Front. Electr. Electron. Eng. China 2011, 6(3): 468–480
DOI 10.1007/s11460-011-0168-5

RESEARCH ARTICLE

Chun-Hung CHEN, Leyuan SHI, Loo Hay LEE

Stochastic systems simulation optimization

Abstract With the advance of new computational technology, stochastic systems simulation and optimization has become increasingly a popular subject in both academic research and industrial applications. This paper presents some of recent developments about the problem of optimizing a performance function from a simulation model. We begin by classifying different types of problems and then provide an overview of the major approaches, followed by a more in-depth presentation of two specific areas: optimal computing budget allocation and the nested partitions method.

Keywords simulation optimization, discrete-event systems, simulation-based decision making, computing budget allocation, ranking and selection

1 Introduction

Simulation is a popular tool for designing large, complex, stochastic engineering systems, since closed-form analytical solutions generally do not exist for such problems. Simulation allows one to accurately specify a system through the use of logically complex, and often nonalgebraic, variables and constraints. Detailed dynamics of complex, stochastic systems can therefore be modeled. This capability complements the inherent limitation of traditional optimization. With the advance of new computational technology, stochastic simulation and optimization has become increasingly a popular subject in both academic research and industrial applications. Optimization models can routinely involve millions of decision variables and constraints. Similarly, simulation models routinely utilize the generation of millions of random variates. However, the combination of these two popular systems engineering tools on this scale is not yet achievable [1]. So one either has a large-scale simulation model with relatively few (perhaps just a handful of) decision variables or a huge mathematical programming model with a relatively small number of random variables (or scenarios) over a couple of recourse stages.

The key difficulty in simulation optimization involves a trade-off between allocating computational resources for **searching** the solution space versus conducting additional simulation replications for better **estimating** the performance of current promising solutions. The searching aspect can involve algorithmic computation, as well as simulation computation for estimating the new candidate solutions. So application of randomized search algorithms in the simulation optimization setting involves two types of sampling: sampling the solution space and sampling the sample path (stochastic simulation) space. The fundamental tradeoff between search and estimation becomes especially pronounced when the cost of simulation is expensive. For example, one simulation replication of a complex semiconductor fab for a month of operations might take as long to run as solving a large linear programming (LP) problem. With millions of random variables generated in the stochastic simulation, a mathematical programming model formulation with millions of decision variables and an objective function involving a quantity that must be estimated using the simulation quickly leads to intractability.

In this paper, we provide an overview of various aspects of simulation optimization. We begin with a classification of problem settings and then present a brief summary of the most well-known approaches, providing some of the key references that the reader can consult for more details. The rest of the paper provides more in-depth description of two specific areas: the optimal computing budget allocation (OCBA) approach and the nested partitions (NP) method. OCBA and NP have partially addressed the tradeoff issue mentioned above.

Received April 28, 2011; accepted June 7, 2011

Chun-Hung CHEN (✉)
Department of Systems Engineering and Operations Research, George Mason University, Fairfax, VA 22030, USA
E-mail: cchen9@gmu.edu

Leyuan SHI
Department of Industrial and Systems Engineering, University of Wisconsin, Madison, WI 53706-1572, USA

Loo Hay LEE
Department of Industrial and Systems Engineering, National University of Singapore, Singapore 117576, Singapore

The goal of OCBA is to determine which are the most worthy designs to take more simulation replications and how many, whereas NP tries to determine which part of the design space is worthiest to explore in more depth by sampling more designs there. That said, OCBA intends to do a best estimation and NP intends to do a best search within a given computing budget.

The paper is organized as follows. We begin with the problem setting and a classification of types of problems, and present an illustrative prototypical example and then an overview of the major approaches. Sections 3 and 4 treat two particular research areas in more depth: optimal computing budget allocation and the nested partitions method. Section 5 concludes the paper.

2 Problem setting and overview

The simulation optimization problem setting is as follows:

$$\min_{\theta\in\Theta} J(\theta), \tag{1}$$

where θ is a p-dimensional vector of all the decision variables and Θ is the feasible region. Our setting presupposes that we have little knowledge on the structure of J (such as linearity or convexity), and moreover that J cannot be obtained directly, but rather is an expectation of another quantity $L(\theta; w)$, to which we have access, i.e.,

$$J(\theta) = \mathrm{E}[L(\theta; w)],$$

where w comprises the randomness (or uncertainty) in the system. In our setting, w represents a simulation replication, and $L(\theta; w)$ is a sample performance estimate obtained from the output of the simulation replication, e.g., the number of customers who waited more than a certain amount of time in a queueing system, or average costs in an inventory control system, or the profit and loss distribution in an investment portfolio or risk management strategy. Most performance measures of interest can be put into this general form, including probabilities by using indicator functions. However, quantiles are an example that cannot be put into this form.

The prototypical example we will use for illustrative purposes in this paper is a single-item (s, S) inventory control problem. Clearly, this example could easily be extended to more realistic systems that include many items and complicated relationships in terms of demand and ordering costs. Recall that under an (s, S) inventory control policy, when the inventory position (which includes inventory on hand plus that on order) falls below s at an order decision point (discrete points in time in a periodic review setting and any point in time in a continuous review setting), then an order is placed in the amount that would bring the inventory position up to S. The usual performance measure of interest involves costs assessed for excess inventory, inventory shortages, and item ordering. Alternatively, the problem can be formulated with costs on excess inventory and item ordering, subject to a service level constraint involving inventory shortages. We will adopt the first formulation for simplicity, and also define $q = S - s$ to use in place of S as one of the decision variables. Thus, in the framework of above, we have $\theta = (s, q), \Theta = \mathcal{R}^+ \times \mathcal{R}^+$ and $J(s, q)$ is the sum of holding costs, backlogging or lost sales costs, and ordering costs (which includes a fixed set-up cost and a per-unit cost). Under certain conditions, it is well known that an (s, S) policy is optimal, but even in cases where such a policy is not guaranteed to be optimal, one might adopt an (s, S) policy for its simplicity in implementation, and the optimization would be over all policies of that form. In certain simplified settings, an analytical form for J can be derived, and the optimal (s, q) values obtained analytically in closed form. Sometimes to test a simulation optimization algorithm, such a simplified setting is also considered to compare the convergence speed of the algorithm to the known true optimum. However, in more general settings, the costs can only be estimated via simulation.

A rough classification of simulation optimization problems can be based on the form of Θ. First, one could distinguish between continuous variable optimization problems and discrete variable (e.g., combinatorial, including binary) optimization problems. The latter could also be divided into categories where the solution space is finite and small, finite and large, or (countably) infinite. In the first of these cases, if it were a deterministic optimization problem, then enumeration would be an easy and obvious solution approach. However, in the simulation setting, even enumeration is not straightforward, as there is still the estimation aspect which leads to the question of how to allocate simulation replications among the feasible solutions. One approach for treating this setting is presented in the next section.

