Arab Journal of Administrative Sciences

2007, Vol. 14, No. 2, 347 - 364

Talal M. Al-Khamis

Kuwait University,
Kuwait

SIMULATED ANNEALING FOR SOLVING A SPECIAL CLASS OF STOCHASTIC OPTIMIZATION PROBLEMS

Key Words

Simulated Annealing; Stochastic Optimization; Production Planning; Simulation.

Abstract

In the field of management and administrative science, operations research and industrial engineering, although estimating the performance of a complex stochastic system is of great value to the decision maker, it is not always sufficient. For example, a production control manager may be interested in finding out the probability that all demands are met from on-hand inventory under a certain system configuration of a fixed safety stock and a fixed order quantity. However, he might be more interested in finding out what values of safety stock and order quantity will maximize this probability. In this paper we propose two variants of Simulated Annealing (SA) algorithm to solve a special class of discrete stochastic optimization problems where the objective function can be represented as the probability involving a performance event of a stochastic system. Similar to the original SA algorithm, both variants have the hill climbing feature to escape the trap of local optima. The first variant selects the last state visited by the algorithm to be the estimate of the optimal solution. The second variant selects the most frequently visited state to estimate the optimal solution. Like the original SA algorithm, the first variant uses decreasing annealing temperature, while the second variant uses constant temperature. Computational results are given to demonstrate the performance of the proposed SA algorithms.

Submitted September 2005, accepted June 2006.

Introduction

Discrete event simulation is widely used in management and administrative science, planning communication networks, computer systems, production assembly lines, and other complex systems. In planning these systems, decisions must be made concerning configurations with which various probabilistic events occur within the system. Planners typically want to know how the system will perform under various configuration settings.

Developing efficient methods of optimizing stochastic production systems by simulation is not easy but is extremely important in practice. In problems where the objective function is not given as an explicit function of system parameters, as is the case in simulation, direct search methods are often applicable. However, applying these search techniques to optimize stochastic simulation production systems would result in a major difficulty. For the majority of applications of search techniques it is assumed that a given set of values for decision variables results in a unique value for the objective function. This is not the case for stochastic simulation optimization. Rather we may observe samples from the objective function for each set of parameter values. Thus, a single evaluation of the objective function in the latter case provides only one realization from that objective function.

Much of the literature on simulation output analysis concentrates on estimating the expected value of system response. However, stochastic simulation optimization on the basis of average system behavior alone can sometimes result in misleading conclusions. Therefore, we may consider measures of performance other than means. One such useful measure of performance is the probability that the system response is less than some value. Let X_i be a random variable defined on the jth replication for $j = 1,2,..., n$. Suppose that we would like to estimate the probability $\beta = P(X \in A)$, where $A \subset R$ is a set of real numbers. By way of example, for queuing system we might want to determine the probability that job waiting time in the system is less than or equal to user threshold value.

In this paper, our search problem is to find optimal points in S, where $S = \{1, 2, ..., s\}$ is a finite set of system configurations, with maximum value of β. We will assume that S has multiple optima and S^* is the set that contains them, i.e. $S^* = \{i : \beta_i = \max_{j \in S} \beta_j\}$. If there is a single point in S^*, denote it by i^* where $\beta_{i^*} > \beta_j \; \forall j \in S, j \neq i^*$. For-

mally, our optimization problem is to choose the best system, that is, the system with the largest value of β_i :

$$\beta_i = \max_{j \in S} \beta_j$$

$$= \max_{j \in S} P\,(\text{performance event} \mid \text{vert configuration } j).$$

Consider, for example, a general production line which can be modeled as open k-station queuing network with measure of performance being, say, the throughput. Here, the performance event may be defined as {throughput is greater than some critical value}. We might be interested to select the configuration with the largest value of β, where $\beta = P$ {throughput is greater than some critical value}.

We focus on the case where the objective function is evaluated through simulation. In such a situation, all the function evaluations will include noise, so conventional (deterministic) optimization methods cannot be used to solve this problem. Kushner (1963) first proposed a method for solving stochastic optimization problems, where the values of the objective function can only be sampled via Monte-Carlo method. He assumed that the sample objective functions are disturbed by a sequence of zero mean i.i.d. Gaussian random variables.

Alkhamis et al. (1999) presented a variant of the SA algorithm for discrete stochastic optimization problems. The basic idea of their modification is to make the comparison between solution point i and solution point j based on whether the objective function value indicates statistically significant difference at each iteration. The differences between objective function values are considered to be statistically significant based on confidence intervals associated with these values. They showed that under suitable conditions on the random noise, the modified annealing algorithm converges in probability to the set of optimal solutions.

Fox and Heine (1995) investigated the application of simulated annealing to solve discrete stochastic optimization problems. Gelfand and Mitter (1989) presented theoretical analysis for the SA algorithm when the objective function includes noise. They showed that under suitable conditions on the noise, the modified annealing algorithm exhibits the same convergence in probability to the globally optimal solution as the original annealing algorithm. Gutjahr and Pflug (1996) generalized the classical convergence result for the SA algorithm to the case where cost function observations are disturbed by random

noise. They showed that for a certain class of noise distributions, the convergence assertion remains valid, provided that the standard deviation of the noise is reduced in the successive steps of cost function evaluation with a speed $O(k^{-\gamma})$, where γ is an arbitrary constant larger than one. Roenko (1990) applied SA algorithm to a stochastic optimization problem. His approach, however, makes it necessary to store all feasible solutions encountered during the execution of the algorithm and to compare them with each newly generated solution. It seems that this approach is not realistic for practical applications due to the computational burden involved.

