Sequential Procedures for Large-Scale Ranking-and-Selection Problems in Parallel Simulation Environments

January 21, 2013

1 Introduction

Selecting the best system with the largest or smallest mean performance from a set of a finite number of alternatives is a common problem in many areas of operations research and management science. For instance, in designing a multi-stage manufacturing line we may need to determine the best allocation of the buffer space to maximize the average throughput, in controlling an inventory system we may need to identify the best reorder point to minimize the average cost, and in managing an emergency center we may need to select the optimal vehicle dispatching policy to minimize the average response time. In all these examples the mean performances of the alternatives may be evaluated by running simulation experiments. This type of problems is known as a rank-and-selection (R&S) problem in the simulation literature.

Many R&S procedures have been developed in the literature to select the best system (see, for instance, Kim and Nelson (2006a) for a comprehensive review). These procedures typically determine how to allocate the simulation effort to all alternatives such that the best can be selected with certain statistical guarantees, e.g., a pre-specified probability of correct selection (PCS). However, these procedures are often designed to handle a relatively small number of alternatives. As pointed out by Kim and Nelson (2006a), the two-stage procedure of Rinott (1978), hereinafter called Rinott's procedure, is often used to handle no more than 20 alternatives, and the fully sequential procedure of Kim and Nelson (2001), hereinafter called KN procedure, is often used to handle up to 500 alternatives. The NSGS procedure of Nelson et al. (2001) is designed specifically to solve large-scale R&S problems. However, the largest test problem reported in their paper has only 500 alternatives.

In practice, however, there are many R&S problems that have thousands to tens of thousands of alternatives. Traditionally, these problems are solved using optimization-via-simulation (OvS) algorithms (see, for instance, Hong and Nelson (2009) for a recent review of OvS). Most of the OvS

algorithms, which handle this type of problems, guarantee global convergence, i.e., they guarantee to select the best system as the simulation effort goes to infinity. To achieve global convergence, however, these algorithms evaluate all alternatives as the simulation effort goes to infinity, which become essentially R&S procedures. When they stop short of infinity, as they always do, there is often no (statistical) guarantee on the quality of the selected solution and the solution may be significantly worse than the optimal solution. Therefore, if there is sufficient computing power, we may actually want to apply R&S procedures on these problems to select the best solution with a statistical guarantee. However, the limited computing power often stops us from doing that.

In the past few years one of the drastic developments in computing environment is the quick adoption of parallel computing. Multiple-core processors are ubiquitous today, they are used not only in servers and personal computers, but also in panel computers and even smart phones. Moreover, large amount of computing resource (e.g., parallel processors) delivered as a service through the Internet, often called cloud computing, is also becoming readily available and often quite affordable to ordinary users. This motivates us to consider how to solve large-scale R&S problems in parallel simulation environments[1] and, in particular, whether current R&S procedures are statistically valid and efficient in parallel simulation environments and, if they are not, how to design statistically valid and efficient new procedures.

R&S problems are in general easy to fit into parallel computing environments. When solving a R&S problem, most of the computing time is typically used to generate independent simulation observations from various alternatives, and this can be done by executing the simulation programs in parallel scheme without requiring any synchronization among different processors. This type of parallelization is often called "embarrassingly parallel" (see, for instance, Foster (1995)) and it makes parallel computing very attractive to solve R&S problems. This advantage of using parallel simulation technology for R&S problems has also been discussed by Chen (2005) and Yücesan et al. (2001).

Large-scale R&S problems are also in general feasible to fit into parallel computing environments. The total computing effort required to solve a R&S problem typically increases only moderately as the problem size increases. Taking Rinott's procedure as an example, we plot the expected total number of samples as a function of the number of systems k in Figure (). We find it grows slightly slower than the order of $k \log(k)$. To make this result more intuitive, suppose that we have 100 parallel processors and each processor can handle a R&S problem with 500 alternatives in an allowable amount of time using a Rinott's procedure. Then, in the same amount of time, Rinott's procedure on all processors can handle the same problem with at least 30,000 alternatives, which significantly enlarges the type of R&S problems that may be solvable. We also plot the

[1]It is worthwhile noting that, when a R&S procedure is not specifically coded to use multiple cores, it uses only a single core no matter how many cores that the computer may have.

maximum (or worst-case) expected total number of samples for the KN procedure in Figure (). We find it grows in the order of $k(k-1)^{\frac{2}{n_0-1}}$ where n_0 is the first-stage sample size. Note that, when the number of systems increases, the proportion of clearly inferior alternatives often increases much faster than that of good alternatives. Therefore, fully sequential procedures, e.g., KN procedure, that allow early eliminations often require much smaller expected sample size than the worst case, which makes sequential procedures even more attractive to large-scale R&S problems than two-stage procedures, e.g., Rinott's procedure.

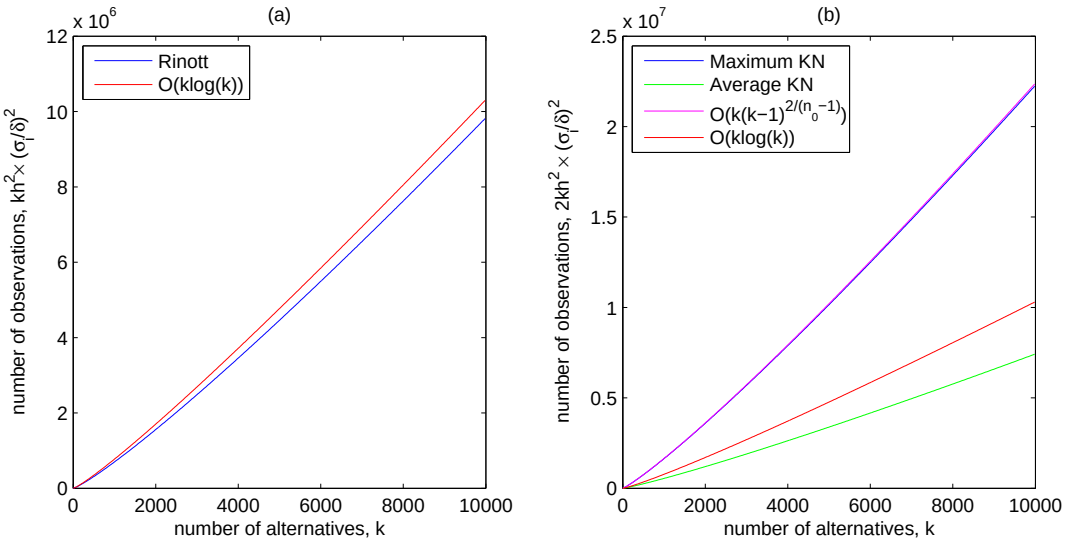

Figure 1: In Rinott's procedure and KN procedure, we assume that the initial sample size $n_0 = 16$, variances across all alternatives $\sigma_i^2 = 1$, and the indifference-zone parameter $\delta = \sigma_i/\sqrt{n_0}$.

From the implementation point of view, two-stage procedures are typically easier to be parallelized compared with sequential procedures. For instance, a naive approach of implementing Rinott's procedure can be done as follows. In the first stage, we distribute kn_0 replications equally among all processors and compute the first-stage sample variances and second-stage sample sizes of all alternatives after all processors finish their jobs. In the second stage, we again distribute all additional samples equally among all processors and select the alternative with best sample mean after all processors finish their jobs. Note that the total time required to complete the procedure is determined by the processor that finishes its job last. When replication times of different alternatives are different or they are random, the total time of this approach may be quite long and cause many processors to be idle. To improve the efficiency, one may estimate the replication time for each alternative after the first stage and formulate the sample allocation problem in the second stage as a stochastic parallel machine scheduling problem to minimize its makespan (i.e., total time to completion). Interested readers may refer to Chapter 12 of Pinedo (2008) for more discussions on the scheduling literature.

When there are multiple processors available to handle large-scale R&S problems, the number of processors are typically a few orders-of-magnitude smaller than the number of alternatives, and many orders-of-magnitude smaller than the expected total sample size. Then essentially, the large number of replications are distributed to these processors on which observations are simulated sequentially. From the view of the entire computing system (containing all processors), the simulation results are also reported sequentially and hence, "the sequential nature of computer simulation" (Kim and Nelson (2006a)) also applies to typical parallel computing environments. Therefore, it makes sense to design sequential R&S procedures that take advantages of the available information to eliminate inferior alternatives earlier. Moreover, as we mentioned earlier, when there are a large number of alternatives, a large percentage of them may be clearly inferior to the best. Then, sequential procedures can eliminate them early and save on the total computing resource.

In summary, to solve large-scale R&S problems, we would like to have fully sequential procedures that fit into parallel computing environments with valid statistical guarantees. There are many different configurations of parallel computing environments, ranging from multi-core personal computers to many-core servers to local computer farms and to clouds on the Internet. In this paper, we focus mainly on designing statistically valid fully sequential procedures for multi-core personal computers and many-core servers. Then, without communications via the Internet, the times of loading simulation programs to different processors and transmitting data among processors are almost negligible for these procedures. When implementing these procedures on clouds on the Internet, however, there may be package delays or even losses on the Internet, which may affect the validity of the procedures. Therefore, we leave the design of fully sequential procedures for cloud implementations as a topic for future research.

When designing fully sequential procedures for parallel computing environment, a critical question is "*what makes fully sequential procedures on multiple processors different from the ones on a single processor?*" A succinct answer to this question is that "*the input and output sequences of observations are different on multiple processors, while they are same on a single processor.*" A single processor system works like a single server queue, the departure (i.e., output) sequence is same as the arrival (i.e., input) sequence; while a multiple processor system works like a multiple server queue, the departure sequence is in general different from the arrival sequence when the service time (i.e., replication time of an observation in our situation) is random. In a simulation study we may control the input sequence deterministically. For instance, in KN procedure, we simulate all systems one at a time according to a predetermined order. Then, the output sequence of a single processor system is also the same deterministic sequence. However, the same input sequence may result in a random output sequence on a multiple processors system.

The randomness in the output sequence creates both implementation issues as well as statistical

issues when designing fully sequential R&S procedures. From an implementation point of view, randomness in output sequence makes sample size synchronization difficult. For instance, when System 1 has 30 observations, System 2 may have 40 while System 3 may have only 20. Then, procedures that require perfect synchronization of sample sizes from all systems are either difficult to implement or inefficient (i.e., using only part of observations, e.g., setting a sample size as 20 in our three-system example). However, implementation issues are often easy to handle as there exist fully sequential R&S procedures that allow unequal sample sizes from different systems (e.g., Hong (2006)). The statistical issues caused by randomness in output sequence are more crucial. First, when the performance of an alternative is correlated to its replication time, observations with shorter replication time tend to be available earlier and the sequence of output observations may not be independent even though they use independent random numbers. This problem even exists when simulating a single alternative using multiple processors. Heidelberger (1988) shows that the output sequence may not be an i.i.d. sequence even when they observations are run independently. Second, even when the performances of the alternatives are independent of their replications times or even the replication times are constant, sample sizes of survived systems depend on elimination decisions which in turn depend on sample-mean information of the systems. This type of dependence destroys the independence between sample means and sample sizes in typical R&S procedures due to Stein (1945) and, thus, make the typical R&S procedures no longer statistical valid. More details on the statistical issues caused by random output sequence will be discussed in Section 2.

