Optimal Computing Cost Budget Allocation For Solving Buffer Allocation Problem

Mahmoud H. Alrefaei* **and Ahmad A. Abubaker†**

* Department of Mathematics and physics
Qatar University
P.O. Box 2713, Doha, QATAR
E-mail: malrefaei@qu.edu.qa
† College of Computer and Information Science
Al-Imam Muhammad Ibn Saud Islamic University
P.O.Box 84880, Alriyadh 11681, Saudia Arabia
E-mail: a_abubaker2000@yahoo.com

In this paper, we study the problem of selecting a system that has the best performance or selecting one of the best systems when the objective function is the expected performance of a complex stochastic system. The focus here is on a huge size but finite feasible solution set, that is not necessarily well structured. We consider the use of the ordinal optimization technique that concerns with the order of the designs in the state space rather than estimating each design alternative accurately. We also assume that the computing cost is limited. We discuss how to allocate the available computational resources on the alternative designs to maximize the probability of making a correct selection (selecting the best or one of the best systems).

Keywords: Simulation Optimization, Optimal Computing Budget Allocation, Simulated Annealing, Ordinal Optimization.

1. Introduction

We study the discrete stochastic optimization problems where the objective function is the expected performance of a complex stochastic system. The problem can be expressed as follows:

$$\min_{\theta\in\Theta} J(\theta) \equiv E[L(\theta,\xi_\theta)], \quad \ldots\ldots\ldots \quad (1)$$

where L is a function of θ and ξ_θ, ξ_θ is a random variable depends on the design θ, and Θ is the set of all potential solution candidates or the search space, is a huge finite set and not necessarily well structured. Θ is said to be well structured if for any $\theta \in \Theta$, $\exists$ a set $N(\theta)$, contains all neighbors of θ and all alternatives can be connected through a sequence of neighbors. This type of problems can be found in many real life complex stochastic systems such as communication, manufacturing, and other systems.

When the sample space Θ is small, then numeral methods can be used, where the objective function values in (1) are estimated using simulation. Random samples of stochastic process $\xi_\theta^1, \xi_\theta^2, \ldots, \xi_\theta^N$ of the random variable ξ_θ are generated, and $E[L(\theta,\xi)]$ is estimated as follows:

$$\bar{J}(\theta) = E[L(\theta,\xi_\theta)] \approx \frac{1}{N}\sum_{i=1}^{N} L(\theta,\xi_\theta^i) \quad \ldots\ldots \quad (2)$$

where ξ_θ^i represents the ith sample of design θ, and N is the number of replications (i.e., the number of simulation runs). Then use any deterministic optimization technique to solve the alternative problem by replacing $J(\theta)$ by $\bar{J}(\theta)$. This method requires large computational time to do simulation runs, especially when the set of feasible solutions is large. Therefore, this method is considered infeasible. Several techniques are proposed to tackle this problem. In this paper, we focus on one of these techniques, particularly the ordinal optimization technique and compare its performance with the simulated annealing algorithm in a simple practical example.

The rest of this paper is organized as follows; in Section 2, we preview a simulated annealing algorithm, in Section 3, we review the ordinal optimization approach and in Section 4, we present the optimal computing cost budget allocation technique. In Section 5, we present numerical results obtained from applying the proposed approach to a buffer allocation problem for homogeneous asymptotically reliable serial production lines. We compare the results and computational time to results obtained by applying the most recent simulated annealing algorithms to the same problem. Finally, in Section 6, we end by some concluding remarks.

2. Simulated annealing with the standard clock technique

Some researchers have studied the use of simulated annealing to solve discrete simulation optimization problems. The simulated annealing algorithm has originally been proposed by Kirkpatrick et.al. [9] who use the idea of simulated annealing to solve deterministic optimization problems. Gelfand and Mitter [6] and Gutjahr and Pflug [7] have proposed and analyzed the simulated annealing for solving stochastic optimization problems. Al-

refaei and Andradóttir [3] have proposed another version of simulated annealing that is different from all other approaches in which it uses different criterion for estimating the optimal solution. Alrefraei [2] proposes using the idea of standard clock simulation combined with the simulated annealing to reduce the simulation time.

