AIAA JOURNAL
Vol. 38, No. 10, October 2000

Improved Simulated Annealing Search for Structural Optimization

Jau-Sung Moh* and Dar-Yun Chiang†
National Cheng-Kung University, Tainan 701, Taiwan, Republic of China

An optimization algorithm based on simulated annealing is proposed, in which the domain of the search is successively reduced based on a probability concept until the stopping criteria are satisfied. By introducing the ideas of probability cumulative distribution function and stable energy, the selection of initial temperature and equilibrium criterion in the process of simulated annealing becomes easy and effective. Numerical studies using a set of standard test functions and an example of a 10-bar truss show that the approach is effective and robust in solving both functional and structural optimization problems.

I. Introduction

THE achievement of optimum design is a goal naturally attractive to the designer. The field of structural design is generally characterized by a large number of variables, which are usually discrete in dimensions or properties, and these variables should satisfy all of the constraints to be a feasible structural design. Some structural optimization problems are fairly amenable to a mathematical programming approach. In such instances, the design space is continuous and convex. The search may be deterministic, and methods are employed using, for example, gradient concepts. However, there is a large class of structural optimization problems with nonconvexities in the design space and with a mix of continuous and discrete variables. Under such circumstances, standard mathematical programming techniques are usually inefficient because they are computationally expensive and are almost assured of locating the relative optimum close to the starting design. To overcome these difficulties, the stochastic search in structural optimization is considered. Many methods have become possible with the powerful computing facilities available in recent years. Among the stochastic algorithms, pure random search[1] is the simplest strategy for optimal design. Some modified versions have been suggested, such as single start, multistart, and random directions.[2,3] Methods in this class generally are quite simple to implement, but the appropriate stopping rules are very difficult to derive.

Recently, two classes of powerful search methods, which have their philosophical basis in processes found in nature, have been widely used in structural optimization. The first class of methods, including genetic algorithms[4] and simulated evolution,[5] is based on the spirit of Darwinian theory of evolution. The second class of methods is generally referred to as simulated annealing techniques[6] because they are qualitatively derived from a simulation of the behavior of particles in thermal equilibrium at a given temperature. Because of their global capabilities, research on the utilization of these search methods in design optimization has been undertaken.[7–11] Some hybrid techniques have also been developed by combining features of these two algorithms, such as using a genetic algorithm to determine better annealing schedules[12] and introducing a Boltzmann-type mutation or selection process into simulated evolution.[13,14]

In this paper, we propose a method based on simulated annealing that searches from a population as in the method of simulated evolution instead of from a single point. The algorithm is called the region-reduction simulated annealing (RRSA) method because it locates the optimum by successively eliminating the regions with low probability of containing the optimum.

A brief review of basic simulated annealing is given in Sec. II, which is helpful for development and explanation of the proposed algorithm. In Secs. III and IV, we describe the concept and procedure of RRSA and test it in functional optimization, respectively. Application of RRSA in structural optimization and the results of numerical experiments are presented in Sec. V, followed by concluding remarks.

Received 11 June 1999; revision received 30 December 1999; accepted for publication 22 February 2000.

*Ph.D. Candidate, Institute of Aeronautics and Astronautics.
†Associate Professor, Institute of Aeronautics and Astronautics.

II. Method of Simulated Annealing

The ideas of simulated annealing (SA) are derived from the principles of statistical thermodynamics. Consider the atoms of a molten metal at some elevated equilibrium temperature T. The state of this system ϕ is characterized by specific spatial locations of the atoms. The probability $p_T(\phi)$ of the event that, at the given equilibrium temperature T, the system is in a state ϕ is given by the Boltzmann distribution[15,16]

$$p_T(\phi) = \exp\left[\frac{-E(\phi)}{kT}\right] \Big/ \sum_{\Phi} \exp\left[\frac{-E(\phi)}{kT}\right] \tag{1}$$

where k is the Boltzmann constant, $E(\phi)$ is the energy of the state ϕ, and Φ are all possible states that the system can assume at that temperature. At a given temperature, random variations of the system state are considered. If a change of state results in a lower energy level, it is immediately accepted. If, however, a higher energy state results from the variation, it is only accepted with a probability p given by

$$p = \frac{p_T(\phi_2)}{p_T(\phi_1)} = \exp\left\{-\frac{[E(\phi_2) - E(\phi_1)]}{kT}\right\} \tag{2}$$

where $p_T(\phi_1)$ and $p_T(\phi_2)$ are probabilities of states ϕ_1 and ϕ_2, respectively, as obtained from the Boltzmann distribution. Equation (2) implies that, even at low temperature, there is still a chance, though very small, for a system to jump into a state with higher energy. This mechanism provides the system with the chance of getting out of a local energy minimum and finding a state with even lower energy. If an annealing schedule is established in which the temperature gets lowered gradually and thermal equilibrium is reached at each temperature, a pure crystalline state corresponding to the lowest energy level is attainable.

SA is an approach simulating the process of annealing for optimization. It takes the objective function of an optimization problem as the energy corresponding to a given state, and the candidates (vectors of design variables) in the search space are treated as the possible states for an equilibrium temperature, which is a control parameter in the process. The general scheme of SA can be stated as follows:

1) Generate randomly a candidate $\boldsymbol{x}$.

2) Choose $T > 0$ be the initial temperature.

3) If a stopping criterion is satisfied, then stop; otherwise repeat the following steps:

a) If equilibrium is reached, then exit this loop.

b) Let $\boldsymbol{x}'$ be a randomly selected neighbor of $\boldsymbol{x}$.

c) Generate a uniform random number U in [0, 1].

d) If $\exp\{-[f(\boldsymbol{x}') - f(\boldsymbol{x})]/T\} > U$, then $\boldsymbol{x} = \boldsymbol{x}'$.

4) Let T be a new (lower) temperature value; go to step 3.

Several decisions have to be made to implement the conceptual algorithm just described, including the following:

1) Choose an initial temperature and the corresponding temperature decrement strategy: A large value of initial control parameter or a slow decrement rule will cause the algorithm to be unacceptably slow; a small initial value or a fast temperature decrement will tend to exclude the possibility of ascent steps, thus losing the global optimization capability of the method.

2) Choose an adequate stopping criterion: It is the crucial and most difficult part of the algorithm and has great influence on the overall performance.

3) Choose a criterion for detecting equilibrium: Ideally, for each value of the control parameter T, the inner loop (steps 3a–3d in the preceding algorithm) should be executed indefinitely to let the system reach equilibrium. A good stopping criterion can save the time of computation without losing the ability of escaping from a local minimum.

III. Concept and Procedures of RRSA

Consider an unconstrained function optimization problem in which the objective function to be minimized is given by

$$f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) \tag{3}$$

Some ideas for improving the approach of SA in solving unconstrained optimization problems will be elucidated in this section. A major disadvantage of SA, as compared with other stochastic methods, such as simulated evolution or genetic algorithms, is that it searches only from a single point, instead of from a population of points. Thus, the benefit of exchanging of notions such as crossover in the genetic algorithms is not available. To improve the performance of SA, we propose that the method searches from a population of points. It cannot only facilitate the search for the optimum through the information of population, but also provides us with a way to judge whether the search should be terminated or not. That is the major advantage of having a multiple-point search in contrast to the multistart of one-point SA.

