A study of simulated annealing variants

Ana I.P.N. Pereira[1], *Edite M.G.P. Fernandes*[2]

[1]apereira@ipb.pt, Braganca Polytechnic Institute, Braganca, Portugal
[2]emgpf@dps.uminho.pt, Minho University, Braga, Portugal

Abstract

Este trabajo presenta un estudio comparativo de cinco variantes del método *simulated annealing* de optimización global. Con la nueva variante aquí propuesta (ASALO), se obtiene el mejor valor de la función y un mayor porcentaje de convergencias.

Keywords: Global optimization; simulated annealing.

1. Introduction

Consider the nonlinear optimization problem in the following mathematical form:

$$\max_{t \in T} g(t) \tag{1}$$

where $g : \mathbb{R}^n \rightarrow \mathbb{R}$ is a given nonlinear function and T is a compact set defined by $T = \{t \in \mathbb{R}^n : a_i \leq t_i \leq b_i, i = 1, ..., n\}$. A global solution to problem (1) is the point $t^* \in T$ such that $\forall t \in T,\ g(t^*) \geq g(t)$.

There are two types of numerical methods to solve this problem. The deterministic methods usually require a great deal of information and conditions on the objective function and they do no guarantee convergence to the global maximum. On the other hand, the stochastic methods can be easily implemented and converge in practice to a global maximum. In particular, for the simulated annealing method, it has been proved that it converges asymptotically to a global solution.

Examples of stochastic methods are: Multistart, clustering, multi level, adaptive random search, genetic algorithms and simulated annealing.

Motivated by analogy with the behavior of physical systems in the presence of a heat bath, Kirkpatrick, Gelatt and Vecchi (1983), and Cerny (1985), proposed the simulated annealing (SA) approach to solve combinatorial optimization problems. Since then, the SA algorithm has been applied in many areas such as the graph partitioning, graph coloring, number partitioning, circuit design, composite structural design, data analysis, image reconstruction, neural networks, biology, geophysics and finance [7, 10, 15].

The main disadvantage of the SA algorithm is that it requires a great deal of function evaluations. To overcome this inefficiency, many authors have been proposing variants of the SA algorithm.

In this paper, we describe five variants of the simulated annealing method and analyze their performances for a standard set of test functions. This paper is organized as follows. Section 2 describes the simulated annealing algorithm and the four crucial phases of the method. In Section 3 we present five variants of the simulated annealing method including the new ASALO variant. The numerical results are shown in Section 4 and Section 5 contains the conclusions.

2. Simulated Annealing method

In 1953, Metropolis *et al.* proposed an algorithm to simulate the behavior of physical systems in the presence of a heat bath. Thirty years later, Kirkpatrick *et al.* [11] applied the Metropolis algorithm to combinatorial optimization problems and named it by simulated annealing.

In 1986, Bohachevsky *et al.* [1] applied the SA algorithm to solve continuous optimization problems. Since then, the SA algorithm has been subject to various modifications in order to improve its efficiency. See, for example, Corana *et al.* [2], Dekkers and Aarts [3], Ingber [8], Romeijn and Smith [14] and Szu and Hartley [16].

The spread use of the SA algorithm is mainly due to the fact that it is easily implemented, it can be applied to any optimization problem, it does not use any derivative information, it does not require specific conditions on the objective function and it has been proved that the SA algorithm asymptotically converges to a global maximum.

2.1. The SA algorithm

The SA algorithm can be easily described using four phases: the generation of a new candidate point, the acceptance criterion, the reduction of the control parameter and the stopping criterion.

Algorithm 2.1 *Given an initial approximation t^0, a control parameter c^0 and the number of iteration with the same control parameter N_c^k*

while *stopping criterion is not reached* ***do***

for $j = 1$ ***to*** N_c^k ***do***

Generate a new candidate point y

Analyze the acceptance criterion

end

Update N_c^k

Reduce the control parameter

end

The remaining of this section is devoted to present in detail the four referred phases of the SA algorithm.