Ahmed *et al.* (1998), developed three search strategies for finding the optimal solution to a stochastic discrete simulation optimization problem. In each iteration of these strategies, two neighboring alternatives are compared and the one that appears to be better is passed on to the next iteration. They showed that these search procedures satisfy local balance equations and their equilibrium distributions give most weight to the optimal point. They also showed that the configuration that has been visited most often in the first m iterations converges almost surely to a globally optimum solution.

In this paper we present two approaches for solving problem (1). The first approach uses decreasing annealing temperature and selects the last state visited by the algorithm to be the estimate of the optimal solution. In the second approach we use constant temperature and the optimal solution is estimated by the state that is visited most often by the algorithm (Andradottir ,1995, 1996). The paper is organized as follows. In the next section, we set up the structure of our optimization problem. Then we present two approaches based on SA algorithm for solving our optimization problem. Then we present computational results and compare the performance of the proposed approaches. Finally, the last section contains some concluding remarks.

Problem Structure and Assumptions

In this work we are interested in solving a stochastic optimization problem involving a finite number of different configurations. Define Y_i to be the indicator function of performance event with configuration i, i.e.

$$Y_i = \begin{cases} 1 & \text{with probability } \beta_i \\ 0 & \text{with probability } 1-\beta_i \end{cases} \qquad 0 < \beta_i < 1.$$

Our goal is to find the solution point that has maximum probability β. Assume $S = \{1, 2, \ldots, s\}$ is a non-empty discrete finite set of configurations and the search is conducted by picking an initial point in S and then comparing a neighboring point according to the following definition.

Definition 2.1: For each $i \in S$ we define a neighborhood $N(i)$ as a subset of $S - \{i\}$, such that each point in $N(i)$ can be reached from i in a single transition.

Our search is organized in such a way that the next solution candidate is found among the neighbors of the present candidate. Hence, to ensure that the search will be able to reach any element of S, we make the following assumption about the system $\{N(i), i \in S\}$ of neighborhoods.

Assumption 2.1: For any pair $(i,j) \in S \times S$, j is reachable from i, i.e. there exists a finite sequence, $\{n_m\}_{m=0}^{\ell}$ for some ℓ, such that $i_{n_0}=i$, $i_{n_\ell}=j$ and $i_{n_{m+1}} \in N(i_{n_m})$, $m = 0, 1, 2, ..., \ell - 1$.

Now we impose a stochastic structure to the selection of a candidate among the neighbors by the following function G. Given $i \in S$, a candidate is selected among $N(i)$ such that the probability of selecting a neighbor $j \in N(i)$ is equal to G_{ij} which is defined as follows:

Definition 2.2: A function $G : S \times S \rightarrow [0,1]$ is said to be a generating probability function for S and N if

(1) $G_{ij} > 0 \Leftrightarrow j \in N(i)$ and

(2) $\sum_{j \in S} G_{ij}=1$ for all $i \in S$.

Here G_{ij} is the probability of generating solution point j as a candidate for the next solution point when the system is in solution point i. We will consider G_{ij} such that the probability is distributed uniformly over $N(i)$, i.e. $G_{ij} = \frac{1}{|N_i|}$, where $|N_i|$ is the cardinality of the set N_i .

At each iteration k, a sample of L_k observations is taken from i and j. A candidate neighbor $j \in N(i)$ is accepted and a move is performed from i to j if $\hat{\beta}_i < \hat{\beta}_j$ where $\hat{\beta}_i$ and $\hat{\beta}_j$ are such that

$$\hat{\beta}_i = \frac{\sum_{m=1}^{L_k} I_m^i}{L_k} \quad \text{where} \quad I_m^i = \begin{cases} 1 & \text{if performance event occurs given point } i \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Simulated Annealing Approachs

Simulated annealing (SA) algorithm was proposed originally by (Kirkpatrick et al.,1983) for solving complex deterministic optimization problems with discrete space. SA has shown successful applications in a wide range of combinatorial optimization problems and this fact has motivated researchers to use SA in simulation optimization. The SA methodology draws its analogy from the annealing process of solids. In the annealing process, a solid is heated to a high temperature and gradually cooled to allow it to crystallize. As the heating process allows the atoms to move randomly, if the cooling is done too rapidly, it gives the atoms enough time to align themselves in order to reach a minimum energy state. This analogy can be used in combinatorial optimization with the state of solid corresponding to the feasible solution, the energy at each state corresponding to the improvement in the objective function and the minimum energy state being the optimal solution. SA allows the search to proceed to a neighboring state even if the move causes the value of the objective function to become worse. To allow the search to escape a local optimum, at iteration k a move that increase the objective function value is accepted with a probability of $\exp(-(\Delta)/T_k$, where Δ is the differences in the objective function value and $T_k \succ 0$ is a sequence of positive real numbers $\{T_k, k = 0, 1, ...\}$ satisfying $T_k > 0$, $T_{k+1} < T_k \ \forall k$, and $lim_{k\to\infty} T_k = 0$. T_k is called the temperature at the kth iteration and the sequence $\{T_k, k = 0, 1, ...\}$is called the cooling schedule. The SA algorithm can be described as a sequence of Markov chain with the state space being the domain of the objective function to be optimized. Let X_k denote the state of the system visited by the SA algorithm at the kth step. Now we present our first SA approach for solving problem (1); this approach uses decreasing annealing temperature and selects the last state visited by the algorithm to be the estimate of the optimal solution. The steps of the first approach are as follows:

Algorithm 1.