In this paper we propose two solutions to deal with these issues. If one insists on making existing sequential R&S procedures suitable for parallel simulation schemes, we may only perform comparisons based on the input sequence of samples. Then, we suggest to create a vector to record the observations exactly in the order of the input sequence and make comparisons based on a pre-determined comparison rule. For instance, one may perform a comparison after all surviving alternatives have their first m observations finished for any $m = n_0, n_0 + 1, \ldots$ and the procedure has the exactly the same statistical validity of KN procedure. We call this type of procedures *vector filling procedures* as they fill the vector of observations based on the input sequence. Although vector filling procedures have finite-sample statistical validity, they may not use all available observations at the time of comparison and may also add complexity in implementation as one needs to track the input order. If one is content with asymptotical validity, we design fully sequential procedures that allow unequal sample sizes for all alternatives and make elimination decisions based on all available observations and that can be shown valid asymptotically as the indifference-zone parameter goes to zero, an asymptotic regime used in Kim and Nelson (2006a). Note that, as the indifference-zone parameter goes to zero, the procedures need an infinite sample size to make a correct decision. As the sample size of each alternative goes to infinity, the difference between the sample means calculated based on input sequence and output sequence also goes to zero, and the statistical

validity of the procedures can thus be guaranteed.

Our work is related to three streams of simulation literature. The first is the literature on R&S. In this paper we take a frequentist's view and consider the indifference-zone (IZ) formulation of the problem. The IZ formulation was first proposed by Bechhofer (1954) and related procedures are summarized in Bechhofer et al. (1995) and Kim and Nelson (2006a). There are also many Bayesian formulations and procedures to the R&S problems. For instance, Chen et al. (2000) and Chick and Inoue (2001a,b) attempt to allocate a finite number of samples to different alternatives to maximize the posterior probability of correct selection. Recently, Frazier (2011) investigates the problem from a different perspective and develops a Bayes-inspired IZ procedure to handle around a problem with 15,000 alternatives (where alternatives are simple function evaluations that can be simulated very quickly). The second is the literature on parallel and distributed simulation (PADS). According to Heidelberger (1988), PADS are two different approaches to parallelizing the simulation experiments, in which the former means each processor simulates multiple independent replications and the latter means multiple processors cooperate on a single realization or replication. There was a vast literature on PADS in 1980s and 1990s, where many of them focus on the synchronization issues by analyzing correct ordering of events in discrete-event simulations (see, for instance, Misra (1986) and Fujimoto (1990)). Recently, cloud computing has also been applied to handle PADS (Fujimoto et al. (2010)). The third is the literature on simulation output analysis in a parallel and distributed simulation environment. Heidelberger (1988) discusses a variety of statistical properties for sample mean estimators calculated by observations from terminating simulations in a parallel simulation environment under three different stopping rules. Glynn and Heidelberger (1991) further study mean performance estimators for both terminating simulations and steady-state regenerative simulations under a completion time constraint. Recently, Hsieh and Glynn (2009) propose two new estimators for steady state simulations by weighting the sample average across replications on multiple processors according to some model selection criterion. To the best of our knowledge, there are only two papers using parallel and distributed simulation to solve R&S problems. The first is by Yücesan et al. (2001), who implement an optimal computing budget allocation (OCBA) algorithm in a web-based parallel environment to select the best system based on a Bayesian approach. The second is by Chen (2005), who applies a multi-stage R&S procedure with the simulation tasks of each stage distributed to multiple processors. Both papers test their procedures only using small scale problems (both with only 10 alternatives), it is not clear whether their procedures are capable of handling large-scale R&S problems.

The rest of this paper is organized as follows: In Section 2, we describe the distinguished difference and difficulty using parallel simulation. Then, we propose a meaningful asymptotic study to overcome the difficulty, and design a fully R&S procedure in Section 3. Some other potential issues are discussed in Section 4 to make our procedure more efficient and easier to implement on

various parallel simulation environments. In Section 5, numerical experiments for the proposed procedure are tested to demonstrate its validity and efficiency, follows by some concluding remarks in Section 6.

2 The Randomness of Output Sequence

Suppose there are k independent systems whose mean performances can be evaluated by simulation on m processors. Let $X_{i\ell}$ denote the ℓth sample from system i, and we assume that $X_{i\ell}$, $\ell = 1, 2, \ldots$, are i.i.d. random variables with finite mean μ_i for each $i = 1, 2, \ldots, k$. Under the IZ formulation, we further assume that $\mu_1 - \delta \geq \mu_2 \geq \ldots \geq \mu_k$, where δ is the IZ parameter, and our goal is to design valid fully sequential procedures that can select system 1 as the best with probability $1 - \alpha$. In particular, in this paper we are mainly interested in implementing these R&S procedures on multi-core personal computers and many-core servers. Then, for mathematical simplification, we also assume that the m processors are homogeneous in terms of the processing speed and the times of loading the simulation into each processors and transmitting data among the processors are negligible.

2.1 Queueing Analogy

To better understand the difference between implementation of fully sequential procedures on a single processor (i.e., $m = 1$) and that on multiple processors (i.e., $m > 1$), we describe the simulation process using queueing models. In the queueing analogy, samples from alternative i are denoted by class i customers, m homogeneous processors are denoted by a server pool with m identical servers. There is no arrival process but all customers are waiting in the queue with a predetermined order, which is called the *input sequence*. The first m customers will be assigned to the m servers at the beginning of simulation. Once a server finishes the service of his current customer (i.e., generating the observation), the first customer waiting in queue will be immediately routed to that server. The departure process captures states of customers after service (i.e., the observations), which is called the *output sequence*. Based on the observations in the output sequence, we perform comparison and elimination among all survived alternatives. Once alternative i is decided to be eliminated, the class i customers in the queue will abandon from the queueing system. Note that the predetermined order in the input sequence is often specified by the sampling rule in the R&S literature. However, for simplicity, in this paper we consider only the round-robin order (i.e., a repeated order of all survived systems) as the predetermined order and leave the discussion of various sampling rules as future research.

It is worthwhile noting that the input sequence is always the same as the out sequence for a fully

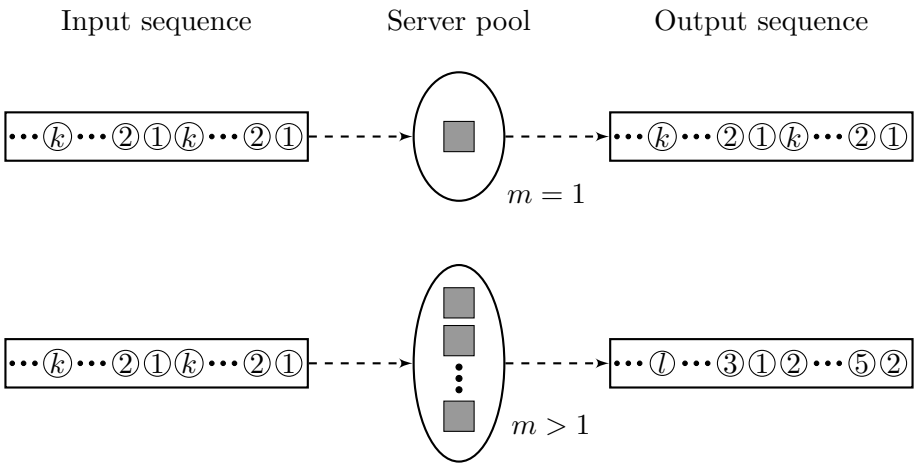

Figure 2: An illustration using queueing models

sequential procedure implemented on a single processor. However, the input and output sequences tends to be different when executing a fully sequential procedure using multiple processors. Taking the KN procedure for example, the input sequence of all samples are in the round-robin order, which is the same as the output sequence using a single processor but could be different of the output sequence using multiple processors (see Figure 2 for an intuitive illustration). Moreover, the output sequence for the multiple-processor configuration may not keep the property of a repeated order.

Since that the output sequence may be diverse from the input sequence, in addition to $X_{i\ell}$, we need to define Y_{ij} as the jth observation from alternative i to complete in the output sequence. Thus if the ℓth replication from alternative i in input sequence is the jth overall replication from alternative i to complete, then $X_{i\ell} = Y_{ij}$, where ℓ is not necessarily equal to j. Let $\Gamma_{i\ell}$ denote the (random) amount of time it takes to run the ℓth replication $X_{i\ell}$ from alternative i, which is denoted as the service time of the ℓth customer of class i. We assume that the service time $\Gamma_{i\ell} > 0$ a.s. and it has finite mean $\gamma_i > 0$. Then $\{(X_{i\ell}, \Gamma_{i\ell}), \ell \geq 1\}$ are i.i.d. random variables. However, $Y_{i\ell}$, $\ell = 1, 2, \ldots$, may be dependent on each other if $X_{i\ell}$ and $\Gamma_{i\ell}$ are correlated.

Like many queueing models, the random service times, $\Gamma_{i\ell}$ for all $i = 1, 2, \ldots, k$ and $\ell = 1, 2, \ldots$, lead to a random sample size of each alternative at any given time point. Let $D_i(t)$ denote the number of replications from alternative i that have been completed by time t (i.e., the number of departures of class i customers). Then, for any given time $t > 0$, $D_i(t)$ and $D_j(t)$ are not only random but also dependent for any $i \neq j$, and the randomness and dependence occurs even for a single-sever queue. However, none of existing R&S procedures discuss about it because that they consider only the time points when a new observation is obtained. For instance, the KN procedure is only interested in the time points when the sample size of each survived alternative is

increased by 1. Then, the sample sizes of all survived alternatives are deterministic at these time points. Considering the "virtual time" created by sample sizes rather than the actual time t is very straightforward for any sequential procedures using a single processor because that the comparison and elimination decision are made only once new samples are obtained.

When there are multiple processors, we still care only about the particular time points when new observations are obtained. However, the situation is a bit more complicated since the sample sizes of all survived are random at these time points. For instance, we simulate 4 alternatives with 8 processors where the time for generating one replication from alternatives i follows an exponential distribution with mean i unit-times. Figure 3 depicts the ratio of the sample sizes between alternative i, $i = 1, 2, 3$, and alternative 4 when the sample size of alternative 4 is increased by one based on one sample path. From the Figure we find that the output sequence is random but the ratio of the sample sizes between any pair of two alternatives approximates to the ratio of that specified in the input sequence as the number of samples increases.

