Efficient Selection of a Set of Good Enough Designs with Complexity Preference

Shen Yan, Enlu Zhou, *Member, IEEE,* and Chun-Hung Chen, *Senior Member, IEEE*

***Abstract*—Many automation or manufacturing systems are large, complex, and stochastic. Since closed-form analytical solutions generally do not exist for such systems, simulation is the only faithful way for performance evaluation. From the practical engineering perspective, the designs (or solution candidates) with low complexity (called simple designs) have many advantages compared with complex designs, such as requiring less computing and memory resources, and easier to interpret and to implement. Therefore, they are usually more desirable than complex designs in the real world if they have good enough performance. Recently, Jia [1] discussed the importance of design simplicity and introduced an adaptive simulation-based sampling algorithm to sequentially screen the designs until one simplest good enough design is found. In this paper, we consider a more generalized problem and introduce two algorithms OCBA-mSG and OCBA-bSG to identify a *subset* of m simplest and good enough designs among a total of K ($K > m$) designs. By controlling the simulation allocation intelligently, our approach intends to find those simplest good enough designs using a minimum simulation time. The numerical results show that both OCBA-mSG and OCBA-bSG outperform some other approaches on the test problems.**

***Note to Practitioners*—This paper was motivated by two problems from the real world: design of ordering policies in inventory control, and design of node activation rules in sensor networks. In designing a good ordering policy or a good sensor activation rule, simple designs have various advantages in practice, e.g., easy to learn, to implement, to operate, and to maintain. More generally, practitioners want to find a design or a set of designs which not only have good performance but are also simple. Due to the complexity of the systems, simulation is a popular tool in industry to evaluate the performance of different alternative designs before actual implementation. While the advance of new technology has dramatically increased computational power, efficiency is still a big concern. Our proposed approach intelligently controls the simulation of alternative designs, so that the simplest good enough designs can be selected using a minimum computation cost. Our numerical experiments show that the proposed approach is very effective.**

***Index Terms*—optimal computing budget allocation, ranking and selection, simulation-based optimization, complexity.**

Shen Yan was a Master student in the Department of Industrial & Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, IL, 61801 USA (yanshen1987@gmail.com).

Enlu Zhou is with the faculty in the Department of Industrial & Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, IL, 61801 USA (enluzhou@illinois.edu).

Chun-Hung Chen (corresponding author) is with the faculty in Department of Electrical Engineering & Institute of Industrial Engineering, National Taiwan University, Taipei, Taiwan (cchen9@gmu.edu).

This work was supported by the National Science Foundation under Grants ECCS-0901543 and CMMI-1130273, the Air Force Office of Scientific Research under Grant FA-9550-12-1-0250, the Department of Energy under Grant DE-SC0002223, National Institutes of Health under Grant 1R21DK088368-01, and National Science Council of Taiwan under Grant NSC-100-2218-E-002-027-MY3.

I. Introduction

The motivation for considering the selection of simplest good enough designs comes from the real world, where simple designs are preferred to the complex ones if the simple designs are good enough to satisfy our requirements. A typical example is to find an ordering policy in inventory control. Consider the problem of ordering a certain amount of products at each period to meet a stochastic demand which follows a probability distribution. In order to minimize the expected cost (including holding cost for excess inventory and shortage cost for unfilled demand), we want to determine the optimal ordering policy in each period. The optimal policy can be found analytically for some models to have the structure of a base-stock policy or an (s,S) policy [2], [3], [4], [5]. The base-stock policy is a threshold function that maps the current stock into the ordering amount. Motivated by this simple structure of the optimal policy for some models, we can approximate the optimal policies for other more complex inventory models by threshold policies. In general, we can approximate the optimal ordering policy better with a greater number of thresholds given the right values of these thresholds. Then the problem is to decide the number and the values of the thresholds. It is clear that with more thresholds in the function we have a more complex ordering policy, making it harder to determine the optimal values of these thresholds and to implement in practice. Conversely, with a small number of thresholds, such as one (base-stock policy) or two, we have a simple ordering policy, which is easier to compute and to implement. If the simple ordering policy can achieve a required cost criterion, it will be more desirable than a complex ordering policy, even though the complex policy may yield a lower cost. Another example is the design of node activation rules in the wireless sensor networks (WSNs), as described in [6], [1]. Each node needs to collaborate with its neighbors in order to have enough power to monitor an area of interest. Similarly, given that the required probability of correct detection can be achieved, we prefer small communication radius of each node (i.e., simple activation rule) to large radius (i.e., complex activation rule).

In this paper, we use the word "design" to refer to the object under consideration, such as the ordering policy and the node activation rule in the previous examples. We consider the problem of selecting m ($m \geq 1$) designs that are simplest (with smallest complexity) and good enough (satisfying a constraint on the performance measure). Selection of multiple designs is sometimes preferred because of at least two reasons. First, a decision maker has to face different objectives and constraints. However, in many cases it is too complicated to include all objectives and constraints in the simulation model. As a result, the decision maker may not like the best design obtained from the model. By offering a set of multiple good designs, the decision maker can choose the one he/she likes by considering more factors. Second, the design space can be extremely large and the total simulation cost is too expensive. A common approach is to first apply a simplified model to screen out some good alternatives before the full-scale simulation modeling analysis. Offering a set of multiple good designs is highly useful for this purpose.

The complexity of a design is represented by an integer number, where simpler design has a smaller integer number. The complexity is a deterministic value known before simulation. However, the performance of a design is subject to system noise, and hence, it can only be estimated from simulation, which is often computationally expensive. For example, it takes a significant amount of computational effort to simulate the inventory system in order to evaluate the cost of a particular ordering policy. Hence, our goal is to allocate a given simulation budget efficiently to the designs so as to maximize the probability of correctly selecting the m simplest good enough designs out of a total of K designs.

The above problem is closely related with many known results in the literature on ranking and selection (R&S). Several recent R&S procedures are discussed and compared by Branke et al. [7]. Some of the procedures can be further extended to more generalized simulation optimization problems (e.g., [8], [9], [10], [11], [12], [13]). For subset selection problem, Gupta [14] proposed the method of selecting a random size subset containing the best design with a given probability of correct selection. Later, Santner [15] extended Gupta's method by imposing a maximum size m on the subset. Koenig and Law [16] developed a two-stage procedure for selecting the top m designs with best performance, following the results in Dudewicz and Dalal [17]. Chen et al. [18], [19], [20] developed the optimal computing budget allocation (OCBA) procedure for the selection of one best design, and later Chen et al. [21] extended OCBA to the selection of the m best designs. However, all of this work has focused on optimizing a single-objective performance measure.

Problems of multi-objective optimization and constraint optimization have also been studied. Lee et al. [22], Teng et al. [23], Chew et al. [24], and Lee et al. [25] extended the OCBA framework to efficiently select designs that optimize multiple performance measures. Branke and Mattfeld [26] proposed to search for solutions that are not only good but also flexible in dynamic scheduling. Branke and Gamer [27] proposed an efficient sampling procedure in interactive multi-criterion selection. In constraint optimization, Andradóttir et al. [28] proposed a two-phase approach which identifies all the feasible systems first and then selects the best from them. Szechtman and Yücesan [29] used large deviation theory to deal with feasibility determination. Most recently, Pujowidianto et al. [30] developed OCBA further for selecting one single best design under multiple constraints of secondary performance measures.

The problem of considering both complexity and performance evaluation has only been considered recently. It appears to be a multi-objective optimization or a constraint optimization problem, but it has its unique problem structure that can be exploited to design a more efficient sampling procedure. The most relevant problem to ours is probably the selection of one simplest good design, for which Jia [1] proposed an Adaptive Sampling Algorithm (ASA) to minimize the Type II error of the chosen design. The relation between complexity and performance in choosing policies has also been explored in the context of Markov decision processes [31], [32].

In this paper, we address this problem of selecting multiple simplest good enough designs by proposing the algorithm OCBA-mSG, abbreviated for optimal computing budget allocation for m simplest good enough designs. Based on OCBA-mSG, we develop another slightly different algorithm called OCBA-bSG for selecting the designs with the best performance from all the simplest good enough designs, with a slight increase of simulation budget than OCBA-mSG. Numerical results indicate that both OCBA-mSG and OCBA-bSG allocate the simulation budget efficiently to achieve a high probability of correct selection.

The rest of the paper is organized as follows. In Section II, we define the two problems of selecting m simplest good enough designs and selecting the *best m* simplest good enough designs, respectively. In Section III, we state the main results (proofs are included in the Appendix) and present the algorithms. In Section IV, we carry out the numerical experiments on several test problems. Finally, we conclude our paper in Section V.