Parameters: N, $\{T_k\}$, $\{L_k\}$.

Step 1. Obtain an initial solution $X_0 \in S$. Set $k = 0$. Obtain an initial temperature $T_k > 0$.

Step 2. Given $X_k = i$, choose a candidate $Z_k \in N(i)$ with probability distribution

$$P[Z_k = j \mid X_k = i] = G_{ij} = \frac{1}{|N_i|},\ j \in N(i).$$

Step 3. Given $Z_k = j$, generate two L_k independent observations from point i and j. Evaluate $\hat{\beta}_i$ and $\hat{\beta}_j$.

Step 4. Given $Z_k = j$, generate $U_k \sim U[0,1]$, and set

$$X_{k+1} = \begin{cases} Z_k & \text{if } U_k \leq exp\left[\frac{-[\hat{\beta}_i - \hat{\beta}_j]^+}{T_k}\right] \\ X_k & \text{otherwise} \end{cases}$$

(where for all $a \in R$, $a^+ = a$ if $a > 0$, and $a^+ = 0$ otherwise)

Step 3. Set $k = k + 1$. Update T_k and L_k and go to step 2.

Remark 1: The above algorithm converges to the set of global optimal solutions provided that the noise in the estimated objective function values in iteration k has the normal distribution with 0 mean and variance $\sigma_k^2 > 0$, where $\sigma_k = o(T_k)$ as $k \to \infty$ and $lim_{k \to \infty} T_k = 0$, see (Gelfand and Mitter 1989).

Remark 2: One of the following stopping criteria may be used for the above algorithm: Stop the search process after performing a predetermined number of iterations or when the current best configuration has not been changed for a predetermined number of iterations.

Remark 3: Our implementation of step 3 is as follows: at each iteration an L_k observations is obtained from points i and j and then the performance measures are calculated as in equation (2).

The stochastic random process $\{X_k\}$ produced by the SA algorithm above is a discrete-time inhomogeneous Markov chain defined over states S, and its state transition probability is given by

$$P_{ij}(k) = P[X_{k+1} = j | X_k = i]$$
$$= \begin{cases} G_{ij} \min[1, exp - (\hat{\beta}_i - \hat{\beta}_j)/T_k] & \text{if } j \in N(i), \\ 1 - \sum_{\ell \in N(i)} P_{i\ell}(T_k) & \text{if } i = j, \end{cases}$$

where G_{ij} denotes the generating probability, and $\min[1, \exp - (\hat{\beta}_i - \hat{\beta}_j)/T_k]$ denotes the acceptance probability, i.e. the probability of accepting state j, once it is generated from state i, and T_k denotes the temperature control parameter. If we fix the temperature T_k at T, i.e., $T_k = T$, for all k, then the Markov chain constructed by algorithm 1 is a time-homogeneous Markov chain with transition probability matrix P defined as follows:

$$P_{ij}(k) = P[X_{k+1} = j | X_k = i]$$
$$= \begin{cases} G_{ij} \min[1, exp - (\hat{\beta}_i - \hat{\beta}_j)/T] & \text{if } j \in N(i), \\ 1 - \sum_{\ell \in N(i)} P_{i\ell}(T) & \text{if } i = j, \end{cases}$$

Now we present our second SA approach for solving problem (1). Algorithm 1 discussed above uses decreasing temperature and selects the state that is visited by the algorithm in iteration k as the estimated optimal solution in that iteration. In the second approach we use a constant temperature and a different approach for estimating the optimal

solution. We select the most frequently visited state by the algorithm (divided by a normalizer) to be the estimate of the optimal solution. Let for all $i \in S$ and $k \geq 0$, V_k(i) is the number of times that the Markov chain $\{X_k, k = 0, 1, ...\}$ has visited state i in the first k iterations, and X_k^* is the state that the search process has visited most often after k iterations. Letting $\tilde{V}_k(i) = \frac{V_k(i)}{|N(i)|}$, then $X_k^* = \{i : \tilde{V}_k(i) \geq \underset{j \in S}{Max}\, \tilde{V}_k(j)\}$. It is clear that the value of X_k^*, say j changes only when the search process visits point i and $\tilde{V}_k(i) > \tilde{V}_k(j)$. Before we state the algorithm we need the following assumptions.

Assumption 3.1 Let $\{L_k\}$ be a sequence of positive integers such that $\lim_{k \to \infty} L_k = \infty$.

Assumption 3.2 The temperature T is a positive (constant) real number.

Algorithm 2.

Parameters: N, T, $\{L_k\}$

Step 1. Select a starting point $i_0 \in S$. Let $V_0(i_0) = 1$ and $V_0(j) = 0$ for all $j \in S, j \neq i_0$. Let $k = 0$ and $X_k^* = i_0$.

Step 2. Given $X_k = i$, choose a candidate $Z_k \in N(i)$ with probability distribution $P[Z_k = j | X_k = i] = G_{ij}, j \in N(i)$.

Step 3. Given $Z_k = j$, generate two L_k independent observations $Y_i^1, Y_i^2, \ldots, Y_i^{L_k}$ and $Y_j^1, Y_j^2, \ldots, Y_j^{L_k}$. Evaluate $\hat{\beta}_i$ and $\hat{\beta}_j$.