2.2. Generation of a new candidate point

The generation of a new candidate point is one of the crucial phases of the SA algorithm. Obviously, the scheme that is used to generate a new point affects the performance of the algorithm. This scheme must be such that a good exploration of the search region and a feasible point are provided.

The initial approximation, t^0, is usually randomly generated. However, some authors suggest that this initial approximation should be constructed throughout a preliminary analysis of the problem.

Then, a new point is found using the current approximation, t^k, and the generating probability density function, $f_{t^k y}(c^k)$. This function establishes how the new candidate point is created and usually depends on the control parameter c^k and/or on the dimension of the problem.

Techniques to generate new candidate points can be found in the following papers: Bohachevsky *et al.* [1], Corana *et al.* [2], Dekkers and Aarts [3], Ingber [8], Romeijn and Smith [14], Szu and Hartley [16] and Tsallis and Stariolo [17].

2.3. Acceptance criterion

The acceptance criterion allows the SA algorithm to avoid getting stuck in local, non-global maximum, when searching for global maximum. This is accomplished by accepting points where a decrease of the objective function is verified. During the process, the probability of negative movements decreases slowly to zero and in the final phase, the algorithm improves the precision of the approximation to a global maximum.

$A_{t^k y}(c^k)$ is the acceptance function and it represents the probability of accepting the point y when t^k is the current point. This function depends on the control parameter and on the difference of the function values at the points y and t^k. The acceptance criterion has the following form

$$t^{k+1} = \begin{cases} y & \text{if } \tau \leq A_{t^k y}(c^k) \\ t^k & \text{otherwise} \end{cases}$$

where t^k is the current approximation to the global maximum, y is the new candidate point and τ is a uniformly random number drawn from $U(0,1)$.

When $A_{t^k y}\left(c^k\right) = \min\left\{1, e^{-\frac{g\left(t^k\right)-g(y)}{c^k}}\right\}$, then the acceptance criterion is denoted by Metropolis criterion.

This criterion accepts all points where the objective function value increases, i.e., $f(t^k) \leq f(y)$, because $e^{-\frac{f(t^k)-f(y)}{c^k}} \geq 1$. However, if $f(t^k) > f(y)$, the point y might be accepted with some probability.

When the control parameter c^k is high, the maximization process searches in all feasible region, looking up for promising regions to find the global maximum. As the algorithm develops, c^k is slowly reduced and the process computes best approximations to a optimum.

The Metropolis criterion is the most used acceptance criterion in all the SA variants. In particular, this criterion is used in all variants presented in the Section 3. Different acceptance criteria are suggested in the literature. See Ingber [8] and Tsallis and Stariolo [17] papers.

2.4. Reduction of the control parameter

The function c^k is called the control parameter, temperature or cooling schedule, and must be a decreasing function that verifies

$$\lim_{k \to \infty} c^k = 0.$$

A crucial phase of the SA algorithm is to determine how the control parameter should be reduced. The procedure must be quick and guarantee that the SA algorithm converges to a global maximum.

For a good performance of the algorithm, the initial control parameter must be sufficiently high (to search for promising regions) but not extremely high because, in this case, the algorithm becomes too slow. To solve this dilemma, some authors suggested that a preliminary analysis of the objective function should be done in order to get an appropriate value, Dekkers and Aarts [3], Ingber [8] and Laarhoven and Aarts [12].

2.5. Stopping criterion

Any iterative process requires a stopping criterion to terminate the algorithm. There are in the literature many stopping criteria that can be used to terminate de SA algorithm. All criteria are based on the idea that the algorithm should terminate when "... *the system "freezes" and no further changes occur*..."[11]. The usual stopping criterion is to limit the number of function evaluations (later denoted in the paper by N_{FE_MAX}). Other used stopping criterion defines a lower limit for the value of the control parameter. In this case, the iterative process terminates when the control parameter verifies $c^k < c_{min}$, where c_{min} is a pre-defined parameter.