We present the approach of Alrefaei [2] that uses the standard clock simulation technique for estimating several objective function values simultaneously, using one sample path. The algorithm has the hill climbing feature as in the simulated annealing algorithm. It starts by an initial state and then searches the neighborhood of the current state in order to locate the best states. An estimate of the objective function value for each state in the neighborhood set of the current state are obtained simultaneously using only one simulation run. Then good states are selected with high probability as the next current state, other states are not neglected, they are selected with probability depending on the estimate of their object function values.

2.1. The SA algorithm

Now we present the simulated annealing algorithm that uses the standard clock simulation technique, see Alrefaei [2].

step 0: Select a starting point $X_0 = \theta_0 \in \Theta$, let $V_0(\theta_0) = 1$ and $V_0(\theta) = 0$ for all $\theta \in \Theta$ and $\theta \neq X_0$, let $k = 0$ and $X_k^* = X_k$.

step 1: Generate independent samples $\xi_1(\theta), \xi_2(\theta), \ldots, \xi_{L_k}(\theta)$ from $\xi(\theta)$ for all $\theta \in N(X_k) \cup \{X_k\}$, simultaneously using the standard clock simulation technique, compute $J_k(\theta)$ as follows:

$$\bar{J_k}(\theta) = \frac{1}{L_k} \sum_{i=1}^{L_k} L(\theta, \xi_i(\theta))$$

step 2: Let $\hat{R}(X_k)$ be defined as follows:

$$\hat{R}(X_k) = \sum_{\theta \in N(X_k) \cup \{X_k\}} \exp\left[-C\left[\bar{J_k}(\theta) - \bar{J_k}(X_k)\right]^+\right],$$

where C is a constant used as a scaler. Set $X_{k+1} = \theta$ with probability

$$P_{X_k,\theta} = E\left[\frac{\exp\left[-C\left[\bar{J_k}(\theta) - \bar{J_k}(X_k)\right]^+\right]}{\hat{R}(X_k)}\right] \quad (3)$$

step 3: Let $k = k+1$, $V_k(X_k) = V_{k-1}(X_k) + 1$ and $V_k(\theta) = V_{k-1}(\theta)$ for all $\theta \in \Theta$ and $\theta \neq X_k$.
If $V_k(X_k)/\hat{R}(X_k) > V_k(X_{k-1}^*)/\hat{R}(X_{k-1}^*)$ then let $X_k^* = X_k$, otherwise let $X_k^* = X_{k-1}^*$. If the stopping criterion is not reached, go to step 1.

Alrefaei [2] has shown that the Markov chain $\{X_k^*\}$, generated by the algorithm converges almost surely to a global optimal solution. If the system we are considering is not a Markovian so that standard clock is not applicable, then one can replace step 1 by:

step 1′: Generate independent samples $\xi_1(\theta)$, $\xi_2(\theta)$, $\ldots, \xi_{L_k}(\theta)$ from $\xi(\theta)$ for all $\theta \in N(X_k) \cup \{X_k\}$ and compute $J_k(\theta)$ as above.

3. The ordinal optimization technique

The ordinal optimization approach has been proposed by Ho et.al. [8] for finding a satisfactory alternative rather than obtaining an accurate estimate of each alternative. The ordinal optimization is particularly used when the number of alternatives is huge (systems that have large number of alternatives). The basic idea of ordinal optimization for discrete event dynamic systems is simple; instead of finding the best solution, ordinal optimization concentrates on finding a good or better systems rather than trying to find the true best system. Ordinal optimization finds a satisfactory "good enough" solution within a significantly reduced computational time and cost. The ordinal optimization has been implemented in many applications in real-life where finding a "good enough" solution is satisfactory. Such applications include, ecology, air traffic control, manufacturing automation, buffer allocation in production lines, queuing networks, parallel networks and computer performance.

To illustrate the idea of ordinal optimization, suppose we have N alternatives in Θ, where N is a large number. Let G be the "good enough" subset, and $|G| = g$, (i.e., G contains the best g states in Θ). Let S be the selected subset from the design space Θ, where S is selected randomly, such that $|S| = s$. The quality of selection is based on the overlap of S with G. Then the probability that at least k of the observed s designs actually belong to the top g designs is given by:

$$P(|G \cap S| \geq k) = \sum_{i=k}^{g} \frac{\binom{g}{i}\binom{N-g}{s-i}}{\binom{N}{s}} \quad . \quad (4)$$

The technique of optimal computing budget allocation has been proposed by Chen et. al. [4] to enhance the use of ordinal optimization. It is assumed that the total computational budget is limited, and the problem is how these samples are allocated to each design in order to maximize the probability of correct selection (CS), denoted by $P(CS)$. The correct selection is to select a design b which is the actual best design or to select a set of designs that contain the best design with very high probability.