Next, Θ itself could be defined either explicitly or implicitly, and deterministically or probabilistically. What we mean by explicit and implicit here is nonstandard, but it refers to the variables being constrained independently, as in $\Theta = \mathcal{R}^+ \times \mathcal{R}^+$ versus what would have been the case had we used s and S as the decision variables, since in that case, we would have the constraint $S \geqslant s$, putting a dependence between the two decision variables, whereas s and q have no such dependence. Typical mathematical programming problems have both of these types of constraints, though the distinction is usually not made. For randomized search algorithms, however, there is an implication in terms of generating components of q independently or in a possibly correlated fashion. Both

of these types of constraints, however, define known deterministic domains for the decision variables, whereas if we consider the setting of a service level constraint in the (s, S) inventory example, then Θ itself would not even be known a priori, but would be implicitly estimated based on simulation, e.g., $\Theta = \{(s, q) : P(\text{stockout}) < 0.05\}$, where $P(\text{stockout})$ requires simulation to estimate for a given value of s and q.

Approaches toward simulation optimization include the following:

• **sample path optimization**, also known as sample average approximation — the main idea is to take a large enough set of samples so that the stochastic problem is basically turned back into a deterministic problem to which the tools of nonlinear programming (or possibly convex optimization) could be applied; in finance settings, this approach is sometimes referred to as "freezing all the random numbers", which typically could easily go into the millions or even billions; "convergence" is in terms of increasing the number of samples taken; for more details, see Refs. [2–4];

• **sequential response surface methodology** — iterative algorithms using statistical methods, primarily regression, in order to improve upon the candidate solution by searching the feasible solution space; there is also a non-sequential metamodel version, as well; for more details and recent developments, see Refs. [5,6];

• **stochastic approximation** — iterative algorithms that mimic gradient methods in deterministic (nonlinear) optimization; such algorithms have provably convergent asymptotic (in the number of iterations) properties under suitable conditions; a comprehensive reference is Ref. [7];

• **deterministic metaheuristics** — broad category including approaches such as genetic algorithms, tabu search, scatter search, Nelder-Mead iteration, and other iterative and possibly population-based (evolutionary) algorithms from deterministic (nonlinear) optimization; there is little probabilistic or statistical consideration incorporated; for more details in the simulation setting, see Ref. [8].

More details on these techniques can be found in other simulation optimization surveys [1,9–12]. The first three of these approaches benefit from the availability of gradient estimates. In the sequential response surface methodology approach, the gradient is obtained via (linear or quadratic) regression. In the other two cases, there are different techniques depending on how much knowledge one has about the system of interest, from "black box" approaches to exploitation of the dynamics of the system. For recent overviews on stochastic gradient estimation techniques, see Refs. [13,14]. Gradient estimation for the (s, S) inventory problem is treated in Refs. [15–20].

Over the past decade, all of the commercial simulation software vendors have begun offering an "optimization" module as an option. Most of the algorithms in these modules are based on the fourth approach, employing metaheuristics on the estimated performance, and thus have little in the way of probabilistic or statistical guarantees [10], although more recently some statistical ranking and selection ideas have begun to be applied in order to be able to say something about the goodness of the solutions obtained.

Many commercial simulation software vendors offer a call center simulation. Using queueing models [21], one could optimize staffing levels, and then use simulation to test the performance and tune the decision variables and/or optimize other decision variables. At this time, however, as far as we are aware, none of the commercial software for simulation optimization has implemented any efficient gradient search, which would require integration between the simulation model and the optimization routine.

The following two sections outline in more detail two different approaches or methodologies that attack the problem of simulation optimization from different aspects. The first approach can be viewed as one methodology in the general research area of statistical ranking and selection. The objective is simply to pick the best and an optimal subset among a fixed set of alternatives, where all alternatives will be tested, so the number of choices has to be relatively small. In the context of simulation optimization, the difficulty of search has been removed, so the focus is on efficiently allocating simulation replications among the alternatives. Then we consider a global optimization approach called the nested partitions method, which can be applied to both deterministic and stochastic optimization problems.

In terms of our earlier classification, the two methodologies discussed in more detail in the following sections attack the following cases:

• Section 3: Θ is finite (and relatively small).

• Section 4: Θ can be continuous or discrete, possibly defined implicitly and probabilistically.

3 Optimal computing budget allocation

When Θ is finite and small enough, enumeration can be used in principle to find the optimum, but in the simulation optimization setting, the value of the objective function is obtained through simulation, so the main question is how to allocate the simulation replications efficiently among the various alternative candidate solutions. This is a classical problem in statistical ranking and selection (see, e.g., Refs. [22,23]).

One direct and intuitive approach is to gradually increase the computing budget (i.e., the number of simulation replications) for each alternative solution until

the variance of the estimated performance is sufficiently small (i.e., the confidence interval for estimation is satisfactorily narrow). The simplest allocation is to use an identical number of replications for each alternative — called the "equal allocation" (Equal), which can be inefficient. For example, if one alternative has very low variance, then it may only require one or two simulation replications to estimate its performance; thus, it seems sensible that the number of total replications allocated to an alternative increase with increasing (estimated) variance. On the other hand, if another alternative has a very large mean, then it may be easily screened out as not being competitive; thus, it seems intuitive that the number of total replications allocated to an alternative decrease with increasing (estimated) mean (for a minimization problem; otherwise, it should increase). To improve efficiency for selecting a best design, several approaches have been explored. Intuitively, to ensure a high probability of correctly selecting an optimal design, a larger portion of the computing budget should be allocated to those designs that are critical in the process of identifying good designs. In other words, a larger number of simulations must be conducted with those critical designs to reduce estimator variance. Similarly, limited computational effort should be expended on noncritical designs that have little effect on identifying the good design (even if these designs have high variances). Overall simulation efficiency is improved as less computational effort is spent on simulating noncritical designs and more is spent on critical designs. The main result in the optimal computing budget allocation (OCBA) approach is that both the means and variances enter into the allocation. We first give a few simple examples to explain the ideas.

Suppose that in an inventory control problem, five alternative values for the control (s, S) are provided to us. For each of these values we want to conduct simulation to find the one with minimum expected cost. Suppose we conduct some initial simulation replications for all five alternative designs, and consider some possible outcomes from the first-stage estimates, in terms of the estimated inventory costs and associated 99% confidence intervals. The question of computing budget allocation is how we should allocate additional simulation replications if we want to continue more simulations to enhance the probability of correctly selecting the best design.

Case 1 A trivial example

Figure 1 gives a hypothetical example of possible results from the initial replications, showing the obtained 99% confidence intervals along with the accompanying mean estimator (represented as the line in the middle of the confidence interval). While there is uncertainty in the estimation of the performance for each design, it is obvious that designs 2 and 3 are much better than the other designs. As a result, it is sensible to allocate few or no simulation replications to designs 1, 4, and 5; instead allocating the bulk of the simulation budget to designs 2 and 3 provides more insight into the optimal solution.

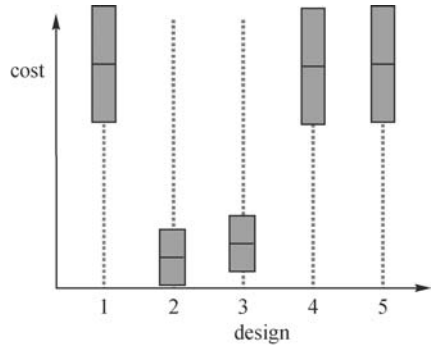

Fig. 1 99% confidence intervals for 5 alternative designs after preliminary simulation in Case 1

It is worth noting that some of earlier R&S procedures allocate the simulation replications to each design proportional to the (estimated) variance, and do not consider the estimated means. In Case 1, the three worst designs have larger variance than the two best design; as a result an allocation based solely on variances may waste resources on significantly inferior designs. In addition, they may not be able to distinguish which of the two better designs is optimal. Thus, intuitively optimal simulation allocation would result in limited simulation of designs 1, 4, and 5, and more extensive simulation of designs 2 and 3.