Step 4. Given $Z_k = j$, generate $U_k \sim U[0, 1]$, and set

$$X_{k+1} = \begin{cases} Z_k & \text{if } U_k \leq exp\left[\frac{-[\hat{\beta}_i - \hat{\beta}_j]^+}{T}\right] \\ X_k & \text{otherwise} \end{cases}$$

(where for all $a \in R$, $a^+ = a$ if $a > 0$, and $a^+ = 0$ otherwise)

Step 3. Set $k = k + 1$, $V_k(X_k) = V_{k-1}(X_k) + 1$ and $V_k(j) = V_{k-1}(j)$, for all $j \in S$ and $j \neq X_k$ if $\frac{V_k(X_k)}{|N(X_k)|} > \frac{V_k(X_{k-1}^*)}{|N(X_{k-1}^*)|}$ then let $X_k^* = X_k$; otherwise let $X_k^* = X_{k-1}^*$. Update L_k, go to step 2.

Remark 4: Alrefaei and Andradottir (1999) have shown that the above algorithm converges to the set of global optimal solutions.

Remark 5: One of the following stopping criteria may be used for the above algorithm: Stop the search process after performing a predetermined number of iterations or when the current best configuration has not been changed for a predetermined number of iterations.

Computational Results

In this section, we present two test cases of two different discrete simulation optimization problems. In the first case we apply the two proposed algorithms to solve a discrete stochastic optimization problem with 50 configurations. In the second case we consider the optimi-

zation of a queuing system , similar to the one presented in (Alrefaei and Andradottir ,1999) and compare the performance of the proposed algorithms.

Test case 1.

In this case we implement each of the two SA algorithms to solve a discrete stochastic optimization problem with 50 configurations. Consider the following optimization problem. Find points in S with maximum value of β:

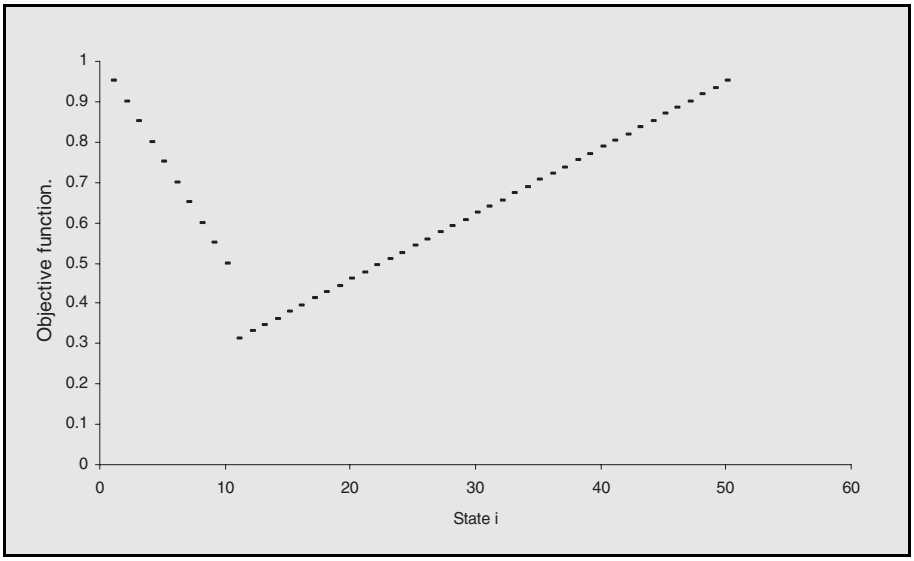
$$\max_{i \in S} \beta_i$$

where $S = \{0, 1, ..., 49\}$ and

$$\beta_i = \begin{cases} 0.95 - 0.05(i) & for\ i = 0, 1, ..., 9 \\ 0.15 + (0.8/49)(i) & for\ i = 10, 11, ..., 49 \end{cases}$$

See Figure 1 for the graph of the function β_i. Note that we have two global maxima at $i = 0$ and at $i = 49$ with values equal to 0.95. As an example we may consider β to be the probability that the throughput of a production line is less than some prespecified value. We apply Algorithm 1 to solve this optimization problem using different annealing sequences $\{T(k)\}$ and different neighborhood structures $\{N(x) : x \in S\}$, to see their effect on the performance of the algorithm. In particular, we consider two annealing sequences $\{T(k)\}$, the first annealing sequence is given by $T_1(k) = 0.1/\ln(10 + k)$ which is slower than the second annealing sequence given by $T_2(k) = 1/\ln(10 + k)$.

Figure 1
Graph of the objective function values of test case 1



Also we use two different neighborhood structures $\{N(x) : x \in S\}$. The first neighborhood structure is given by $N1(i) = \{j \in S : |j - i| \leq 5\}$ which is smaller than the second neighborhood structure given by $N2(i) = \{j \in S : |j - i| \leq 10\}$.

Figure 2 shows the performance of Algorithm 1 as a function of the annealing sequence $\{T(k)\}$ when it is applied to solve the above problem using neighborhood structure *N1*. The x-axis shows the number of iterations that were used in our simulation, while the y-axis shows the average estimated optimal objective value at the estimated optimal solution over 100 replications. It is clear from this figure that choosing $T_1(k) = 0.1/\ln(10 + k)$ gives better performance than $T_2(k) = 1/\ln(10 + k)$. Figure 3 presents the performance of Algorithm 1 as a function of the annealing sequence $\{T(k)\}$ when it is applied to solve the above problem using neighborhood structure *N2*. As in Figure 2, Figure 3 indicates that choosing slower annealing sequence $T_1(k)$ gives better performance than $T_2(k)$.