II. PROBLEM STATEMENT

A. Selecting m Simplest Good Enough Designs

Let θ denote a design, and Θ denote the set of all the K ($K > m$) designs, i.e.,

$$\Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}.$$

To simplify notations, we will also use the integers $1, 2, \ldots, K$ to denote the designs in the following when there is no ambiguity. The performance of the design θ_k is measured by

$$J_k = E[L(\theta_k, \zeta)],$$

where ζ is a random vector that represents the uncertainty in the system, and $L(\theta_k, \zeta)$ can only be evaluated through simulation of the system. The underlying assumption is that such simulation is expensive. A design is considered better if its performance measure J is smaller. A *good enough design* is one that satisfies $J_k < J_0$, where J_0 is a given threshold on the performance. In practice, J_0 can be set by the user or chosen based on a few pilot runs which return a rough estimate of the performance of the designs. Please note that the definition of "good enough" here is the same as "feasible", which is different from the definition in the literature on ordinal optimization (c.f. [33]). In the rest of the paper we will use the words "good enough" and "feasible" interchangeably. Hence, the good enough set (or the feasible set) is defined as

$$F = \{k | J_k < J_0, k = 1, 2, \ldots, K\}.$$

The complexity of the design θ_k is represented by the complexity $C(\theta_k)$, which is a deterministic value in the set $\{0,1,\ldots,n\}, n<K$, and is known before simulation. Note that $C(\theta)$ is the result of the user mapping his/her definition of complexity to integer numbers. For example, the user could map the number of thresholds of the ordering policy to an integer; or the user could map a range of communication radius in the WSN problem to an integer. The complexity set C_i contains all the designs with complexity i, defined as

$$C_i = \{k | C(\theta_k) = i, k = 1,2,\ldots,K\}.$$

The set of m simplest and good enough designs is defined as

$$S_m = \{m_1, m_2, \ldots, m_m \ \mid \ C(\theta_{m_i}) \leqslant C(\theta_k), \forall k \in F \setminus S_m\},$$

where $F \setminus S_m = \{k \in F | k \notin S_m\}$. Notice that the set S_m may not be unique, because it is possible that multiple designs in the set F have the same complexity. For example, if there are $\underline{m}$ ($\underline{m} < m$) designs in F with complexity 0 and $\bar{m}$ ($\bar{m} > m - \underline{m}$) designs in F with complexity 1, then S_m includes all the $\underline{m}$ designs with complexity 0 and *any* $m-\underline{m}$ designs of those $\bar{m}$ designs with complexity 1.

Fig. 1 gives a pictorial view of the general case. Suppose that all the designs in the complexity sets C_0, C_1, $\ldots$, C_{t-1} ($1 \le t \le n$) are not good enough (or infeasible) and the first feasible design appears in the set C_t. Moreover, suppose that the total number of feasible designs in $C_t, C_{t+1}, \ldots, C_{t'}$ is less than m until t' reaches $t+p$ ($0 \le p \le n-t$). Hence, in general we need to consider three types of subsets, which we refer to as: infeasible simplest subsets S_{d_i}, $i = 0,1,\ldots,t-1$; simplest good enough subsets S_{s_i}, $i = 0,1,\ldots,p$; and infeasible non-simplest subsets S_{e_i}, $i = 0,1,\ldots,p$. In particular, the simplest good enough subsets $S_{s_0}, S_{s_1}, \ldots, S_{s_p}$ satisfy

$$\sum_{i=0}^{p-1} |S_{s_i}| < m, \quad \sum_{i=0}^{p} |S_{s_i}| \geqslant m,$$

where $|\cdot|$ denotes the cardinality of the set. Therefore, according to the definition of the m simplest and good enough designs, S_m should include all the designs in the subsets $S_{s_0}, S_{s_1}, \ldots, S_{s_{p-1}}$ and *any* $(m - \sum_{i=0}^{p-1} |S_{s_i}|)$ designs in the subset S_{s_p}. Since there are already m designs selected in the lower complexity sets $C_0, \ldots, C_{t+p}$, there is no need to consider the higher complexity sets.

In simulation, we compute the sample mean $\bar{J}$ based on the samples on hand to estimate the performance J for each design, and then order the designs to find the subsets $\{\hat{S}_{s_i}, i = 0,1,\ldots,\hat{p}\}$, $\{\hat{S}_{d_i}, i = 0,1,\ldots,\hat{t}-1\}$ and $\{\hat{S}_{e_i}, i = 0,1,\ldots,\hat{p}\}$ as estimates for S_{s_i}, S_{d_i} and S_{e_i} respectively according to the relationship shown in Fig. 1. Hence, the selected set $\hat{S}_m$ should include all the designs in $\hat{S}_{s_0}, \hat{S}_{s_1}, \ldots, \hat{S}_{s_{\hat{p}-1}}$ and *any* $(m - \sum_{i=0}^{\hat{p}-1} |\hat{S}_{s_i}|)$ designs in $\hat{S}_{s_{\hat{p}}}$. To further classify the subsets, we denote

$$\hat{S}_s = \bigcup_{i=0}^{\hat{p}} \hat{S}_{s_i}, \quad \hat{S}_I = \left\{ \cup_{i=0}^{\hat{p}} \hat{S}_{e_i} \right\} \bigcup \left\{ \cup_{i=0}^{\hat{t}-1} \hat{S}_{d_i} \right\}.$$

We define the correct selection CS_m as the event that the designs in $\hat{S}_s$ are the true simplest good enough designs, i.e.,

$$CS_m = \{J_i < J_0 \ \& \ J_j \geqslant J_0, \forall i \in \hat{S}_s, \forall j \in \hat{S}_I\},$$

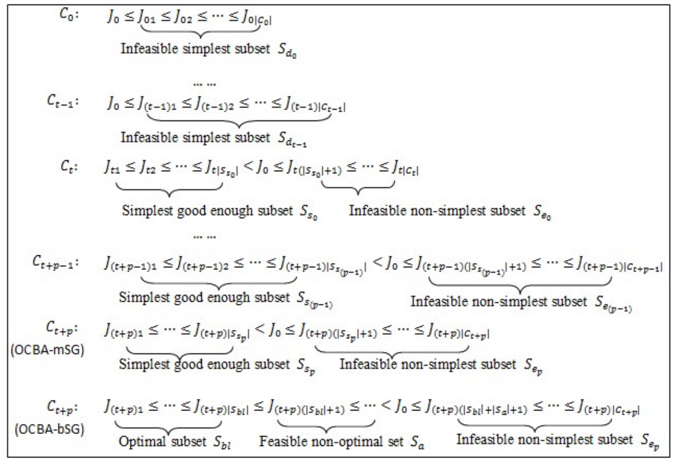


Fig. 1: Relationship between subsets. J_{ij} denotes the performance of a design whose complexity is i and whose performance is the j^{th} smallest in its complexity set C_i.

where $\hat{S}_s$ includes all the simplest good enough designs, $\hat{S}_I$ includes all the infeasible (either simplest or non-simplest) designs.

The determination of $\hat{S}_s$ and $\hat{S}_I$ is based on the estimate (sample mean) of the performance of every design, and the accuracy of the estimate is determined by the number of simulations carried out for that design. Therefore, the decision variables that determine the probability of correct selection $P(CS_m)$ are the number of simulations $N_1, N_2, \ldots, N_K$ allocated for the designs $\theta_1, \theta_2, \ldots, \theta_K$, respectively. This will be more clearly shown in the explicit expression (3) for $P(CS_m)$ later. Given a fixed total simulation budget T, we want to find the optimal budget allocation $N_1, N_2, \ldots, N_K$ to the K designs in order to maximize the probability of correct selection:

$$\begin{aligned} \max_{N_1,N_2,\ldots,N_K} \quad & P(CS_m) \\ \text{s.t.} \quad & N_1 + N_2 + \ldots + N_K = T. \end{aligned} \tag{1}$$

B. Selecting the Best m Simplest Good Enough Designs

Consider a simple example that there are two good enough designs with complexity 0, say A and B, so they are both simplest good enough designs. If we only need one simplest good enough design, then we can choose either A or B. However, if A and B have different performance, say $J_A < J_B$, then we would prefer A, because it is better than B in performance and as simple as B. This is what we refer to as the "best simplest good enough design". The formal definition of the *best m* simplest good enough designs is as follows:

$$\begin{aligned} S_b \quad = \quad & \{b_1, \ldots, b_m \in F \ \mid \ C(\theta_{b_i}) < C(\theta_k) \text{ OR} \\ & J_{b_i} < J_k \text{ if } C(\theta_{b_i}) = C(\theta_k), \forall k \in F \setminus S_b\}, \end{aligned}$$

where $F \setminus S_b = \{k \in F | k \notin S_b\}$. The key difference from S_m is that S_b includes all the feasible designs in the subsets $S_{s_0}, S_{s_1}, \ldots, S_{s_{p-1}}$ and the *best* $(m - \sum_{i=0}^{p-1} |S_{s_i}|)$ designs in the subset S_{s_p}. That implies that the subset S_{s_p} should be further divided into two subsets: optimal subset S_{bl}, and feasible non-optimal set S_a. Fig. 1 gives a pictorial view of the relationship between the subsets. Therefore, the optimal set S_b satisfies

$$S_b = \left\{ \cup_{i=0}^{p-1} S_{s_i} \right\} \bigcup S_{bl}, \quad |S_b| = m.$$

In simulation, we find estimates for these subsets based on the sample means of the designs, and similarly we denote

$$\hat{S}_{b^-} = \bigcup_{i=0}^{\hat{p}-1} \hat{S}_{s_i}, \quad \hat{S}_I = \left\{ \cup_{i=0}^{\hat{p}} \hat{S}_{e_i} \right\} \bigcup \left\{ \cup_{i=0}^{\hat{t}-1} \hat{S}_{d_i} \right\}.$$

