Optimizing Maintenance Policies based on Discrete Event Simulation and the OCBA Mechanism

Dingzhou Cao, Ph.D., ReliaSoft Corporation
Yu Sun, Ph.D., Siena College
Huairui Guo, Ph.D., ReliaSoft Corporation

Key Words: Maintenance Policies; Discrete Event Simulation; Optimal Computing Budget Allocation; Simulation Optimization

SUMMARY & CONCLUSIONS

In an age of high competition, strict safety and environmental regulations, maintenance which acted as an effective tool to improve system availability so as to increase profit margin has played an important role in modern industry. In this paper, the system costs (availability) are estimated by the discrete event simulation technique and the Optimal Computing Budget Allocation (OCBA) mechanism is implemented to try to find the optimal maintenance policies for the system. The OCBA is an effective algorithm that can identify the optimal maintenance policies while significantly improving the simulation efficiency. The main purpose of this article is to validate the efficiency of this method in the application of maintenance policy optimization. An example taken from literature is presented to show the effectiveness of this method.

1 INTRODUCTION

Reliability and maintainability play a crucial role in ensuring the successful operation of a repairable system (such as a chemical process plant), as they determine system availability and thus contribute significantly to system economics and safety. Equipment maintenance is mainly classified into two types: 1) Corrective Maintenance (CM), which means repair when equipment fails, restoring it to normal function; and 2) Preventive Maintenance (PM), which is maintenance or replacement that occurs during normal functioning of the equipment in order to restore it to a better functioning condition and reduce the probability of equipment failure. The primary goal of PM is to prevent the failure of equipment before it actually occurs. It is designed to preserve and enhance equipment reliability so as to increase the system availability by replacing worn components before they actually fail. If implemented properly, PM can be an effective method to maintain the equipment at optimal functioning and to maintain the highest system availability. Frequent PM (PM interval is too short) increases the system reliability and availability, but the resulting costs are also increased. Rare PM (PM interval is too long) will keep the maintenance costs lower, but may lead to large system losses due to long downtime of the system, as CM durations are usually longer than PM duration. The problem is to find the "sweet spot" between these two, the optimal PM interval that will maximize the system availability while keeping the maintenance costs at an acceptable level.

Numerous methods have been proposed to study this problem, including analytical approaches [1] and simulation-based approaches [2]. The analytical approaches in general suffer severe modeling limitations and can be applied only to simple system with few components, while the simulation-based methods have the weakness of requiring extensive computing time. In this article, the simulation-based method is studied. For a system with a large number of components, the simulation cost of attempting to find the optimal PM interval for each component becomes prohibitively high. It actually becomes a simulation-based optimization problem. Due to the complexity of the simulation, the objective function of the simulation-based optimization problem is typically a) subject to various levels of noise, and b) not necessarily differentiable. The Genetic Algorithm [2, 3] is a popular method for solving this problem. In this paper, we discretize the decision variables to simplify the PM interval optimization problem and put it as a design selection problem. The goal of the design selection problem is to identify the best system design from a collection of candidate system designs. In our problem, each component has several PM interval alternatives for choose. The goal is to select a combination of PM intervals for all components (one PM interval for each component) to maximize the system availability or minimize the system costs.

Discrete event simulation is used to estimate the system metric in the paper. As mentioned above, efficiency is a significant concern in simulation-based optimization. To obtain a good statistical estimation for a design decision, a large number of simulation samples or replications are usually required for each design alternative. To improve the efficiency of simulation optimization, Chen [4] proposed the Optimal Computing Budget Allocation (OCBA) mechanism. The underlying philosophy is that, to ensure a high probability of correctly selecting a good 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

978-1-4673-4711-2/13/$31.00 ©2013 IEEE

those critical designs in order to reduce estimator variance. On the other hand, limited computational effort should be expended on non-critical designs that have little effect on identifying the good designs. Overall simulation efficiency is improved as less computation effort is spent on simulating non-critical designs and more is spent on critical designs. In this paper, the OCBA mechanism is implemented for trying to find the optimal maintenance policies for the system. The main purpose of this article is to validate the efficiency of these methods in the application of maintenance policy optimization.