The following theorem by Chen et.al. [4] determines the number of samples that need to be allocated for each design in the design space in order to select the actual best design with high probability.

Theorem 1: Given a total number of simulation samples TC to be allocated to k computing designs whose performance is depicted by random variables with means $J(\theta_1), J(\theta_2), \ldots, J(\theta_k)$, and finite variances $\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2$ respectively. As $TC \longrightarrow \infty$, the approximate probability of correct selection can be asymptotically maximized

when

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

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

Where N_i is the number of samples allocated to design i, $\delta_{b,i}$, the estimated difference between the performance of the two designs ($\delta_{b,i} = \bar{J_b} - \bar{J_i}$), and $\bar{J_b} \geq \max \bar{J_i}$ for all i (Here we assume that we are maximizing the function $J(\theta)$). Here $\bar{J_i} = \frac{1}{N_i}\sum_{j=1}^{N_i} L(\theta_i, \xi_{ij})$, where ξ_{ij} is a sample from the distribution of ξ_i for $j = 1,2,\ldots,N_i$.

4. Optimal computing cost budget allocation (OCCBA)

In this section, we propose a procedure that determines the best allocation of simulation trials or samples necessary to maximize the probability of correct selection, when the budget of the total cost of simulation runs is limited. Suppose that it costs c_i for each simulation run for design i, and we have a limited budget for the total cost of simulation runs, (TC). The budget constraint can be written as $\sum_{i=1}^{k} c_i N_i = TC$, where N_i, $i = 1,2,...,k$ is the number of samples for the ith design.

The proof of the following theorem is given by Abubaker [1].

Theorem 2: Given a total costs of simulation runs TC to be allocated to k computing designs whose performance is depicted by random variables with means $J(\theta_1), J(\theta_2), \ldots, J(\theta_k)$, and finite variances $\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2$ respectively. As $TC \longrightarrow \infty$, the approximate probability of correct selection can be asymptotically maximized when

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

b) $N_b = N_s \delta_{b,s} \frac{\sigma_b}{\sigma_s} \sqrt{\frac{c_s}{c_b} \sum_{i=1,i\neq b}^{k} \frac{1}{\delta_{b,i}^2}}$.

Where N_i is the number of samples allocated to design i, $\delta_{b,i}$ the estimated difference between the performance of the two designs ($\delta_{b,i} = \bar{J_b} - \bar{J_i}$), and $\bar{J_b} \geq \max \bar{J_i}$ for all i. Here $\bar{J_i} = \frac{1}{N_i}\sum_{j=1}^{N_i} L(\theta_i, \xi_{ij})$, where ξ_{ij} is a sample from ξ_i for $j = 1,2,\ldots,N_i$.

Remark: Chen et.al. [5] have derived a formula for the relation between N_i and $N_j, i \neq j \neq b$ and between N_b and $N_i, i \neq b$. The formula depends on the following bound for normal distribution; If Y is normally distributed random variable with mean $\delta_{b,i}$ and variance $\sigma_{b,i}^2$, then $P(Y < 0) \leq \exp\left(\frac{-\delta_{b,i}^2}{2\sigma_{b,i}^2}\right)$. When $x = \frac{\delta_{b,i}}{\sigma_{b,i}}$ is large, then this bound is not good; for example if $x = 1$ then $P(Y < 0) = 0.15866$ but $\exp(-x^2/2) = 0.60653$.

If we take any design s in the above theorem where $s \neq b$, we get

$$N_i = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_s/\delta_{b,s}}\right)^2 N_s,$$

where $i = 1,2,\ldots,k$, and $s \neq i \neq b$. Suppose

$$D_i = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_s/\delta_{b,s}}\right)^2$$

then

$$N_i = D_i N_s. \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5)$$

We also know that

$$N_b = N_s \delta_{b,s} \frac{\sigma_b}{\sigma_s} \sqrt{\frac{c_s}{c_b} \sum_{i=1,i\neq b}^{k} \frac{1}{\delta_{b,i}^2}}$$