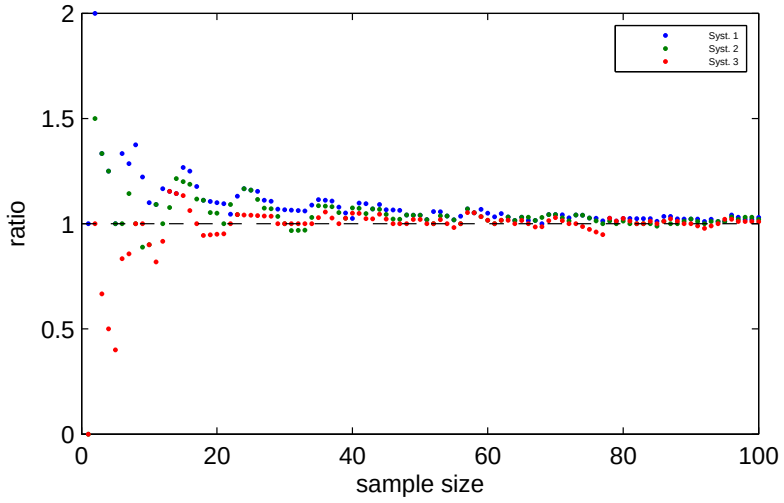

Figure 3: Ratio of the sample sizes between alternative i, $i = 1, 2, 3$, and alternative 4 when the number of processors is $m = 8$ and the number of alternatives is $k = 4$.

If random sample sizes were the only problem, then it would be easy to handle, e.g., using only part of the observations to make the sample sizes the same as required. However, the statistical issues caused by the parallel simulation scheme are more critical, which are discussed in the following subsections.

2.2 Biased Estimators

If $X_{i\ell}$ and $\Gamma_{i\ell}$, $\ell = 1, 2, \ldots$, are correlated, then the sample mean estimator

$$\bar{Y}_i(n) = \frac{1}{n}\sum_{\ell=1}^{n} Y_{i\ell} \tag{2.1}$$

tends to be biased. Taking a simple example in Heidelberger (1988) for illustration, suppose there is only one alternative to be simulated and $X_{1\ell} = \Gamma_{1\ell}$ follows an exponential distribution with mean μ_1. Then the first observation from the output sequence $Y_{11} = \min\{X_{11}, X_{12}, \ldots, X_{1m}\}$ is exponentially distributed with mean μ_1/m, i.e.,

$$\mathrm{E}[Y_{11}] = \frac{\mu_1}{m} \neq \mu_1 = \mathrm{E}[X_{11}].$$

By the Markovian property, we can further derive that

$$\mathrm{E}[Y_{1i}] = \mu_1 - \mu_1\left(1 - \frac{1}{m}\right)^i,$$

and

$$\sum_{\ell=1}^{n} \mathrm{E}[Y_{1i}] = n\mu_1 - \mu_1(m-1)\left[1 - \left(1 - \frac{1}{m}\right)^n\right]. \tag{2.2}$$

Then $\bar{Y}_1(n)$ is a downward biased estimator, but $\lim_{n\to\infty} \bar{Y}_1(n) = \mu_1$ is asymptotically unbiased. This is a classic biased sampling problem and the sampling bias is caused by the way in which samples are collected in orders.

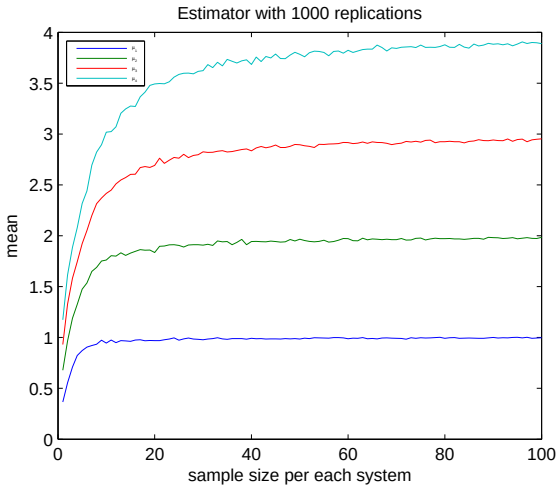

Figure 4: The sample mean estimators with 1000 sample paths when the number of processors is $m = 8$ and the number of alternatives is $k = 4$.

The kind of bias still exists when there are multiple alternatives, seeing Figure 4 for example. In Figure 4, we simulate 4 alternatives on 8 processors where $X_{i\ell} = \Gamma_{i\ell}$ follows an exponential

distribution with mean i time-units. The sample mean of system i is calculated based on Equation (2.1) where the sample size n varies from 1 to 100 for $i = 1, 2, 3, 4$. Note that the sample-mean estimator given by Equation (2.1) is downward biased with finite samples, but asymptotically unbiased as sample size goes to infinity.

2.3 Elimination Causing Dependence

If $X_{i\ell}$ is independent of $\Gamma_{i\ell}$, or even $\Gamma_{i\ell}$ is constant for all $i = 1, 2, \ldots, k$, then $\bar{Y}_i(n)$ become an unbiased estimator. However, the elimination decisions could introduce the dependence between sample sizes of survived systems, and thus the dependence between their sample means.

Consider another example that there are three systems to be simulated on two processors, where the service times of systems $1, 2, 3$ are fixed as $2, 1, 1$ time units respectively. Suppose that the input sequence is in the round-robin order, then the simulation process can be described as Figure 5. At each point t_n where a new observation from system 3 is obtained, we do the comparison with equal sample size for all systems. Suppose at some time point t_n, system 2 has been eliminated by either systems 1 or 3. Then at next time point t_{n+1} where a new observation from system 3 is obtained, the survived systems, 1 and 3 have unequal sample sizes. The elimination decision changes the sample size of some survived system at the following comparison time points.

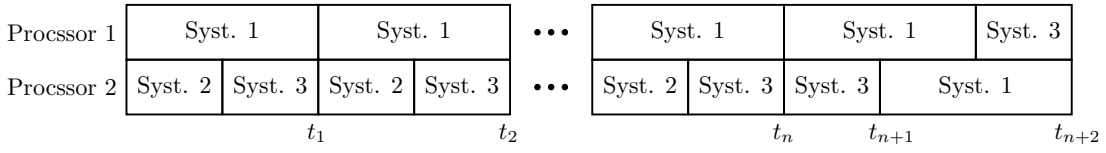

Figure 5: constant scheduling.

When there are a large amount of systems with random replication times simulated on many processors, the dynamics of the survived systems could be more complicated as more systems have been eliminated at these random time points, leading to the dependence between sample sizes and sample means of survived systems. Note that early elimination never causes any problem for a sequential procedure using a single processor because the output sequence of remaining systems remains the same as that without elimination. It is also worthwhile pointing out that the ratio of the sample sizes between any pair of two survived systems approximates to the ratio of them specified in the input sequence, as shown in Figure 5 and also in Figure 3.

Thus, to design a valid finite-time R&S procedure using parallel simulation schemes, we need to pre-specify the virtual time points to perform the pairwise comparisons. This is the idea behind finite-time procedures. However, the effect that the randomness and dependence "vanish" as sample sizes goes to infinity encourages us to investigate the asymptotic behavior of a R&S procedure that

can take all available samples into account at any elimination points.

3 Framework Construction and Procedure Design

To the best our knowledge, this is the first time that replication (service) times $\Gamma_{i\ell}$, $i = 1, 2, \ldots, k$ and $\ell = 1, 2, \ldots$, are considered in R&S literature. The random replication times and the elimination decision could introduce the randomness and dependence of samples of all systems in the output sequence, thus lead both the implementation and statistical issues of fully sequential R&S procedures. To deal with these issues, we propose two solutions in this section. If we restrict the comparisons by a pre-determined rule and the statistics formulated for comparison are based on the input sequence rather than the output sequence, then all existing sequential R&S procedures are valid to be implemented on multiple processors. To achieve this, we need to create a table to record the observations exactly in the order of the input sequence. We call this type of procedures with finite-time statistical validity *table filling procedures*. On the other hand, the asymptotic results shown in Section 2 indicates that it is possible to design asymptotically valid sequential procedures in which the statistics formulated for comparison are based on the output sequence. We call this type of procedures *parallel sequential procedures*. In the following we propose two types of statistics that are used for a table filling procedure and a parallel sequential procedure, respectively, in Section 3.2.

As pointed out by Kim and Nelson (2006a), many sequential procedures are based on the results for Brownian motion processes when comparing the means of two systems. Let $\{\mathcal{B}_\Delta(t) : t \geq 0\}$ denote a Brownian motion process with a constant drift Δ. Let $\bar{X}_i(n) = \frac{1}{n}\sum_{\ell=1}^{n} X_{i\ell}$ be the sample mean with sample size n for $i = 1, 2, \ldots, k$. For instance, when $\sigma_i^2 = \sigma_j^2 = 1$, Robbins and Siegmund (1974) define the statistic $\frac{mn}{m+n}\left[\bar{X}_i(m) - \bar{X}_j(n)\right]$ which has the same joint distribution with $\mathcal{B}_{\mu_i-\mu_j}\left(\frac{mn}{m+n}\right)$; when $m = n$, Kim and Nelson (2001) shows the statistic $n\left[\bar{X}_i(n) - \bar{X}_j(n)\right]/\sqrt{\sigma_i^2+\sigma_j^2}$ has the same joint distribution with $\mathcal{B}_{(\mu_i-\mu_j)/\sqrt{\sigma_i^2+\sigma_j^2}}(n)$; Hong (2006) provides a generalized approach allowing both unequal sample sizes and unequal variances by defining the statistic as

$$Z(m,n) = \left(\frac{\sigma_i^2}{m} + \frac{\sigma_j^2}{n}\right)^{-1}\left[\bar{X}_i(m) - \bar{X}_j(n)\right], \tag{3.1}$$

whose joint distribution is the same as $\mathcal{B}_{\mu_i-\mu_j}\left(\left(\frac{\sigma_i^2}{m} + \frac{\sigma_j^2}{n}\right)^{-1}\right)$. Note that all the three statistics require the assumption that $X_{i\ell}$ are normally distributed for $i = 1, 2, \ldots, k$, and $\ell = 1, 2, \ldots$. Since their procedures are implemented on a single processor, the output sequence is the same as the input sequence which is predetermined by a sampling rule. In this paper, since we only consider the

round-robin order, we narrow the design of a table filling procedure based on statistic in Equation (3.1) with $m = n$, which is essentially the KN procedure of Kim and Nelson (2001). Various table filling procedures with different sampling rules (e.g., the procedures in Hong (2006)) can be easily developed under parallel simulation scheme.