Then the correct selection CS_b is defined as

$$\begin{aligned} CS_b = \{ & J_i < J_0 \ \& \ J_j \leqslant J_k < J_0 \ \& \ J_s \geqslant J_0, \\ & \forall i \in \hat{S}_{b^-}, \forall j \in \hat{S}_{bl}, \forall k \in \hat{S}_a, \forall s \in \hat{S}_I \}. \end{aligned}$$

Our goal is to find the optimal budget allocation $N_1, N_2, \ldots, N_K$ to the K designs in order to maximize the probability of correct selection given a fixed total simulation budget T:

$$\begin{aligned} \max_{N_1, N_2, \ldots, N_K} \quad & P(CS_b) \\ \text{s.t.} \quad & N_1 + N_2 + \ldots + N_K = T \end{aligned} \tag{2}$$

III. MAIN RESULTS

A. Selecting m Simplest Good Enough Designs

We estimate $P(CS)$ using the same Bayesian model presented in [34] and [35]. Assuming that the performance of each design, J_i, has a noninformative normal prior distribution $N(0, \nu^2)$ with ν^2 extremely large, and a sample $\hat{J}_i$ for J_i is normally distributed as $N(J_i, \sigma_i^2)$, then the posterior distribution of J_i has been shown in [34] to be

$$\tilde{J}_i \sim N(\bar{J}_i, \frac{{\sigma_i}^2}{N_i}),$$

where $\bar{J}_i = \frac{1}{N_i} \sum_{k=1}^{N_i} \hat{J}_i(k)$, and $\hat{J}_i(1), \hat{J}_i(2), \ldots, \hat{J}_i(N_i) \overset{\text{iid}}{\sim} N(J_i, \sigma^2)$. Thus, $P(CS_m)$ is as follows:

$$\begin{aligned} P(CS_m) &= P\{\tilde{J}_i < J_0 \ \& \ \tilde{J}_j \geqslant J_0, \forall i \in \hat{S}_s, \forall j \in \hat{S}_I\} \\ &= \prod_{i \in \hat{S}_s} P\{\tilde{J}_i < J_0\} \prod_{j \in \hat{S}_I} P\{\tilde{J}_j \geqslant J_0\}, \end{aligned} \tag{3}$$

where the second equation is due to the independence between designs. The results are stated in the following theorem, and the proof is contained in the Appendix.

Theorem 1. *$P(CS_m)$ is asymptotically (as $T \to \infty$) maximized by the following allocation rule:*

$$\frac{N_i}{\sigma_i^2 / (\bar{J}_i - J_0)^2} = \frac{N_j}{\sigma_j^2 / (\bar{J}_j - J_0)^2}, \tag{4}$$

for all $i \in \hat{S}_s$ and $j \in \hat{S}_I$. $N_k = 0$ for all other $k \in \{1, 2, \ldots, K\}$.

Remark 1. From (4), we know that the simulation budget for each design increases proportionally to its corresponding sample variance. If a design has a larger sample variance, more simulation budget will be allocated to it in order to obtain a more accurate estimate for the performance in the next iteration. On the other hand, the simulation budget for each design decreases proportionally to the difference between its sample mean and the good enough threshold J_0. The design whose sample mean of the performance is closer to J_0 will be assigned more simulation budget, since it is more sensitive to the feasibility test. As there are already m designs selected from $\hat{S}_s \bigcup \hat{S}_I$ in the lower complexity sets, there is no need to consider the higher complexity sets, and hence, there is no more simulation budget allocated to the designs that are not in $\hat{S}_s \bigcup \hat{S}_I$. However, as more simulation is carried out and the sample means are updated, $\hat{S}_s$ and $\hat{S}_I$ may become different at the next iteration and contain some of the higher-complexity sets that are not considered in the previous iteration.

B. Selecting the Best m Simplest Good Enough Designs

It is hard to maximize $P(CS_b)$ (problem (2)) analytically, and hence we maximize a lower bound of $P(CS)_b$, which is called Approximate Probability of Correct Selection $APCS_b$ [18]. $APCS_b$ is defined as follows.

$$\begin{aligned} P(CS)_b &= P\{\tilde{J}_i < J_0 \ \& \ \tilde{J}_j \leqslant \tilde{J}_k < J_0 \ \& \ \tilde{J}_s \geqslant J_0, \\ &\quad \forall i \in \hat{S}_{b^-}, \forall j \in \hat{S}_{bl}, \forall k \in \hat{S}_a, \forall s \in \hat{S}_I\} \\ &\geqslant P\{\tilde{J}_i < J_0 \ \& \ \tilde{J}_j \leqslant \mu \ \& \ \mu \leqslant \tilde{J}_k < J_0 \ \& \ \tilde{J}_s \geqslant J_0, \\ &\quad \forall \theta_i \in \hat{S}_{b^-}, \forall j \in \hat{S}_{bl}, \forall k \in \hat{S}_a, \forall s \in \hat{S}_I\} \\ &= \prod_{i \in \hat{S}_{b^-}} P\{\tilde{J}_i < J_0\} \prod_{j \in \hat{S}_{bl}} P\{\tilde{J}_j \leqslant \mu\} \prod_{k \in \hat{S}_a} \\ &\quad P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{s \in \hat{S}_I} P\{\tilde{J}_s \geqslant J_0\} \qquad (5) \\ &\triangleq APCS_b, \end{aligned}$$

where the second equation is due to the independence between the designs. It is easy to see that a larger $APCS_b$ yields a better approximation for $P(CS_b)$. Following a similar procedure as in [21], we determine the value of μ as stated in the following Lemma.

Lemma 2. *Let $\theta_{[r]}$ denote the design with the largest sample mean in the subset $\hat{S}_{bl}$, and $\theta_{[r+1]}$ denote the design with the smallest sample mean in the subset $\hat{S}_a$. Then the μ value introduced in $APCS_b$ is given by*

$$\mu = \frac{\hat{\sigma}_{[r+1]} \bar{J}_{[r]} + \hat{\sigma}_{[r]} \bar{J}_{[r+1]}}{\hat{\sigma}_{[r]} + \hat{\sigma}_{[r+1]}}, \tag{6}$$

where $\hat{\sigma}_i = \sigma_i / \sqrt{N_i}$.

Therefore, instead of solving the maximization problem (2), we consider the following maximization problem.

$$\begin{aligned} \max_{N_1, N_2, \ldots, N_K} \quad & APCS_b \\ \text{s.t.} \quad & N_1 + N_2 + \ldots + N_K = T. \end{aligned} \tag{7}$$

The results are given in the following theorem, and the proof is contained in the Appendix.

Theorem 3. *$APCS_b$ is asymptotically (as $T \to \infty$) maximized by the following allocation rule:*

Case 1: If $\hat{S}_a \neq \varnothing$ (i.e., the total number of feasible designs is greater than m), then

$$\begin{aligned} \frac{N_i}{\sigma_i^2 / (\bar{J}_i - J_0)^2} &= \frac{N_j}{\sigma_j^2 / (\bar{J}_j - \mu)^2} = \frac{N_s}{\sigma_s^2 / (\bar{J}_s - J_0)^2} \\ &= \frac{N_x}{\sigma_x^2 / (\bar{J}_x - \mu)^2} = \frac{N_y}{\sigma_y^2 / (\bar{J}_y - J_0)^2}, \end{aligned} \tag{8}$$

for all $i \in \hat{S}_{b^-}$, $j \in \hat{S}_{bl}$, $s \in \hat{S}_I$, $x \in \{k \in \hat{S}_a | \bar{J}_k \leqslant \frac{\mu+J_0}{2}\}$, $y \in \{k \in \hat{S}_a | \bar{J}_k > \frac{\mu+J_0}{2}\}$. $N_k = 0$ *for all other* $k \in \{1,2,\ldots,K\}$.

Case 2: If $\hat{S}_a = \varnothing$ *(i.e., the total number of feasible designs is less than or equal to m), then*

$$\frac{N_i}{\sigma_i^2/(\bar{J}_i - J_0)^2} = \frac{N_s}{\sigma_s^2/(\bar{J}_s - J_0)^2} \tag{9}$$

for all $i \in \hat{S}_{b^-} \bigcup \hat{S}_{bl}$ *and* $s \in \hat{S}_I$. $N_k = 0$ *for all other* $k \in \{1,2,\ldots,K\}$.

Remark 2. Theorem 2 provides some intuitive results. We notice that at the two critical points μ and J_0: μ is the threshold for the optimality, J_0 is the threshold for the feasibility. The designs closer to these two points will be assigned more simulation budget. For the subsets $\hat{S}_{b^-}$ and $\hat{S}_I$, we are only interested in determining wether the designs are good enough, and indeed more simulation budget is assigned to the designs near J_0. Similarly, for the subset $\hat{S}_{bl}$, we are only interested in comparing the performance of the designs, and more simulation budget is assigned to the designs around μ. For the subset S_a where both μ and J_0 are critical points, the last two terms in (8) imply that we should divide the set into two parts by the midpoint $\frac{\mu+J_0}{2}$: the designs with sample means in the range $\mu \leqslant \bar{J}_x \leqslant \frac{\mu+J_0}{2}$ will be compared with μ, and the ones closer to μ will get more simulation budget; the designs falling into the range $\frac{\mu+J_0}{2} < \bar{J}_y \leqslant J_0$ will be compared with J_0, and be assigned more simulation budget if closer to J_0. Please see fig. 2 for a pictorial view of the budget allocation in the complexity set $C_{\hat{t}+\hat{p}}$, the highest complexity set under consideration.