To test the performance of Algorithm 2 we apply Algorithm 2 to solve this optimization problem using several choices of the parameter T and different neighborhood structures $\{N(x) : x \in S\}$, to see their effect on the performance of the algorithm. In particular, we take $T \in \{0.01, 0.1, 1\}$. Figure 4 shows the performance of Algorithm 2 as a function of the parameter T when it is applied to solve the above problem using neighborhood structure $N1$. It is clear from this figure that small values of T give better performance than larger values. Figure 5 presents the performance of Algorithm 2 as a function of the parameter T when it is applied to solve the above problem using neighborhood structure $N2$. As in Figure 4, Figure 5 indicates that choosing small values for T gives better performance than larger values of T. Also, Figures 4 and 5 indicate that choosing larger neighborhood structure gives better performance than small size neighborhood structure.

Table 1 compares the performance of Algorithm 1 and Algorithm 2 when they are applied to solve the above optimization problem using the following choices of the parameter T, the annealing sequence $\{T_k\}$ (for Algorithm 1), and the sequence $\{L_k\} : T = 0.1$ for Algorithm 2, and for all $k \in N$, $T_k = 0.1/ln(10 + k)$ for Algorithm 1, $L_k = 10 + \lfloor \ln(k) \rfloor$ for Algorithm 1 and Algorithm 2. As we mentioned earlier, Algorithm 1 uses a decreasing annealing schedule $\{T_k\}$ while Algorithm 2 uses

Figure 2
Optimization trajectories of test case 1 using Algorithm 1 and N1 for different annealing sequences

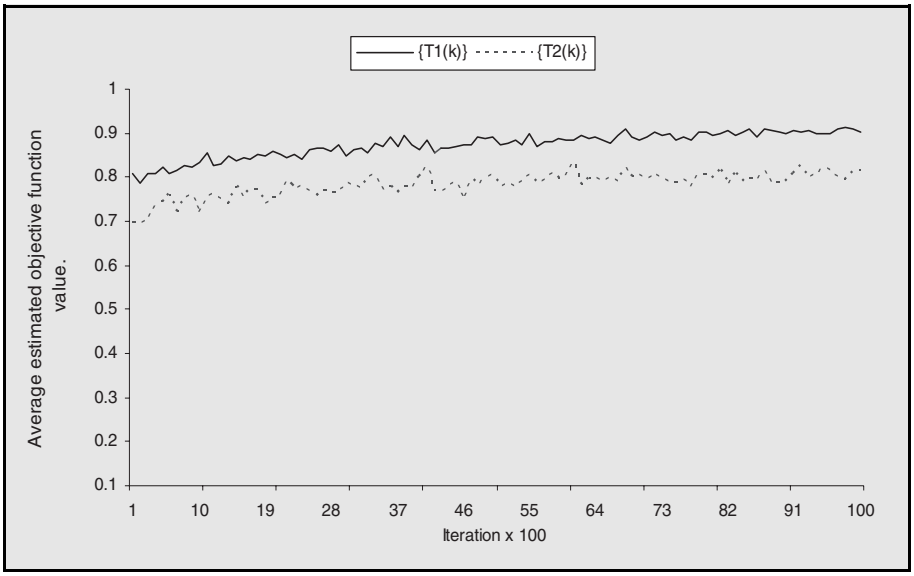


Figure 3
Optimization trajectories of test case 1 using Algorithm 1 and N2 for different annealing sequences

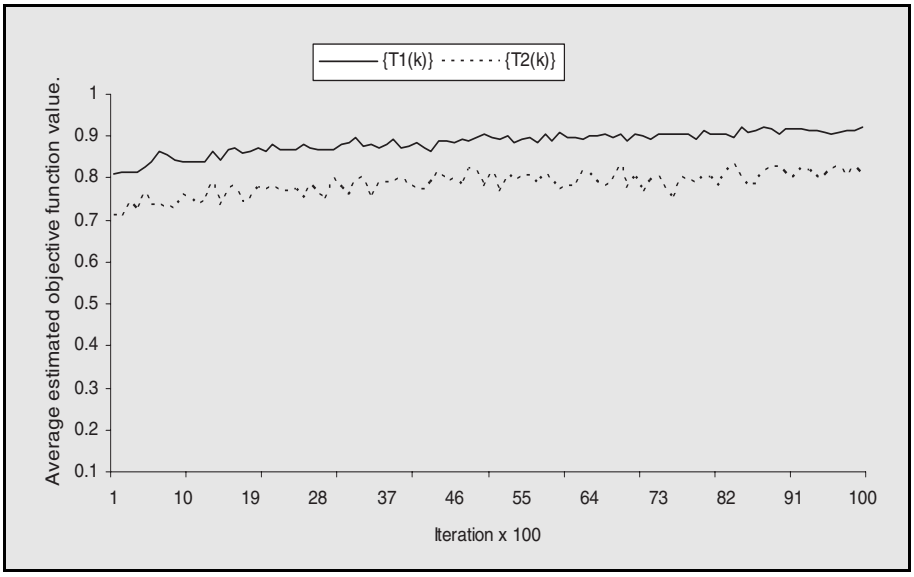


Figure 4
Optimization trajectories of test case 1 using Algorithm 2 and N1 for different T