The remainder of this paper is organized as follows: Section 2 provides a complete description of the problem and an introduction to the OCBA mechanism. Section 3 presents the simulation model used for the evaluation of system performance. Section 4 presents a case study to show the efficiency of the OCBA method compared with the equal allocation approach. Section 5 discusses the issues of OCBA and wraps up the conclusion.

2 PROBLEM DESCRIPTION AND OCBA

Suppose we have a complex system with n components. Each component has its lower bound LB_i and upper bound $\mathrm{UB_i}$ of PM interval $, i = 1,2, \dots, n$. In simulation optimization, the decision variables for this problem are the PM interval $p \in [LB, UB]$ of each component. They could be continuous variables and thus it is a simulation optimization problem with the continuous decision variables as a vector $P = \{p_1, p_2, \dots p_n\}$. Usually it is more difficult to solve a simulation optimization with continuous decision variables than one with discrete decision variables. Here we discretize the decision variables and put it as a design selection problem (discrete simulation optimization problem). For the PM interval of component $i, p_i \in [LB_i, UB_i]$, we do the discretization in this way: suppose the discretization factor is K_i , then the new decision variable p_i becomes $p_i \in \left\{LB_i + \frac{k_i(UB_i - LB_i)}{K_i},\ k_i = 0, 1, 3, \dots, K_i\right\}$. The total number of system design alternatives is $Q = \prod_{i=1}^{n}(K_i + 1)$. A system design q can be defined as $D_q = \{p_{1*}, p_{2*}, \dots p_{N*}\}, q = 1,2, \dots Q$, where p_{i*} denotes an arbitrary PM interval alternative of component i. A simulation-based optimization problem for such a system can be defined as:

$$\min_{D_q \in U} S(D_q) \equiv E\left[L(D_q, \xi)\right] \tag{1}$$

where $\mho$ is the search space (the total design alternatives). S is the performance criterion (system unavailability or costs). It is the expectation of L and as a function of system design D and ξ (a random vector that represents uncertain factors in the system). Note that for the complex systems considered in this paper, $L(D_q, \xi)$ is obtained only via simulation. The way to estimate $E(L(D_q, \xi))$ is by the sample mean performance measure:

$$\overline{S} = \frac{1}{N_q}\sum_{j=1}^{N_q} L(D_q, \xi_{qj}) \tag{2}$$

where ξ_{qj} represents the jth sample of ξ and N_q represents the number of simulation samples for design q.

Based on the sample mean performance above, the ordinal comparison is used to identify the best design. It should be noted that the behavior of ordinal comparison studied here is sample-varying as opposed to the conventional (static with possible replications) order statistics. With some samples of simulations for each design, the probability of correct selection or $P\{CS\}$ is defined as follows to characterize the behavior of ordinal comparison.

$$\begin{aligned}&\Pr\{CS\}\\ &= \Pr\{\textit{observed best design b is actually the best design}\}\\ &= \Pr\{S(D_b < S(D_i)\},\quad i \neq b \mid L(D_i, \xi_{ij}), \forall i, j\end{aligned} \tag{3}$$

Here b is the observed best design. Instead of equally simulating all designs, the OCBA will improve the performance of ordinal comparison by determining the best number of simulation samples for each design. Precisely, it chooses $N_1, N_2, \dots, N_Q$ such that $P\{CS\}$ is maximized, subject to a limited computing budget T,

$$\begin{aligned}&\max_{N_1, N_2, \cdots, N_Q} \Pr\{CS\}\\ &st. N_1 + N_2 + \cdots + N_Q = T\end{aligned} \tag{4}$$

where N is the set of non-negative integer and $N_1 + N_2 + \cdots + N_Q$ denotes the total computational cost, assuming the simulation times for different designs are roughly the same. $P\{CS\}$ can be approximated based on a Bayesian model when the number of designs is large.

