parallel Computing Environments". *Working Paper*.

Luo, J., and L. J. Hong. 2011. "Large-scale ranking and selection using cloud computing". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 4051–4061. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Luo, Y.-C., C.-H. Chen, E. Yücesan, and I. Lee. 2000. "Distributed web-based simulation optimization". In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 1785–1793. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Nelson, B. L., and F. J. Matejcik. 1995. "Using Common Random Numbers for Indifference-Zone Selection and Multiple Comparisons in Simulation". *Management Science* 41 (12): 1935–1945.

Nelson, B. L., J. Swann, D. Goldsman, and W. Song. 2001. "Simple Procedures for Selecting the Best Simulated System When the Number of Alternatives is Large". *Operations Research* 49 (6): 950–963.

Ni, E. C. 2014. "mpiRnS: Parallel Ranking and Selection Using MPI". https://bitbucket.org/ericni/mpirns.

Ni, E. C., S. R. Hunter, and S. G. Henderson. 2013. "Ranking and selection in a high performance computing environment". In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, 833–845. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pasupathy, R., and S. Ghosh. 2013. "Simulation Optimization: A concise overview and implementation guide". In *TutORials in Operations Research*, edited by H. Topaloglu, Chapter 7, 122–150. INFORMS.

R. Pasupathy and S. G. Henderson 2011. "SimOpt". http://www.simopt.org.

Pichitlamken, J., B. L. Nelson, and L. J. Hong. 2006, 8/16. "A sequential procedure for neighborhood selection-of-the-best in optimization via simulation". *European Journal of Operational Research* 173 (1): 283–298.

Rinott, Y. 1978. "On two-stage selection procedures and related probability-inequalities". *Communications in Statistics - Theory and Methods* 7 (8): 799–811.

Texas Advanced Computing Center 2014. "TACC Stampede User Guide". Accessed May. 11, 2014. https://www.tacc.utexas.edu/user-services/user-guides/stampede-user-guide.

Yoo, T., H. Cho, and E. Yücesan. 2009. "Web services-based parallel replicated discrete event simulation for large-scale simulation optimization". *SIMULATION* 85 (7): 461–475.

AUTHOR BIOGRAPHIES

ERIC C. NI is a Ph.D. student in the School of Operations Research and Information Engineering at Cornell University. He received a B.Eng. in Industrial and Systems Engineering and a B.Soc.Sci. in Economics from the National University of Singapore in 2010. His research interests include simulation optimization, emergency services and queuing theory. His webpage is http://people.orie.cornell.edu/cn254/.

SHANE G. HENDERSON is a professor in the School of Operations Research and Information Engineering at Cornell University. His research interests include discrete-event simulation and simulation optimization, and he has worked for some time with emergency services. He co-edited the Proceedings of the 2007 Winter Simulation Conference. His web page is http://people.orie.cornell.edu/∼shane.

SUSAN R. HUNTER is an assistant professor in the School of Industrial Engineering at Purdue University. Her research interests include Monte Carlo methods and simulation optimization. Her email address is susanhunter@purdue.edu, and her webpage is http://web.ics.purdue.edu/∼hunter63/.