Suppose

$$D_b = \delta_{b,s} \frac{\sigma_b}{\sigma_s} \sqrt{\frac{c_s}{c_b} \sum_{i=1,i\neq b}^{k} \frac{1}{\delta_{b,i}^2}}$$

then

$$N_b = D_b N_s. \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (6)$$

Since $\sum_{i=1}^{k} c_i N_i = TC$ then $N_s \sum_{i=1}^{k} c_i D_i = TC$, where we assume that $D_s = 1$ this implies that $N_s = \frac{TC}{\sum_{i=1}^{k} c_i D_i}$, and that

$$N_i = \frac{D_i TC}{\sum_{j=1}^{k} c_j D_j}, \quad i = 1,2,\ldots,k \quad \ldots\ldots \quad (7)$$

To enhance the performance of ordinal optimization, we combine the optimal computing cost budget allocation in order to allocate the number of allowed samples on the competent alternatives smartly.

4.1. The OCCBA Algorithm

The OCCBA Algorithm is described as follows:

step 0: Initialization:

- 0.1: Determine the size of the subset G, let $|G| = g$.
- 0.2: Determine the number of initial simulation sample t_0.
- 0.3: Determine the number of increment simulation samples Δ.
- 0.4: Determine the total budget of computing cost TC.
- 0.5: Determine the cost c_θ associated to alternative θ, for all $\theta \in \Theta$.
- 0.6: Select a starting point $\theta_0 \in \Theta$, let $l = 0$, where l is the iteration number.

step 1: Select a subset G of g alternatives randomly from the feasible solution set Θ.

step 2: Let $N_\theta^l = t_0$ for all $\theta \in G \cup \{\theta_l\}$.

step 3: Generate independent samples $\xi_1^\theta, \xi_2^\theta, \ldots, \xi_{N_\theta^l}^\theta$, for all $\theta \in G \cup \{\theta_l\}$

step 4: If $\sum_{\theta \in G \cup \{\theta_l\}} c_\theta N_\theta^l \geq TC$, then go to step 8.

step 5: Compute $\bar{J_l}(\theta)$, $\delta_{\theta_{l+1},\theta}$ and σ_θ for all $\theta = G \cup \{\theta_l\}$ as follows:

$$\bar{J_l}(\theta) = \frac{1}{N_\theta^l} \sum_{i=1}^{N_\theta^l} L(\theta, \xi_i^\theta)$$

$$\theta_{l+1} = \arg \max_{\theta \in G \cup \{\theta_l\}} \{\bar{J_l}(\theta)\}$$

$$\delta_{\theta_{l+1},\theta} = \bar{J_l}(\theta_{l+1}) - \bar{J_l}(\theta)\,, \qquad \theta_{l+1} \neq \theta$$

$$\sigma_\theta = \frac{1}{N_\theta^l - 1} \sum_{i=1}^{N_\theta^l} \left(\bar{J_l}(\theta) - L(\theta, \xi_i^\theta)\right)^2$$

Step 6: Increase the computing budget by Δ and compute the new budget allocation, N_θ^{l+1} for all $\theta \in G \cup \{\theta_l\}$, using equations (5), (6) and (7).

Step 7: Perform additional $\max\{0, N_\theta^{l+1} - N_\theta^l\}$ simulations for all design $\theta \in G \cup \{\theta_l\}$. Let $l \longleftarrow l+1$. Go to step 4.

Step 8: If the stopping criteria is not reached, set $\theta_0 = \theta_l$, $l = 0$ and go to step 1

Note that the parameters given in Step 0, depend on the availability of resources and the way the user would like to proceed. An appropriate setting is to divide the simulation to more than 30 iterations. Using the ordinal optimization technique in step 1, makes the problem easier and accelerates the process of reaching the optimal solution. The OCCBA technique is used in steps 4-7 to select a good alternative from $G \cup \{\theta_l\}$ as the next current alternative, the procedure continues until termination is reached.

5. Numerical example

We consider the buffer allocation problem for homogeneous asymptotically reliable serial production lines. The machines of production lines are assumed to have a positive probability of working and a positive probability of failure. The proposed algorithm is applied in production lines to select the best design that has the maximum production rate when the total buffer capacities are given. Since the set of feasible solutions is well structured in this example, we can use also simulated annealing, so we compare the performance of these two algorithms on this example.