In our problem, $E(L(D_q, \xi))$ is estimated by the sample mean performance measure. Thus the simulation output tends to be normally distributed. After the simulation is performed, a posterior distribution of $S(D_q), p(S(D_q)|L(D_q, \xi_{qj}), j = 1, 2, \dots, N_q)$, can be constructed as:

$$\tilde{S}_q = N\left(\overline{S}_q, \frac{\sigma_q^2}{N_q}\right) \tag{5}$$

Here σ_q^2 is the variance for design q, i.e., $\sigma_q^2 = Var(L(D_q, \xi))$, which can be approximated by sample variance. $\tilde{S}_q$ denotes the random variable whose probability distribution is the posterior distribution for design q . According to the Bonferroni inequality $P\{\bigcap_{i=1}^{Q}(Y_i < 0)\} \geq 1 - \sum_{i=1}^{Q}[1 - P\{Y_i < 0\}]$, the lower bound for the probability of correct selection can be obtained as:

$$\begin{aligned}\Pr\{CS\} &= \Pr\left\{\bigcap_{i=1, i\neq b}^{Q}(\tilde{S}_b - \tilde{S}_i) < 0\right\}\\ &\geq 1 - \sum_{i=1, i\neq b}^{Q}\left(1 - \Pr\{\tilde{S}_b - \tilde{S}_i < 0\}\right)\\ &= 1 - \sum_{i=1, i\neq b}^{Q}\Pr\{\tilde{S}_b > \tilde{S}_i\} = APCS\end{aligned} \tag{6}$$

The lower bound of the correct selection probability is referred as Approximate Probability of Correct Selection (APCS). As the simulation proceeds, APCS is used to approximate $P\{CS\}$ and the problem in (4) becomes:

$$\max_{N_1,N_2,\cdots,N_Q} 1-\sum_{i=1,i\neq b}^{Q}\Pr\{\tilde{S}_b > \tilde{S}_i\}$$
$$st.\ N_1+N_2+\cdots+N_Q = T \tag{7}$$

The APCS can be asymptotically maximized when the following asymptotic allocation rule is met [4]:

a) $\frac{N_i}{N_j}=\left(\frac{\sigma_i/\delta_{b,i}}{\sigma_j/\delta_{b,j}}\right)^2, i,j \in \{1,2,\ldots,Q\}\ and\ i\neq j\neq b$

b) $N_b=\sigma_b\sqrt{\sum_{i=1,i\neq b}^{Q}\frac{N_i^2}{\sigma_i^2}}$

where N_i is the number of samples allocated to design i, $\delta_{b,i}=\bar{S}_b-\bar{S}_i$ and $\bar{S}_b \leq min_i\,\bar{S}_i$

Based on the discussion above, the OCBA algorithm works in this way: At the beginning, n_0 simulation replications are conducted on each design to get some information about the performance of each design during the first phase. As simulation proceeds, the sample means and the sample variance of each design are computed from the data already collected up to that phase. Based on the simulation output, the new computing budget of the next phase (the computing budget of the previous phase plus an extra computing budget, Δ) is allocated according to the asymptotic allocation rule above. The procedure is continued until the total budget T is exhausted. The algorithm is summarized as follows:

Algorithm of OCBA

Step 01: Perform n_0 simulation replications for all designs; $l=0; N_1^l=N_2^l=\cdots=N_Q^l=n_0$

Step 02: If $T^l=\sum_{i=1}^{Q}N_i^l > T$, then stop.

Step 03: Increase the computing budget (i.e., number of additional simulation) by Δ. $T^{l+1}=T^l+\Delta$

Step 04: $\alpha_s=1, \alpha_i=\frac{N_i}{N_s}=\left(\frac{\frac{\sigma_i}{\delta_{b,i}}}{\frac{\sigma_s}{\delta_{b,s}}}\right)^2, i,s \in \{1,2,\ldots,Q\}$ $and\ i\neq s\neq b.$

Step 05: $\alpha_b=\sigma_b\sqrt{\sum_{i=1,i\neq b}^{Q}\frac{\alpha_i^2}{\sigma_i^2}}$.

Step 06: Indicate$(i)=1, i=1,2,\ldots,Q$

Step 07: If Indicate$(i)=1,\ i=1,2,\ldots,Q$, then

Step 08: $N_i^{l+1}=\left\lfloor T^{l+1}\times\frac{\alpha_i}{\sum_{j=1}^{Q}\alpha_j}\right\rfloor, i=1,2,\ldots,Q$

Step 09: If $N_i^{l+1}<N_i^l, i=1,2,\ldots,Q$, then,

Step 10: $N_i^{l+1}=N_i^l$;

Step 11: Indicate(i) $=0$;

Step 12: $T^{l+1}=T^{l+1}-N_i^{l+1}$;

Step 13: Go to ***Step 07***.