To use the information of all available samples, we may define a nature statistic in a similar form of Equation (3.1) as

$$Z_{ij}\left(D_i(t), D_j(t)\right) = \begin{cases} \left(\frac{\sigma_i^2}{D_i(t)} + \frac{\sigma_j^2}{D_j(t)}\right)^{-1} \left[\bar{Y}_i\left(D_i(t)\right) - \bar{Y}_j\left(D_j(t)\right)\right] & D_i(t) > 0 \text{ and } D_j(t) > 0, \\ 0 & D_i(t) = 0 \text{ or } D_j(t) = 0, \end{cases} \tag{3.2}$$

where $\bar{Y}_i\left(D_i(t)\right) = \frac{1}{D_i(t)} \sum_{\ell=1}^{D_i(t)} Y_{i\ell}$ for all $i = 1, 2, \ldots, k$. There are several differences between the statistics defined in Equations (3.1) and (3.2). First, the sequence $(D_i(t), D_j(t))$ is a counting process while (m, n) is any given pair of positive integers. Second, $Y_{i1}, Y_{i2}, \ldots$ for $i = 1, 2, \ldots, k$ could be dependent on each other while $X_{i1}, X_{i2}, \ldots$ are i.i.d. for $i = 1, 2, \ldots, k$. In other words, the statistic in Equation (3.1) has the nice properties of deterministic sample size and i.i.d. random sequence, which may not hold for the statistic in Equation (3.2). We next show how to "retrieve" these properties for Equation (3.2) in some meaningful asymptotic regime.

3.1 Brownian Motion Construction

Under the IZ formulation, it is often difficult for a R&S procedure to select the best system for the slippage configuration (SC), where the difference between the best and all other systems are all equal to the IZ parameter δ. In this section, we focus on the SC setting and construct a Brownian motion process assuming that σ_i^2 for all $i = 1, 2, \ldots, k$ are known.

Suppose that $\mu_1 = \delta$ and $\mu_i = 0$ for $i = 2, 3, \ldots, k$ under the SC setting. Then we are interested in $Z_{1j}(D_1(t), D_j(t))$, $j = 2, 3, \ldots, k$ until a random time when either system 1 or system j is eliminated. To design a sequential R&S procedure with asymptotic statistical guarantee, we want to "remove" the randomness and dependence of sample sizes of any pair of alternatives in the statistic of Equation (3.2) and link it to a Brownian motion process. Even the output sequence may be different from the input sequence, the random output sequence can be approximated by the predetermined input sequence.

Define $Q_i(t)$ denote the number of samples from alternative i that have been taken away from the queue by the time t. Note that given any time t, $Q_i(t)$ is still not deterministic. As mentioned earlier, we are interested in some particular time points rather than arbitrary time t. Define the particular time points $\{t_n\}$ as the end of the nth cycle of the input sequence. Then $Q_i(t_n) = n$ for any survived alternative i. For notational simplicity, we let $n = Q_i(t_n)$ and $D_i^n = D_i(t_n)$. Then, $0 \le n - D_i^n \le m$ for any $n \ge 1$ and $\frac{D_i^n}{n} \to 1$ a.s. as $n \to \infty$.

Before defining the Brownian motion process, we first define the meaningful limiting regime, where the IZ parameter $\delta \to 0$, which is summarized as the asymptotic regime of "PCS as $\delta \to 0$" in Kim and Nelson (2006a). The asymptotic PCS analysis is not new in literature, see Kim and Nelson (2006a) and the references therein for more discussion and Kim and Nelson (2006b) for one practical application on R&S for steady-state simulation. Generally speaking, the asymptotic validity of a procedure can approximately guarantee the desired probability of selecting the best even when the unknown differences are vanishingly small.

Let $N^\delta = \lceil \frac{2a\sigma^2_{\max}}{\delta^2} \rceil$ as the maximum number of observations from each alternative, where $\sigma^2_{\max} = \max\{2\sigma_i^2 : i = 1, 2, \ldots, k\}$ and $a = -\ln[2\alpha/(k-1)]$, and $\lceil x \rceil$ is the smallest integer not less than x. The superscript of N^δ means the maximum sample size is a function of the IZ parameter such that N^δ is driven to infinity as δ goes to zero. Let $s = n/N^\delta \in [0, 1]$, which is used to construct the time of the Brownian motion. Note that we view s as continuous element. Let $\mathcal{P}_i^n$ denote the set of index ℓ such that the ℓth replication $X_{i\ell}$ is processing (not obtained yet) at time t_n. Then,

$$\begin{aligned} \bar{Y}_i(D_i^n) &= \frac{n}{D_i^n} \cdot \frac{1}{n} \left[\sum_{\ell=1}^{n} X_{i\ell} - \sum_{\ell \in \mathcal{P}_i^n} X_{i\ell} \right] \\ &= \frac{n}{D_i^n} \cdot \frac{1}{N^\delta s} \sum_{\ell=1}^{N^\delta s} (X_{i\ell} - \mu_i) + \mu_i \frac{n}{D_i^n} - \frac{1}{D_i^n} \sum_{\ell \in \mathcal{P}_i^n} X_{i\ell}. \end{aligned} \tag{3.3}$$

For mathematical purpose, we also redefine the discrete process $Z_{1j}(\cdot, \cdot)$ in (3.2) as

$$Z_{1j}\left(D_1^n, D_j^n\right) = \begin{cases} \left[\frac{\sigma^2_{\max}}{N^\delta} \left(\frac{\sigma_1^2}{D_1^n} + \frac{\sigma_j^2}{D_j^n} \right)^{-1} \right] \left[\frac{\sqrt{N^\delta}}{\sigma_{\max}} \left(\bar{Y}_1\left(D_1^n\right) - \bar{Y}_j(D_j^n) \right) \right] & D_1^n > 0 \text{ and } D_j^n > 0, \\ 0 & D_1^n = 0 \text{ or } D_j^n = 0. \end{cases} \tag{3.4}$$

Note that $Z_{1j}(\cdot, \cdot) = 0$ when $D_1^n = 0$ or $D_j^n = 0$ is defined for the purpose that limiting process of $Z_{1j}(\cdot, \cdot)$, a Brownian motion process, always starts from the original point, making the Brownian motion well-defined at time 0. The term $\frac{\sigma^2_{\max}}{N^\delta}$ in the first bracket is to scale the time into the interval $[0, 1]$, and the term $\frac{\sqrt{N^\delta}}{\sigma_{\max}}$ is for using a functional central limit theorem (Donsker's Theorem 14.1 in Billingsley (1999)), which will be shown later. Since neither system will be eliminated without any observations, in the rest of the paper we only focus on the situation where all $D_i^n > 0$. Let $\mathbf{D}[0, 1]$ be the Skorohod space of all right-continuous real-valued functions on $[0, 1]$ with limits from the left everywhere, endowed with the Skorohod topology (see Billingsley (1999)). Then we further write $Z_{1j}(\cdot, \cdot)$ in (3.4) as an element of $\mathbf{D}[0, 1]$ such that

$$Z_{1j}\left(t_{1j}\right) = t_{1j} \cdot \frac{\sigma_1^2/n + \sigma_j^2/n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \cdot \left[\frac{\sqrt{N^\delta}}{\sigma_{\max}} \left(\bar{Y}_1\left(D_1^n\right) - \bar{Y}_j(D_j^n) \right) \right], \tag{3.5}$$

where t_{1j} is defined by

$$t_{1j} = \sigma^2_{\max} \left(\frac{\sigma_1^2}{s} + \frac{\sigma_j^2}{s} \right)^{-1}. \tag{3.6}$$

We are now ready to state the main result of constructing Brownian motion processes.

Theorem 3.1 (Convergence to A Brownian Motion Process)**.** *Let $Z_{1j}(\cdot)$ be an element of the Skorohod space $\mathbf{D}[0,1]$ defined by* (3.5)*. Then,*

$$Z_{1j}\left(t_{1j}\right) \Rightarrow \mathcal{B}_{\Delta}(t_{1j}), \quad \text{as } \delta \to 0, \tag{3.7}$$

where $\mathcal{B}_{\Delta}(\cdot)$ is a Brownian motion process with constant drift $\Delta = \sqrt{2a}$.

Proof. This proof relies on Converging Together Lemma (Theorem 11.4.5 in Whitt (2002)) and Donsker's Theorem (Theorem 14.1 in Billingsley (1999)). We break the right-hand-side of $Z_{1j}(\cdot)$ in (3.5) into several parts, and show the convergence of them one by one. First, as $\delta \to 0$, then $N^{\delta} \to \infty$, $n \to \infty$, and $D_i^n \to \infty$ with $\frac{D_i^n}{n} \to 1$ a.s. for all $i = 1, 2, \ldots, k$. Then,

$$\frac{\sigma_1^2/n + \sigma_j^2/n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \leq \frac{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \cdot \max\left\{\frac{D_1^n}{n}, \frac{D_j^n}{n}\right\} \to 1 \quad a.s.$$

and

$$\frac{\sigma_1^2/n + \sigma_j^2/n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \geq \frac{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \cdot \min\left\{\frac{D_1^n}{n}, \frac{D_j^n}{n}\right\} \to 1 \quad a.s.$$

imply that

$$\frac{\sigma_1^2/n + \sigma_j^2/n}{\sigma_1^2/D_1^n + \sigma_j^2/D_j^n} \to 1 \quad a.s. \quad \text{as } \delta \to 0.$$

By Equation (3.3),

$$\begin{aligned}\frac{\sqrt{N^{\delta}}}{\sigma_{\max}}\left[\bar{Y}_1\left(D_1^n\right) - \bar{Y}_j(D_j^n)\right] = {} & \frac{n}{D_1^n} \cdot \frac{\sigma_1}{\sigma_{\max} s} \cdot \frac{\sum_{\ell=1}^{N^{\delta}s}\left(X_{1\ell} - \mu_1\right)}{\sigma_1\sqrt{N^{\delta}}} - \frac{n}{D_j^n} \cdot \frac{\sigma_j}{\sigma_{\max} s} \cdot \frac{\sum_{\ell=1}^{N^{\delta}s}\left(X_{j\ell} - \mu_j\right)}{\sigma_j\sqrt{N^{\delta}}} \\ & + \left[\frac{\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \frac{n}{D_1^n} \cdot \mu_1 - \frac{\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \frac{n}{D_j^n} \cdot \mu_j\right] - \frac{1}{D_1^n}\sum_{\ell \in \mathcal{P}_1^n} X_{k\ell} + \frac{1}{D_j^n}\sum_{\ell \in \mathcal{P}_j^n} X_{j\ell}.\end{aligned}$$