The Buffer allocation model, we consider is described as follows Lim et.al. [10]. A serial production line consists of M machines $M_i, i = 1, \ldots, M$, and $M-1$ buffers B_j, $j = 1, \ldots, M-1$ of finite capacity. It is assumed that each buffer stores at least 1 unit and that the total buffer capacity that can be allocated over all buffers is a constant integer N.

Machine M_i is starved if buffer B_{i-1} is empty, and blocked if buffer B_i is full and machine M_{i+1} is either down, blocked, or busy. An unlimited supply of work pieces is available up stream machine M_1, and an unlimited storage area is present downstream machine M_M. Thus, the first machine is never starved and the last machine is never blocked. Machines have identical service time distribution. A Machine M_i that is not blocked and not starved, can handle a part of production during any time slot with probability δ_i and fails to do so with probability $1-\delta_i$ (production lines in which machines have this property are called homogeneous).

It is assumed that the production line under consideration is homogeneous, asymptotically reliable, that is

$$\delta_i = 1 - \varepsilon \Lambda_i \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad (8)$$

where $0 < \varepsilon \ll 1$ and $\Lambda_i, i = 1, ..., M$, is independent of ε. Λ_i is known as the loss parameters.

Let L_i be the cumulative losses per hour for machine M_i due to its failure and let A_i be the production rate of machine M_i. If M_i does not break down then the probability of handling a job by this machine is $\delta_i = \frac{A_i - L_i}{A_i} = 1 - \frac{L_i}{A_i}$. Since $\delta_i = 1 - \varepsilon\Lambda_i$ where $0 < \varepsilon \ll 1$, it follows that $L_i \ll A_i$. Thus, $\varepsilon\Lambda_i$ refers to the fraction between the cumulative losses of machine M_i and the production rate of machine M_i (i.e., Λ_i depends on δ_i).

Major decisions in designing production lines involve the workload allocation and the buffer allocation problems with respect to an objective function such as, the profit maximization or the production rate maximization for a given total buffer capacity. Our objective in this example is to find the optimal buffer allocations in order to maximize the average production rate of the production line.

Assume we have M machines and $M-1$ buffers, and the total buffer capacities is N. Then we need to find the optimal vector $\theta = (N_1^\theta, \ldots, N_{M-1}^\theta) \in \Theta$ that maximizes $PR_M(\theta)$ given that $\sum_{i=1}^{M-1} N_i^\theta = N$, where Θ is the feasible region and $PR_M(\theta)$ is the mean production rate of a production line consisting of M machines and $M-1$ buffers that is related to design θ. N_i^θ is the capacity of buffer B_i related to design θ, $i = 1, ..., M-1$. For any $\theta \in \Theta$, we can estimate the production rate that depends on the service time μ_i and δ_i=Prob(M_i is working) for all $i = 1, \ldots, M$. We assume that the service time is fixed for all designs and equals one unit time. In other words, the system changes its state at discrete points. To generate an estimate of the production rate $PR_M(\theta)$, we generate $u_i \in Uniform[0,1]$, if $u_i < \delta_i$ then machine M_i is working, otherwise the machine is not working, for all $i = 1, \ldots, M$.

We apply OCCBA Algorithm for production lines with five machines M_i, $i = 1, 2, \ldots, 5$, ($M = 5$) and loss parameters Λ_i, $i = 1, 2, \ldots, 5$, respectively, see Table (1), where the total buffer capacities are given. we compute δ_i by using Equation (8), for all $i = 1, 2, \ldots, 5$, where $\varepsilon = 0.01$, (see Table (2)).

Λ_1	Λ_2	Λ_3	Λ_4	Λ_5
3.1	2.4	1.5	4.4	3.5

Table 1. The loss parameters for the numerical example.

	δ_1	δ_2	δ_3	δ_4	δ_5
$\varepsilon = 0.01$	0.969	0.976	0.985	0.956	0.965

Table 2. δ_i, $i = 1,2\ldots,5$, for $\varepsilon = 0.1$ and 0.01.

The number of designs in the state space can be computed by considering distributing N units on B buffers spaces, which can be considered by placing $B-1$ bars in to the spaces between the N units; this can be done by $\begin{pmatrix} N-1 \\ B-1 \end{pmatrix}$ ways. Therefore the number of different alternative designs is:

$$|\Theta| = \frac{(N-1)!}{(N-B)!(B-1)!} \quad \ldots\ldots\ldots\ldots \quad (9)$$

where N is the total buffer capacities, B is the number of buffers (i.e., $M-1$).