To have a fast rate of convergence and to obtain an appropriate stopping criterion, we introduce the concept of eliminating the regions of search that have low probability of containing the global optimum. To identify whether a region is to be eliminated, the probability cumulative distribution function for the objective function values (energy levels) is needed, as will be described. We will also discuss the important issues such as neighborhood structure for generating new candidates, the equilibrium criterion for the rule of decreasing temperature, the exchanging of notions and retaining the best candidate for fast convergence, and the reliability and stopping criterion for terminating the process. If it is a problem with multiple global optima, a strategy based on simulated evolution is developed within RRSA. The concepts and procedures are described in detail as follows.

We define the probability cumulative distribution function for the objective function values as

$$P(f_r) = \text{prob}[f(\boldsymbol{x}) \leqq f_r]$$

where f_r is a reference value and $\boldsymbol{x}$ is arbitrarily selected within the region of interest. Without loss of clarity, $P(f_r)$ is more conveniently denoted as $P(f)$.

Suppose that $P(f)$ is known in a problem; then for any current state $\boldsymbol{x}$, the probability of getting a new, randomly selected state with a lower value of f can be obtained. Although the true $P(f)$ is usually unknown for an engineering optimization problem, we can use a statistical method through sampling and grouping to get the cumulative distribution function (CDF) of the grouped samples. Using the method of curve fitting, we can then obtain an approximate function $C(\bar{f})$ for further calculation. The procedure for finding $C(\bar{f})$ is described in the following algorithm:

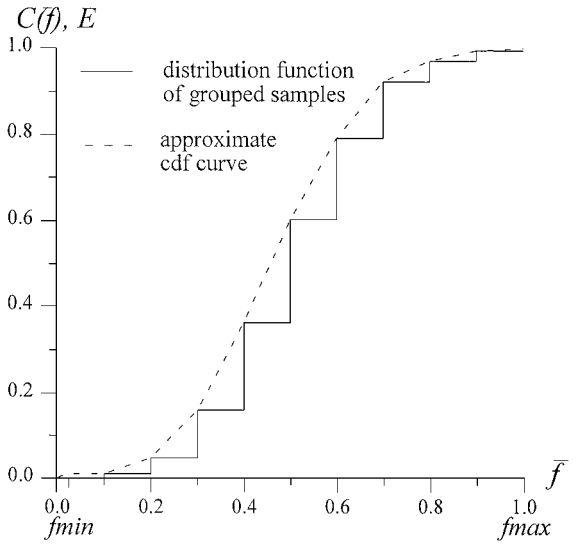

Fig. 1 Example of obtaining approximate probability distribution function $C(\bar{f})$.

1) Generate N candidates $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ randomly in the search space and calculate the corresponding function values $f_1, f_2, \ldots, f_N$.

2) Among the function values, let the minimum be $f_{\min}$ and the maximum be $f_{\max}$.

3) Normalize the function value f_k into $\overline{f_k}$ so that $\overline{f_k}$ lies in [1/N, 1]:

$$\overline{f_k} = \frac{f_k - f_{\min} + (1/N)(f_{\max} - f_k)}{f_{\max} - f_{\min}}$$

4) Take the normalized function values as the samples.

a) Subdivide the interval $I = [0, 1]$ into class intervals I_c.

b) Compute the class frequency N_c, which is the number of samples whose values lie in the class interval I_c.

c) Divide N_c by the sample size N to give the relative class frequency.

d) Compute the cumulative relative class frequency to obtain the distribution function of the grouped samples.

5) Use the method of curve fitting to obtain $C(\bar{f})$.

An example is given in Fig. 1, where the fitted curve can be simply chosen as polynomials or other orthogonal functions.

In conventional SA algorithms, the function value f instead of energy E is usually used to calculate the accepted probability as defined by Eq. (2). Because the range of function values is different in each problem, it is difficult to determine an appropriate value for the initial temperature and for the decrement of temperature in a particular problem. Observing that $C(\bar{f})$ always lies in [0, 1] for any problem gives us an idea that we may treat $C(\bar{f})$, rather than f itself, as the energy E of a state. In this way, the CDF $C(\bar{f})$ serves two purposes: as a probability distribution function for calculating reliability and for reducing search space and as the energy level of a state for calculating the accepted probability.

The RRSA search procedure starts from a population of points, so that we need to choose N_p initial candidates. Normalize the search space into [0, 1] in each and every dimension. Performing SA once for each of all candidates in the population is called a generation. If the criterion of system equilibrium (to be discussed in Sec. III.A) is satisfied after several generations, we then decrease the temperature level kT. Because all of the candidates concentrate into a small region and the convergent condition (cf. Sec. III.B) is satisfied after several temperature levels, we say that it is convergent in this cycle. Then a new cycle starts in the reduced and renormalized search space. It is remarkable that the CDF must be calculated at the beginning of each new cycle because the CDF calculated by sampling from the previous region will not be effective around values close to zero due to the limited size of samples used. These procedures will be performed repeatedly until the stopping criteria (cf. Sec. III.C) are satisfied. In the following, we discuss the important issues involved in the procedures of RRSA.

A. Equilibrium Criterion

It is important but difficult to find a criterion for detecting system equilibrium in the process of SA. We introduce the concept of stable energy to derive the equilibrium criterion for decreasing temperature level.

If the energy level E is continuously distributed in [0, 1], then Eq. (1) becomes

$$p_T(E) = \exp(-E/kT)/C \tag{4}$$

where C is a normalization constant. To satisfy the normalization condition

$$\int_0^1 p_T(E)\,\mathrm{d}E = 1 \tag{5}$$

one can derive that

$$C = kT[1 - \exp(-1/kT)] \tag{6}$$

The probability CDF of E will be

$$P(E) = \text{prob}(0 < X \le E) = \int_0^E \frac{\exp(-\xi/kT)}{kT[1-\exp(-1/kT)]}\,\mathrm{d}\xi = \frac{1-\exp(-E/kT)}{1-\exp(-1/kT)} \tag{7}$$