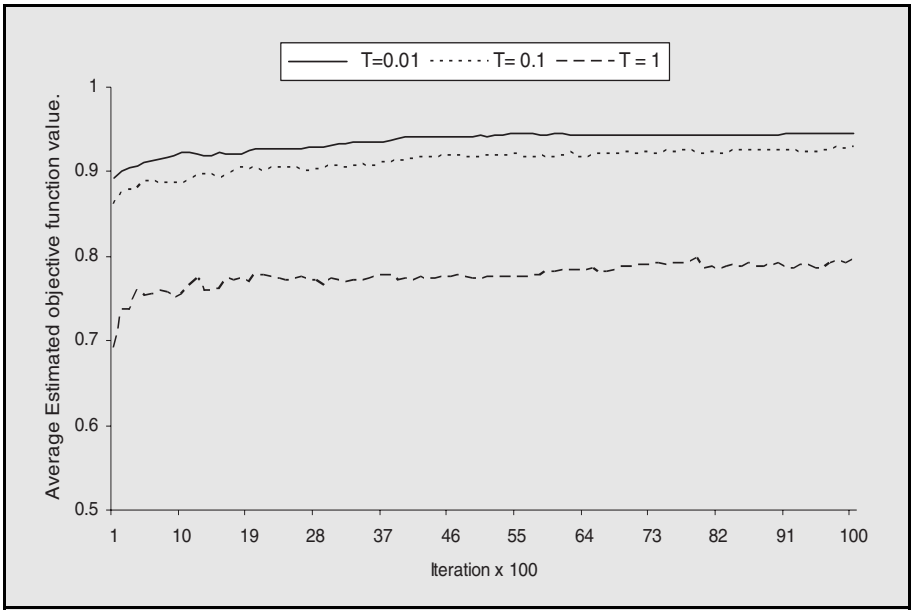
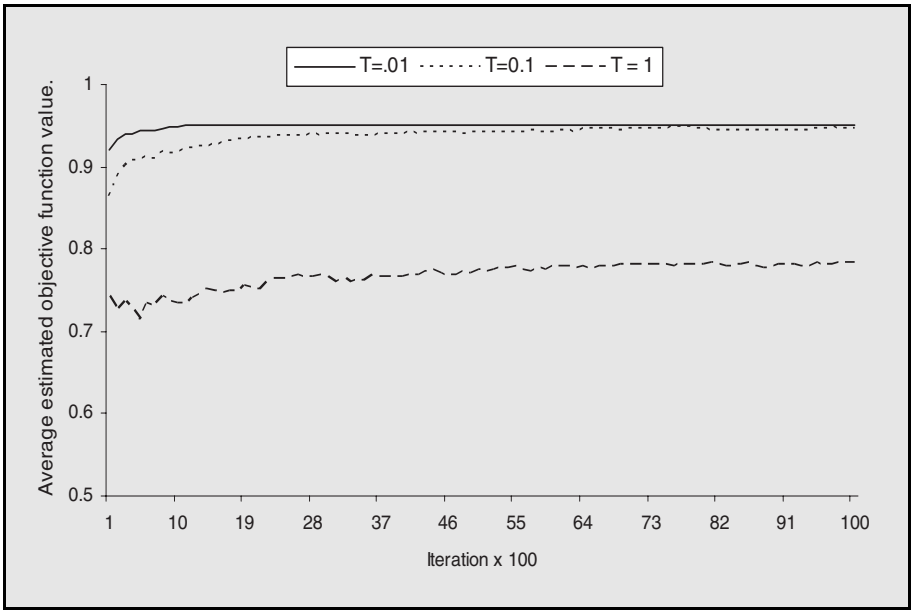


Figure 5
Optimization trajectories of test case 1 using Algorithm 2 and N2 for different T



fixed temperature T. Table 1 presents the number of convergent paths these algorithms have done so far out of a total of 100 replications (i.e., the number of replications in which the estimated optimal solution equals the (true) global optimal solution) using neighborhood structures $N1$ and $N2$. For both neighborhoods Algorithm 2 converges more rapidly to the optimal global solution than Algorithm 1. Using neighborhood structure $N1$, 92 replications have converged after 10000 iterations when Algorithm 2 is used, compared to 18 replications for Algorithm 1. For the same parameters setting and using $N2$, 99 replications have converged after 10000 iterations when Algorithm 2 is used, compared to 23 replications for Algorithm 1. Table 1 indicates that Algorithm 2 is more efficient than Algorithm 1 when solving the above discrete optimization problem using the parameter values given previously.

Test Case 2.

Many practical production planning problems can be modeled as queuing systems. Our next test example involve the optimization of M/M/1 queuing system similar to the one presented in (Alrefaei and Andradóttir, 1999). Consider an M/M/1 queuing system with job arrival rate λ and service rate μ. For

Table 1
A comparison of the performance of Algorithm 1 and Algorithm 2 for test case 1 with different neighborhood structures

	Number of convergent paths.			
	Algorithm 1		Algorithm 2	
Iteration #	N1	N2	N1	N2
100	5	8	32	37
500	5	12	49	62
1000	5	12	59	74
2000	11	15	68	91
3000	11	17	74	95
4000	12	17	81	96
5000	14	22	84	97
6000	15	22	85	99
7000	15	22	86	99
8000	15	23	89	98
9000	18	23	92	99
10000	18	23	92	99

$i \in \mathbb{N}$, let W_i be the time in the system for job i, B_i be the service time of job i, and A_i be the inter-arrival time between job $i-1$ and i, where $A_0 = 0$ and $W_0 = 0$. To generate the system time of job c, W_c, one can generate B_i, A_i, for $i = 1, \ldots, c$ and then use the following well known recursive formula:

$$W_i = max\{B_i, W_{i-1} + B_i - A_i\}$$

to obtain W_i for $i = 1, ..., c$. We are interested in maximizing the event that the waiting time in the system for job i does not exceed threshold value (δ) to be set by the user. Then we are interested in solving the following optimization problem:

$$\max_{x \in S} P\{W(x) \leq \delta\}$$

where $S = \{1, ..., 50\}$ and $W(x)$ is the average time a job spends in the system for fixed arrival rate $\lambda = 1$ and service rate $\mu(x) > 1$ for all $x \in S$, where the values of $\mu(x)$ for all $x \in S$, are given in Table 2.