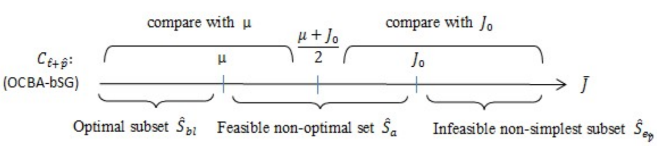


Fig. 2: Simulation allocation in the set $C_{\hat{t}+\hat{p}}$.

Remark 3. Comparing selecting S_b with S_m, the difference is in C_{t+p}, where the subset S_{s_p} in selecting S_m is divided into two subsets S_{bl} and S_a in selecting S_b. Theorem 3 implies that in addition to allocating more simulation budget to the designs near J_0 in the sets $C_0, C_1 \ldots, C_{t+p}$, we also allocate simulation budget to designs near μ in the set C_{t+p}. As a result, the extra simulation budget assigned to designs near μ in selecting S_b is approximately $2/(t+2p+2)$ of the total simulation budget in selecting S_m. If $t=0$ and $p=0$ (i.e., there are more than m feasible designs in the lowest complexity set C_0), then selecting S_b needs approximately double simulation budget of that in selecting S_m for the same accuracy of the sample means of the design performance. On the other hand, if $t+2p$ is large, selecting S_b requires little extra simulation budget, and will be preferred since it yields the *best m* designs among all simplest and good enough designs.

C. OCBA-mSG and OCBA-bSG

Based on the above results, we propose the Optimal Computing Budget Allocation procedure for selecting m Simplest and Good enough designs (OCBA-mSG) and that for selecting the Best m Simplest and Good enough designs (OCBA-bSG). Since the two algorithms are similar, we describe them together to save space and specify the different steps in the description.

OCBA-mSG and OCBA-bSG

Input: the total number of the designs K, the number of designs needed m, the total simulation budget T, the simulation budget increase at each iteration Δ, the initial simulation budget for every design n_0, the good enough performance constraint J_0, and the upper bound of the total simulation budget for one design NU.

Initialize: $l = 0$.

- Group the designs according to their complexities to obtain the complexity sets $C_0, C_1, \ldots, C_n$.
- Perform n_0 simulation replications for all designs to generate samples X_i^k, $k = 1,2,\ldots,n_0$, $i = 1,2,\ldots,K$. Set $N^l = Kn_0$.

Loop: while $N^l < T$, do

1) Update:
   - For each design i, compute the sample mean $\bar{J}_i = \frac{1}{N_i^l}\sum_{k=1}^{N_i^l} X_i^k$, and the sample standard deviation $\sigma_i = \sqrt{\sum_{k=1}^{N_i^l}(X_i^k - \bar{J}_i)^2/(N_i^l - 1)}$. Sort the designs in each complexity set according to their sample means in the increasing order.
   - Increase the computing budget $N^{l+1} = \min\{N^l + \Delta, T\}$.
2) Allocate:

   OCBA-mSG
   - For each design θ_i, compute the simulation budget N_i^{l+1} according to (4).

   OCBA-bSG
   - If the total number of feasible designs is greater than m, compute μ according to (6), and compute the simulation budget N_i^{l+1} for each design θ_i according to (8).
   - Otherwise, compute the simulation budget N_i^{l+1} for each design θ_i according to (9).
3) Simulate:
   - If $N_i^{l+1} \geqslant NU$ or $N_i^{l+1} \leqslant N_i^l$, we set $N_i^{l+1} = N_i^l$, and do not simulate design θ_i at this iteration.
   - Otherwise, perform $(N_i^{l+1} - N_i^l)$ simulations for design θ_i to generate more samples X_i^k, $k = N_i^l + 1, N_i^l + 2, \ldots, N_i^{l+1}$.
4) Update: $l \to l+1$.

End of loop

Output: output the feasible designs starting from the lowest complexity set in the increasing order of their sample means until the total number of such designs reaches m or all the designs have been examined.

Remark 4. In the above algorithms, we introduce an upper bound NU on the simulation budget for one single design: if $N_i \geqslant NU$, we stop allocating new simulation budget to that design. That is because we obtain the simulation budget allocation rules under the asymptotic limit $T \to \infty$ but the actual total simulation budget T is finite. When T is infinity, we can assign a large amount of budget to one design at one iteration, and there will always be enough budget left for other designs if needed at future iterations. This is not true when T is finite. Thus, we introduce NU and determine its value in the following way. Since the designs near the critical points need more simulation budget, we need to ensure each of such critical designs will be simulated at least once. Hence, we approximate the upper bound by counting the number of subsets related to the critical points after initialization, where those subsets are $\hat{S}_{s_i}$, $\hat{S}_{d_i}$, $\hat{S}_{e_i}$ in OCBA-mSG or $\hat{S}_{s_i}$, $\hat{S}_{bl}$, $\hat{S}_a$, $\hat{S}_{d_i}$, $\hat{S}_{e_i}$ in OCBA-bSG (c.f. Fig. 1).

$$NU = \frac{(T-Kn_0)}{total\ number\ of\ sets} + n_0.$$

This choice of upper bound works well in our numerical experiments.

IV. NUMERICAL EXPERIMENTS

In this section, we demonstrate our methods OCBA-mSG and OCBA-bSG on some examples and also compare them with two other methods - Equal Allocation and Levin Search.

Equal Allocation (EA) allocates the simulation budget equally among all the designs and do not use any information such as the mean, the variance or the complexity of the design. At iteration l, it allocates Δ simulation budget according to

$$N_i^{l+1} - N_i^l = \Delta/K, \quad \forall i \in \{1, 2, \dots, K\}.$$

Levin Search (LS) method [36] allocates simulation budget to the designs sequentially in the order of the complexity. It is useful when applied to find one simplest and good enough design [1]. LS first simulates the designs with smallest complexity until obtaining a certain accuracy for the estimates of the performance, based on which the good enough designs are selected. If only less than m good enough designs are found, it then continues to simulate the designs in the next complexity set until eventually finding m simplest good enough designs eventually. In our implementation, we simulate every design for n_0 times at initialization, and order them according to their sample means and complexities. Since it is hard to specify a given accuracy for the estimates in our examples, we evenly allocate the total remaining budget $(T - Kn_0)/K$ to all the designs beforehand, but simulate the designs sequentially, i.e., start simulating the first simplest design for $(T - Kn_0)/K$ times and then move on to the next one to repeat the same procedure. Please note LS often exhibits some "jump" behavior in the $P(CS)$, because the $P(CS)$ stays flat if the design currently under simulation is not good enough and the $P(CS)$ increases otherwise. If the desirable set of designs is found before utilizing all computing budget, LS will terminate and the corresponding $P(CS)$ curve will level off in the figures. LS is the same as EA when utilizing all the T simulation budget, but LS often achieves the final $P(CS)$ earlier. In general, LS method performs better if the performance deteriorates as the complexity increases.

In the numerical experiments, we test three generic examples which mimic different scenarios in real world. In Example 1, good designs are also simple designs. In contrast, bad designs are simple ones in Example 2. In Example 3, we consider a problem with a larger number of alternative designs. We use $P(CS)$ as the efficiency measurement: for a given total simulation budget, the faster the $P(CS)$ converges, the better the corresponding method is. Here we estimate $P(CS)$ using Monte Carlo simulation by computing the ratio of the number of simulation runs with correct selections to the total number of simulation runs. In addition, for convenience, we assume that design θ_i has complexity $\lfloor \log_2 i \rfloor$, so the complexity is non-decreasing in i.

1) Example 1 (Mean increases as complexity increases): There are 20 designs in total, with the i^{th} design having $L(\theta_i, \zeta)$ distributed according to the normal distribution $N(i, (0.5i)^2)$. We want to find 5 simplest good enough designs with good enough constraint $J_0 = 6.3$. The initial simulation budget $n_0 = 20$, simulation budget increment $\Delta = 200$, total simulation budget $T = 8000$, and total number of simulation runs = 10^4. The complexity sets are $C_0 = \{\theta_1\}$, $C_1 = \{\theta_2, \theta_3\}$, $C_2 = \{\theta_4, \theta_5, \theta_6, \theta_7\}$, $C_3 = \{\theta_8, \dots, \theta_{15}\}$ and $C_4 = \{\theta_{16}, \dots, \theta_{20}\}$. The mean increases as the complexity increases, and the variance increases as the mean increases.

a) OCBA-mSG: The correct selection of the five desirable designs should include $\{\theta_1, \theta_2, \theta_3\}$ and any two from $\{\theta_4, \theta_5, \theta_6\}$. Fig. 3 shows that OCBA-mSG converges faster than EA and LS. EA performs well in this example because of the small total number of designs K and the small variance σ^2. LS searches from the simplest sets $\{\theta_1\}$, $\{\theta_2, \theta_3\}$, ..., and in this example the correct selection is $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\}$, so LS terminates in about 7 iterations.