For a current state $\boldsymbol{x}$ with corresponding energy E, the expectation of possible energy levels E' associated with the next state can be calculated as follows:

$$E' = \int_0^E \xi p(\xi)\,\mathrm{d}\xi + \int_E^1 p(\xi)\left\{\xi \exp\left[-\frac{(\xi - E)}{kT}\right]\right\}\mathrm{d}\xi + \int_E^1 E\left\{1 - \exp\left[-\frac{(\xi - E)}{kT}\right]\right\}\mathrm{d}\xi \tag{8}$$

where, as from Eqs. (4) and (6),

$$p(\xi) = \frac{\exp(-\xi/kT)}{kT[1-\exp(-1/kT)]}$$

The first term in Eq. (8) signifies the case that the energy level of the new state is lower than that of the current state and that the new state is certainly accepted to replace the current state. The second term indicates the case that the new state has higher energy level but is still accepted. The third term signifies the case of rejecting the new state. Equation. (8) can be evaluated to obtain

$$E' = 1/[1-\exp(-1/kT)]\Big(kT\left\{1 - \tfrac{3}{4}\exp(-E/kT) - \tfrac{1}{4}\exp[-(2-E)/kT]\right\} + \left\{\tfrac{1}{2}E\exp[-(2-E)/kT] - \tfrac{1}{2}\exp[-(2-E)/kT] - E\exp(-1/kT)\right\}\Big) \tag{9}$$

Using Eq. (9), we can calculate the energy expectation E' for any given kT and E. A diagram of E' vs E at different values of kT is shown in Fig. 2. It is clear that the curves above the line of $E' = E$ correspond to the states for which the energy expectation of the new state will be higher than the energy level of current state. Obviously, if a state is in that condition, the potential of further decreasing energy may become poor.

If we obtain the energy expectation E' from a state energy E at a certain kT and repeatedly treat E' as a new E to calculate the next E', then we can obtain the diagram of E' vs the number of repeated calculations N, as shown in Fig. 3. It can be seen that, even if we select the highest energy level $E = 1$ or the lowest energy $E = 0$ as the initial value, the energy expectation E' will approach the same constant value for a given kT. This constant value is given by the intersection point of the line $E' = E$ and the $E' - E$ curve corresponding a given kT in Fig. 2. This value is the energy level that the system achieves equilibrium for a given kT. Thus, we may define a stable energy E_S that satisfies $E' = E$, and from Eq. (9),

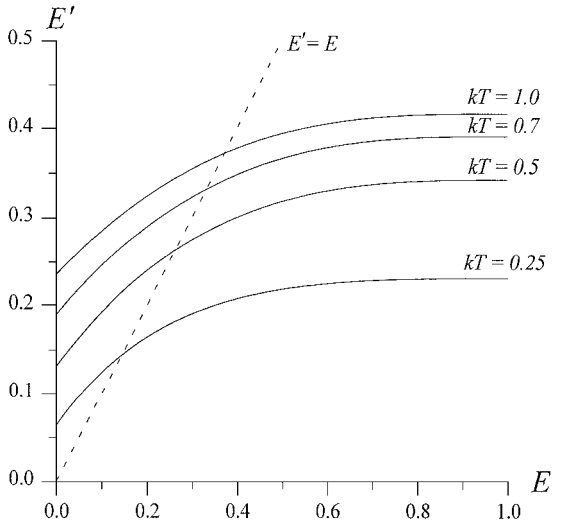

Fig. 2 Energy expectation at different temperatures.

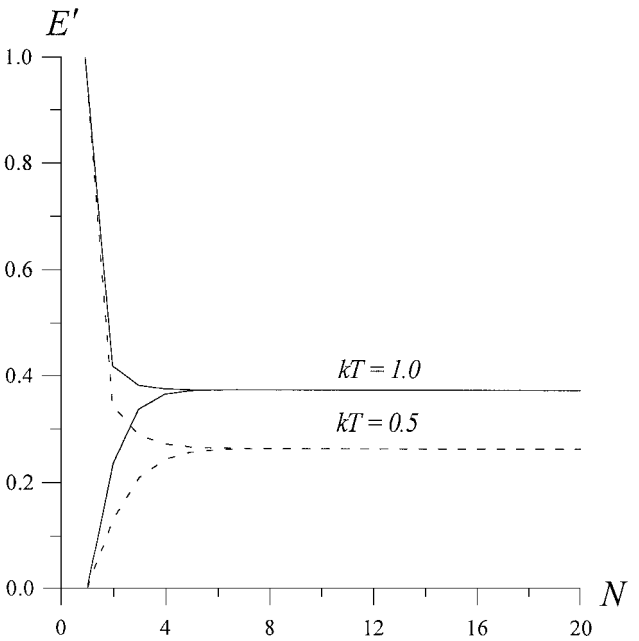

Fig. 3 Stable energy at different temperatures.

$$E_S - \{1/[1-\exp(-1/kT)]\}\Big(kT\left\{1 - \tfrac{3}{4}\exp(-E_s/kT) - \tfrac{1}{4}\exp[-(2-E_S)/kT]\right\} + \left\{\tfrac{1}{2}E_S\exp[-(2-E_S)/kT] - \tfrac{1}{2}\exp[-(2-E_S)/kT] - E_S\exp(-1/kT)\right\}\Big) = 0 \tag{10}$$

For a given kT, we may get the corresponding stable energy E_S from solving Eq. (10) numerically. Once the average of the population energy is less than E_S, we claim that the system equilibrium is achieved at the current temperature. We may then reduce kT to a smaller value to further decrease the energy level effectively.

B. Convergent Condition

Once we decrease the temperature parameter kT, we check whether the convergent conditions are satisfied at the same time. As the SA process evolves, the population of candidates should concentrate toward the global optimum and their energy levels should become lower. Therefore, we may use these two properties to derive the convergent conditions. To avoid confusion, we use $\boldsymbol{y}$ to express the position vector relative to the reduced and normalized search space, and $\boldsymbol{x}$ to denote the position vector relative to the original search space. If all of the candidates are enclosed in a small region in the normalized search space, we defined the reduced region as a hypercube whose side lengths are given by

$$\Delta y_k = y_{k\max} - y_{k\min}, \qquad k = 1, 2, \ldots, n \tag{11}$$

where $y_{k\max}$ and $y_{k\min}$ are the largest and smallest values of coordinates among the candidates in the kth dimension, respectively. The convergence conditions for terminating a cycle can then be defined as

$$\Delta y_k \le L_c, \quad \forall k, \qquad E_{\max} \le E_c \tag{12}$$

where $E_{\max}$ is the highest energy level in the candidates and L_c and E_c are two control parameters to be prescribed for a particular problem. Note that the two parameters can be specified easily because L_c is a fraction of the search region in each dimension and E_c is a probability of finding a better candidate in the original space.

If the convergent condition (12) is satisfied, then all of the candidates would have concentrated into a small region with side length less than L_c in each dimension, and the probability of finding a better candidate in the original domain would be less than E_c [because $E = C(\bar{f})$]. Thus, the reduced region in the original search space is bounded by $[x_{k\min}, x_{k\max}]$, $k = 1, 2, \ldots, n$. The length of the region in each dimension can be denoted as

$$\Delta x_k = x_{k\max} - x_{k\min}, \qquad k = 1, 2, \ldots, n \tag{13}$$

Although the global optimum should be enclosed in the reduced region with high probability, it is still possible that the global optimum is close to but outside the reduced region. To overcome this difficulty, we may choose a larger search space than the reduced region defined by Eq. (11) or Eq. (13) for the next cycle as

$$\boldsymbol{D} = \{(x_1, x_2, \ldots, x_n) \mid x_{k\min} - c \cdot \Delta x_k \le x_k \le x_{k\max} + c \cdot \Delta x_k, k = 1, 2, \ldots, n\} \tag{14}$$

where c is a constant and is chosen as 0.2.