Step 14: $N_b^{l+1}=N_b^{l+1}+(T^{l+1}-\sum_{i=1}^{Q}N_i^{l+1})$

Step 15: Perform additional $max\ (0, N_i^{l+1}-N_i^l)$ simulations for design $i=1,2,\ldots,Q$

Step 16: $l=l+1$, go to ***Step 02***.

In the above algorithm, s is the second best design; l is the iteration number; $\lfloor . \rfloor$ is the highest lower integer function (the slightly different budget allocated to design b at ***Step 14***). As simulation evolves, the design with the best sample mean, design b, may change from iteration to iteration, although it will converge to the optimal design as l goes to infinity.

3 THE SIMULATION MODEL FOR SYSTEM EVALUATION

In this section we present the basic assumptions behind the simulation model used for the evaluation of system performance.

As we mentioned in the introduction, in this paper, only two types of maintenance policies are considered in the simulation model: PM and CM. Within our simulation model, the maintenance strategies mainly refer to finding the optimal PM interval for each component so that the total cost will be minimal. However, it is easy to extend our model to account for other maintenance strategies such as spare part policy, repair crew policy, etc.

Preventive maintenance is meaningful only when the following two conditions are met:

a) The components have an increasing failure rate. In other words, the failure rate of the component increases with time, thus implying wear out.
b) The overall cost of the preventive maintenance action must be less than the overall cost of a corrective action.

In the simulation model, there are two kinds of preventive maintenance intervals available, the PM interval based on system age and the PM interval based on item age. In simulation, the system and each component of the system maintain separate clocks within the simulation. The system clock is the simulation elapsed time, while each item clock is the age of the item since last renewal (for example, a failure of a component causes the system to go down; when the system is down, other components in the system stop operating and their ages do not change during the system down time). If the system clock is used, the PM will be performed every X time units. If the item clock is used, the PM will be performed every time the component reaches that age. For example, if the PM is set to be performed at a system age of 100, then a PM will be performed at 100, 200, 300 and so forth. If the PM is set based on an item's age of 100, then the PM will be performed when the item reaches an age of 100.

For the evaluation of the various maintenance strategies we consider the system overall cost during the mission time T_M. The system overall cost can be calculated as follows:

$$\begin{aligned} Cost_{Total} &= CostRate_{System} * DownTime_{System} \\ &+ CostPerFailure_{System} * NumberOfFailures_{System} \\ &+ \sum_{i=1}^{n} CostRate_i * Downtime_i \\ &+ \sum_{i=1}^{n} CostPerFailure_i * NumberofFailures_i \\ &+ \sum_{i=1}^{n} PM\_CostRate_i * PM_{Downtime_i} \\ &+ \sum_{i=1}^{n} CostPerPM_i * NumberOfPM_i \end{aligned} \tag{8}$$

$CostRate_{Sytem} * DownTime_{System}$: The product of system down time rate and system down time; can be used to account for costs such as production loss cost.

$CostPerFailure_{System} * NumberOfFailure_{System}$: Can be used to account for the cost of system disruption due to system failures.

$CostRate_i * DownTime_i$: Component level down time cost.

$CostPerFailure_i * NumberOfFailure_i$: Component level disruption cost due to component failure.

$PM\_DownTimeRate_i * PM\_DownTime_i$: Down time cost due to PM actions; can be used to account for costs such as labor costs of the repair crews.

$PM\_CostPerFailure_i * NumberOfPM_i$: The one-time cost for each PM action, which can be used to account for the part cost or handling cost, etc. The CM cost is similar explanation as that of PM cost.

As discussed in the previous section, for a system with n components, the search space would be $\prod_i^n (K_i + 1)$. In order to reduce the search space and simplify the problem further, here we group components into several groups and perform the PMs based on groups. For components within the same group, they share the same k and K. This means there are only K solutions within a group. For example, n components are grouped into m groups. For components within group j, they share the same k' and K'_j. Here $0 < j \leq m$ and $m \leq n$. The search space (total designs) then would become $\prod_j^m (K'_j + 1)$. As in Table 2, 7 types of components are grouped into 3 groups. For group 1, only 10 solutions are available. When $k'_1 = 0$, there is a solution {14, 16, 14.4} for blocks A, E, and F. When $k'_1 = 1$, another solution {28, 32, 28.8} is available, et al. Of course, reducing the search space would also reduce the possibility to get a more optimal design.