One natural question that results from this "screening" approach is whether to always stop simulating poor performing designs or to simulate a very low number of replications of these designs. The answer is not necessarily always the same for all problem sets. This question is further explored in the following example.

Case 2 Most common scenario

In Case 1, some designs dominate the others and so we have a good idea about how the additional simulation budget should be allocated. However, most cases are not as trivial as it. We consider a more common scenario here.

Figure 2 shows the output after initial simulations. In this case, some designs seem better, but none are clearly better than all the others, since all the confidence intervals overlap. In situations such as this, it is not straightforward to determine which designs can be eliminated and which designs should receive more simulation budget. Ideally, we would like to allocate simulation replications to designs in a way that maximizes the simulation quality or minimizes some objective function. As discussed above, this approach is intuitively dependent upon the mean and variance of each design.

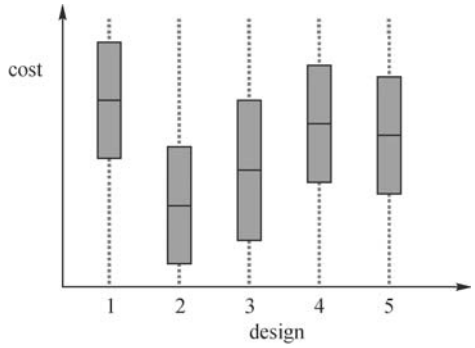

Fig. 2 99% confidence intervals for 5 alternative designs after preliminary simulation in Case 2

Moreover, a good computing budget allocation depends not only on the relative means and variances of all designs, but also the specific objective that we would like to achieve. In previous cases, our objective has been to choose the best design correctly. If we employ different objectives, the computing budget allocation may be entirely different. This issue is illustrated in the following case.

Case 3 Selecting an optimal subset

In this example, instead of choosing the best design, we would like to choose all the top-m designs (i.e., to determine which designs are among the best top-m, where $m \geqslant 2$). Similarly, we conduct initial simulation replications for all five alternative designs. The results from these preliminary simulations are shown in Fig. 3. In this case, the inventory cost estimations are in ascending order and all designs have similar variance.

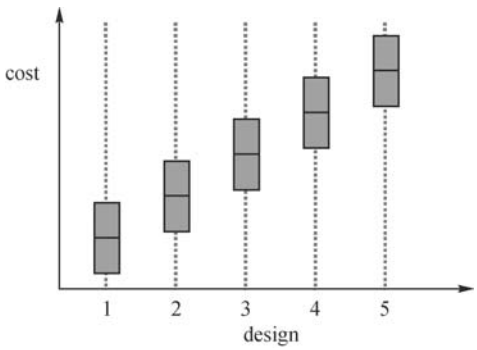

Fig. 3 99% confidence intervals for 5 alternative designs after preliminary simulation in Case 3

Design 1 seems to be the best design and design 2 is a close competitor. All other designs have steadily decreasing performance. If our objective is to correctly select the best design, it is reasonable to stop simulating designs 3, 4, and 5, and only continue simulating designs 1 and 2.

However, if the decision-maker wants to identify the top-2 designs, instead of selecting the best design, how should we continue allocating simulation budget? Since design 1 is much better than designs 3, 4, and 5, it is clear that design 1 is probably one of the top-2 designs. As a result, there is minimal need to continue simulating design 1. On the other hand, designs 4 and 5 are much worse than other designs and so they have little chance to be ranked within top-2. Therefore, there is not much value to simulate designs 4 and 5 either. To allocate computing budget efficiently, it is reasonable to concentrate simulation time on designs 2 and 3 because they are the "critical designs" in determining the top-2 designs.

Following this same idea, if we are interested in selecting the subset of top-3 designs, it is best to simulate most on designs 3 and 4. If we want to determine the worst design, an efficient simulation budget allocation would allocate most of the simulation budget to designs 4 and 5. These various scenarios demonstrate that an efficient simulation budget allocation also depends on the objective.

If the sample performances $L(\theta; w)$ are assumed to be independent and normally distributed, then the means and variances should suffice to characterize an efficient allocation of simulation resources (otherwise, higher moments and correlations might be required). The OCBA — introduced in Refs. [24–26] — maximizes an approximation of the probability of correct selection $P\{\mathrm{CS}\}$, where correct selection (CS) indicates choosing the alternative with minimum mean, leading to an efficient allocation algorithm that includes both means and variances. Extensions of the OCBA approach consider correlated sampling [27]; non-normal distributions [28,29]; using a regression model [30]; multiple objective functions [31–33]; using expected opportunity cost instead of the probability of correct selection [34,35]; minimizing variance instead of maximizing the probability of correct selection [36]; transient means ranking and selection [37,38]; selecting an optimal subset of top-m solutions rather than the single best solution [39]; selecting the best design with consideration of stochastic constraints or design complexity [40,41].

The approach by Chick and Inoue [42,43] estimates the correct selection probability with Bayesian posterior distributions, and allocates further samples using decision-theory tools to maximize the expected value of information in those samples. The procedure by Kim and Nelson [23] allocates samples in order to provide a guaranteed lower bound for the frequentist probability of correct selection integrated with ideas of early screening. Branke, Chick, and Schmidt [44] provide an overview and extensive computational comparison between many of the aforementioned selection procedures.

3.1 Selecting the best design

Now we briefly describe the OCBA approach, one of the top performers in the computational tests of Branke,

Chick, and Schmidt [44]. Chen et al. [25] formulate the problem of simulation computing budget allocation as an optimization problem. Let N_i denote the number of simulation replications allocated to alternative design i. As motivated earlier, instead of equally simulating all designs, we want to choose $N_1, N_2, \ldots, N_k$ more intelligently so that the simulation efficiency can be enhanced. First, we consider choosing $N_1, N_2, \ldots, N_k$ such that $P\{\text{CS}\}$ is maximized, subject to a limited computing budget T,

$$\max_{N_1,N_2,\ldots,N_k} \quad P\{\text{CS}\}$$

$$\text{s.t. } N_1 + N_2 + \cdots + N_k = T. \tag{2}$$

Here $N_1+N_2+\cdots+N_k$ denotes the total computational cost assuming the simulation execution times for different alternatives are roughly the same. Using a Bayesian model [45], an approximation of $P\{\text{CS}\}$ is found and then an asymptotic solution to the approximation is obtained. Basically, aside from the best alternative, the solution prescribes an allocation proportional to variance and inversely proportional to the squared difference in mean from the best.

Let design b be the design with the best sample average thus far, σ_i^2 be the variance for design i, μ_i be the mean of design i, and $\delta_{b,i} \equiv \mu_i - \mu_b$. Based on the work by Chen et al. [25], $P\{\text{CS}\}$ is asymptotically maximized when the relationship between two non-best designs i and j, $i \neq j \neq b$, is

$$\frac{N_i}{N_j} = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_j/\delta_{b,j}}\right)^2, \tag{3}$$

and the number of simulation replications for the best design is given as

$$N_b = \sigma_b \sqrt{\sum_{i=1,i\neq b}^{k} \frac{N_i^2}{\sigma_i^2}}. \tag{4}$$

It shows that the noisier the simulation output, the more replications are allocated. More replications are also given to the design of which mean is closer to that of the best design.