C. Stopping Criterion

A new cycle starts in a unit hypercube, which is obtained by normalizing each dimension of the reduced search space $\boldsymbol{D}$. Searching in the reduced region $\boldsymbol{D}$ can, in general, locate the optimum faster. Besides, we may set up the stopping criterion by using a modified objective function in each new cycle. The procedure is described as follows.

Let $\boldsymbol{x}_i^*$ be the best candidate found in the ith cycle, and the corresponding function value $f(\boldsymbol{x}_i^*)$ be f_i^*, then we may replace the objective function $f(\boldsymbol{x})$ by $F_{i+1}(\boldsymbol{x})$ for the $(i+1)$th cycle, where

$$F_{i+1}(\boldsymbol{x}) = f(\boldsymbol{x}) - f_i^* \tag{15}$$

Note that the value of the modified objective function is always negative when a better state is found, and it approaches zero as the cycle number increases. If a cycle is terminated with a positive value of the modified objective function, which implies that the candidates converged to a local optimum, we may restart this cycle. If the value of the modified objective function is very close to zero, which means that it is hard to find a better state, we may then terminate the search process.

In general, the criterion for terminating the search process can be defined by the following two conditions:

1) All candidates converge to a small region, that is,

$$\frac{x_{k\max} - x_{k\min}}{x_k^U - x_k^L} \le \varepsilon_x, \qquad k = 1, 2, \ldots, n \tag{16}$$

where $x_{k\max}$ and $x_{k\min}$ are the largest and the smallest values of the kth coordinate component, respectively, in the population at the end of a cycle; x_k^U and x_k^L are the upper and lower limits of the kth coordinate component, respectively, in the optimization problem; and ε_x is a positive small number.

2) The value of the modified objective function approaches zero, that is,

$$-\varepsilon_f \le \frac{F_n(\boldsymbol{x}_n^*)}{|f(\boldsymbol{x}_n^*)|} \le 0 \tag{17}$$

where $\boldsymbol{x}_n^*$ is the best state found in the current cycle and ε_f is a positive small number.

When both of the two conditions are satisfied, the region of convergence relative to the original search space is very small, and it would be difficult to find a better candidate, and so the searching process can be terminated.

D. Neighborhood Structure

The choice of an appropriate neighborhood structure is important for the efficiency of the SA approach. There are many ways to generate a new state $\boldsymbol{y}' = (y_1', y_2', \ldots, y_n')$ from a current state $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$. Herein, we consider a random direction as well as a random step length, so that

$$y_k' = y_k + r d_k, \qquad k = 1, 2, \ldots, n \tag{18}$$

where $r = k_r \sqrt{E}$ is the step length and $d_k \sim N(0, 1)$ is a Gaussian random number generated from a Gaussian distribution with zero mean and unit variance. The distance from a new state to the current state in a certain direction is controlled by two parameters: a Gaussian random number d_k and the parameter of step length r. The Gaussian random numbers are introduced such that a direction is selected at random, and the closer the new state is to the current state, the higher the probability is that it would be selected. The parameter of step length r is set to be proportional to $\sqrt{E}$, which follows from the idea that the contour surface constructed from the objective function can be approximated as a quadratic function of design variables at points close to the optimum; also the lower the energy is, the shorter distance the state will move. The proportional constant k_r should be chosen large enough to allow state variations to distribute throughout the domain of search, that is, the unit hypercube for the current cycle, to avoid getting trapped in a local minimum easily. In this paper, k_r is chosen to be 5 for all of the following numerical experiments.

Note that although we reduce the domain of search in each cycle, the search step is still allowed to go beyond the reduced region. Thus, the search steps are not limited in the reduced space only, and this can somewhat avoid losing the global optima as the domain of search is reduced.

E. Reliability

Because all of the candidates start from different locations in the search space but converge into a small region and all of their CDF values are lower than E_c as a cycle is terminated, we may deduce that the probability of the event that the optimum is contained in this small region would be larger than $1 - E_c$, or the reliability that the optimum will be in this region is R_i, which is defined as

$$R_i = 1 - E_{i,\max} \tag{19}$$

where $E_{i,\max}$ is the highest energy of all of the candidates as they converge in the ith cycle. Because searches in each cycle could be treated as independent events, the reliability that the optimum is contained in the new region after n cycles can be calculated approximately as

$$R_n = \prod_{i=1}^{n} (1 - E_{i,\max}) \tag{20}$$

F. Exchanging of Notions and Retaining the Best

Exchanging of notions by the procedure crossover gives genetic algorithms much of their power. RRSA may have the advantage by generating a new candidate as the centroid of the existing candidates and replacing the worst one. The kth component of coordinate of the centroid, $\bar{y}_k$, is defined as

$$\bar{y}_k = \frac{1}{N_p} \sum_{i=1}^{N_p} y_{ik} \tag{21}$$

where N_p is the population size.

Because all candidates are supposed to concentrate from different locations gradually, the centriod of them is usually closer to the optimum than the worst candidate. This procedure improves the convergence rate, but should not be executed too frequently to avoid

getting trapped into a local minimum. Once a cycle has been completed, the best candidate is retained for the next cycle to improve the search efficiency.

G. Multiple Optima

For the current cycle, it is possible that kT has decreased to a very small value, but the candidates still fail to converge. This may be due to the situation that there exist several global optima or there are some local optima whose function values are close to the global optimum. To solve this problem, we introduce the method of simulated evolution[5] into RRSA. The evolution process consists of four major steps: reproduction, mutation, competition, and selection. To avoid getting trapped into a local optimum and to improve the convergence rate, Chiang and Huang[17] propose a new rule for competition and use an adjustable deviation for mutation. In RRSA, we use the simulated evolution in case a cycle fails to converge as kT is smaller than a prescribed small value T_{cv}. Note that we use the method of evolution just to identify the regions in which optima are possibly enclosed, but not to find the exact location of optima. After several generations of evolution, we may obtain regions where many organisms concentrate. Using the RRSA algorithm in these regions we may find the global optima or some local optima whose function values are close to those of the global optima. The procedure is described as follows:

1) Generate N_E organisms randomly in the search space for the current cycle.

2) Perform N_G generations of simulated evolution.

3) Let $\boldsymbol{y}_1 = (y_{11}, y_{12}, \ldots, y_{1n})$ be the best organism in the last generation.

4) Define a hypercube

$$\boldsymbol{D}_1 = \{(y_1, y_2, \ldots, y_n) \mid y_{1k} - d \le y_k \le y_{1k} + d, k = 1, 2, \ldots, n\}$$

where d is selected as a small percentage of each dimension, and then delete the organisms that are included in $\boldsymbol{D}_1$.