4 CASE STUDY: A CHEMICAL PROCESS PLANT

In the following, we illustrate the application of the proposed optimization approach on a case study taken from literature (some necessary adjustments are made).

The system under consideration is a chemical process plant [2]. The model contains standby units, load sharing units and k-out-of-n units. Components A and B are warm standby units. Component B is in a standby status with reduced failure rate (dormant factor = 0.5) when component A is operational. If component A fails, component B would activate immediately. A perfect switch is assumed. The components C1 through C5 are arranged into a 4-out-of-5 cold standby configuration. Components F and G work in a 2-out-of-2 configuration. The two H components are in parallel but they are related through a load connection according to which failure of one of them leads to an increased load on the surviving component, which is then forced to work in more stressful conditions so that its failure rate increases. The reliability diagram for this system is shown in Figure 1. The components failure, repair and cost data are shown in Table 1.

There are 12 components in this system. C1 to C5 are the same type of component; H1 and H2 are the same type of component too. Here we assume that, for the same type of components, their PM intervals are the same. Thus the number of PM intervals we need to consider reduces to 7 (corresponding to the 7 types of components here). To further simply the problem, we group these 7 types of components into 3 $(m = 3)$ groups $\{A, F, G\}, \{B, H\}, \{C, E\}$ mainly based on the Eta of their failure distribution. The total number of designs becomes $\prod_j^3 (K_j + 1) = 10 * 5 * 5 = 250$ (there is not combination within a group; for components in the same groups, they share the same K and k). The lower bound, upper bound and K for each block are listed in Table 2. All PM intervals in this example are based on the block's item age.

In order to evaluate the efficiency of the OCBA approach, the experiments are conducted in this way: 20,000 simulations are run on each of the 250 designs. Based on the total costs of these 250 designs, we pick the minimal one as the true best design. Given the true best design, we can estimate the probability of correct selection ($P\{CS\}$) for the OCBA approach and equal allocation approach. The sample size for estimating the $P\{CS\}$ is 1000, which means for each computing budget level, we run 1000 times for both methods and see how many times they pick the true best design as their final best solution.

Mission time is 100 hours. With 20,000 simulations, design 33 (k_1=2, $k_2 = k_3$=3) corresponding solution {A:28, F:32, G:28.8}{B:25.2, H:27.6} {C:10.8, E:16.8 } comes out as the true best design with total cost = 4126.032. At the system level, there are 19.86 system failures (number of CMs) and 8.2419 PMs are conducted. The total system down time is 29.795 hours.

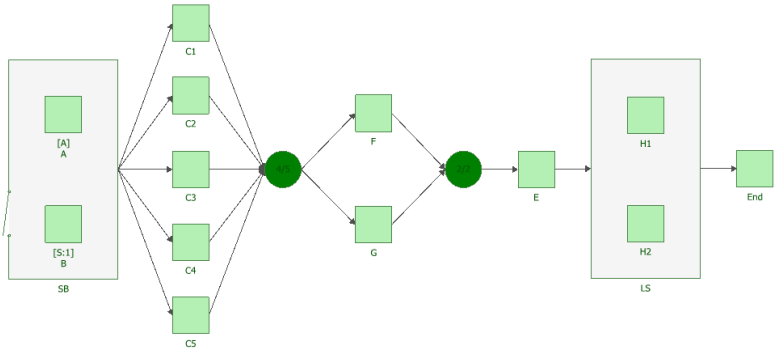

Figure 1: Reliability block diagram for the chemical process plant

Table 1: Components failure, repair and cost data

Blocks	Failure Distribution		CM			PM			Block Level		System Level	
	Beta	Eta	Duration	Cost Rate	Cost per CM	Duration	Cost Rate	Cost per PM	Cost Rate	Cost per Failure	Cost Rate	Cost per Failure
A	1.5	35	1.5	12	15	.75	8	12	2	7	30	50
B	1.5	21	1.6	15	13	.8	13	11	3	5		
C	1.5	9	1.2	7	17	.6	6	14	1	4		
F	1.5	40	2.5	10	18	1.25	8	15	2	6		
G	1.5	36	1.28	8	9	.64	5	7	4	3		
E	1.5	14	2.3	7	17	1.15	7	14	4	2		
H	1.5	23	3.28	13	14	1.64	10	12	6	5		