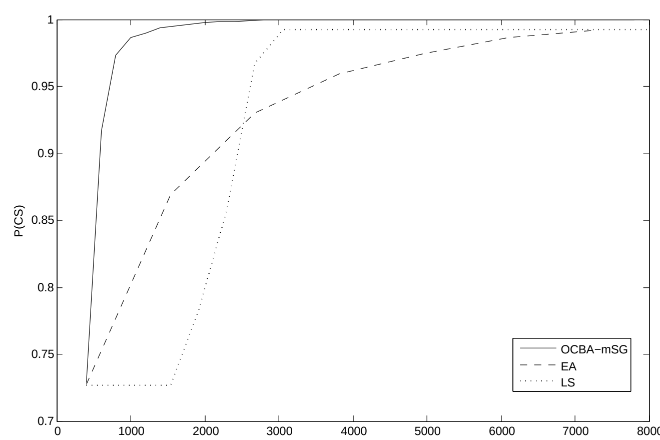


Fig. 3: Example 1 - selecting 5 simplest good enough designs from 20 designs with distribution $N(i, (0.5i)^2)$ and $J_0 = 6.3$.

b) OCBA-bSG: The correct selection is $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\}$. Fig. 4 shows the simulation result.

2) Example 2 (Mean decreases as complexity increases): There are 20 designs, with the i^{th} design having $L(\theta_i, \zeta)$

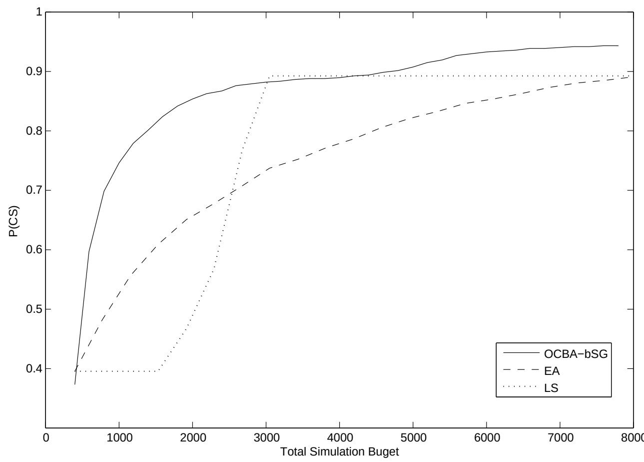


Fig. 4: Example 1 - selecting the best 5 simplest good enough designs from 20 designs with distribution $N(i,(0.5i)^2)$ and $J_0 = 6.3$.

distributed according to the normal distribution $N((21-i),(0.5i)^2)$. We want to find 5 simplest good enough designs with good enough constraint $J_0 = 7.3$. The initial simulation budget $n_0 = 20$, simulation budget increment $\Delta = 200$, total simulation budget $T = 8000$, and total number of simulation runs = 10^4. The complexity sets are the same as in Example 1. The mean decreases as the complexity increases, and the variance increases as the mean decreases.

a) OCBA-mSG: Correct selection of the five desirable designs should include $\{\theta_{14}, \theta_{15}\}$ and any three from $\{\theta_{16}, \theta_{17}, \theta_{18}, \theta_{19}, \theta_{20}\}$. Fig. 5 shows the simulation result. All three methods converge slower than Example 1, but OCBA-mSG still converges faster than EA and LS. Designs with smaller means have larger variances, and the correct selection is in the set $\{\theta_{14}, \theta_{15}, \theta_{16}, \theta_{17}, \theta_{18}, \theta_{19}, \theta_{20}\}$ which have relatively large variances compared to other designs, so OCBA-mSG converges slower than that in Example 1. LS still searches from the simplest sets while the correct selection is in the higher complexity sets, so LS method also terminates later than Example 1.

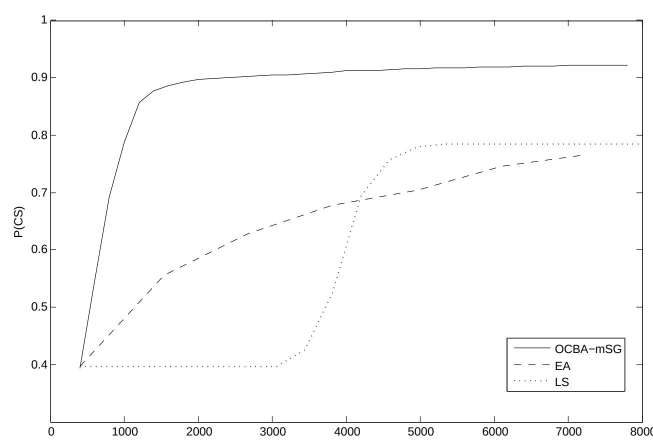


Fig. 5: Example 2 - selecting 5 simplest good enough designs from 20 designs with distribution $N((21-i),(0.5i)^2)$ and $J_0 = 7.3$.

b) OCBA-bSG: The correct selection is $\{\theta_{14}, \theta_{15}, \theta_{18}, \theta_{19}, \theta_{20}\}$. Fig. 6 shows the simulation result.

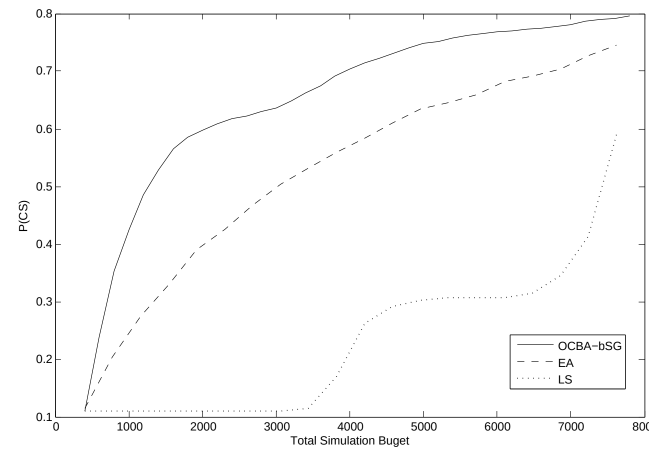


Fig. 6: Example 2 - selecting the best 5 simplest good enough designs from 20 designs with distribution $N((21-i),(0.5i)^2)$ and $J_0 = 7.3$.

3) Example 3 (Mid-scale problem): There are 65 designs, with the i^{th} design having $L(\theta_i, \zeta)$ distributed according to the normal distribution $N((66-i),(0.05i)^2)$. We want to find 5 simplest good enough designs with good enough constraint $J_0 = 6.3$. The initial simulation budget $n_0 = 20$, simulation budget increment $\Delta = 200$, and total simulation budget $T = 8000$. The complexity sets are $C_0 = \{\theta_1\}$, $C_1 = \{\theta_2, \theta_3\}$, $C_2 = \{\theta_4, \ldots, \theta_7\}$, $C_3 = \{\theta_8, \ldots, \theta_{15}\}$, $C_4 = \{\theta_{16}, \ldots, \theta_{31}\}$, $C_5 = \{\theta_{32}, \ldots, \theta_{63}\}$ and $C_6 = \{\theta_{64}, \theta_{65}\}$.

a) OCBA-mSG: The correct selection of the five desirable designs should include $\{\theta_{60}, \theta_{61}, \theta_{62}, \theta_{63}\}$ and any one from $\{\theta_{64}, \theta_{65}\}$. For this mid-scale problem, OCBA-mSG performs much better than EA and LS as shown in Fig. 7. Detailed explanation is similar to that for OCBA-bSG in the following.

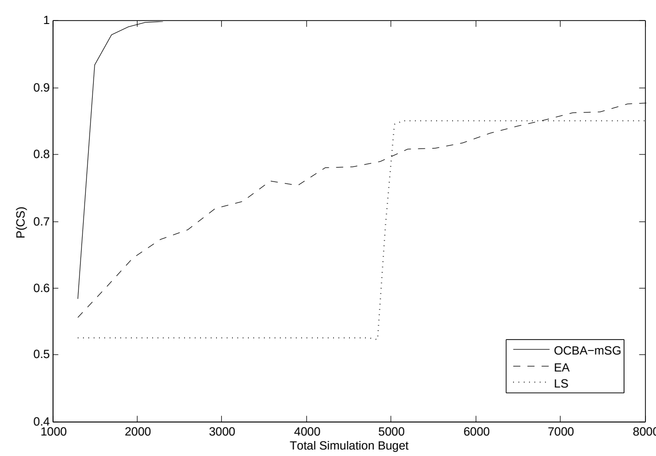


Fig. 7: Example 3 - selecting 5 simplest good enough designs from 65 designs with distribution $N((66-i),(0.05i)^2)$ and $J_0 = 6.3$.

b) OCBA-bSG: The correct selection is $\{\theta_{60}, \theta_{61}, \theta_{62}, \theta_{63}, \theta_{65}\}$. For this mid-scale problem, OCBA-bSG performs much better than EA and LS as shown in

Fig. 8. When the total design number K is large, EA converges slowly since each design is assigned with less simulation budget at every iteration compared to Examples 1 and 2. For LS, the first time LS jumps in $P(CS)$ is the time that the total simulation budget reaches 4900, which is when it first starts to simulate designs in the set $\{\theta_{32},\ldots,\theta_{63}\}$ with means $\{34,\ldots,3\}$. As we assign the simulation budget according to the order of the designs in the same complexity set, here we simulate designs in the order of $\theta_{63},\theta_{62},\ldots$. Since designs $\theta_{63},\theta_{62},\theta_{61}$ and θ_{60} belong to the correct selection set, LS jumps in $P(CS)$ at this point. The second jump in the $P(CS)$ for LS happens in the end due to the simulation budget allocation to the design θ_{65}.