With the asymptotic solution in Eqs. (3) and (4), we now present a cost-effective sequential approach based on OCBA to select the best design from k alternatives with a given computing budget. Each design is initially simulated with n_0 replications in the first stage, and additional replications are allocated incrementally with Δ replications to be allocated in each iteration.

OCBA procedure

INPUT k, T, Δ, n_0 ($T - kn_0$ a multiple of Δ and $n_0 \geqslant 5$);

INITIALIZE $l \leftarrow 0$;
Perform n_0 simulation replications for all designs; $N_1^l = N_2^l = \cdots = N_k^l = n_0$.

LOOP WHILE $\sum_{i=1}^{k} N_i^l < T$ **DO**

UPDATE Calculate sample means $\bar{J}_i$, and sample standard deviation s_i, $i = 1, 2, \ldots, k$, using the new simulation output; find $b = \arg\min_i \bar{J}_i$.

ALLOCATE Increase the computing budget by Δ and calculate the new budget allocation, $N_1^{l+1}, N_2^{l+1}, \ldots, N_k^{l+1}$, according to

1) $\dfrac{N_i^{l+1}}{N_j^{l+1}} = \left(\dfrac{s_i(\bar{J}_b - \bar{J}_j)}{s_j(\bar{J}_b - \bar{J}_i)}\right)^2$, for all $i \neq j \neq b$, and

2) $N_b^{l+1} = s_b \sqrt{\sum_{i=1,i\neq b}^{k} \left(\dfrac{N_i^{l+1}}{s_i}\right)^2}$.

SIMULATE Perform additional $\max(N_i^{l+1} - N_i^l, 0)$ simulations for design i, $i = 1, 2, \ldots, k$; $l \leftarrow l + 1$.

END OF LOOP

When the simulation is relatively expensive, the computation cost of the ALLOCATE step becomes negligible. In this case, Δ should be as small as possible and can be set as 1. To avoid the numerical error occurred when rounding N_i^{l+1} into an integer, we suggest a numerically robust alternative. The ALLOCATE and SIMULATE steps are revised as follows.

ALLOCATE Increase the computing budget by one and calculate the new budget allocation, $N_1^{l+1}, N_2^{l+1}, \ldots, N_k^{l+1}$, according to

1) $\dfrac{N_i^{l+1}}{N_j^{l+1}} = \left(\dfrac{s_i(\bar{J}_b - \bar{J}_j)}{s_j(\bar{J}_b - \bar{J}_i)}\right)^2$, for all $i \neq j \neq b$, and

2) $N_b^{l+1} = s_b \sqrt{\sum_{i=1,i\neq b}^{k} \left(\dfrac{N_i^{l+1}}{s_i}\right)^2}$,

leave N_i^{l+1} as a decimal number and find $i^* = \arg\max_i (N_i^{l+1} - N_i^l)$.

SIMULATE Perform additional one simulation for design i^*; $N_{i^*}^{l+1} = N_{i^*}^l + 1$; $N_i^{l+1} = N_i^l$, for $i \neq i^*$; $l \leftarrow l + 1$.

Intuitively, we determine which design is the most starving one in terms of the need of additional simulation, and then simulate that design for one additional replication. This procedure is iteratively continued until the total budget T is exhausted. Our numerical testing indicates that this simpler procedure performs equally well.

Chen and Lee [46] test the OCBA procedure on a worker allocation problem described below. In summary, this problem represents the challenge of allocating 31 workers within a two-stage queue where each stage of the queue can contain no less than 11 workers as shown in Fig. 4.

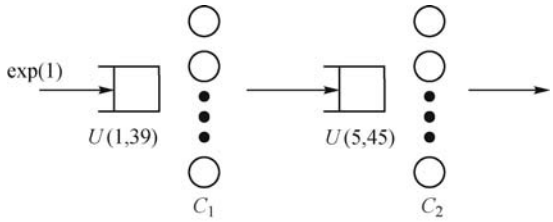

Fig. 4 A two-stage queuing system where both C_1 and C_2 must be greater than 10

Denote C_1 and C_2 as the numbers of workers allocated to nodes 1 and 2. Thus $C_1 + C_2 = 31, C_1 \geqslant 11$, and $C_2 \geqslant 11$. There are 10 alternative combinations of (C_1, C_2). We want to choose the best alternative of (C_1, C_2) so that the average system time for the first 100 customers is minimized. Since there is no closed-form analytical solution for the estimation of the system time, stochastic (discrete-event) simulation can be performed to find the best design.

To characterize the performance of different procedures as a function of T, we vary T between 200 and 8000 for all of the sequential procedures and the estimated achieved $P\{\text{CS}\}$ as a function of T is shown in Fig. 5. We see that all procedures obtain a higher $P\{\text{CS}\}$ as the available computing budget increases. However, OCBA achieves a same $P\{\text{CS}\}$ using the lowest mount of computing budget. Table 1 shows the computation costs to attain $P\{\text{CS}\} = 0.95$ and 0.99 for OCBA, Equal, and PTV, respectively.

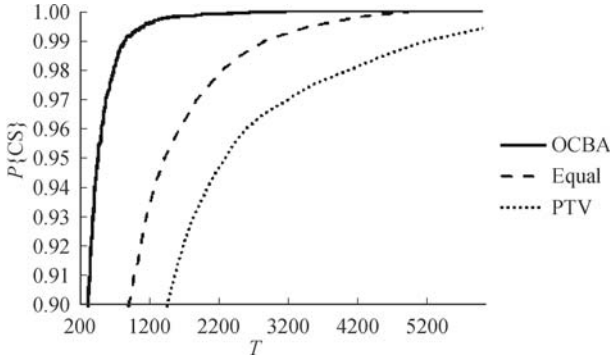

Fig. 5 $P\{\text{CS}\}$ vs. T using three sequential allocation procedures

Table 1 Computation costs to attain $P\{\text{CS}\} = 0.95$ or 0.99 using different sequential procedures

$P\{\text{CS}\}$	OCBA	Equal	PTV
0.95	470	1450	2270
0.99	850	2890	5230

It is worth noting that PTV (a sequential version of the procedure by Rinott [47]) performs worse than simple equal allocation for this problem. This performance is because PTV determines the number of simulation samples for all designs using only the information of sample variances. For this particular worker allocation problem, the best designs have the lowest variance and the worst designs have the largest variance. As a result, PTV generally allocates less time to designs that perform well, thus achieving lower $P\{\text{CS}\}$ for the same T values.

Overall, both Equal and PTV are much slower than OCBA. Figure 6 shows the average total number of simulation replications allocated, N_i, for all i, to achieve a $P\{\text{CS}\}$ of 99% using OCBA, PTV, and the Equal procedure, respectively. Note that the budget allocation by OCBA is quite different from that using PTV. It is worthy to note that OCBA simulates the top three designs (designs 3, 4, and 5) much more than all of the other designs combined whereas PTV simulates these designs less than other designs because they have smaller variance.