Table 2: Lower bound, upper bound and K for each block

Groups	Blocks	Lower Bound	Upper Bound	*K*	Possible PM Intervals
1	**A**	14	140	9	{14, 28, 42, 56, 70, 84, 98, 112, 126, 140}
	F	16	160		{16, 32, 48, 64, 80, 96, 112, 128, 144, 160}
	G	14.4	144		{14.4, 28.8, 43.2, 57.6, 72, 86.4, 100.8, 115.2, 129.6, 144}
2	**B**	8.4	42	4	{8.4, 16.8, 25.2, 33.6, 42}
	H	9.2	46		{9.2, 18.4, 27.6, 36.8, 46}
3	**C**	3.6	18	4	{3.6, 7.2, 10.8, 14.4, 18}
	E	5.6	28		{5.6, 11.2, 16.8, 22.4, 28}

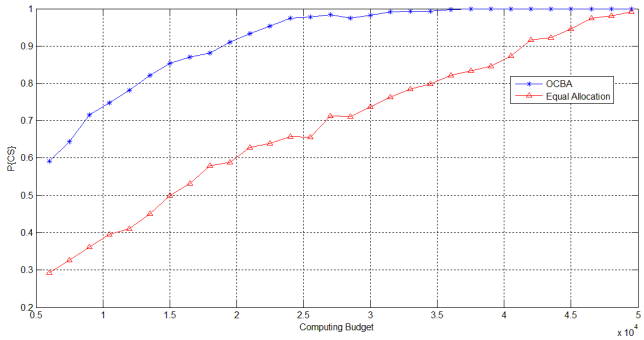

Figure 2: $P\{CS\}$ vs. computing budget for OCBA and equal allocation method for case study

Table 3: $P\{CS\}$ vs. computing budget for OCBA and equal allocation method for case study

Computing Budget (10^3)	6	7.5	9	10.5	12	13.5	15	16.5	18	**19.5**	21	22.5	24	25.5	27
OCBA, $P\{CS\}$	.591	.643	.716	.748	.781	.822	.853	.871	.881	**.911**	.933	.954	.975	.978	.984
Equal Allocation, $P\{CS\}$	.292	.327	.361	.396	.410	.450	.499	.532	.580	**.589**	.629	.639	.657	.656	.713
Computing Budget (10^3)	28.5	30	31.5	33	34.5	36	37.5	39	40.5	**42**	43.5	45	46.5	48	49.5
OCBA, $P\{CS\}$	.974	.983	.991	.993	.993	.998	1	1	1	**1**	1	1	1	1	1
Equal Allocation, $P\{CS\}$	.711	.737	.763	.784	.799	.822	.834	.846	.874	**.916**	.923	.945	.975	.981	.991

Figure 2 depicts the simulation results for picking the best design based on the OCBA and equal allocation methods. We can see the relative performance of these two methods. Using OCBA is about twice as fast as using the equal allocation method. From Table 3, in order to get a $P\{CS\}$) greater than .9, a computing budget of 42,000 is needed for the equal allocation method, while only 19,500 is needed for OCBA (the numbers in bold). In addition, we can see that OCBA is much more efficient even when the total computing budget is small.

5 DISCUSSION AND CONCLUSION

The case study results in the previous section show that, compared with the traditional equal allocation approach, the OCBA mechanism does improve the simulation optimization process significantly. However, two issues need to be discussed here.