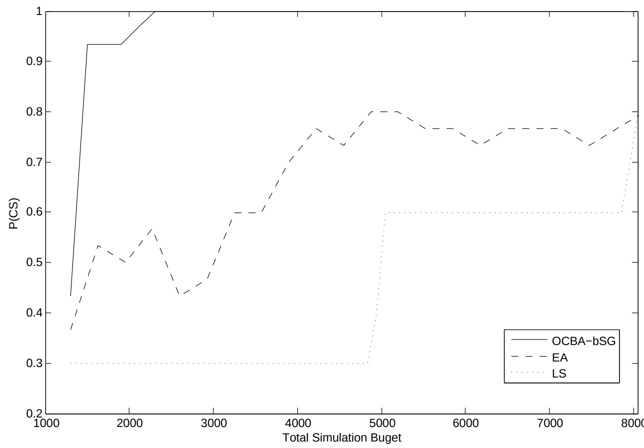


Fig. 8: Example 3 - selecting the best 5 simplest good enough designs from 65 designs with distribution $N((66-i),(0.05i)^2)$ and $J_0=6.3$.

V. Conclusion

In this paper, we considered the simulation-based selection of simplest good enough designs, which is motivated by real-life applications. We proved the optimal simulation budget allocation rules to asymptotically maximize the probability of correct selection (or the approximate probability of correct selection in the case of OCBA-bSG). Based on the asymptotic results, we proposed the algorithm OCBA-mSG to efficiently allocate the simulation budget for selecting m simplest good enough designs out of a total of K designs, and also proposed a slightly different algorithm OCBA-bSG in order to find the *best m* simplest good enough designs. Numerical results show that both methods converge fast on all the test problems, which indicates OCBA-mSG and OCBA-bSG indeed allocate simulation budget efficiently. While our algorithms are motivated by the asymptotic results, an important future direction is to analyze the finite-time performance of our algorithms.

VI. Appendix

A. Appendix A: Proof of Theorem 1

Since $\tilde{J}_k \sim N(\bar{J}_k,\frac{\sigma_k^2}{N_k})$, we have

$$P(\tilde{J}_k<J_0)=\Phi\left(\frac{J_0-\bar{J}_k}{\sigma_k/\sqrt{N_k}}\right),$$

where Φ is the error function (i.e., the cumulative distribution function (c.d.f.) of the standard normal distribution). By Lagrangian relaxation of $P(CS_m)$ and Karush-Kuhn-Tucker (KKT) condition (c.f. [37]) for the maximization problem (1), we get

$$F=\prod_{k\in S_s}P\{\tilde{J}_k<J_0\}\prod_{k\in S_I}P\{\tilde{J}_k\geqslant J_0\}-\lambda(\sum_{k=1}^{K}N_i-T).$$

For $i\in S_s$,

$$\begin{aligned}\frac{\partial F}{\partial N_i} &= \frac{1}{2}\prod_{k\in S_s,k\neq i}P\{\tilde{J}_k<J_0\}\prod_{k\in S_I}P\{\tilde{J}_k\geqslant J_0\}\cdot\\ &\quad \phi\left(\frac{J_0-\bar{J}_i}{\sigma_i/\sqrt{N_i}}\right)\frac{J_0-\bar{J}_i}{\sigma_i\sqrt{N_i}}-\lambda=0. \qquad (10)\end{aligned}$$

For $i\in S_I$,

$$\begin{aligned}\frac{\partial F}{\partial N_i} &= -\frac{1}{2}\prod_{k\in S_s}P\{\tilde{J}_k<J_0\}\prod_{k\in S_I,k\neq i}P\{\tilde{J}_k\geqslant J_0\}\cdot\\ &\quad \phi\left(\frac{J_0-\bar{J}_i}{\sigma_i/\sqrt{N_i}}\right)\frac{J_0-\bar{J}_i}{\sigma_i\sqrt{N_i}}-\lambda=0, \qquad (11)\end{aligned}$$

where ϕ denotes the probability density function (p.d.f.) of the standard normal distribution.

In order to find the relationship between N_i and N_j, we need to consider $\binom{2}{1}+\binom{2}{2}=3$ cases that i, j belong to different sets.

Case 1: $i\in S_s,\ j\in S_I$. Equating (10) and (11),

$$\begin{aligned}&P\{\tilde{J}_j\geqslant J_0\}\phi\left(\frac{J_0-\bar{J}_i}{\sigma_i/\sqrt{N_i}}\right)\frac{J_0-\bar{J}_i}{\sigma_i\sqrt{N_i}}\\ =\ &P\{\tilde{J}_i<J_0\}\phi\left(\frac{J_0-\bar{J}_j}{\sigma_j/\sqrt{N_j}}\right)\frac{\bar{J}_j-J_0}{\sigma_j\sqrt{N_j}}.\end{aligned}$$

Taking logarithm on both sides, we have

$$\begin{aligned}&\log P\left\{\tilde{J}_j\geqslant J_0\right\}-\frac{(J_0-\bar{J}_i)^2}{2\sigma_i^2/N_i}+\log\left(\frac{J_0-\bar{J}_i}{\sigma_i}\right)-\frac{1}{2}\log N_i\\ =\ &\log P\left\{\tilde{J}_i<J_0\right\}-\frac{(J_0-\bar{J}_j)^2}{2\sigma_j^2/N_j}+\log\left(\frac{\bar{J}_j-J_0}{\sigma_j}\right)-\frac{1}{2}\log N_j.\end{aligned}$$

Assuming N_i takes continuous values, let $N_i=\alpha_i T$. Taking the asymptotic limit of the above equation as $T\rightarrow\infty$, we have

$$\begin{aligned}&\lim_{T\to\infty}\frac{1}{T}\{\log P(\tilde{J}_j\geqslant J_0)-\frac{(J_0-\bar{J}_i)^2\alpha_i T}{2\sigma_i^2}+\\ &\log\left(\frac{J_0-\bar{J}_i}{\sigma_i}\right)-\frac{1}{2}\log(\alpha_i T)\}\\ =\ &\lim_{T\to\infty}\frac{1}{T}\{\log P(\tilde{J}_i<J_0)-\frac{(J_0-\bar{J}_j)^2\alpha_j T}{2\sigma_j^2}+\\ &\log\left(\frac{J_0-\bar{J}_j}{\sigma_j}\right)-\frac{1}{2}\log(\alpha_j T)\},\end{aligned}$$

and we obtain

$$\frac{\alpha_i}{\alpha_j} = \frac{(\bar{J}_j-J_0)^2}{(\bar{J}_i-J_0)^2}\cdot\frac{\sigma_i^2}{\sigma_j^2}.$$

Case 2: $i\in S_s,\ j\in S_s,\ i\neq j$. By equating (10) and (10), similarly as above we obtain

$$\frac{\alpha_i}{\alpha_j} = \frac{(\bar{J}_j-J_0)^2}{(\bar{J}_i-J_0)^2}\cdot\frac{\sigma_i^2}{\sigma_j^2}.$$

Case 3: $i \in S_I,\ j \in S_I,\ i \neq j$. By equating (11) and (11), similarly as above we obtain

$$\frac{\alpha_i}{\alpha_j} = \frac{(\bar{J}_j - J_0)^2}{(\bar{J}_i - J_0)^2} \cdot \frac{\sigma_i^2}{\sigma_j^2}.$$

Combining all three cases, we prove Theorem 1.

B. Proof of Lemma 2

Our derivation of the value of μ follows the idea and method in Section 3.3 in [21]. Specifically, if we assume that all the designs have equal variances, then we know

$$\begin{aligned} P\left(\tilde{J}_{[r]} \leqslant \mu\right) &\leqslant P\left(\tilde{J}_i \leqslant \mu\right), \quad \forall i \in S_{bl}, \\ P\left(\tilde{J}_{[r+1]} \geqslant \mu\right) &\geqslant P\left(\tilde{J}_i \leqslant \mu\right), \quad \forall i \in S_a. \end{aligned}$$

To maximize $APCS_b$ is equivalent to maximizing the product of all the above terms. The smallest terms $P\left(\tilde{J}_{[r]} \leqslant \mu\right)$ and $P\left(\tilde{J}_{[r+1]} \geqslant \mu\right)$ have the most impact on the value of the product. Hence, to simplify the problem, we consider the maximization of the product of these two terms. A good choice of μ can be determined by solving the following maximization problem

$$\begin{aligned} \max_{N_{[r]},N_{[r+1]}} \quad & P\left(\tilde{J}_{[r]} \leqslant \mu\right) P\left(\mu \leqslant \tilde{J}_{[r+1]}\right) \\ s.t. \quad & N_{[r]} + N_{[r+1]} = T. \end{aligned}$$

Following the same approach in the proof of Theorem 1, we obtain the asymptotically (as $T \to \infty$) optimal solution (6).