5) Let $\boldsymbol{y}_2 = (y_{21}, y_{22}, \ldots, y_{2n})$ be the best one of the remaining organisms; do the same procedure as in step 4 and obtain $\boldsymbol{D}_2$.

6) Repeat step 5 until there is no organism left, then search in those hypercubes $\boldsymbol{D}_i$ by using RRSA, respectively.

We select $N_E = 50$, $N_G = 200$, and $d = 0.05$.

H. RRSA Procedure

The complete procedure of RRSA is summarized in the following algorithm:

1) Normalize the search space $\boldsymbol{D}$.

2) Evaluate the approximate probability CDF by sampling from $\boldsymbol{D}$, and set $kT = 1$.

3) Compute stable energy E_S by using Eq. (10).

4) Choose N_p states as the initial population, and calculate the function values and corresponding energy levels.

5) Do the loop for $i = 1 \sim N_p$.

a) generate a new state using Eq. (18).

b) Calculate the accepted probability p by using Eq. (2).

c) Generate a uniform random number U in [0, 1].

d) If $U < p$, then replace the current state with the new state.

6) Calculate the average energy of population $\bar{E}$.

7) If $\bar{E} > E_S$, then go to step 5.

8) If the region convergent conditions are satisfied, then go to step 10.

9) If $kT > T_{cv}$, then $kT = kT \cdot T_r$ and go to step 3, otherwise perform simulated evolution to find candidate regions, then go to step 1.

10) If the stopping criteria (16) and (17) are satisfied, then go to step 12.

11) Determine the search space $\boldsymbol{D}$ using Eq. (14) and go to step 1 for a new cycle.

12) If all candidate regions found by simulated evolution have been searched, then terminate the program, otherwise go to step 1 to search the next region.

The flowchart is shown in Fig. 4. It is remarkable that, through the outlined procedures, we have significantly reduced the dependence of SA on problem-dependent parameters and overcome the difficulty of nonunique global optima.

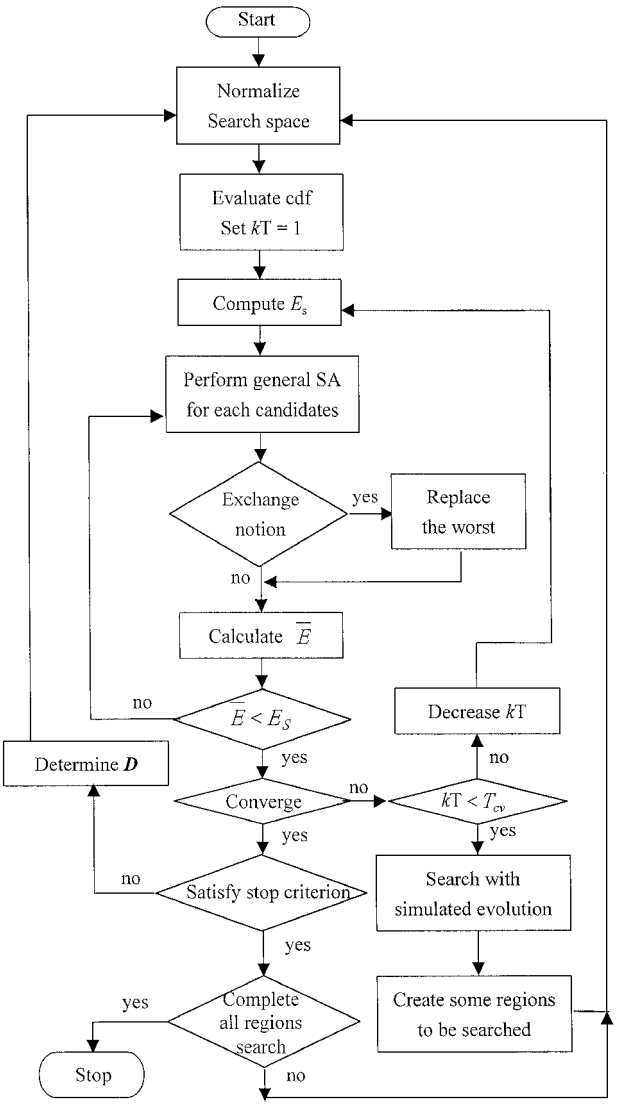

Fig. 4 Flowchart of RRSA.

IV. Numerical Experiments for Functional Optimization

We will test the RRSA algorithm using a set of standard problems available in the literature.[18] The following control parameters are used in RRSA for all of the following problems: 1) the number of sample states selected in each cycle for estimating probability distribution $N = 100$; 2) the number of candidates in a population $N_p = 3$; 3) temperature decreasing ratio $T_r = 0.6$; 4) region convergent conditions $L_c = 0.20$ and $E_c = 0.005$; 5) limit of temperature $T_{cv} = 1 \times 10^{-12}$; 6) frequency of notion exchange, once per 500 function value evaluations; 7) proportional constant of step length $k_r = 5$; and 8) stopping criterions $\varepsilon_x = 1 \times 10^{-2}$ and $\varepsilon_f = 1 \times 10^{-3}$.

The test problems are given next along with their domains of interest and optimal function values f^*. Each test problem has been solved with 10 different seeds for the random number generator. The test results are reported in Tables 1–3 by specifying the following quantities:

N_{opt} = number of times that the optimum has been found out of 10 times
$\boldsymbol{x}^*$ = optimal candidate
f^* = optimum function value
$\boldsymbol{x}^+$ = highest value of $\boldsymbol{x}^*$ in the 10 times
$\boldsymbol{x}^-$ = lowest value of $\boldsymbol{x}^*$ in the 10 times

as well as the number of function value evaluations (NFV).

Table 1 Result of test problem 1[a]

Average of x^*	N_{opt}
(−7.708379, 5.482880)	2
(−7.708424, −0.800481)	2
(−7.708192, −7.083494)	4
(−7.083759, 4.857993)	1
(−7.084535, −1.424979)	5
(−7.083453, −7.708098)	5
(−1.425045, 5.483011)	3
(−1.425124, −0.800610)	4
(−1.425444, −7.083274)	2
(−0.800353, 4.858189)	7
(−0.800206, −1.425278)	3
(−0.800250, −7.708132)	5
(4.858126, 5.482900)	7
(4.858344, −0.800422)	3
(4.857841, −7.083517)	1
(5.483077, 4.858139)	3
(5.481580, −1.424131)	3
(5.482778, −7.708371)	3

[a]Total NFV = 714707 (10 runs) average $f^* = -186.729$, and average reliability = 96.63%.

Problem 1, Shubert function:

$$f(x_1, x_2) = \left\{\sum_{k=1}^{5} k\cos[(k+1)x_1 + k]\right\}\left\{\sum_{k=1}^{5} k\cos[(k+1)x_2 + k]\right\}$$

$$-10 \le x_i \le 10, \qquad i = 1, 2, \qquad f^* \approx -186.73091$$

In the region considered, this function has 760 minima, 18 of which are also global. Table 1 presents the result of optimization using RRSA. It is seen that all of the 18 global optima have been found in the 10 tests, and each test individually found 1–12 optima. This demonstrates the ability of RRSA to find multiple optima in one search procedure. It is remarkable that because we chose a small population size ($N_p = 3$), it was not too difficult for all of the candidates to concentrate into a region containing only a few of the 18 global optima.