The first one is the application of OCBA in a difficult scenario where the best design in the maintenance policy optimization problem is not stable within a limited computing budget. For example, suppose there are 1000 computing

budget available for 100 designs. This means that on average, a computing budget of 10 can be allocated to each design. However, for some maintenance policy problems, even with a computing budget of 1000 for each design (total computing budget would be 100*1000), the best design cannot be identified due to there being several similar designs with closed objective values (the best design is not stable). In this scenario, with a limited computing budget (1000), the best design is not guaranteed. An acceptable solution could be a solution from the good solution set, such as the top-10 best solution set, etc. Theoretically, the OCBA is designed to pick the single best design. Thus, in the future, more experiments need to be conducted to evaluate the ability of the OCBA for picking one of the good solutions instead of always picking the best one. An extension of OCBA for selecting an optimal subset is available in [6]. However, the application background in [6] is slightly different from the issue discussed here. It is designed to select all good solutions to from an optimal subset. The $P\{CS\}$ in [6] is defined as $P\{observed\ optimal\ subset\ is\ actually\ the\ optimal\ subset\}$, while the $P\{CS\}$ for the issue discussed here should be defined as $P\{observed\ best\ design\ is\ actually\ among\ the\ optimal\ subset\}$. Further research would focus on the extension of OCBA to account for this issue.

The second issue refers to the trade-off between the computing cost of each replication and the cost of application of OCBA. The execution of OCBA involves extra computing cost, so OCBA is only cost effective when the computing cost of each replication is high.

REFERENCES

1. J. K. Vaurio, "Optimization for test and maintenance intervals based on risk and cost," *Reliability Engineering and System Safety*, vol. 49, no. 1, pp. 23-36, 1995.
2. M. Marseguerra and E. Zio, "Optimizing maintenance and repair policies via a combination of genetic algorithms and Monte Carlo simulation,". *Reliability Engineering and System Safety*., vol. 68, no. 1, pp. 69-83, April 2000.
3. M. Marseguerra, E. Zio, and L. Podofillini, "Condition-based maintenance optimization by means of genetic algorithm and Monte Carlo simulation," *Reliability Engineering and System Safety*., vol. 77, no. 2, pp. 151-166, August 2002.
4. Chun-Hung Chen, Jainwu Lin, Enver Yucesan, and Stephen E. Chick, "Simulation budget allocation for further enhancing the efficiency of ordinal optimization," *Discrete Event Dynamic Systems: Theory and Applications*, vol, 10, pp. 252-270, 2000.
5. Dingzhou Cao, Shaobai Kan and Yu Sun, "Design of reliable system based on Dynamic Bayesian Networks and Genetic Algorithm", *Proc. Ann. Reliability & Maintainability Symp.*, (Jan.) 2012.
6. Chun-Hung Chen, Donghai He, Michael Fu, and Loo Hay Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579-595, Fall 2008.

ACKNOWLEDGEMENTS

We are thankful to our colleague Ms. Melinda Caroline for her helps in improving the presentation clarity of this article.

BIOGRAPHIES

Dingzhou Cao
ReliaSoft Corporation
1450 S. Eastside Loop
Tucson, AZ 85710

e-mail: Dingzhou.Cao@ReliaSoft.com

Dr. Dingzhou Cao is a Research Scientist at ReliaSoft Corporation. He received his Ph.D. in Industrial Engineering from Wayne State University. He is the key person with full responsibility in developing the simulation engine for ReliaSoft's products: BlockSim, RCM and Reno.

Yu Sun
Siena College
515 Loudon Road
Loudonville, NY 12211

e-mail: ysun@siena.edu

Dr. Yu Sun is an Assistant Professor at Quantitative Business Analysis Department, Siena College. She received her M.S. in applied Mathematics and Ph.D. in Management Science from Donghua Univeristy, China; M.A. in Mathematical Statistics and Ph.D. in Mathematics from Wayne State University, USA. Her research interests include mathematical statistics, stochastic approximation and optimization, stochastic control and applied probability, modeling, as well as quantitative business analysis.

Huairui Guo
ReliaSoft Corporation
1450 S. Eastside Loop
Tucson, AZ 85710

e-mail: Harry.Guo@ReliaSoft.com

Dr. Huairui Guo is the Director of the Theoretical Development Department at ReliaSoft Corporation. He received his Ph.D. in System & Industrial Engineering and M.S. in Reliability & Quality Engineering; both from the University of Arizona. His research and publications cover reliability areas, such as life data analysis, repairable system modeling and reliability test planning, and quality areas, such as process monitoring, analysis of variance and design of experiments. He is a certified reliability professional (C.R.P.), ASQ certified CQE. He is a member of IIE, SRE and ASQ.