In this example, we assume that $N = 15$ then we have 364 alternatives designs in Θ. Figure (1) shows the production rate when we apply OCCBA Algorithm and compare its performance with the performance of SA Algorithm. We repeat the implementation for 25 iterations and in each iteration, we use the ordinal optimization to select a set G of 12 designs (i.e., $g = 12$), and use optimal computing budget allocation to select the best design. We assume that the total number of samples is $TC = 13,000$ with equal cost, the increment $\Delta = 50$ and the initial sample size $t_0 = 20$. We use SA Algorithm with the following neighborhood structure: the neighborhood of any design θ can be obtained by moving one buffer slot to another buffer, i.e.,

$$N(\theta) = \{\theta' \in \Theta : \exists\, i, j \in \{1,2,\ldots,B\} \quad \ldots \quad (10)$$

where $i \neq j$ such that $\theta' = \theta + e_i - e_j\}$, $\forall\ \theta \in \Theta$

where e_i is a unit vector whose ith term is 1. It is clear from Figure (1) that the optimal computing budget allocation locate the optimal solution faster than the simulated annealing. The optimal vector $\theta = (3,3,4,5)$ with production rate (0.938456). OCCBA takes 9 minutes and 54 seconds, but SA takes 32 minutes and 2 seconds to complete the whole course of simulation. Convergence time of OCCBA takes 6 minutes and 13 seconds, but SA takes 24 minutes and 37 seconds.

Now, we consider a larger size problem by increasing the number of machines to $M = 10$ with $B = 9$ buffers, and a total buffer capacities $N = 20$ and loss parameters vector $(\Lambda_i,\ i = 1,\ldots,10)$ is given by (3.1, 2.4, 1.5, 4.4, 3.5, 1.2, 5.3, 2.9, 1.9, 2.5). By formula (9) we have $|\Theta| = 75,582$ designs. Figure (2) depict the result when the two algorithms are applied. We repeat the algorithms for 35 iterations and in each iteration we use the ordinal optimization to select a set G of 72 designs (i.e., $g = 72$), and use optimal computing budget allocation to select the best design, where $TC = 73,000$, $\Delta = 200$ and $t_0 = 100$. In SA Algorithm we use the neighborhood structure $N(\theta)$, given by Equation (10).

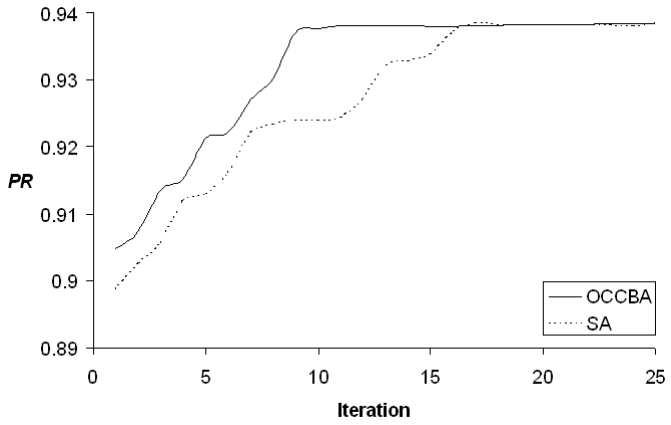

Fig. 1. The Production rate obtained when OCCBA Algorithm and SA Algorithm are applied for 5 machines

It is also clear from Figure (2) that the optimal computing budget allocation algorithm behaves better than the simulated annealing algorithm. The optimal vector obtained by the proposed method is $\theta^* = (2,2,3,3,2,3,2,2,1)$ with production rate (0.909586). The convergence time of OCCBA is 32 minutes and 26 seconds, but for SA is 144 minutes and 57 seconds. By applying the procedure of Lim et.al. [10] on the same vector θ^* we obtain the production rate of (0.911766).