For any $i = 1, 2, \ldots, k$, by Donsker's Theorem and Converging Together Lemma, as $\delta \to 0$,

$$\frac{n}{D_i^n} \cdot \frac{\sigma_i}{\sigma_{\max} s} \cdot \frac{\sum_{\ell=1}^{N^{\delta}s}\left(X_{i\ell} - \mu_i\right)}{\sigma_i\sqrt{N^{\delta}}} \Rightarrow 1 \cdot \frac{\sigma_i}{\sigma_{\max} s} \cdot \mathcal{B}^i\left(s\right),$$

where $\mathcal{B}^i(\cdot)$ is a standard Brownian motion process. Because that

$$\begin{aligned}\frac{\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \frac{n}{D_1^n} \cdot \mu_1 - \frac{\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \frac{n}{D_j^n} \cdot \mu_j &\leq \frac{\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \left(\mu_1 - \mu_j\right) \cdot \max\left\{\frac{n}{D_1^n}, \frac{n}{D_j^n}\right\} \\ &= \frac{\delta\sqrt{N^{\delta}}}{\sigma_{\max}} \cdot \max\left\{\frac{n}{D_1^n}, \frac{n}{D_j^n}\right\} \to \sqrt{2a}\end{aligned}$$

and

$$\frac{\sqrt{N^\delta}}{\sigma_{\max}} \cdot \frac{n}{D_1^n} \cdot \mu_1 - \frac{\sqrt{N^\delta}}{\sigma_{\max}} \cdot \frac{n}{D_j^n} \cdot \mu_j \quad \geq \quad \frac{\delta\sqrt{N^\delta}}{\sigma_{\max}} \cdot \min\left\{ \frac{n}{D_1^n}, \frac{n}{D_j^n} \right\} \to \sqrt{2a},$$

we have $\left[\frac{\sqrt{N^\delta}}{\sigma_{\max}} \cdot \frac{n}{D_1^n} \cdot \mu_1 - \frac{\sqrt{N^\delta}}{\sigma_{\max}} \cdot \frac{n}{D_j^n} \cdot \mu_j\right] \to \sqrt{2a}$ as $\delta \to 0$. By definition of $O_{\mathbb{P}}(1)$, for any $\epsilon > 0$, there exists $c = m\mu_i/\epsilon$, such that

$$\mathbb{P}\left\{ \left| \sum_{\ell \in \mathcal{P}_i^n} X_{i\ell} \right| > c \right\} \leq \mathrm{E}\left[\left| \sum_{\ell \in \mathcal{P}_i^n} X_{i\ell} \right| \right] / c \leq m\mathrm{E}\left[|X_{i\ell}| \right] / c = \epsilon,$$

which means $\sum_{\ell \in \mathcal{P}_i^n} X_{i\ell} = O_{\mathbb{P}}(1)$ for any $i = 1, 2, \ldots, k$. Then,

$$\frac{1}{D_i^n} \sum_{\ell \in \mathcal{P}_i^n} X_{i\ell} \to 0 \quad \text{ in probability.}$$

Using Converging Together Lemma again,

$$\begin{aligned} Z_{1j}\left(t_{1j}\right) \quad &\Rightarrow \quad t_{1j} \cdot \left[\frac{\sigma_1}{\sigma_{\max} s} \cdot \mathcal{B}^1\left(s\right) - \frac{\sigma_j}{\sigma_{\max} s} \cdot \mathcal{B}^j\left(s\right) - \sqrt{2a} \right] \\ &\triangleq \quad \mathcal{B}_{\sqrt{2a}}\left(\sigma_{\max}^2 \left(\frac{\sigma_1^2}{s} + \frac{\sigma_j^2}{s} \right)^{-1} \right) = \mathcal{B}_\Delta\left(t_{1j}\right) \end{aligned} \tag{3.8}$$

where (3.8) is due to independence of two standard Brownian motion processes, $\mathcal{B}^1(\cdot)$ and $\mathcal{B}^j(\cdot)$, and the definition of a Brownian motion with drift Δ, $\mathcal{B}_\Delta(\cdot)$. That concludes the proof. □

Remark 3.1. *There is a critical assumption for deriving this result that the number of servers m is finite (fixed in our model) and the running time t_n is driven to infinity to make the sample size N^δ to infinity. Then the scale of time in Equation* (3.5) *to construct the Brownian motion can be viewed as the* ***conventional Heavy-Traffic*** *regime in queueing literature (see, for instance, Whitt (2002)). However, the expression of the Brownian motion here is different a lot of the Brownian motions in queueing models because the proposed Brownian motion is typically used to solve R&S problems. The problem could be harder if we fixed the running time but let number of servers goes to infinity (known as the* ***Halfin-Whitt Heavy-Traffic*** *regime in queueing literature). Interested readers may be referred to Heidelberger (1988) and Glynn and Heidelberger (1991) for the discussion of the difficulties for point estimation problems.*

3.2 The Procedures

As we mentioned before, a finite-time procedure is possible if we perform comparisons exactly according to the input sequence. In this section, we first provide a finite-time procedure, *table filling procedure*, without showing its statistical validity, and then we design an asymptotic procedure

called *parallel sequential procedure*. The table filling procedure is essentially the same as the KN procedure. Hence, to ensure its validity, we assume that each system is normally distributed, which is not necessary for the parallel sequential procedure since that FCLT indicates approximated normality of sample mean as sample size is large.

Replication	Syst. 1	Syst. 2	...	Syst. k
1	X_{11}	X_{21}	...	X_{k1}
2	X_{12}	X_{22}	...	X_{k2}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r	X_{1r}	X_{2r}	...	X_{kr}
$r+1$	$X_{1,r+1}$	$X_{2,r+1}$	...	$X_{k,r+1}$
$r+2$	$X_{1,r+2}$	$X_{2,r+2}$	...	$X_{k,r+2}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 1: Sample Storage Table for TFP.

Procedure 1 (Table Filling Procedure)**.**

Step 0. **Setup:** Select confidence level $1/k < 1-\alpha < 1$, IZ parameter $\delta > 0$, and the first-stage sample size $n_0 \geq 2$. Let $h^2 = (n_0 - 1)\left[\left(\frac{2\alpha}{k-1}\right)^{-2/(n_0-1)} - 1\right]$.

Step 1. **Initialization:** Let $I = \{1, 2, \ldots, k\}$ be the set of alternatives still in contention. All replications from all systems in I will be processed by m processors in a round-robin way. The ℓth replication from system i is denoted as $X_{i\ell}$. Let $n_i = \max\{n \geq 0 : X_{i\ell} \text{ is obtained for all } \ell \leq n\}$. Note that $n_i = 0$ for all $i \in I$ at the beginning. Let $I_r = \emptyset$ be the set of systems that is ready for comparison.

Step 2. **Variance Estimation:** If $n_j \geq n_0$ and $j \in I \setminus I_r$[2], then do the following things:

a. Record the new observation $X_{j\ell}$ and add j into I_r;

b. Calculate $\sum_{\ell=1}^{n_0} X_{j\ell}$ and $\sum_{\ell=1}^{n_0} X_{j\ell}^2$, and record the triple $(n_0, \sum_{\ell=1}^{n_0} X_{j\ell}, \sum_{\ell=1}^{n_0} X_{j\ell}^2)$;

c. Compute first-stage sample variance of system j.

$$S_j^2 = \frac{1}{n_0 - 1}\left[\sum_{\ell=1}^{n_0} X_{j\ell}^2 - \frac{1}{n_0}\left(\sum_{\ell=1}^{n_0} X_{j\ell}\right)^2\right],$$

Otherwise, just record the new observation $X_{j\ell}$.

[2] $A \setminus B = \{x : x \in A \text{ and } x \notin B\}$.

Step 3. **Checking Status:** If $I \setminus I_r = \emptyset$, then set $r = n_0$ and $I_r = \emptyset$. Otherwise, wait for next new observation from any system in I, saying $X_{j\ell}$, update n_j and go to Step 2.

Step 4. **Elimination:** Set $I^{\text{old}} = I$. Let

$$I = I^{\text{old}} \setminus \left\{ i \in I^{\text{old}} : \sum_{\ell=1}^{r} X_{i\ell} - \sum_{\ell=1}^{r} X_{j\ell} < \min\left\{0, -\frac{h^2}{2\delta}\left(S_i^2 + S_j^2\right) + \frac{\delta}{2} r\right\} \text{ for some } j \in I^{\text{old}}, j \neq i \right\}.$$

Step 5. **Stopping Rule:** If $|I| = 1$, stops all processors and selects the system whose index is in I as the best. Otherwise, check whether it is ready for next comparison.

(I). For any $j \in I \setminus I_r$, if $n_j \geq r + 1$, then add j into I_r. If $I \setminus I_r = \emptyset$, then set $I_r = \emptyset$ and $r = r + 1$, and go to Step 4;

(II). Otherwise, wait for a new observation from any system j in I, record it and update n_j, and go to (I).

Remark 3.2. *The table filling procedure is designed to implement the KN procedure on parallel simulation environments. Since we do not consider CRN in this paper, in table filling procedure we can compute the sample variance of each system rather than the sample variance of the difference between any two systems in KN procedure. The procedure can be further simplified when $X_{i\ell}$ and $\Gamma_{i\ell}$ are independent. In that case, $Y_{i\ell}$ is unbiased for all $i = 1, 2, \ldots, k$ and $\ell = 1, 2, \ldots$, then we can use $Y_{i\ell}$ instead of $X_{i\ell}$ and let n_i be the number of available samples in the procedure.*

The key idea behind the table filling procedure is to do comparison exactly according to the input sequence. To achieve this goal, we need to create a table to record all observations in the order of the input sequence, which may additional space to storage these information. In fact, using that table, we are able to implement all existing sequential procedures in the parallel simulation scheme.

We next present an asymptotic procedure which can use all available samples for comparison.

Procedure 2 (Parallel Sequential Procedure)**.**

Step 0. **Setup:** Select confidence level $1/k < 1 - \alpha < 1$, IZ parameter $\delta > 0$, and the first-stage sample size $n_0 \geq 2$. Let $a = -\ln[2\alpha/(k-1)]$.

Step 1. **Initialization:** Let $I = \{1, 2, \ldots, k\}$ be the set of alternatives still in contention. All replications will be processed by m processors in a round-Robin way. The ℓth completed observation from system i is denoted as $Y_{i\ell}$ and the number of overall completed observation from that system is denoted as D_i. Then record the following triple $(D_i, \sum_{\ell=1}^{D_i} Y_{i\ell}, \sum_{\ell=1}^{D_i} Y_{i\ell}^2)$ for system i, $i = 1, 2, \ldots, k$. Note that $(D_i, \sum_{\ell=1}^{D_i} Y_{i\ell}, \sum_{\ell=1}^{D_i} Y_{i\ell}^2) = (0, 0, 0)$ for all $i \in I$ at the beginning. Let $I_r = \emptyset$ be the set of systems that is ready for comparison.