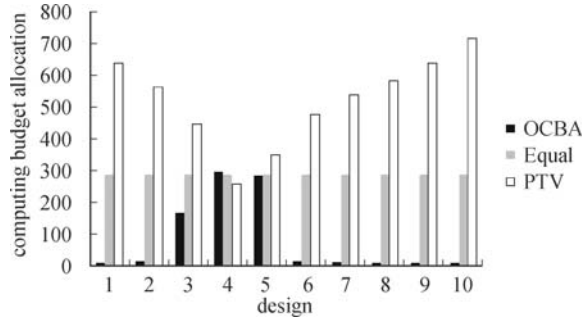

Fig. 6 Computing budget allocation as determined by OCBA, Equal, and PTV procedure for worker allocation problem to achieve a $P\{\text{CS}\}$ of 99%

Chen and Lee [46] also study how well OCBA performs for larger numbers of design alternatives. In this case, instead of providing only 31 workers, we increase the number of workers up to 121 workers where there must be at least 11 workers at each station. As a result, the number of possible design alternatives varies from 10 to 100. In this test, we compare OCBA and equal allocation, and focus on the "speedup factors" under OCBA. For both procedures, we record the minimum computation cost to reach $P\{\text{CS}\} = 99\%$: T_{OCBA} and T_{EA}. The "speedup factor" using OCAB is given by the ratio $T_{\text{EA}}/T_{\text{OCBA}}$. It is also useful to measure the so-called *Equivalent Number of Alternatives with a Fixed Computing Budget*, ENAFCB(k), which is defined as the number of alternatives that can be simulated under the equal allocation procedure using the computing budget that is needed for OCBA to simulate k alternatives for reaching $P\{\text{CS}\} = 99\%$.

Based on the approach described above, the speedup factors and ENAFCB factors are shown below in Table

2. We see that OCBA is even more efficient as the number of designs increases. The higher efficiency is obtained because a larger design space gives the OCBA algorithm more flexibility in allocating the computing budget. In particular, ENAFCB(50) = 3.94 means that with an equivalent effort of less than 4 alternatives, OCBA can simulate 50 design alternatives. This advantage demonstrates that OCBA can provide the ability to simulate much many more designs for limited available computational time (further enhancing the probability of identifying the correct design).

Table 2 Speedup factor of using OCBA as compared with the use of equal allocation

number of designs, k	5	10	25	50	75	100
speedup factor using OCBA	2.08	3.40	7.86	12.69	16.50	20.05
ENAFCB(k)	2.40	2.94	3.18	3.94	4.55	4.99

3.2 Selecting an optimal subset

Instead of selecting the best design, in some cases it is more useful to provide a set of good designs than a single best design for decision maker to choose, because he/she may have other concerns which are not modeled in the simulation. Such efficient subset selection procedures are also beneficial to some recent developments in simulation optimization that require the selection of an "elite" subset of good candidate solutions in each iteration of the algorithm, such as evolutionary population-based algorithm. A subset with good performing solutions will result in an update that leads the search in a promising direction.

Let S_m be the set of m (distinct) indices indicating designs in selected subset. Naturally we take S_m to be the m designs with the smallest *sample* means. Thus,

$$S_m \equiv \{(1), (2), \ldots, (m)\}.$$

Since the simulation output is stochastic, the set S_m is a random set. In terms of our notation, the correct selection event is defined by S_m containing all of the m smallest mean designs:

$$\mathrm{CS}_m \equiv \left\{ \bigcap_{i \in S_m} \bigcap_{j \notin S_m} (\mu_i \leqslant \mu_j) \right\}.$$

The objective here is to find a simulation budget allocation that maximizes the probability of selecting the *optimal subset,* defined as the set of $m(< k)$ best designs, for m a fixed number. Note that rank order within the subset is not part of the objective. The optimal computing budget allocation (OCBA-m) problem is given by

$$\max_{N_1, N_2, \ldots, N_k} \quad P\{\mathrm{CS}\}_m$$
$$\text{s.t. } N_1 + N_2 + \ldots + N_k = T.$$

Chen et al. [39] offer an approximation to $P\{\mathrm{CS}\}_m$ and show that the approximation can be asymptotically (as $T \to \infty$) maximized when

$$\frac{N_i}{N_j} = \left(\frac{\sigma_i / \delta_i}{\sigma_j / \delta_j} \right)^2,$$

for any $i, j \in \{1, 2, \ldots, k\}$, and $i \neq j$, where $\delta_i = \mu_i - c$ and c is a chosen value between $\mu_{(m)}$ and $\mu_{(m+1)}$. For practical implementation, we provide a sequential procedure presented as follows. Extensive numerical testing can be found in Ref. [46], which shows that OCBA-m is much more efficient than traditional procedures.

OCBA-m allocation procedure

INPUT k, m, T, Δ, n_0 ($T - kn_0$ a multiple of Δ and $n_0 \geqslant 5$);

INITIALIZE $l \leftarrow 0$;
Perform n_0 simulation replications for all designs; $N_1^l = N_2^l = \cdots = N_k^l = n_0$.

LOOP WHILE $\sum_{i=1}^{k} N_i^l < T$ **DO**

UPDATE Calculate sample means $\bar{J}_i$, and sample standard deviation s_i, $i = 1, 2, \ldots, k$, using the new simulation output; compute $\hat{\sigma}_i = \frac{s_i}{\sqrt{N_i^l}}$, $i = 1, 2, \ldots, k$, and $c = \frac{\hat{\sigma}_{i_{m+1}} \bar{J}_{i_m} + \hat{\sigma}_{i_m} \bar{J}_{i_{m+1}}}{\hat{\sigma}_{i_m} + \hat{\sigma}_{i_{m+1}}}$; update $\delta_i = \bar{J}_i - c$, $i = 1, 2, \ldots, k$.

ALLOCATE Increase the computing budget by Δ and calculate the new budget allocation, $N_1^{l+1}, N_2^{l+1}, \ldots, N_k^{l+1}$, according to

$$\frac{N_1^{l+1}}{\left(\frac{s_1}{\delta_1}\right)^2} = \frac{N_2^{l+1}}{\left(\frac{s_2}{\delta_2}\right)^2} = \ldots = \frac{N_k^{l+1}}{\left(\frac{s_k}{\delta_k}\right)^2}. \quad (10)$$

SIMULATE Perform additional $\max(N_i^{l+1} - N_i^l, 0)$ simulations for design i, $i = 1, 2, \ldots, k$; $l \leftarrow l + 1$.

END OF LOOP

3.3 A general framework of integration of OCBA with metaheuristics

When the design space Θ is large, instead of simulating all possible designs, some optimization search methods can be applied to search the design space. By exploring the information in the design space, the expensive enumeration can be avoided. One nice approach is the NP

method presented in the next section. A search method is applied to iteratively search the design space to find the optimum as depicted in Fig. 7. At each iteration of the search, several designs are selected for simulation. The simulation output information is used to guide the search for next iteration. Such simulation and search iteratively proceeds until a good or optimal design is found. Such a framework involves two major issues:

• How should we iteratively search the design space Θ?

• How should we efficiently simulate those designs generated in each iteration in order to provide the necessary information to guide the new search? Ideally, we want to simulate the generated designs in an intelligent way so that the total simulation cost is minimized.

A search method like NP aims to address this issue. OCBA focuses more on the second issue by determining an effective simulation budget allocation so that the necessary simulation output can be obtained efficiently to guide the search effectively given a metaheuristic or search method. The integration framework of OCBA and metaheuristics is shown in Fig. 7. Most of the metaheuristics start with a population of designs, and elite designs will be selected from this population of designs in order to generate the better population during the search process. Evaluation via simulation is critical in determining the elite designs, and OCBA is able to help to conduct the evaluation effectively by allocating computing budget intelligently.