C. Proof for Theorem 3

Since $\tilde{J}_i \sim N(\bar{J}_i, \frac{\sigma_i^2}{N_i})$, we have

$$\begin{aligned} &\text{for } i \in S_{b^-}, \quad P(\tilde{J}_i < J_0) = \Phi\left(\frac{J_0 - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right); \\ &\text{for } i \in S_{bl}, \quad P(\tilde{J}_i \leqslant \mu) = \Phi\left(\frac{\mu - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right); \\ &\text{for } i \in S_a, \quad P(\mu \leqslant \tilde{J}_i < J_0) = \Phi\left(\frac{J_0 - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right) - \Phi\left(\frac{\mu - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right); \\ &\text{for } i \in S_I, \quad P(\tilde{J}_i \geqslant J_0) = \Phi\left(\frac{\bar{J}_i - J_0}{\sigma_i/\sqrt{N_i}}\right), \end{aligned}$$

where Φ is the error function. By Lagrangian relaxation of $APCS_b$ and KKT condition, we have

$$F = \prod_{k\in S_{b^-}} P\{\tilde{J}_k < J_0\} \prod_{k\in S_{bl}} P\{\tilde{J}_k \leqslant \mu\} \prod_{k\in S_a} P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{k\in S_I} P\{\tilde{J}_k \geqslant J_0\} - \lambda\left(\sum_{k=1}^{K} N_k - T\right).$$

Let ϕ denote the p.d.f. of the standard normal distribution. We obtain the following conditions. For $i \in S_{b^-}$,

$$0 = \frac{\partial F}{\partial N_i} = -\lambda + \frac{1}{2} \prod_{k\in S_{b^-},k\neq i} P\{\tilde{J}_k < J_0\} \prod_{k\in S_{bl}} P\{\tilde{J}_k \leqslant \mu\} \prod_{k\in S_a} P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{k\in S_I} P\{\tilde{J}_k \geqslant J_0\} \cdot \phi\left(\frac{J_0 - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right) \cdot \frac{J_0 - \bar{J}_i}{\sigma_i\sqrt{N_i}}. \quad (12)$$

For $i \in S_{bl}$,

$$0 = \frac{\partial F}{\partial N_i} = -\lambda + \frac{1}{2} \prod_{k\in S_{b^-}} P\{\tilde{J}_k < J_0\} \prod_{k\in S_{bl},k\neq i} P\{\tilde{J}_k \leqslant \mu\} \prod_{k\in S_a} P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{k\in S_I} P\{\tilde{J}_k \geqslant J_0\} \cdot \phi\left(\frac{\mu - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right) \cdot \frac{\mu - \bar{J}_i}{\sigma_i\sqrt{N_i}}. \quad (13)$$

For $i \in S_a$,

$$0 = \frac{\partial F}{\partial N_i} = -\lambda + \frac{1}{2} \prod_{k\in S_{b^-}} P\{\tilde{J}_k < J_0\} \prod_{k\in S_{bl}} P\{\tilde{J}_l \leqslant \mu\} \prod_{k\in S_a,k\neq i} P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{k\in S_I} P\{\tilde{J}_k \geqslant J_0\} \cdot \left[\phi\left(\frac{J_0 - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right) \cdot \frac{J_0 - \bar{J}_i}{\sigma_i\sqrt{N_i}} - \phi\left(\frac{\mu - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right) \cdot \frac{\mu - \bar{J}_i}{\sigma_i\sqrt{N_i}}\right]. \quad (14)$$

For $i \in S_I$,

$$0 = \frac{\partial F}{\partial N_i} = -\lambda - \frac{1}{2} \prod_{k\in S_{b^-}} P\{\tilde{J}_k < J_0\} \prod_{k\in S_{bl}} P\{\tilde{J}_k \leqslant \mu\} \prod_{k\in S_a} P\{\mu \leqslant \tilde{J}_k < J_0\} \prod_{k\in S_I,k\neq i} P\{\tilde{J}_k \geqslant J_0\} \cdot \phi\left(\frac{\bar{J}_i - J_0}{\sigma_i/\sqrt{N_i}}\right) \cdot \frac{\bar{J}_i - J_0}{\sigma_i\sqrt{N_i}}. \quad (15)$$

In order to find the relationship between N_i and N_j, we need to consider $\binom{4}{1} + \binom{4}{2} = 10$ cases that θ_i and θ_j belong to different sets.

Case 1: $\theta_i \in S_{b^-},\ \theta_j \in S_a$. Equating (12) and (14), we have

$$P\{\tilde{J}_i < J_0\}\left[\phi\left(\frac{J_0 - \bar{J}_j}{\sigma_j/\sqrt{N_j}}\right)\frac{J_0 - \bar{J}_j}{\sigma_j\sqrt{N_j}} - \phi\left(\frac{\mu - \bar{J}_j}{\sigma_j/\sqrt{N_j}}\right)\frac{\mu - \bar{J}_j}{\sigma_j\sqrt{N_j}}\right] = P\{\mu \leqslant \tilde{J}_j < J_0\}\phi\left(\frac{J_0 - \bar{J}_i}{\sigma_i/\sqrt{N_i}}\right)\frac{J_0 - \bar{J}_i}{\sigma_i\sqrt{N_i}}.$$

Assuming N_i takes continuous values, let $N_i = \alpha_i T$. Taking the asymptotic limit of the above equation as $T \to \infty$, we have

$$\begin{aligned} &\lim_{T\to\infty} \frac{1}{T}\left[\log P\{\tilde{J}_i < J_0\} + \log A - \log\sigma_j - \frac{1}{2}\log(\alpha_j T)\right] \\ = &\lim_{T\to\infty} \frac{1}{T}\{\log P\{\mu \leqslant \tilde{J}_j < J_0\} - \frac{(J_0 - \bar{J}_i)^2 \alpha_i T}{2\sigma_i^2} + \log\left(\frac{J_0 - \bar{J}_i}{\sigma_i}\right) - \frac{1}{2}\log(\alpha_i T)\}, \quad (16) \end{aligned}$$

where

$$A = \phi\left(\frac{J_0 - \bar{J}_j}{\sigma_j/\sqrt{\alpha_j T}}\right)(J_0 - \bar{J}_j) - \phi\left(\frac{\mu - \bar{J}_j}{\sigma_j/\sqrt{\alpha_j T}}\right)(\mu - \bar{J}_j).$$

By L'H$\hat{o}$pital's Rule,

$$\begin{aligned} &\lim_{T\to\infty} \frac{1}{T}\log A \\ = &\lim_{T\to\infty} \frac{dA/dT}{A} \\ = &\lim_{T\to\infty} \frac{\left(\frac{-(J_0-\bar{J}_j)^2}{2\sigma_j^2/\alpha_j} - \frac{-(\mu-\bar{J}_j)^2}{2\sigma_j^2/\alpha_j}\right)(\mu - \bar{J}_j)}{\exp\left(\frac{-(J_0-\bar{J}_j)^2 T}{2\sigma_j^2/\alpha_j} - \frac{-(\mu-\bar{J}_j)^2 T}{2\sigma_j^2/\alpha_j}\right)(J_0 - \bar{J}_j) - (\mu - \bar{J}_j)} + \\ &\frac{-(J_0 - \bar{J}_j)^2}{2\sigma_j^2/\alpha_j}. \end{aligned}$$

1) If $J_0-\bar{J}_j \geqslant \bar{J}_j-\mu$, then $\exp\left(\frac{-(J_0-\bar{J}_j)^2T}{2\sigma_j{}^2/\alpha_j} - \frac{-(\mu-\bar{J}_j)^2T}{2\sigma_j{}^2/\alpha_j}\right) \to 0$ as $T \to \infty$. Hence,

$$\lim_{T\to\infty}\frac{1}{T}\log A = -\frac{(\bar{J}_j-\mu)^2\alpha_j}{2\sigma_j^2}.$$

2) If $J_0-\bar{J}_j < \bar{J}_j-\mu$, then $\exp\left(\frac{-(J_0-\bar{J}_j)^2T}{2\sigma_j{}^2/\alpha_j} - \frac{-(\mu-\bar{J}_j)^2T}{2\sigma_j{}^2/\alpha_j}\right) \to \infty$ as $T \to \infty$. Hence,

$$\lim_{T\to\infty}\frac{1}{T}\log A = -\frac{(J_0-\bar{J}_j)^2\alpha_j}{2\sigma_j^2}.$$

By applying the above results of A to equation (16), we get

1) If $\bar{J}_j \leqslant \frac{J_0+\mu}{2}$, $\frac{\alpha_i}{\alpha_j} = \frac{(\mu-\bar{J}_j)^2}{(J_0-\bar{J}_i)^2}\cdot\frac{\sigma_i^2}{\sigma_j^2}$.
2) If $\bar{J}_j > \frac{J_0+\mu}{2}$, $\frac{\alpha_i}{\alpha_j} = \frac{(J_0-\bar{J}_j)^2}{(J_0-\bar{J}_i)^2}\cdot\frac{\sigma_i^2}{\sigma_j^2}$.