Step 2. **Checking Status:** Check the status of system j:

a. If $D_j \geq n_0$ and $j \in I \backslash I_r$, then add j into I_r, compute sample mean $\bar{Y}_j(D_j) = \frac{1}{D_j}\sum_{\ell=1}^{D_j} Y_{j\ell}$ and sample variance

$$S_j^2(D_j) = \frac{1}{D_j - 1}\left[\sum_{\ell=1}^{D_j} X_{j\ell}^2 - \frac{1}{D_j}\left(\sum_{\ell=1}^{D_j} X_{j\ell}\right)^2\right],$$

and go to Step 3;

b. If $j \in I_r$, then compute sample mean $\bar{Y}_j(D_j) = \frac{1}{D_j}\sum_{\ell=1}^{D_j} Y_{j\ell}$ and sample variance $S_j^2(D_j) = \frac{1}{D_j-1}\left[\sum_{\ell=1}^{D_j} X_{j\ell}^2 - \frac{1}{D_j}\left(\sum_{\ell=1}^{D_j} X_{j\ell}\right)^2\right]$, and go to Step 3;

c. If $D_j \leq n_0$, then go to Step 4.

Step 3. **Elimination:** When $|I_r| \geq 2$, set $I^{\text{old}} = I_r$. Let

$$I_r = I^{\text{old}} \setminus \left\{ i \in I^{\text{old}} : i \neq j \text{ and } \bar{Y}_i(D_i) - \bar{Y}_j(D_j) < \min\left\{0, -\frac{a}{\delta}\left[\frac{S_i^2(D_i)}{D_i} + \frac{S_j^2(D_j)}{D_j}\right] + \frac{\delta}{2}\right\}\right\}$$
$$\setminus \left\{ j : \bar{Y}_j(D_j) - \bar{Y}_i(D_i) < \min\left\{0, -\frac{a}{\delta}\left[\frac{S_i^2(D_i)}{D_i} + \frac{S_j^2(D_j)}{D_j}\right] + \frac{\delta}{2}\right\} \text{ for some } i \in I^{\text{old}}, i \neq j \right\}.$$

Set $I = I \setminus \left(I^{\text{old}} \setminus I_r\right)$.

Step 4. **Stopping Rule:** If $|I| = 1$, stops all processors and selects the system whose index is in I as the best. Otherwise, wait for a new observation from any system in I, saying system j, is obtained, update the triple

$$(D_j, \sum_{\ell=1}^{D_j} Y_{j\ell}, \sum_{\ell=1}^{D_j} Y_{j\ell}^2) = (D_j + 1, \sum_{\ell=1}^{D_j} Y_{j\ell} + Y_{jD_j}, \sum_{\ell=1}^{D_j} Y_{j\ell}^2 + Y_{jD_j}^2),$$

and go to Step 2.

Remark 3.3. *The first-stage sample size n_0 is often set to $10-30$ to make the estimation accurate.*

Compared to table filling procedure, there are several advantages of parallel sequential procedure. First, parallel sequential procedure uses the information of every observation more efficiently, especially when simulation effort of generating each observation is quite expensive. Second, parallel sequential procedure requires less space to store the data. Third, we do not require normality assumption of the output performance and we can update the sample variances as long as sample sizes increase. We end this section by stating the theorem.

Theorem 3.2. *Let $X_{i\ell}$ denote the performance output of the ℓth replication from system i with finite mean $\mu_i = \mathrm{E}[X_{i\ell}]$. Under the IZ formulation, we assume that $\mu_1 - \delta \geq \mu_2 \geq \ldots \geq \mu_k$. Then, as the IZ parameter $\delta \to 0$, the parallel sequential procedure selects system 1 as the best system with probability at least $1 - \alpha$.*

3.3 The Asymptotic Statistical Validity

The asymptotic validity of parallel sequential procedure is based on Theorem 3.1. For clear presentation, we first restrict our discussion on the slippage configuration, i.e., $\mu_1 - \delta = \mu_2 = \ldots = \mu_k$. To use the statistic $Z_{1j}(t_{1j})$ defined in (3.5) to handle the unknown variance case, we redefine $Z_{1j}(t_{1j})$ as

$$Z_{1j}(t_{1j}) = t_{1j} \cdot \frac{\sigma_1^2/n + \sigma_j^2/n}{S_1^2(D_1)/D_1 + S_j^2(D_j)/D_j} \cdot \left[\frac{\sqrt{N}}{\sigma_{\max}} \left(\bar{Y}_1(D_1) - \bar{Y}_j(D_j) \right) \right], \tag{3.9}$$

where t_{1j} is defined by (3.6). We omit the superscripts δ of N^δ and n of D_i^n to make a clean presentation. As $n \to \infty$, $S_i^2(D_i) \to \sigma_i^2$ w.p.1 for all $i = 1, 2, \ldots, k$, then

$$\frac{\sigma_1^2/n + \sigma_j^2/n}{S_1^2(D_1)/D_1 + S_j^2(D_j)/D_j} \to 1 \quad \text{w.p.1.} \tag{3.10}$$

Therefore, the conclusion for Theorem 3.1 still holds for $Z_{1j}(\cdot)$ defined by (3.9). Note that we use a deterministic time t_{1j} rather than a random time $\tau_{1j} = \left[\frac{S_1^2(D_1)}{D_1} + \frac{S_j^2(D_j)}{D_j} \right]$ in order to "move" the randomness to the boundary of the continuation region C^δ which will be defined soon.

Let $U_{1j}^\delta(t) = \max\left\{0, \frac{a\sigma_{\max}}{\delta\sqrt{N}} - \frac{\delta\sqrt{N}}{2\sigma_{\max}} \cdot \frac{\sigma_1^2/n + \sigma_j^2/Q_j}{\sigma_1^2/D_1 + \sigma_j^2/D_j} \cdot t\right\}$. The symmetric continuation region C_{1j}^δ for $Z_{1j}(\cdot)$ is formed by the upper boundary $U_{1j}^\delta(t)$ and lower boundary $-U_{1j}^\delta(t)$. Note that the continuation region C_{1j}^δ is not a triangular region (seeing Figure 6). However, by (3.10), it is easy to show that

$$U_{1j}^\delta(t) \to U(t) = \max\left\{0, \frac{a}{\Delta} - \frac{\Delta}{2} \cdot t\right\}, \quad \text{w.p.1 as } \delta \to 0,$$

where $\Delta = \sqrt{2a}$. Then, the symmetric triangular continuation region C for $\mathcal{B}_\Delta(\cdot)$, the limiting process of $Z_{1j}(\cdot)$ defined by (3.8), is formed by $U(t)$ and $-U(t)$.

Define the stopping time T_{1j}^δ that $Z_{1j}(\cdot)$ first exits the continuation region C^δ as

$$T_{1j}^\delta = \inf\left\{ t_{1j} : |Z_{1j}(t_{1j})| \geq U_{1j}^\delta(t_{1j}) \right\}, \tag{3.11}$$

and the stopping time T_{1j} that $\mathcal{B}_\Delta(\cdot)$ first exits the triangular region as

$$T_{1j} = \inf\left\{ t_{1j} : |\mathcal{B}_\Delta(t_{1j})| \geq U(t_{1j}) \right\}. \tag{3.12}$$

Theorem 3.1 establishes the weak convergence of $Z_{1j}(\cdot)$ to $\mathcal{B}_\Delta(\cdot)$ at every time point. To bound the probability of incorrect selection, we need a stronger result that ensure the value at stopping time $Z_{1j}(T_{1j}^\delta)$ can be approximated by $\mathcal{B}_\Delta(T_{1j})$, which is summarized as follows.

Theorem 3.3 (Convergence to A Brownian Motion Process at the Stopping Time)**.** *Let $Z_{1j}(t_{1j}^\delta)$ defined in* (3.9) *be an element of the Skorohod space $\mathbf{D}[0,1]$, where t_{1j}^δ is defined in* (3.6)*. Then,*

$$Z_{1j}(\cdot) \Rightarrow \mathcal{B}_\Delta(\cdot), \quad \text{as } \delta \to 0, \tag{3.13}$$

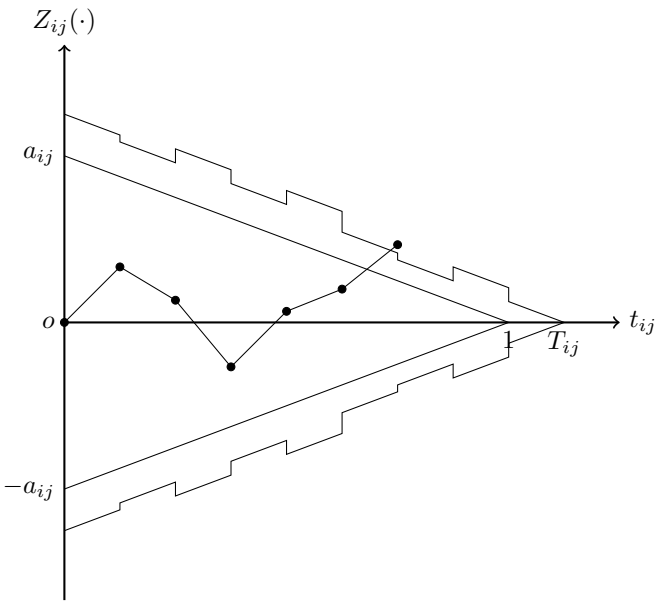

Figure 6: Continuation region C^δ.

where $\mathcal{B}_\Delta(\cdot)$ *is a Brownian motion process with constant drift* $\Delta = \sqrt{2a}$. *Moreover,* T_{1j}^δ *in* (3.11) *and* T_{1j} *in* (3.12) *are the stopping times that* $Z_{1j}(\cdot)$ *and* $\mathcal{B}_\Delta(\cdot)$ *first exit their own continuation regions, respectively. Then,*

$$Z_{1j}(T_{1j}^\delta) \Rightarrow \mathcal{B}_\Delta(T_{1j}), \quad \text{as } \delta \to 0. \tag{3.14}$$

Remark 3.4. *Note that* (3.13) *has already been proved in Theorem 3.1. The key idea of proving* (3.14) *is exactly the same as Proposition 3.2 of Kim et al. (2005). We summarize the sketch of proof in Appendix A.*

We also need the following lemma to prove the validity of the procedure.

Lemma 3.1 (Fabian (1974))**.** *For a fixed triangular continuation region* C *defined by* $U(t) = \max\{0, A - Bt\}$ *and* $-U(t)$ *, if* $B = \Delta/2$ *and* $\Delta > 0$*, then*

$$\mathbb{P}[\mathcal{B}_\Delta(T) < 0] = \frac{1}{2} e^{-A\Delta}, \tag{3.15}$$

where $T = \inf\{t > 0, \mathcal{B}_\Delta(t) \notin C\}$*, the random time that* $\mathcal{B}_\Delta(t)$ *first exits* C.