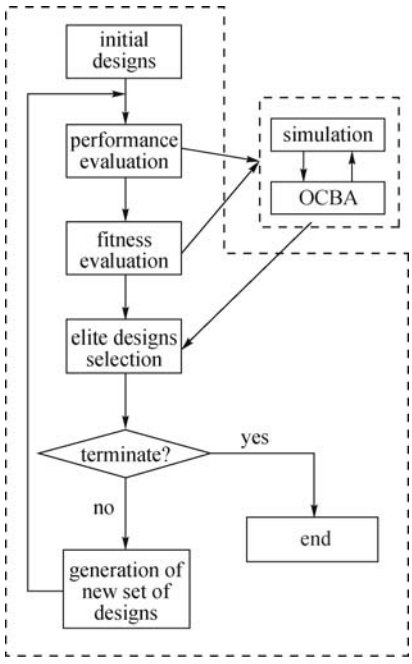

Fig. 7 An example illustrating how OCBA helps a search method for large-scale simulation optimization problems

4 Nested partitions method

The nested partitions (NP) method has recently been proposed to solve global optimization problems [48–50]. Reference [51] concentrates on the deterministic setting, whereas here the emphasis is on how it can be applied to the simulation setting, which we illustrate with the (s, S) inventory control example. The method can be briefly described as follows. In each iteration, a region considered most promising is assumed. We then partition this region into M subregions and aggregate the entire surrounding region into one. Therefore, within each iteration, we only look at $M + 1$ disjoint subsets that cover the feasible region. Each of these $M + 1$ regions is sampled using some random sampling scheme and the estimated performance function values at randomly selected points are used to approximate a so-called promising index for each region. This index determines which region becomes the most promising one in the next iteration. The sub-region scoring highest on the promising index automatically becomes the most promising region. The method backtracks to a larger region, if the surrounding region rather than a sub-region is found to have the best promising index. Here, a fixed backtracking rule is employed. The new most promising region is then partitioned and sampled in a similar fashion. This methodology divides into four main steps that constitute the NP method. Not only can each of these steps be implemented in a generic fashion, but it can also be combined with other optimization methods, thus making it adaptive to any special structure of a given problem.

1) Partitioning. The first step is to partition the current most promising region into several sub-regions and aggregate the surrounding region into one. The partitioning strategy imposes a structure on the feasible region and is therefore important for the rate of convergence of the algorithm. If the partitioning is such that most of the good solutions tend to be clustered together in the same subregions, it is likely that the algorithm quickly concentrates the search in these subsets of the feasible region. If the partitioning is completely unrelated to the performance function, then it is called generic partitioning. The advantage of generic partitioning is that the partitioning tree is usually predictable in terms of branching degrees and searching depths. More efficient partitions could be constructed if the performance function is considered. This type of partitioning technique is called knowledge-based clustered partitioning. For detailed discussion on both partitions, see Ref. [52].

2) Random sampling. The next step of the algorithm is to randomly sample from each of the subregions and from the aggregated surrounding region. This can be done in almost any fashion, provided that each solution in a given sampling region should be selected with a positive probability. Although uniform sampling can always be used, it may often be worthwhile to incorporate special structures into the sampling procedure so

that the sampling quality can be improved, e.g., some kind of weighted sampling scheme.

3) Calculation of promising index and determination of the most promising region. Once each region has been sampled, the next step is to use the sample points to calculate the promising index of each region and then determine the most promising region based on the promising indices. It should be noted that the determination is based on order comparison of the indices. In terms of the implication for simulation optimization, the key thing to note is that an accurate estimation of the promising index value for each region is not critical, i.e., the relative order of promising indices, or the relative order of the considered solutions, is more essential than the value of the promising index itself. This is an ideal situation for the use of ordinal optimization [53,54]. Again the NP methodology offers a great deal of flexibility. The only requirement imposed on a promising index is that it should agree with the original performance function on singleton regions.

4) Further partition or backtracking. If one of the subregions has the best promising index, the algorithm moves to this region and considers it to be the most promising region in the next iteration. If the surrounding region has the best promising index the algorithm backtracks to a larger region. The NP method described here can be applied, in its generic form, to a wide range of problems. The method is also capable of taking advantage of special structures by incorporating them into the partitioning, sampling, and promising index steps. The partitioning approach also makes the algorithm compatible with emerging parallel computing capabilities. Each region can be treated independently and in parallel, with only a minor coordination overhead. The method also uniquely combines global search through partitioning and sampling and local search through estimation of the promising index.

We use the (s, S) inventory example to illustrate how the NP method can be applied to this problem in simulation optimization. In the kth iteration we assume that there is a subregion of the feasible region that may be considered most promising. Initially we assume no knowledge about the most promising region and let it be the entire feasible region Θ. The first step is to partition the most promising region into disjoint subregions and aggregate the surrounding region (if any) into one. In the two-dimensional (s, S) inventory example, the partition can be done on one variable only or on both simultaneously. For example, in Fig. 8, we first partition the solution space into two subregions based on the value of s, without any surrounding region. Next, if $\{(s < 10, q)\}$ becomes the most promising region, we then partition it into two subregions, and the surrounding region is $\Theta \backslash \{(s < 10, q)\}$. We can further partition each such subregion by subdividing the second variable into intervals or by further partitioning the variable s into subintervals as is done in the example shown in Fig. 8. Assuming that s and q are discrete (e.g., integer-valued), this procedure can be repeated until the singleton region is reached, when all the s and q are fixed. It can be seen from Fig. 8 that at the same level, each subregion contains a different number of feasible points. Given a solution space, there exist many partitions. For example, we could also have divided Θ into three subregions initially by dividing the real-line interval for s into three different intervals (as in the third level in Fig. 8). This partition would provide a completely different set of subregions.

The second step is to use some sampling method to obtain a set of solutions from each region. This should be done in such a way that each point has a positive probability of being selected. It should be noted that many heuristics may be incorporated into the sampling step through a weighted sampling scheme, which we will demonstrate. The only requirement is that each point in a sampling region should have a positive probability of being selected. While uniform sampling scheme works well in most cases, from our experience, incorporation of a simple heuristic into the sampling scheme can drastically improve the sampling quality. For example, suppose the current sampling region has the form of $\{([0, R], q)\}$, meaning that the variable s could take any value between $[0, R]$ and q could take any nonnegative value. Generating a random sampling point in this sampling region means to randomly select on both values. However, assume we want to weight the probabilities for

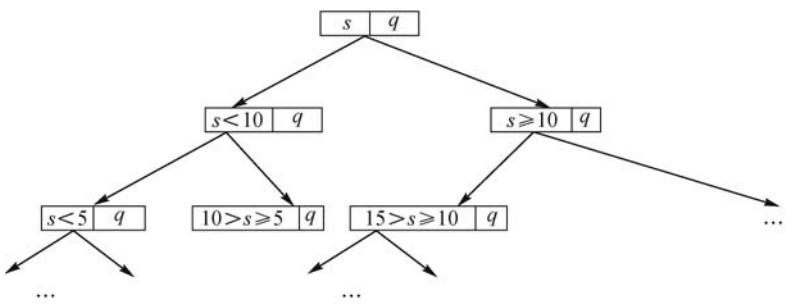

Fig. 8 Partitioning of (s, S) inventory problem example

the numbers between 0 and 2 three times higher than values greater than 2. Of course, there are many other ways to assign the weights. Numerical results indicate that the NP method performs well using such simple heuristics.