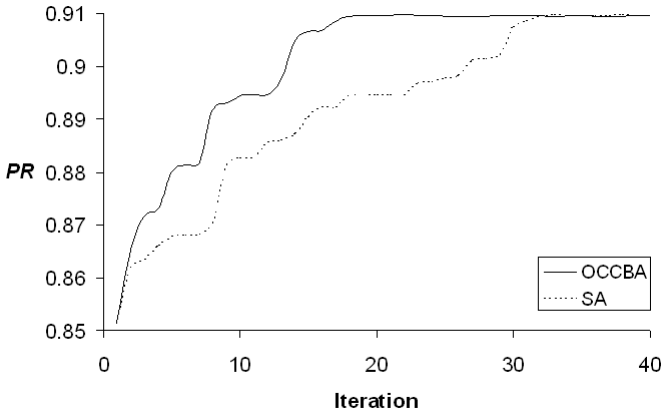

Fig. 2. Production rate by OCCBA Algorithm and SA Algorithm , δ_i, $i = 1,2\ldots,10$ with $\varepsilon = 0.01$

6. Conclusion

We have presented an algorithm that can be used to solve stochastic optimization problems even if the feasible solution set is not well structured. This algorithm is able to locate one of the best alternatives with high probability. The proposed algorithm consists of two sequential stages; in the first stage, the ordinal optimization technique is used to select a small subset from the state space. The optimal computing cost budget allocation (OCCBA) is used to optimally allocate the available computational budget in order to select the best state from the small subset with high probability. Other states are then replaced by newly generated states, the procedure is repeated until the available budget is used. The ordinal optimization technique is used to reduce computational time and cost since it is concerned with the ordinal rather than cardinality of alternative designs. We compare the performance of the proposed algorithm with the performance of a version of the simulated annealing algorithm. It is clear on this example, the proposed algorithm performs well and locate a solution faster.

Acknowledgements

We would like to thank the deanship of Scientific Research at Jordan University of Science and Technology for supporting this research.

References:

[1] Abubaker, A. A., 2005, "Optimal Computoing Cost Budget Allocation for Simulation Optimization". M.Sc. Thesis Jordan Univercity of Science and Technology.

[2] Alrefaei, M. H., 2002, "Stochastic Optimization Using the Standard Clock Simulation". *International Journal of Applied Mathematics*, 8: 317–333.

[3] Alrefaei M. and S. Andradóttir, 1999, "A Simulated Annealing Algorithm with Constant Temperature for Discrete Stochastic Optimization". *Management science*, 45: 748-764.

[4] Chen C. H., J. Lin, E. Yucesan and S. E. Chick 2000, "Simulation Budget Allocation for Further Enhancing the Efficiency of Ordinal Optimization". *Journal of Discrete Event Dynamic Systems: Theory and Application*, 10: 251-270.

[5] Chen C. H., J. Lin, E. Yucesan and K. Donohue 2003, "Optimal Computing Budget Allocation for Monte Carlo Simulation with Application to Product Design". *Simulation Modeling Practice Theory*, 11: 57-74.

[6] Gelfand S. B. and S. K. Mitter 1989, "Simulated Anneling with Noisy or Imprecise Energy Measurements". *Journal of Optimization Theory and Applications*, 62: 49-62.

[7] Gutjahr W. J. and G. CH. Pflug 1996, "Simulated Anneling for Noisy Cost Functions". *Journal of Global Optimization*, 8: 1-13.

[8] Ho Y. C., R. S. Sreenivas and P. Vakili 1992, "Ordinal Optimization of DEDS". *Journal of Discrete Event Dynamic System*, 2: 61-88.

[9] Kirkpatrick S., C. D. Gelatt and M. P. Vecchi 1983, "Optimization by Simulated Annealing". *Science*, 220: 671-680.

[10] Lim J. T., S. M. Meerkov and F. Top 1990, "Homogeneous Asymptotically Reliable Serial Production Lines, Theory and a Case Study". *IEEE Trans. Automat. Control*, 5: 524-534.

Mahmoud Alrefaei is currently an Associate Professor at the Department of Mathematics and Physics at Qatar University. He has received an MA. in Mathematics and a joint Ph.D in Mathematics and Industrial Engineering in the field of Operations Research from the University of Wisconsin-Madison (USA) in 1997. He also worked as a research assistant at the School of Industrial and Systems Engineering at Georgia Institute of Technology. His research interest includes Simulation Optimization, Simulated Annealing, Simulation and Optimization of Queuing Networks. Statistical Selections.

Ahmad Asad Abubaker is currently a lecturer at the College of Computer and Information Science at Al-Imam Muhammad Ibn Saud Islamic University. He has received his MSc. in Mathematics from the Jordan University of Science and Technology. His research interests includes Statistical Selection and Optimization of Queuing Networks.