Proof of Theorem 3.2. We start from the slippage configuration.

$$\begin{aligned}
& \lim_{\delta\to 0}\mathbb{P}\left\{\text{selecting system 1}\right\} \\
= & \lim_{\delta\to 0}\left[1-\mathbb{P}\left\{\bigcup_{j=1}^{k-1}\left\{\text{system } j \text{ eliminates system 1}\right\}\right\}\right] \\
\geq & 1-\lim_{\delta\to 0}\sum_{j=1}^{k-1}\mathbb{P}\left\{\text{system } j \text{ eliminates system 1}\right\}, \qquad (3.16)
\end{aligned}$$

(3.16) is due to Bonferroni inequality.

Next, we need only to compute the limiting probability of system 1 is eliminated by system j. Conditioning on the stopping time T_{1j}^{δ},

$$\begin{aligned}
& \lim_{\delta\to 0}\mathbb{P}\left\{\text{system } j \text{ eliminates system 1}\right\} \\
= & \lim_{\delta\to 0}\mathbb{P}\left\{Z_{1j}\left(T_{1j}^{\delta}\right)\leq -U_{1j}^{\delta}\left(T_{1j}^{\delta}\right)\right\} \\
= & \lim_{\delta\to 0}\mathbb{P}\left\{Z_{1j}\left(T_{1j}^{\delta}\right)\leq 0\right\} \qquad (3.17) \\
= & \mathbb{P}\left\{\mathcal{B}_{\Delta}\left(T_{1j}\right)\leq 0\right\} \qquad (3.18) \\
= & \frac{1}{2}e^{-\frac{a}{\Delta}\Delta}=\frac{\alpha}{k-1} \qquad (3.19)
\end{aligned}$$

(3.17) is the event system j eliminates system 1 since $Z_{1j}(\cdot)$ exits the continuation region through the lower boundary, (3.18) is due to Theorem 3.3, (3.19) is due Lemma 3.1 and the definition of a. Plugging (3.19) into (3.16), yields

$$\lim_{\delta\to 0}\mathbb{P}\left\{\text{selecting system 1}\right\}=1-\sum_{j=1}^{k-1}\frac{\alpha}{k-1}=1-\alpha. \qquad (3.20)$$

For general cases under IZ formulation that $\mu_1-\delta\geq\mu_2\geq\ldots\geq\mu_k$, $Z_{1j}(\cdot)$ defined in (3.9) does not converge in distribution to a Brownian motion process with positive drift Δ any more. However, we can define

$$V_{1j}\left(t_{1j}\right)=t_{1j}\cdot\frac{\sigma_1^2/n+\sigma_j^2/n}{S_1^2(D_1)/D_1+S_j^2(D_j)/D_j}\cdot\left[\frac{\sqrt{N}}{\sigma_{\max}}\left(\bar{Y}_1\left(D_1\right)-\bar{Y}_j(D_j)\right)-\frac{\sqrt{N}}{\sigma_{\max}}\left(\mu_1-\mu_j-\delta\right)\right].$$

Then,

$$V_{1j}\left(t_{1j}\right)\leq Z_{1j}\left(t_{1j}\right), \quad a.s. \qquad (3.21)$$

and

$$V_{1j}\left(t_{1j}\right)\Rightarrow\mathcal{B}_{\Delta}(t_{1j}), \quad \text{as } \delta\to 0. \qquad (3.22)$$

Define $T_{1j}^{\delta,Y} = \inf\left\{t_{1j} : |Y_{1j}(t_{1j})| \geq U_{1j}^{\delta}(t_{1j})\right\}$. Conditioning on T_{1j}^{δ},

$$\begin{aligned}
& \lim_{\delta\to 0}\mathbb{P}\left\{\text{system } j \text{ eliminates system } 1\right\} \\
= & \lim_{\delta\to 0}\mathbb{P}\left\{Z_{1j}\left(T_{1j}^{\delta}\right) \leq -U_{1j}^{\delta}\left(T_{1j}^{\delta}\right)\right\} \\
\leq & \lim_{\delta\to 0}\mathbb{P}\left\{V_{1j}\left(T_{1j}^{\delta,Y}\right) \leq -U_{1j}^{\delta}\left(T_{1j}^{\delta,Y}\right)\right\} \qquad (3.23) \\
= & \lim_{\delta\to 0}\mathbb{P}\left\{V_{1j}\left(T_{1j}^{\delta,Y}\right) \leq 0\right\} \\
= & \mathbb{P}\left\{\mathcal{B}_{\Delta}\left(T_{1j}\right) \leq 0\right\} \\
= & \frac{1}{2}e^{-a} = \alpha/(k-1),
\end{aligned}$$

(3.23) is due to (3.21) which means that given that $Z_{1j}(\cdot)$ exits the continuation region from below, then $Y_{1j}(\cdot)$ must exit the continuation region from below at an earlier time. □

4 Numerical Experiments

Most of the test experiments are conducted on a server with 48 cores, which is installed with Unbutu 10.04 system, a Linux-based operating system. (Jun: say more about the server configuration.) To demonstrate the procedure can be easily implemented on various parallel computing environments, we test some examples on both a quad-core desktop and a cluster of two identical servers.

To test both the statistical validity and efficiency of PSP, we consider serveral configurations of experiments involving different means μ_i and different variances σ_i^2 for $i = 1, 2, \ldots, k$. As mentioned earlier, the replication time of generating one sample from system i, γ_i, for $i = 1, 2, \ldots, k$, also become an important factor in parallel simulation environments. Multi-stage procedures, e.g., the Rinott's procedure, can be easily parallelizable, but may lead to low efficiency. Fully sequential procedures, e.g., the KN procedure, can also be implemented by the way of FTP. Then, we compare PSP with both the Rinott's procedure and the KN procedure to emphasize the advantage of PSP in many ways. In order to make both the Rinott's procedure and the KN procedure valid, we assume the performance output of all systems are normally distributed without additional specification.

4.1 Validity and Efficiency Testing

Since our procedures are designed for large scale R&S problems, the number of systems in each experiment varies over $k = 10^4, 10^5, 10^6$. The first-stage sample size is fixed to $n_0 = 20$, and the IZ parameter is set to $\delta = \lfloor \sigma_k/\sqrt{n_0} \rfloor$, where σ_k^2 is the variance of an observation from the best system.

We consider two configurations of the true means, the slippage configuration (SC) and the grouped increasing means (GIM). The SC, in which $\mu_k = \delta$ and $\mu_2 = \mu_3 = \ldots = \mu_{k-1} = 0$, is considered as a difficult configuration since all the inferior systems are close to the best. In

reality, most of the inferior systems could be far way from the best and only a small percentage of all feasible systems contributes to the good solutions. To capture this effect, we define GIM by $\mu_1 = \mu_2 = \ldots = \mu_{k-l-1} = \mu_k - r_2\delta$ and $\mu_{k-l} = \mu_{k-l+1} = \ldots = \mu_{k-1} = \mu_k - r_1\delta$ and $\mu_k = \delta$, where $l = \lfloor 0.1k \rfloor$ and $r_2 = 3, 4$, $r_1 = 1, 2$.

For each configuration of the means, we test the effect of variances under three different configurations. We set all $\sigma_i^2 = 10^2$ in the equal-variance configuration, and $\sigma_i^2 = 100 \times (|\mu_i - \delta| + 1)$ and $\sigma_i^2 = 100/(|\mu_i - \delta| + 1)$ in the unequal-variance configuration. we are also interested in how the replication time Γ_i impacts on the performance of the procedure. [3] For the 6 combinations of configurations of means and variances, we consider the following three scenarios where $X_{i\ell}$ and $\Gamma_{i\ell}$ are independent, positive correlated and negative correlated, respectively. In the independent case, we consider two sub-scenarios, $\Gamma_{i\ell}$, $i = 1, 2, \ldots, k$, are constant (equaling to $d > 0$) and $\Gamma_{i\ell}$, $i = 1, 2, \ldots, k$, are random (following an exponential distribution with mean d). We assume that $\Gamma_{i\ell} = \max\{0, X_{i\ell} + 10\sigma_{\max}\}$ in the positive correlated case and $\Gamma_{i\ell} = \max\{0, -X_{i\ell} + 10\sigma_{\max}\}$ in the negative correlated case.

To test the robustness of our procedure, we also test experiments where the performance outputs are not normally distributed. To save the space, we consider the following four scenarios, $X_{i\ell}$ and $\Gamma_{i\ell}$ are positive or negative correlated under SC or GIM. Specifically, in the positive correlated case we assume that $X_{i\ell} = \Gamma_{i\ell}$ and $\Gamma_{i\ell}$ follows an exponential distribution with mean μ_i, where $\mu_k = d + \delta$ and $\mu_2 = \mu_3 = \ldots = \mu_{k-1} = d$ for SC and $\mu_1 = \mu_2 = \ldots = \mu_{k-l-1} = \mu_k - r_2\delta$ and $\mu_{k-l} = \mu_{k-l+1} = \ldots = \mu_{k-1} = \mu_k - r_1\delta$ and $\mu_k = d + \delta$, where $d > 5\delta$, $l = \lfloor 0.1k \rfloor$ and $r_2 = 3, 4$, $r_1 = 1, 2$ for GIM. In the negative correlated case we assume that $X_{i\ell} = -\Gamma_{i\ell}$ and $\Gamma_{i\ell}$ follows an exponential distribution with mean μ_i, where $\mu_k = d - \delta$ and $\mu_2 = \mu_3 = \ldots = \mu_{k-1} = d$ for SC and $\mu_1 = \mu_2 = \ldots = \mu_{k-l-1} = \mu_k - r_1\delta$ and $\mu_{k-l} = \mu_{k-l+1} = \ldots = \mu_{k-1} = \mu_k - r_2\delta$ and $\mu_k = d - \delta$, where $d > 5\delta$, $l = \lfloor 0.1k \rfloor$ and $r_2 = 3, 4$, $r_1 = 1, 2$ for GIM.

5 Conclusions and Future Work

References

Bechhofer, R. E. (1954). A single-sample multiple decision procedure for ranking means of normal populations with known variances. *The Annals of Mathematical Statistics 25*, 16–39.

Bechhofer, R. E., T. J. Santner, and D. M. Goldsman (1995). *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons.* John Wiley & Sons.

[3] There is no data transmission via Internet on a server or a desktop computer. As we tested, the communication time of data transmission between the two servers counts only for less than 0.1% of the total time, then we do not consider the communication time here.