The third step is to rank and select the best from the set of sampled alternatives (solutions), comprising the union of all sampled solutions from each disjoint (sub)region. In the simulation setting, this step is connected to the previous step and the next step, because simulation must be used to estimate the performance (promising index) of each sampled solution. Clearly, the OCBA method is one way to carry this out more efficiently, and combining NP with OCBA in these steps is proposed in Ref. [55].

The final step is to determine the most promising region for the next iteration. The subregion estimated to have the best promising index becomes the most promising region in the next iteration. In the simulation setting, the estimation error comes from two sources:

• the error from possibly not sampling the entire subregion (except when they are singletons), and

• the error from simulation estimation of performance at a particular solution point.

The new most promising region is thus nested within the last. By extension, if the surrounding region is found to have the best promising index, the algorithm backtracks to a larger region that contains the best solution. The partitioning continues until singleton regions are obtained and no further partitioning is possible.

5 Conclusions

Simulation optimization is an active research area, both in terms of research into new and improved algorithms and theoretical results, and in commercial software implementation. In this paper, we provide an overview of various aspects of simulation optimization. We begin with a classification of problem settings, and then provide an overview of the most well-known approaches, providing some of the key references that the reader can consult for more details. The rest of the paper provides more in-depth description of two specific areas: the optimal computing budget allocation (OCBA) approach and the nested partitions (NP) method. OCBA and NP have partially addressed the tradeoff issue mentioned above. OCBA intends to determine which are the most worthy designs to take more simulation replications and how many, whereas NP intends to determine which part of the design space is the most worthy place to search more by sampling more designs. That said, OCBA intends to do a best estimation and NP intends to do a best search within a given computing budget.

Acknowledgements Some of this material was presented at the 2008 INFORMS Annual Meeting and 2008 Winter Simulation Conference [56,57]. This work was supported in part by Department of Energy under Award DE-SC0002223, and NIH under Grant 1R21DK088368-01.

References

1. Fu M C. Are we there yet? The marriage between simulation & optimization. OR/MS Today, 2007, 16–17
2. Rubinstein R Y, Shapiro A. Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method. New York, NY: John Wiley & Sons, 1993
3. Homem-de-Mello T, Shapiro A, Spearman M L. Finding optimal material release times using simulation-based optimization. Management Science, 1999, 45(1): 86–102
4. Kleywegt A, Shapiro A, Homem-de-Mello T. The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization, 2002, 12(2): 479–502
5. Barton R R, Meckesheimer M. Metamodel-based simulation optimization. In: Henderson S G, Nelson B L, eds. Handbooks in Operations Research and Management Science: Simulation. Chapter 18. Amsterdam: Elsevier, 2006, 535–574
6. Kleijnen J. Design and Analysis of Simulation Experiments. New York, NY: Springer, 2008
7. Kushner H J, Yin G G. Stochastic Approximation Algorithms and Applications. 2nd ed. New York, NY: Springer-Verlag, 2003
8. Ólafsson S. Metaheuristics. In: Henderson S G, Nelson B L, eds. Handbooks in Operations Research and Management Science: Simulation. Chapter 21. Amsterdam: Elsevier, 2006, 633–654
9. Fu M C. Optimization via simulation: A review. Annals of Operations Research, 1994, 53(1): 199–247
10. Fu M C. Optimization for simulation: Theory vs. practice (Feature Article). INFORMS Journal on Computing, 2002, 14(3): 192–215
11. Andradóttir S. Simulation optimization. In: Banks J, ed. Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice. Chapter 9. New York, NY: John Wiley & Sons, 1998
12. Andradóttir S. An overview of simulation optimization with random search. In: Henderson S G, Nelson B L, eds. Handbooks in Operations Research and Management Science: Simulation. Chapter 20. Amsterdam: Elsevier, 2006, 617–632
13. Fu M C. Gradient estimation. In: Henderson S G, Nelson B L, eds. Handbooks in Operations Research and Management Science: Simulation. Chapter 19. Amsterdam: Elsevier, 2006, 575–616
14. Fu M C. What you should know about simulation and derivatives. Naval Research Logistics, 2008, 55(8): 723–736
15. Fu M C. Sample path derivatives for (s, S) inventory systems. Operations Research, 1994, 42(2): 351–364
16. Fu M C, Hu J Q. (s, S) inventory systems with random lead times: Harris recurrence and its implications in sensitivity analysis. Probability in the Engineering and Informational Sciences, 1994, 8(3): 355–376
17. Bashyam S, Fu M C. Application of perturbation analysis to a class of periodic review (s, S) inventory systems. Naval Research Logistics, 1994, 41(1): 47–80