Problem 2, penalized Shubert function:

$$f(x_1, x_2) = \left\{\sum_{k=1}^{5} k\cos[(k+1)x_1 + k]\right\} \times \left\{\sum_{k=1}^{5} k\cos[(k+1)x_2 + k]\right\} + \beta\left[(x_1 + 1.42513)^2 + (x_2 + 0.80032)^2\right], \qquad \beta = 0.5$$

$$-10 \le x_i \le 10, \qquad i = 1, 2, \qquad f^* \approx -186.73091$$

This function has the same characteristics as the function considered in problem 1 but has a unique global minimum at (−1.42513, −0.80032).

In Table 2 we observe that besides the optimum being found in every test, a local optimum whose function value is close to the optimum was also found four times in the 10 tests.

Problem 3:

$$f(\boldsymbol{x}) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{k=1}^{n-1}(y_k - 1)^2 \times \left[1 + 10\sin^2(\pi y_{k+1})\right] + (y_n - 1)^2\right\}$$

$$y_i = 1 + \frac{1}{4}(x_i - 1), \qquad -10 \le x_i \le 10$$

$$i = 1, 2, \ldots, n, \qquad f^* = 0$$

Table 2 Result of test problem 2[a]

Design variable	Average of x^*	x^-	x^+
	N_{opt} = 10, average f^* = −186.729		
x_1	−1.424898	−1.440378	−1.409482
x_2	−0.800448	−0.817332	−0.783578
	N_{opt} = 4, average f^* = −186.340		
x_1	−0.800566	−0.808972	−0.792476
x_2	−1.425014	−1.433029	−1.416500

[a]Total NFV = 31,6201 (10 runs) and average reliability = 96.67%.

Table 3 Result of test problem 3[a]

Design variable	Average of x^*	x^-	x^+
x_1	0.999959	0.992802	1.007227
x_2	1.001668	0.983674	1.021328
x_3	0.999543	0.983353	1.021048
x_4	0.999961	0.981213	1.019995
x_5	0.998815	0.980727	1.017686
x_6	0.998311	0.977114	1.017590
x_7	1.000319	0.984788	1.016967
x_8	0.999289	0.978128	1.022293
x_9	0.998295	0.976061	1.016361
x_{10}	1.001267	0.978044	1.019580

[a]Average of $f^* = 5.002591E-06$, average of reliability = 95.69%, and average of NFV = 16428.9.

Table 4 Comparison of RRSA with SE and GESA[a]

Function value range of the best solution	Percentage of the best solution found in 50 runs by different methods		
	SE[a]	GESA[a]	RRSA
1.0000–1.0001	66%	100%	100%
1.0024–1.0026	34%	——	——
Average NFV	800,000	800,000	92,254

[a]From Yip and Pao.[14]

This function has roughly a 5^n local minima and a unique global minimum located at $x_i^* = 1$, $i = 1, 2, \ldots, n$. Let $n = 10$. The result of the optimization search using RRSA is listed in Table 3, which indicates that, even for a problem of such high dimension, RRSA can solve it effectively.

In addition to these test problems, we compare the performance of RRSA with that of the methods proposed by Yip and Pao.[14] Consider the following function:

$$y = \tfrac{1}{2}\left(x_1^2 + x_2^2\right) - \cos(20\pi x_1)\cos(20\pi x_2) + 2$$

This function has 40,000 local minimum points in the region where x_1 and x_2 are within $[-10, 10]$. The value of the global minimum is one at $x_1 = x_2 = 0$. Yip and Pao[14] used the methods of guided evolutionary SA (GESA) and general simulated evolution (SE) to solve this problem. They started with 400 organisms in each generation and terminated the process after 2000 generations. For the GESA, SE, and RRSA approaches, 50 random runs were conducted. The results are summarized in Table 4. The results indicate that RRSA can find not only global optimum, but also many suboptima in each search with good efficiency. It is remarkable that there is a big difference in the number of function evaluations between RRSA and the methods employed by Yip and Pao. This is due to the convergence criteria employed for terminating the different search methods. In RRSA, the stopping criteria are easily set up and checked automatically in the search process, whereas the methods employed by Yip and Pao did not have general convergence criteria specified, and the searches stopped only when a larger number of generations have been executed.

V. Application of RRSA in Structural Optimization

Generally a structural design is required to conform to a number of inequality or equality constraints related to stress, deflection,

dimensional relationships, and other variables. The general mathematical statement of an optimization problem can be written as follows:

Minimize $f(\boldsymbol{x})$ subject to

$$g_j(\boldsymbol{x}) \le 0, \qquad j = 1 \sim m$$

$$h_k(\boldsymbol{x}) = 0, \qquad k = 1 \sim p$$

$$x_i^L \le x_i \le x_i^U, \qquad i = 1 \sim n$$

where $f(\boldsymbol{x})$ is the objective function to be minimized, $g_j(\boldsymbol{x})$ and $h_k(\boldsymbol{x})$ are the m-inequality and p-equality constraints for the problem, and x_i are the n-design variables with lower and upper bounds x_i^L and x_i^U, respectively.

To use RRSA in a structural optimization problem, we may conveniently use the penalty-function approach. The constrained minimization problem is converted into an unconstrained problem by introducing the transformed objective function as

$$\phi(\boldsymbol{x}, \boldsymbol{r}) = f(\boldsymbol{x}) + P[\boldsymbol{h}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x}), \boldsymbol{r}] \tag{22}$$

where $\boldsymbol{r}$ is a vector of controlling (penalty) parameters and P is a real-valued function whose action of imposing the penalty is controlled by $\boldsymbol{r}$. The form of the penalty function P depends on the method used. In conjunction with the RRSA method, we use the exterior penalty function approach and choose the penalty function as

$$P[\boldsymbol{h}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x}), \boldsymbol{r}] = r \times \left\{ \sum_{k=1}^{p} [h_k(\boldsymbol{x})]^2 + \sum_{j=1}^{m} g_j^+(\boldsymbol{x}) \right\} \tag{23}$$

where

$$g_j^+(\boldsymbol{x}) = \begin{cases} 0 & g_j(\boldsymbol{x}) \le 0 \\ |g_j(\boldsymbol{x})| & \text{for} \quad 0 < g_j(\boldsymbol{x}) < 1 \\ [g_j(\boldsymbol{x})]^2 & g_j(\boldsymbol{x}) \ge 1 \end{cases} \tag{24}$$

Because RRSA is a zero-order search method, the discontinuity in the derivative of the objective function is a matter of no concern. Thus, we use an absolute valued function instead of the quadratic function for $0 < g_j(\boldsymbol{x}) < 1$ because the former gives better effect of penalty than the latter in this region.