We simulate the above system under transient state conditions and estimate the probability that the average waiting time of the first 100 jobs is less than $\delta = 1.1$. We apply Algorithm 1 and Algorithm 2 to solve this optimization problem using the following choices of the parameter T, the annealing sequence $\{T_k\}$ and the sequence $\{L_k\}$. $T = 0.01$, and for all k, $T_k = 0.1/\ln(10+k)$, $L_k = 10 + \lfloor \ln(10+k) \rfloor$ for Algorithm 1 and Algorithm 2. We use two neighborhood structures. The first neighborhood structure is given by:

$$N1(i) = \begin{cases} \{2\} & \text{if } i = 1, \\ \{49\} & \text{if } i = 50, \\ \{i-1, i+1\} & \text{otherwise.} \end{cases}$$

The second neighborhood structure is given by:

$$N2(i) = \{j \in S : |j - i| \leq 3\}.$$

Note that in the transient setting, the estimated optimal objective function value is 0.756 which occurs at $x = 28$. We select the initial state x_0, randomly (uniformly) over S and run 100 replications to estimate the average performance of Algorithm 1 and Algorithm 2.

Table 2
The values of the service rates $\mu(x)$ for all $x \in S$.

$\mu(1), ..., \mu(10)$	1.65	1.6	1.5	1.6	1.7	1.75	1.65	1.6	1.55	1.5
$\mu(11), ..., \mu(20)$	1.47	1.45	1.5	1.55	1.6	1.65	1.6	1.55	1.5	1.47
$\mu(21), ..., \mu(30)$	1.45	1.5	1.55	1.6	1.65	1.7	1.75	2.0	1.7	1.6
$\mu(31), ..., \mu(40)$	1.55	1.5	1.47	1.5	1.6	1.65	1.7	1.75	1.65	1.6
$\mu(41), ..., \mu(50)$	1.55	1.5	1.47	1.5	1.6	1.65	1.7	1.6	1.5	1.45

Figures 6 and 7 compare the performance of Algorithm 1 and Algorithm 2 when applied to solve this optimization problem with the choice of parameters described above using neighborhood structure N1 and N2,

Figure 6
Optimization trajectories of Algorithm 1 and 2 for the M/M/1 test example using neighborhood structure N1

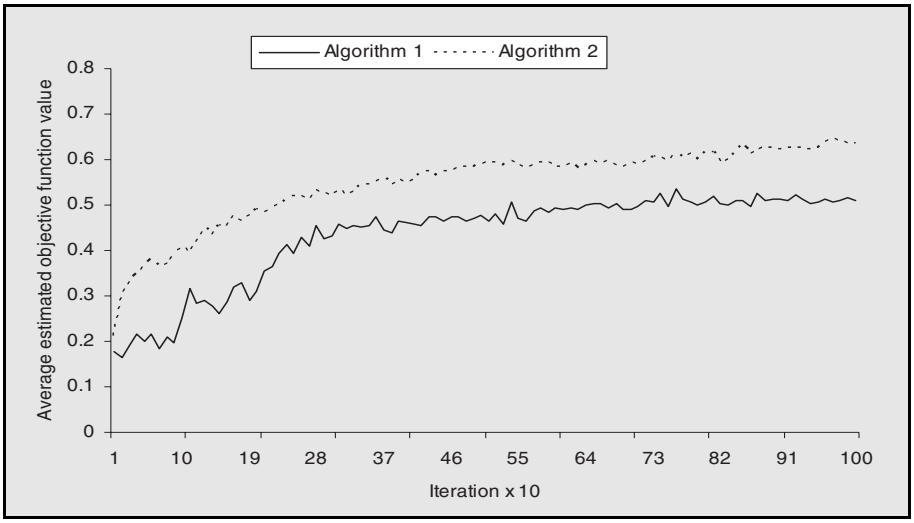
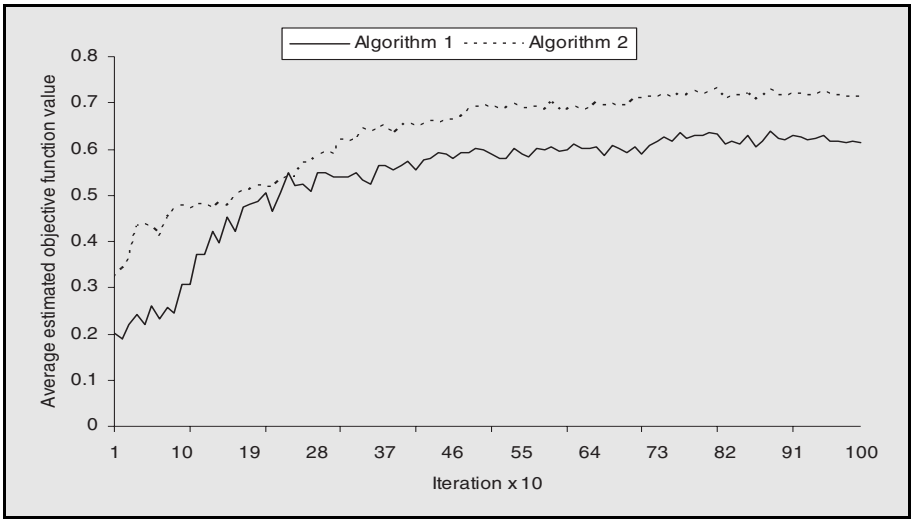