18. Bashyam S, Fu M C. Optimization of (s, S) inventory systems with random lead times and a service level constraint. Management Science, 1998, 44(12): S243–S256
19. Pflug G C, Rubinstein R Y. Inventory processes: Quasi-regenerative property, performance evaluation and sensitivity estimation via simulation. Stochastic Models, 2002, 18(3): 469–496
20. Zhang H, Fu M C. Sample path derivatives for (s, S) inventory systems with price determination. In: Golden B L, Raghavan S, Wasil E A, eds. The Next Wave in Computing, Optimization, and Decision Technologies. Boston, MA: Kluwer Academic Publisher, 2005, 229–246
21. Whitt W. What you should know about queuing models to set staffing requirements in service systems. Naval Research Logistics, 2007, 54(5): 476–484
22. Bechhofer R E, Santner T J, Goldsman D M. Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons. New York, NY: John Wiley & Sons, 1995
23. Kim S H, Nelson B L. Selecting the best system. In: Henderson S G, Nelson B L, eds. Handbooks in Operations Research and Management Science: Simulation. Chapter 17. Amsterdam: Elsevier, 2006, 501–534
24. Chen H C, Chen C H, Dai L, Yücesan E. New development of optimal computing budget allocation for discrete event simulation. In: Proceedings of the 1997 Winter Simulation Conference. 1997, 334–341
25. Chen C H, Lin J, Yücesan E, Chick S E. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. Discrete Event Dynamic Systems: Theory and Applications, 2000, 10(3): 251–270
26. Chen C H, Yücesan E, Dai L, Chen H C. Optimal budget allocation for discrete-event simulation experiments. IIE Transactions, 2010, 42(1): 60–70
27. Fu M C, Hu J Q, Chen C H, Xiong X. Simulation allocation for determining the best design in the presence of correlated sampling. INFORMS Journal on Computing, 2007, 19(1): 101–111
28. Glynn P, Juneja S. A large deviations perspective on ordinal optimization. In: Proceedings of the 2004 Winter Simulation Conference. 2004, 577–585
29. Fu M C, Hu J Q, Chen C H, Xiong X. Optimal computing budget allocation under correlated sampling. In: Proceedings of the 2004 Winter Simulation Conference. 2004, 595–603
30. Brantley M W, Lee L H, Chen C H, Chen A. Optimal sampling in design of experiment for simulation-based stochastic optimization. In: Proceedings of 2008 IEEE Conference on Automation Science and Engineering. 2008, 388–393
31. Lee L H, Chew E P, Teng S Y, Goldsman D. Optimal computing budget allocation for multi-objective simulation models. In: Proceedings of the 2004 Winter Simulation Conference. 2004, 586–594
32. Chew E P, Lee L H, Teng S Y, Koh C H. Differentiated service inventory optimization using nested partitions and MOCBA. Computers & Operations Research, 2009, 36(5): 1703–1710
33. Lee L H, Chew E P, Teng S Y, Goldsman D. Finding the non-dominated Pareto set for multi-objective simulation models. IIE Transactions, 2010, 42(9): 656–674
34. Chick S E, Wu Y Z. Selection procedures with frequentist expected opportunity cost. Operations Research, 2005, 53(5): 867–878
35. He D, Chick S E, Chen C H. The opportunity cost and OCBA selection procedures in ordinal optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2007, 37(5): 951–961
36. Trailovic L, Pao L Y. Computing budget allocation for efficient ranking and selection of variances with application to target tracking algorithms. IEEE Transactions on Automatic Control, 2004, 49(1): 58–67
37. Morrice D J, Brantley M W, Chen C H. An efficient ranking and selection procedure for a linear transient mean performance measure. In: Proceedings of the 2008 Winter Simulation Conference. 2008, 290–296
38. Morrice D J, Brantley M W, Chen C H. A transient means ranking and selection procedure with sequential sampling constraints. In: Proceedings of the 2009 Winter Simulation Conference. 2009, 590–600
39. Chen C H, He D, Fu M C, Lee L H. Efficient simulation budget allocation for selecting an optimal subset. INFORMS Journal on Computing, 2008, 20(4): 579–595
40. Pujowidianto N A, Lee L H, Chen C H, Yep C M. Optimal computing budget allocation for constrained optimization. In: Proceedings of the 2009 Winter Simulation Conference. 2009, 584–589
41. Yan S, Zhou E, Chen C H. Efficient simulation budget allocation for selecting the best set of simplest good enough designs. In: Proceedings of the 2010 Winter Simulation Conference. 2010, 1152–1159
42. Chick S E, Inoue K. New two-stage and sequential procedures for selecting the best simulated system. Operations Research, 2001, 49(5): 732–743
43. Chick S E, Inoue K. New procedures to select the best simulated system using common random numbers. Management Science, 2001, 47(8): 1133–1149
44. Branke J, Chick S E, Schmidt C. Selecting a selection procedure. Management Science, 2007, 53(12): 1916–1932
45. Chen C H. A lower bound for the correct subset-selection probability and its application to discrete event system simulations. IEEE Transactions on Automatic Control, 1996, 41(8): 1227–1231
46. Chen C H, Lee L H. Stochastic Simulation Optimization: An Optimal Computing Budget Allocation. Singapore: World Scientific Publishing Company, 2011
47. Rinott Y. On two-stage selection procedures and related probability inequalities. Communications in Statistics, 1978, 7(8): 799–811
48. Shi L, Ólafsson S. Nested partitions method for global optimization. Operations Research, 2000, 48(3): 390–407
49. Shi L, Ólafsson S. Nested partitions method for stochastic optimization. Methodology and Computing in Applied Probability, 2000, 2(3): 271–291
50. Shi L, Ólafsson S. Nested Partitions Optimization: Methodology and Applications. New York, NY: Springer, 2008
51. Shi L, Ólafsson S. Nested partitions optimization. In: Tutorials in Operations Research. Chapter 1. Hanover, MD: INFORMS, 2007, 1–22
52. Shi L, Ólafsson S, Sun N. New parallel randomized algorithms for the traveling salesman problem. Computers & Operations Research, 1999, 26(4): 371–394
53. Ho Y C, Sreenivas R S, Vakili P. Ordinal optimization of DEDS. Discrete Event Dynamic Systems: Theory and Applications, 1992, 2(4): 61–88
54. Ho Y C, Cassandras C G, Chen C H, Dai L. Ordinal optimization and simulation. Journal of the Operational Research Society, 2000, 51(4): 490–500

55. Shi L, Chen C H. A new algorithm for stochastic discrete resource allocation optimization. Discrete Event Dynamic Systems: Theory and Applications, 2000, 10(3): 271–294
56. Chen C H, Fu M C, Shi L. Simulation and optimization. In: Tutorials in Operations Research. Chapter 11. Hanover, MD: INFORMS, 2008, 247–260
57. Fu M C, Chen C H, Shi L. Some topics for simulation optimization. In: Proceedings of the 2008 Winter Simulation Conference. 2008, 27–38

Chun-Hung CHEN received his Ph.D degree in engineering sciences from Harvard University in 1994. He is a Professor of Systems Engineering and Operations Research at George Mason University (GMU) and is also affiliated with Taiwan University. Dr. Chen was an Assistant Professor of Systems Engineering at the University of Pennsylvania before joining GMU. Sponsored by NSF, NIH, DOE, NASA, MDA, and FAA, he has worked on the development of very efficient methodology for stochastic simulation optimization and its applications to air transportation system, semiconductor manufacturing, healthcare, security network, power grids, and missile defense system. Dr. Chen received the Best Automation Paper Award from the 2003 IEEE International Conference on Robotics and Automation, 1994 Eliahu I. Jury Award from Harvard University, and the 1992 MasPar Parallel Computer Challenge Award. Dr. Chen has served as Co-Editor of the Proceedings of the 2002 Winter Simulation Conference and Program Co-Chair for 2007 Informs Simulation Society Workshop. He is currently a Department Editor for IIE Transactions, Associate Editor of IEEE Transactions on Automatic Control, Area Editor of Journal of Simulation Modeling Practice and Theory, and Associate Editor of International Journal of Simulation and Process Modeling.

Leyuan SHI received the B.S. degree in mathematics from Nanjing Normal University in 1982, the M.S. degree in applied mathematics from Tsinghua University in 1985, and the M.S. degree in engineering and the Ph.D degree in applied mathematics from Harvard University in 1990 and 1992, respectively. She is a Professor with the Department of Industrial and Systems Engineering, University of Wisconsin-Madison. She has been involved in undergraduate and graduate teaching, as well as research and professional service. Her research is devoted to the theory and applications of large-scale optimization algorithms, discrete-event simulation and modeling, and analysis of discrete dynamic systems. She has published many papers in these areas. Her work has appeared in Discrete Event Dynamic Systems, Operations Research, Management Science, the IEEE Transactions, and the IIE Transactions. Dr. Shi is an IEEE Fellow and member of the Institute for Operations Research and the Management Sciences (INFORMS). She is currently an Associate Editor of the IEEE Transactions on Automation Science and Engineering, INFORMS Journal on Computing, and the Journal of Discrete Event Dynamic Systems.

Loo Hay LEE is an Associate Professor in the Department of Industrial and Systems Engineering at National University of Singapore and was a Visiting Professor at the Department of Systems Engineering and Operations Research at George Mason University in 2007. He is currently the deputy head of research and graduate studies for the department. He received his B.S. (Electrical Engineering) degree from Taiwan University in 1992 and his S.M. and Ph.D degrees in 1994 and 1997 from Harvard University. He has sat in the editorial boards for journals such as IIE Transactions, Flexible Services and Manufacturing Journal, the Asia Pacific Journal of Operational Research, Journal of Simulation and is a member in the advisory board for OR Spectrum. He is currently a senior member of IEEE, and a council member of the simulation society of INFORMS. His research focuses on the simulation-based optimization, maritime logistics which includes port operations and the modeling and analysis for the logistics and supply chain system. He has co-authored a book: Stochastic Simulation Optimization — An Optimal Computing Budget Allocation (World Scientific).