To illustrate the procedure of solution by RRSA method, consider the design problem of a 10-bar truss shown in Fig. 5. This example has been used by many investigators to demonstrate various methods of solution.[19] Although this problem can be solved by traditional optimization methods in a small number of function evaluations, the results obtained are highly dependent on the initial point chosen and are usually local optima only. Note that it is not our intention simply to compare the accuracy and computational efficiency of the RRSA approach to those of traditional optimization methods. Instead, we want to show that even in such a simple structural optimization problem a global optimization approach may still easily get trapped in a local optimum. The truss is subjected to a single-loading condition, and the objective function represents the total volume of the truss members:

$$f(\boldsymbol{x}) = \sum_{i=1}^{10} l_i x_i \tag{25}$$

where l_i is the length of each member as given in Fig. 5 and the design variable x_i is the cross-sectional area of each member. The

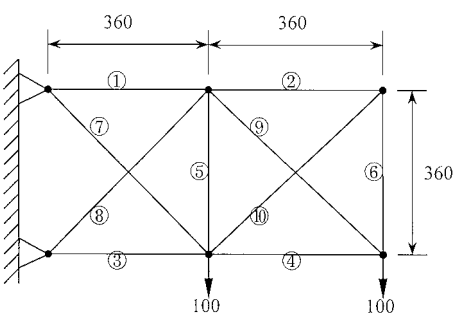

Fig. 5 Example: 10-bar truss.

member-size constraints and stress constraints for $i = 1, 2, \ldots, 10$ are

$$0.1 \le x_i, \qquad -25 \le \sigma_i \le 25$$

There is no equality constraint in this problem, and so the transformed objective function can be defined as

$$\phi(\boldsymbol{x}, r) = f(\boldsymbol{x}) + r \cdot \sum_{j=1}^{10} g_j^+(\boldsymbol{x}) \tag{26}$$

where

$$g_j(\boldsymbol{x}) = \begin{cases} \left(\sigma_j^L - \sigma_j\right) \Big/ \left|\sigma_j^L\right| & \sigma_j < \sigma_j^L \\ 0 \quad \text{for} & \sigma_j^L \le \sigma_j \le \sigma_j^U \\ \left(\sigma_j - \sigma_j^U\right) \Big/ \sigma_j^U & \sigma_j > \sigma_j^U \end{cases} \tag{27}$$

and $g_j^+(\boldsymbol{x})$ is defined as in Eq. (24).

This problem has been solved with 10 different seeds for the random number generator. The test results are reported in Tables 5 and 6. The initial value of parameter r for each cycle is chosen as 200, and it changes as equilibrium is reached, that is, at the time when the temperature parameter kT is reduced. In this problem, the penalty parameter changes according to

$$r = f^* \cdot (N_{RT} + 1)$$

where N_{RT} is the number of times that kT has been reduced in the current cycle and f^* is the lowest function value found in the previous cycle. Through this formula, the penalty parameter can be determined directly without changing with different problems.

In Table 5, the result presented by Hajela and Yoo[20] is also listed for comparison. The problem is the same as the one just stated, but Hajela and Yoo used weight as the objective function, and the problem was solved using the genetic algorithm (GA). It can be seen in Table 5 that the number of function evaluations in RRSA is generally a little higher than that in GA, but the present results are much more accurate. If we carefully examine the result of each design point, it can be found that most of the optima are only local minima. Even though their function values are very close to the exact solution, some of the design variables are quite different from those of the global optimum. In Table 6, the detailed result of a typical 1 of the 10 tests is presented. It is shown that even if the relative error of function value is small (0.34%), it still could be a relative optimum. From the point of view of structural design, the result is acceptable, but the optimal design is the ultimate goal. Note that some variables near the size constraints are so small, for example, x_{10}, that they hardly affect the objective function value. Thus, the search is virtually dominated by the variables that have larger values. To solve this problem, we modify the transformed objective function by adding an extra penalty term and using the augmented Lagrangian method[21] as follows:

$$\phi(\boldsymbol{x}, r) = f(\boldsymbol{x}) + r \cdot \sum_{j=1}^{10} [g_j(\boldsymbol{x}) + \theta]^+ + w \cdot \sum_{j=1}^{10} l_j \cdot \frac{\left(x_j - x_{ij}^*\right)}{x_{ij}^*} \tag{28}$$

where θ and w are two adjustable parameters. They are updated along with the penalty parameter r with initial values of one in each cycle and approach zero as the design point gets close to the optimum. In this paper, we choose the factors of $\frac{1}{10}$ and $\frac{9}{10}$, respectively, to decrease θ and w in each cycle. The temporary optimal design x_{ij}^* is the best design point in the $(i-1)$th cycle. The last term in Eq. (28) has the effect of not only giving the penalty as $x_j > x_{ij}^*$ but also giving the credit as $x_j < x_{ij}^*$.

The results obtained by using the modified objective function given by Eq. (28) are also summarized in Table 5, which shows that RRSA method can solve this problem more accurately and more efficiently by using a proper objective function. Similarly, we present the result of one typical test in Table 6, where we observe that all of the constraints are satisfied and all of the design variables are close to the exact solutions. Although we have shown the

Table 5 Results of the 10-bar truss problem

	Hajela and Yoo,[20] best f = 1635			Objective function using Eq. (26), exact f = 15931.8			Objective function using Eq. (28), exact f = 15931.8		
Number	f^*	Error, %	NFV	f^*	Error, %	NFV	f^*	Error, %	NFV
1	1678	2.630	23721	15982.9	0.321	26070	15938.9	0.045	21383
2	1677	2.568	34971	15947.4	0.098	26270	15934.3	0.016	20125
3	1639	0.245	14803	15993.8	0.389	23653	15938.1	0.040	26733
4	1635	0	17906	15971.7	0.250	22989	15934.2	0.015	21021
5	1688	3.241	16862	15968.7	0.232	24072	15936.2	0.028	22816
6	1666	1.896	26401	15966.1	0.215	22982	15934.5	0.017	23013
7	1692	3.486	23399	15970.2	0.241	23875	15946.5	0.092	19603
8	1725	5.505	14803	15946.7	0.094	27279	15934.9	0.019	24615
9	——	——	——	16065.3	0.838	23272	15986.2	0.341	16531
10	——	——	——	16047.8	0.728	28528	15953.2	0.134	20366
Average	1675	2.446	23992	15986.1	0.341	24899	15943.7	0.075	21621