Similarly, we can obtain the relationship between N_i and N_j for the other 9 cases. If $S_a = \varnothing$, it reduces to the case of OCBA-mbG. By combining the results of all the 10 cases, we prove Theorem 3.

Acknowledgment

A preliminary version of the manuscript was presented at the 2010 Winter Simulation Conference [38].

References

[1] Q. S. Jia, "An adaptive sampling algorithm for simulation-based optimization with descriptive complexity preference," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 4, pp. 720 – 731, 2011.

[2] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 2005.

[3] H. P. Galliher, P. M. Morse, and M. Simond, "Dynamics of two classes of continuous-review inventory systems," *Operations Research*, vol. 7, no. 3, pp. 362–384, 1959.

[4] I. Sahin, "On the objective function behavior in (s, S) inventory models," *Operations Research*, vol. 30, no. 4, pp. 709–724, 1992.

[5] A. Federgruen and Y. S. Zheng, "An efficient algorithm for computing an optimal (r, Q) policy in continuous review stochastic inventory system," *Operations Research*, vol. 40, no. 4, pp. 808–813, 1992.

[6] K. Kar, A. Krishnamurthy, and N. Jaggi, "Dynamic node activation in networks of rechargeable sensors," *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 15–26, 2006.

[7] J. Branke, S. E. Chick, and C. Schmidt, "Selecting a selection procedure," *Management Science*, vol. 53, no. 12, pp. 1916 – 1932, 2007.

[8] S. H. Jacobson and E. Yucesan, "Computational issues for accessibility in discrete event simulation," *ACM Transactions on Modeling and Computer Simulation*, vol. 6, no. 1, pp. 53 – 75, 1996.

[9] ——, "Common issues in discrete-event simulation and discrete optimization," *IEEE Transactions on Automatic Control*, vol. 47, no. 2, pp. 341 – 345, 2002.

[10] M. C. Fu, "Optimization for simulation: Theory vs. practice," *INFORMS Journal on Computing*, vol. 14, no. 3, pp. 192 – 215, 2002.

[11] L. Pi, Y. Pan, and L. Shi, "Hybrid nested partitions and mathematical programming approach and its applications," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 5, p. 573 586, 2008.

[12] W. Chen, L. Pi, and L. Shi, "An enhanced nested partitions algorithm using solution value prediction," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 2, pp. 412 – 419, 2011.

[13] W. P. Wong, L. H. Lee, and W. Jaruphongsa, "Budget allocation for effective data collection in predicting an accurate DEA efficiency score," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1235 – 1246, 2011.

[14] S. S. Gupta, "On some multiple decision (selection and ranking) rules," *Technometrics*, vol. 7, no. 2, pp. 225–245, 1965.

[15] T. J. Santner, "A restricted subset selection approch to ranking and selection problems," *The Annals of Statistics*, vol. 3, no. 2, pp. 334–349, 1975.

[16] L. W. Koenig and A. M. Law, "A procedure for selecting a subset of size *m* containing the *l* best of *k* independent normal populations," *Communications in Statistics - Simulation and Computation*, vol. 14, pp. 719–734, 1985.

[17] E. J. Dudewicz and S. R. Dalal, "Allocation of observation in ranking and selection with unequal variances," *Sankhya: The Indian Journal of Statistics*, vol. 37, pp. 28–78, 1975.

[18] C.-H. Chen, J. Lin, E. Yücesan, and S. E. Chick, "Simulation budget allocation for further enhancing the efficiency of ordinal optimization," *Journal of Discrete Event Dynamic Systems: Theory and Applications*, vol. 10, pp. 251–270, 2000.

[19] C.-H. Chen, D. He, and M. Fu, "Efficient dynamic simulation allocation in ordinal optimization," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 2005–2009, 2006.

[20] C.-H. Chen, E. Yücesan, L. Dai, and H. Chen, "Efficient computation of optimal budget allocation for discrete event simulation experiment," *IIE Transactions*, vol. 42, no. 1, pp. 60 – 70, 2010.

[21] C.-H. Chen, D. H. He, M. Fu, and L. H. Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579–595, 2008.

[22] L. H. Lee, E. P. Chew, S. Y. Teng, and D. Goldsman, "Optimal computing budget allocation for multi-objective simulation models," in *Proceedings of 2004 Winter Simulation Conference*, 2004, pp. 586 – 594.

[23] S. Y. Teng, L. H. Lee, and E. P. Chew, "Multi-objective ordinal optimization for simulation optimization problems," *Automatica*, vol. 43, no. 11, pp. 1884 – 1895, 2007.

[24] E. P. Chew, L. H. Lee, S. Y. Teng, and C. H. Koh, "Differentiated service inventory optimization using nested partitions and MOCBA," *Computers and Operations Research*, vol. 36, no. 5, pp. 1703 – 1710, 2009.

[25] L. H. Lee, E. P. Chew, S. Y. Teng, and D. Goldsman, "Finding the pareto set for multi-objective simulation models," *IIE Transactions*, vol. 42, no. 9, pp. 656 – 674, 2010.

[26] J. Branke and D. Mattfeld, "Anticipation and flexibility in dynamic scheduling," *International Journal of Production Research*, vol. 43, no. 15, pp. 3103 – 3129, 2005.

[27] J. Branke and J. Gamer, "Efficient sampling in interactive multi-criteria selection," in *Proceedings of the 2007 INFORMS Simulation Society Research Workshop*, 2007.

[28] S. Andradóttir, D. Goldsman, B. W. Schmeiser, L. W. Schruben, and E. Yücesan, "Analysis methodology: Are we done?" in *Proceedings of the 37th conference on Winter simulation*, 2005, pp. 790–796.

[29] R. Szechtman and E. Yücesan, "A new perspective on feasibility determination," in *Proceedings of the 2008 Winter Simulation Conference*, 2008, pp. 273–280.

[30] N. A. Pujowidianto, L. H. Lee, C.-H. Chen, and C. M. Yap, "Optimal computing budget allocation for constrained optimization," in *Proceedings of 2009 Winter Simulation Conference*, 2009, pp. 584–589.

[31] Q.-S. Jia and Q.-C. Zhao, "Strategy optimization for controlled markov process with descriptive complexity constraint," *Science in China Series F: Information Sciences*, vol. 52, no. 11, pp. 1993 – 2005, 2009.

[32] Q.-S. Jia, "On state aggregation to approximate complex value functions in large-scale Markov decision processes," *IEEE Transactions on Automatic Control*, vol. 56, no. 2, pp. 333 – 344, 2011.

[33] Y.-C. Ho, Q.-C. Zhao, and Q.-S. Jia, *Ordinal Optimization: Soft Optimization for Hard Problems*. Springer, 2007.

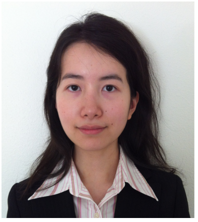
[34] C.-H. Chen, "A lower bound for the correct subset-selection probability and its application to discrete-event system simulations," *IEEE Transactions on Automatic Control*, vol. 41, no. 8, pp. 1227 – 1231, 1996.

[35] D. He, S. E. Chick, and C. H. Chen, "The opportunity cost and OCBA selection procedures in ordinal optimization," *IEEE Transactions on Systems, Man, and Cybernetics–Part C*, vol. 37, no. 5, pp. 951 – 961, 2007.

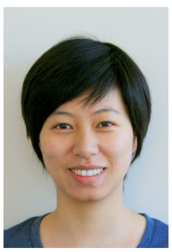
[36] L. A. Levin, "Universal sequential search problems," *Problems of Information Transmission*, vol. 9, no. 3, pp. 265–266, 1973.

[37] R. C. Walker, *Introduction to Mathematical Programming*. Upper Saddle River, NJ: Prentice & Hall, 1999.

[38] S. Yan, E. Zhou, and C.-H. Chen, "Efficient simulation budget allocation for selecting the best set of simplest good enough designs," in *Proceedings of the 2010 Winter Simulation Conference*, 2010, pp. 1152 – 1159.



Shen Yan received her Bachelors of Engineering degree with highest honors from Chinese University of Hong Kong, Hong Kong in 2009, and received the Master of Science degree in Industrial Engineering from the University of Illinois at Urbana-Champaign in 2011. Her research interest is simulation optimization.



Enlu Zhou received the B.S. degree with highest honors in electrical engineering from Zhejiang University, China, in 2004, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2009. Since then she has been an Assistant Professor at the Industrial & Enterprise Systems Engineering Department at the University of Illinois Urbana-Champaign. Her research interests include Markov decision processes, stochastic control, and simulation optimization. She is a recipient of the Best Theoretical Paper award at the 2009 Winter Simulation Conference and the 2012 AFOSR Young Investigator award.

Chun-Hung Chen received his Ph.D. degree in Engineering Sciences from Harvard University in 1994. He is a Professor of Systems Engineering & Operations Research at George Mason University and is also affiliated with National 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 "National Thousand Talents" Award from the central government of China in 2011, 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 has served as a department editor for IIE Transactions, associate editor of IEEE Transactions on Automatic Control, area editor of Journal of Simulation Modeling Practice and Theory, associate editor of International Journal of Simulation and Process Modeling, and associate editor of IEEE Conference on Automation Science and Engineering.