Figure 7
Optimization trajectories of Algorithm 1 and 2 for the M/M/1 test example using neighborhood structure N2



respectively. The x-axis shows the iteration number, while the y-axis shows the average estimated optimal objective value at the estimated optimal solution over the 100 replications. Figures 6 and 7 show that Algorithm 2 converges faster than Algorithm 1. This implies that Algorithm 2 performs more efficiently than Algorithm 1 for this problem setting. Also, Figure 6 and 7 confirm the fact that larger neighborhood structure always produces better results than small neighborhood structure.

Conclusion

In this paper, we have proposed two variants of Simulated Annealing algorithm for solving a special class of discrete stochastic optimization problems where the objective function can be represented as the probability involving a performance event of a stochastic system and can be evaluated only through Monte Carlo simulation. These variants are important when either the objective function cannot be computed exactly or such an evaluation is computationally expensive. Similar to the original SA algorithm, both variants have the hill climbing feature to escape the trap of local optima. While the first variant selects the last state visited by the algorithm to be the estimate of the optimal solution, the second variant selects the most frequently visited state to be the estimate of the optimal solution. Like the original SA algorithm, the first variant uses decreasing annealing temperature, while the second variant uses constant temperature. Computational results indicate that Algorithm 2 performances are better than Algorithm 1 for the test examples presented in this paper.

References

Alkhamis, T. M., M. A. Ahmed, V. K. Tuan. 1999. Simulated Annealing for Discrete Optimization with Estimation. *European Journal of Operational Research*, 116: 530-544.

Ahmed, M. A, Alkhamis, T. M. and D.R. Miller. 1998. Discrete Search Methods for Optimizing Stochastic Systems. *Computers and Industrial Engineering*, 34 (4): 703 - 716.

Alrefaei, M. H., S. Andradottir. 1999. A simulated Annealing Algorithm with Constant Temperature for Discrete Stochastic Optimization. *Management Science*, 45 (5): 748-764.

Andradottir, S. 1995. A Method for Discrete Stochastic Optimization. *Management Science,* 41 (12): 1946 - 1961.

Andradottir, S. 1996. ''Global Search for Discrete Stochastic Optimization'', *SIAM J.* Optimization, 6 (2): 513 - 530.

Fox, B. L., and G.W. Heine. 1995. ''Probabilistic Search with Overrides.'' *The Annals of Applied Probability*, 1995, 5: 1087-1094.

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

Gutjahr, W. J. and G.C. Pflug. 1996. ''Simulated Annealing for Noisy Cost Functions'', *Journal of Global Optimization,* **8**: 1-13.

Kirkpatrick, S., C.D. Gelatt, Jr. and M.P. Vecchi. 1983. Optimization by Simulated Annealing, *Science,* **221**: 671-680.

Kushner, H. J. 1963. Hill Climbing Methods for the Optimization of Multi-parameter Noise Disturbed Systems. Trans. ASME, *J. Basic Engineering*, 85: 157-164.

Roenko N, 1990. *Simulated Annealing under Uncertainty*. Technical Report, Inst. F. Operations Research, Univ. Zurich.

الملخص

الخوارزميات البطيئة لإيجاد الحل الأمثل لنوعية خاصة من المسائل العشوائية المتقطعة

طلال ماضي الخميس
جامعة الكويت

في حقل العلوم الإدارية، وبحوث العمليات، والهندسة الصناعية لايكون كافياً تقدير الأداء وحسابه في الأنظمة العشوائية المعقدة. فمثلا: مسؤول مراقبة الإنتاج يود معرفة احتمال تغطية الطلب على سلعة في ظل مستوى من المخزون و مستوى ثابت من الطلب. ولكن سيكون له اهتمام أكثر بمعرفة المستوى الأمثل لقيم المخزون وقيم الطلب، تلك التي من شأنها أن تعظم قيمة الاحتمال. في هذه الدراسة تم تقديم صيغتين لخوارزميات الأنظمة البطيئة لإيجاد الحل الأمثل لنوعيه خاصة من المسائل العشوائية المتقطعة، مثل النظام الخوارزمي الأساسي؛ حيث إن المقترحين يستخدمان ظاهرة الانتقال لحل غير مثالي في مراحل البحث. المقترح الأول يختار آخر مرحلة يقف عندها الخوارزمي لتكون هي نقطة الحل الأمثل، بينما يختار المقترح الثاني أكثر نقطة تمت زيارتها من قبل الخوارزمي لتكون هي نقطة الحل الأمثل. وقد تم عرض نتائج حسابية لتطبيق المقترحين.

Talal M. Alkhamis (Ph.D. in Operations Research from the Florida Institute of Technology, 1989, USA). Associate professor in the Department of Statistics and Operations Research at Kuwait University. His research interests include discrete stochastic simulation optimization and meta-heuristics optimization.