Configuration		Equal Variance			
		Constant	Independent	Positive Corr.	Negative Corr.
Sample Average	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
Time Span	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
PCS	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
Configuration		Increasing Variance			
		Constant	Independent	Positive Corr.	Negative Corr.
Sample Average	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
Time Span	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
PCS	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
Configuration		Decreasing Variance			
		Constant	Independent	Positive Corr.	Negative Corr.
Sample Average	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
Time Span	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4
PCS	Rinott's	1	2	3	4
	KN	1	2	3	4
	PSP	1	2	3	4

Table 2: Summary under SC configuration when $k = 10^4$

Billingsley, P. (1999). *Convergence of Probability Measures* (Second ed.). John Wiley & Sons.

Chen, C.-H., J. Lin, E. Yücesan, and S. E. Chick (2000). Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynamic Systems 10*, 251–270.

Chen, E. J. (2005). Using parallel and distributed computing to increase the capability of selection procedures. In *Proceedings of the 2005 Winter Simulation Conference*, pp. 723731.

Chick, S. E. and K. Inoue (2001a). New procedures to select the best simulated system using common random numbers. *Management Science 47*, 1133–1149.

Chick, S. E. and K. Inoue (2001b). New two-stage and sequential procedures for selecting the best simulated system. *Operations Research 49*, 732–743.

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein (2009). *Introduction To Algorithms* (Third Edition ed.). MIT Press.

Fabian, V. (1974). Note on andersons sequential procedures with triangular boundary. *The Annals of Statistics 2*, 170–176.

Foster, I. (1995). *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*. Parallel programming / scientific computing. Addison-Wesley.

Frazier, P. I. (2011). Indifference-zone ranking and selection for more than 15,000 alternatives. Working paper.

Fujimoto, R. M. (1990). Parallel discrete event simulation. *Communications of the ACM 33*, 30–53.

Fujimoto, R. M., A. W. Malik, and A. J. Park (2010). Parallel and distributed simulation in the cloud. *SCS Modeling and Simulation Magazine, Society for Modeling and Simulation, Intl. 1*.

Glynn, P. W. and P. Heidelberger (1991). Analysis of parallel replicated simulations under a completion time constraint. *ACM Transactions on Modeling and Computer Simulation 1*, 3–23.

Heidelberger, P. (1988). Discrete event simulation and parallel processing: statistical properties. *SIAM Journal on Scientific and Statistical Computing 6*, 1114–1132.

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

Hong, L. J. and B. L. Nelson (2009). A brief introduction to optimization via simulation. *Proceedings of the 2009 Winter Simulation Conference*.

Hsieh, M. and P. W. Glynn (2009). New estimators for parallel steady-state simulations. *Proceedings of the 2009 Winter Simulation Conference*, 469–474.

Jennison, C., I. M. Johnstone, and B. W. Turnbull (1980). Asymptotically optimal procedures for sequential adaptive selection of the best of several normal means. *Technical Report, Department of Operations Research and Industrial Engineering, Cornell University*.

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*, 251–273.

Kim, S.-H. and B. L. Nelson (2006a). *Elsevier Handbooks in Operations Research and Management Science: Simulation*, Chapter 17. Selecting the best system, pp. 501–534. Elsevier.

Kim, S.-H. and B. L. Nelson (2006b). On the asymptotic validity of fully sequential selection procedures for steady-state simulation. *Operations Research 54*, 475–488.

Kim, S.-H., B. L. Nelson, and J. R. Wilson (2005). Some almost-sure convergence properties useful in sequential analysis. *Sequential Anal. 24(4)*, 411–419.

Luo, J. and L. J. Hong (2011). Large-sclale randking and selection using cloud computing. *Proceedings of the 2011 Winter Simulation Conference*.

Misra, J. (1986). Distributed discrete-event simulation. *ACM Computing Surveys 18*, 39–65.

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*, 950–963.

Pinedo, M. L. (2008). *Scheduling: Theory, Algorithms, and Systems* (Third edition ed.). Springer.

Rinott, Y. (1978). On two-stage selection procedures and related probability-inequalities. *Communications in Statistics - Theory and Methods A7*, 799–811.

Robbins, H. and D. O. Siegmund (1974). Sequential tests involving two populations. *Journal of the American Statistical Association 69*, 132–139.

Stein, C. (1945). A two-sample test for a linear hypothesis whose power is independent of the variance. *The Annals of Mathematical Statistics 16*, 243–258.

Whitt, W. (2002). *Stochastic-process limits*. Springer Series in Operations Research. New York: Springer-Verlag.

Yücesan, E., Y.-C. Luo, C.-H. Chen, and I. Lee (2001). Distributed web-based simulation experiments for optimization. *Simulation Practice and Theory 9*, 73–90.

A Sketch of Proving (3.14) in Theorem 3.3

Recall that $\mathbf{D}[0,1]$ is the Skorohod space. Let Λ be the set of strictly increasing functions λ mapping the domain $[0,1]$ onto itself, such that both λ and its inverse λ^{-1} are continuous. Then, the Skorohod metric ρ on $\mathbf{D}[0,1]$ can be defined by

$$\rho(X,Y) = \inf_{\lambda\in\Lambda}\left\{d : \sup_{t\in[0,1]}|\lambda(t)-t| \le d, \text{ and } \sup_{t\in[0,1]}|X(t)-Y(\lambda(t))| \le d\right\}. \tag{A.1}$$

Definition A.1. *On the Skorohod space* $\mathbf{D}[0,1]$*,*

(a) For $Y \in \mathbf{D}[0,1]$*, Let* $T_Y(U^\delta) = \inf\left\{t : |Y(t)| \ge U^\delta(t)\right\}$*, and define the function* $p^\delta : Y \in \mathbf{D}[0,1] \to p^\delta(Y) \in \mathbb{R}$ *by* $p^\delta(Y) = Y\left(T_Y(U^\delta)\right)$*.*

(b) For $Y \in \mathbf{D}[0,1]$*, Let* $T_Y(U) = \inf\{t : |Y(t)| \ge U(t)\}$*, and define the function* $p : Y \in \mathbf{D}[0,1] \to p(Y) \in \mathbb{R}$ *by* $p(Y) = Y(T_Y(U))$*.*

Proposition A.1. *If* $p^\delta(\cdot)$ *and* $p(\cdot)$ *are defined in Definition A.1, then*

$$\mathbb{P}[\mathcal{B}_\Delta \in \mathbf{D}_p] = 0, \tag{A.2}$$

where $\mathbf{D}_p$ *is the set of* $x \in \mathbf{D}[0,1]$ *such that* $p^\delta(x^\delta) \to p(x)$ *fails for some sequence* $\{x^\delta\}$ *with* $\rho(x^\delta, x) \to 0$ *in* $\mathbf{D}[0,1]$ *as* $\delta \to 0$*.*

Then by (3.13) and (A.2), and generalized continuous-mapping theorem (Theorem 3.4.4 in Whitt (2002)), we show that

$$p^\delta\left(Z_{1j}(\cdot)\right) \Rightarrow p\left(\mathcal{B}_\Delta(\cdot)\right), \quad \text{as } \delta \to 0, \tag{A.3}$$

which is essentially (3.13).

B Derivation of Equation 2.2

Proof. Since that the m servers are homogeneous and the service time of each customer is exponentially distributed with mean μ_1, by the Markovian property, we know that the inter-departure times, denoted by A_i, are i.i.d. with mean $\mu = \mu_1/m$. At time $t_0 = 0$, there are m homogeneous customers assigned to the server pool; at time t_i, $i = 1, 2, \ldots$, the ith customer leaves the system and additional customer is admitted to that server who just finishes service for the departure customer (as shown in Figure 7). Then, $A_i = t_i - t_{i-1}$ and $\mathrm{E}[A_i] = \mu$ for $i = 1, 2, \ldots$.

We intend to calculate the expectation of the service time of the ith customer who leaves the system, i.e., $\mathrm{E}[Y_{1i}]$, $i = 1, 2, \ldots$. The service time of the ith customer who leaves the system depends

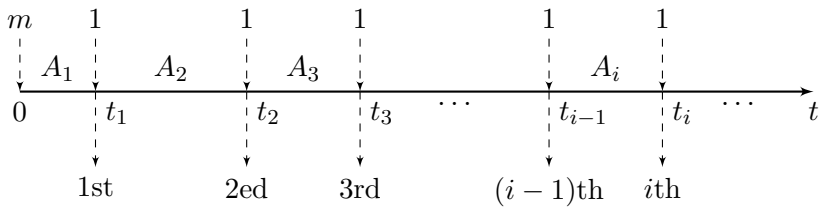

Figure 7: The inter-departure time.

on the time points he enters the system. For instance, the 3rd departure customer could enter the system at time t_0, t_1 or t_2. The probability he enter the system at t_2 is $\frac{m\cdot 1\cdot m}{m^3}$, the probability he enter the system at t_1 is $\frac{m\cdot(m-1)\cdot 1}{m^3}$, and the probability he enter the system at t_0 is $\frac{m\cdot(m-1)\cdot(m-1)}{m^3}$. So,

$$\begin{aligned}
\mathrm{E}[Y_{13}] &= \frac{m\cdot 1\cdot m}{m^3}\mathrm{E}[A_3] + \frac{m\cdot(m-1)\cdot 1}{m^3}\mathrm{E}[(A_2+A_3)] + \frac{m\cdot(m-1)\cdot(m-1)}{m^3}\mathrm{E}[(A_1+A_2+A_3)] \\
&= \frac{1}{m}\cdot\mu + \frac{m-1}{m^2}\cdot(2\mu) + \frac{(m-1)^2}{m^2}\cdot(3\mu).
\end{aligned}$$

Similarly, we can obtain the closed-form of $\mathrm{E}[Y_{1i}]$ as follows,

$$\begin{aligned}
\mathrm{E}[Y_{1i}] &= \frac{1}{m}\mathrm{E}[A_i] + \frac{m-1}{m^2}\mathrm{E}[(A_{i-1}+A_i)] + \ldots + \frac{(m-1)^{i-2}}{m^{i-1}}\mathrm{E}[\sum_{j=2}^{i} A_j] + \frac{(m-1)^{i-1}}{m^{i-1}}\mathrm{E}[\sum_{j=1}^{i} A_j] \\
&= \frac{1}{m}\sum_{j=1}^{i-1}\left[\left(\frac{m-1}{m}\right)^j (j\mu)\right] + \left(\frac{m-1}{m}\right)^{i-1}(i\mu) \quad \text{by plugging } \mu=\mu_1/m, \\
&= \mu_1 - \mu_1\left(1-\frac{1}{m}\right)^i.
\end{aligned}$$

Then, with some basic algebra,

$$\sum_{\ell=1}^{n}\mathrm{E}[Y_{1i}] = n\mu_1 - \mu_1(m-1)\left[1-\left(1-\frac{1}{m}\right)^n\right].$$

□