Table 6 Detailed result of one typical test for the 10-bar truss problem

		Objective function using Eq. (26)			Objective function using Eq. (28)		
x	Exact solution	No. 1 in Table 6	Error, %	Stress	No. 1 in Table 6	Error, %	Stress
x_1	7.94	7.9100	−0.38	+24.9520	7.9410	+0.01	+24.9948
x_2	0.10	0.1000	0.00	+24.3150	0.1005	+0.50	+15.5698
x_3	8.06	8.1524	+1.15	−25.0014	8.0732	+0.16	−25.0000
x_4	3.94	3.9081	−0.81	−25.0048	3.9390	−0.03	−24.9813
x_5	0.10	0.1000	0.00	+0.1820	0.1000	0.00	−0.0152
x_6	0.10	0.2187	+118.70	+11.0472	0.1000	0.00	+15.4840
x_7	5.74	5.7883	+0.84	+25.0036	5.7450	+0.08	+24.9749
x_8	5.57	5.5270	−0.77	−24.9771	5.5706	+0.01	−24.9877
x_9	5.57	5.5286	−0.74	+25.0028	5.5697	−0.01	+24.9869
x_{10}	0.10	0.1584	+58.40	−19.8893	0.1000	0.00	−21.9506

Table 7 Results of the 10-bar truss problem with different frequencies of notion exchange[a]

	N = 50			N = 5000		
Number	f^*	Error, %	NFV	f^*	Error, %	NFV
1	16173.2	1.515	15666	15935.8	0.025	28337
2	17349.2	8.900	20776	15944.3	0.078	25973
3	15935.1	0.021	19674	15934.9	0.019	28566
4	16366.4	2.728	17271	15949.9	0.114	23429
5	17098.4	7.322	20435	15939.0	0.045	25995
6	16338.4	2.552	21385	15944.9	0.082	23742
7	16570.6	4.010	17417	15946.1	0.089	26163
8	17682.9	10.991	17234	15939.9	0.051	25717
9	15932.5	0.004	18979	15943.0	0.070	23795
10	16550.0	3.880	19429	15964.1	0.203	23773
Average	16599.7	4.192	18827	15944.2	0.078	25549

[a]Frequency of notion exchange: once per N function value evaluations.

performance of RRSA through the examples presented, comparison among global optimization methods is usually difficult. The accuracy and efficiency of an approach are generally controlled by several parameters, and some of the parameters may have adverse effects on the accuracy or efficiency. For example, if we use different frequencies of notion exchange in solving the 10-bar truss problem, the results are quite different, as presented in Table 7. Note that the more frequently we introduce notion exchange, the larger the error in the optimal function value is generally. Nevertheless, the total number of function evaluations get reduced. Thus, the potential effectiveness of centroid-based notion exchange must be utilized with caution in practice to prevent premature convergence.

VI. Conclusions

By including the advantages of SE and GA and introducing the idea of stable energy and region reduction, we propose a global optimization method based on SA called RRSA. In addition to faster convergence than conventional SA, advantages of RRSA include the capability of searching multiple optima and the ease of determining the initial temperature, equilibrium criterion, and the stopping rule. By automating these decisions, we have reduced the dependence of SA on problem-dependent parameters. All of these characteristics make the RRSA algorithm a strong competitor to the existing algorithms in global optimization. Further studies are merited of the characteristics of the RRSA approach and its application to various engineering optimization problems.

Acknowledgments

This study was supported by the National Science Council of the Republic of China under Grant NSC 86-2612-E-006-004. The authors also wish to thank anonymous reviewers for giving valuable comments and suggestions.

References

[1]Brooks, K. H., "A Discussion of Random Methods of Seeking Optima," *Operations Research*, Vol. 6, No. 3, 1958, pp. 244–251.

[2]Solis, F. J., and Wets, R. J.-B., "Minimization by Random Search Techniques," *Mathematics of Operations Research*, Vol. 6, No. 1, 1981, pp. 19–30.

[3]Fabio, S., "Stochastic Techniques for Global Optimization: A Survey of Recent Advances," *Journal of Global Optimization*, Vol. 1, No. 2, 1991, pp. 207–228.

[4]Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison–Wesley, Reading, MA, 1989, Chap. 1.

[5]Fogel, L. J., Owens, A. J., and Walsh, M. J., *Artificial Intelligence Through Simulated Evolution*, Wiley, New York, 1966, Chap. 1.

[6]Aarts, E., and Korst, J., *Simulated Annealing and Boltzmann Machines*, Wiley, New York, 1989, Chap. 1.

[7]Jenkins, W. M., "Towards Structural Optimization Via the Genetic Algorithm," *Computers and Structures*, Vol. 40, No. 5, 1991, pp. 1321–1327.

[8]Riche, R. L., and Haftka, R. T., "Optimization of Laminate Stacking Sequence for Buckling Load Maximization by Genetic Algorithm," *AIAA Journal*, Vol. 31, No. 5, 1993, pp. 951–956.

[9]Junjiro, O., and Yoji, H., "Actuator Placement Optimization by Genetic and Improved Simulated Annealing Algorithms," *AIAA Journal*, Vol. 31, No. 6, 1992, pp. 1167–1169.

[10]Atiqullah, M. M., and Rao, S. S., "Parallel Processing in Optimal Structural Design Using Simulated Annealing," *AIAA Journal*, Vol. 33, No. 12, 1995, pp. 2386–2392.

[11]Srichander, R., "Efficient Schedules for Simulated Annealing," *Engineering Optimization*, Vol. 24, No. 2, 1995, pp. 161–176.

[12]Davis, L., and Ritter, F., "Schedule Optimization with Probabilistic Search," *Proceedings of the 3rd IEEE Conference on Artificial Intelligence*, Inst. of Electrical and Electronics Engineers, New York, 1987, pp. 231–236.

[13]Boseniuk, T., and Ebeling, W., "Optimization of NP-Complete Problems by Boltzmann–Darwin Strategies Including Life Cycles," *Europhysics Letters*, Vol. 6, No. 2, 1988, pp. 107–112.

[14]Yip, P. P. C., and Pao, Y. H., "Combinatorial Optimization with Use of Guided Evolutionary Simulated Annealing," *IEEE Transactions on Neural Networks*, Vol. 6, No. 2, 1995, pp. 290–295.

[15]Jaynes, E. T., "Information Theory and Statistical Mechanics," *Physical Reviews*, Vol. 106, No. 6, 1957, pp. 620–630.

[16]Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by Simulated Annealing," *Science*, Vol. 220, May 1983, pp. 671–680.

[17]Chiang, D. Y., and Huang, S. T., "Modal Parameter Identification Using Simulated Evolution," *AIAA Journal*, Vol. 35, No. 7, 1997, pp. 1204–1208.

[18]Lucidi, S., and Piccioni, M., "Random Tunneling by Means of Acceptance–Rejection Sampling for Global Optimization," *Journal of Optimization Theory and Applications*, Vol. 62, No. 2, 1989, pp. 255–277.

[19]Kirsch, U., *Optimum Structural Design*, McGraw–Hill, New York, 1981, Chap. 6.

[20]Hajela, P., and Yoo, J., "Constraint Handling in Genetic Search Using Expression Strategies," *AIAA Journal*, Vol. 34, No. 12, 1996, pp. 2414–2420.

[21]Belegundu, A. D., and Arora, J. S., "A Computational Study of Transformation Methods for Optimal Design," *AIAA Journal*, Vol. 22, No. 4, 1984, pp. 535–542.

E. R. Johnson
Associate Editor