The stopping criterion proposed in this paper pretends to terminate the algorithm when successive approximations to a global maximum are similar,

i.e, the algorithm stops if the following condition is verified for N^* successive iterations

$$|f^* - f^*_{ant}| < \varepsilon$$

where f^*_{ant} represents the previous approximation to an optimum value. Different termination criteria were proposed by Corana *et al.* [2], Dekkers and Aarts [3] and Ingber [8].

3. Variations on original simulated annealing

To accelerate the convergence of the SA algorithm many variants have been appeared in the literature. In this section we describe five variants of the simulated annealing method: the standard simulated annealing, herein denoted by SSA, the variant of the SA algorithm presented by Corana *et al.* (CSA), the ASA variant suggested by Ingber, the SALO algorithm proposed by Desai *et al.* and our variant, named ASALO algorithm, which combines some ideas from the ASA and SALO variants.

3.1. Standard SA variant

The standard simulated annealing (SSA) or Boltzmann annealing algorithm for continuous optimization was proposed by Bohachevsky *et al.* [1]. In this algorithm, the generation of a new candidate point is based on the current approximation and on a direction vector. The new point is computed by $y = t^k + \lambda$.

The control parameter c^k decreases through the reduce factor μ by the following way

$$c^{k+1} = \mu^k c^k$$

where the reduce factor $\mu \in (0, 1)$. In the SSA algorithm the N_c^k is constant.

3.2. Corana SA variant

In 1987, Corana *et al.* [2] suggested one variant of the simulated annealing algorithm (CSA). Later, Goffe *et al.* [5] proposed some modifications to the CSA algorithm.

This variant consists of using adaptive moves along the coordinate directions. For that, each new candidate point is obtained through the current approximation changing only one coordinate. The new point is given by $y = t^k + d_i^k \lambda_i^k e_i$, where d_i^k is a uniformly distributed random variable in $(-1, 1)$, λ_i^k is the component of the step vector λ^k and e_i is the euclidian vector. After N_λ iterations, each step vector component λ_i^k is updated to better adjust the optimization problem.

The value d_i^k is given by $d_i^k = 2u - 1$ where u is a uniformly distributed random variable in $(0, 1)$. The adjustment of the step vector component is done

as follows

$$\lambda_i^k = \begin{cases} \lambda_i^k * \left[1 + V_i * (\frac{r-0.6}{0.4})\right] & 0.6 < r \\ \lambda_i^k & 0.4 \leq r \leq 0.6 \\ \frac{\lambda_i^k}{\left[1+V_i*(\frac{0.4-r}{0.4})\right]} & r < 0.4 \end{cases}$$

where r represents the percentage of points accepted according to the coordinate i, i.e., $r =$ (number of the accepted points according to coordinate i)$/(N_\lambda)$ and V_i is a fixed value throughout the process.

The main idea for this adjustment is to accept 50% of the generated points. To accomplish this, the algorithm proceeds as follows: if $0.6 < r$ then more than 60% of generated points were accepted. This behavior indicates that the generated points are far away from the global maximum. In this situation, the ith coordinate of the step vector should proportionally increase according to the factor $\eta \in (1, 1 + V_i]$. On the other hand, if the algorithm accepted less than 40% of the generated points, then the ith coordinate of the step vector should proportionally decrease according to the factor $\eta \in \left[\frac{1}{1+V_i}, 1\right)$. Finally, when the algorithm accepted between 40 % and 60% of the generated points, then the step vector should not be updated.

Goffe *et al.* suggested that the number of the iteration with the same control parameter value should be constant during the process [5].

3.3. ASA variant

Ingber [6], in 1989, introduced some alterations to the Fast Annealing algorithm proposed by Szu and Hartley, and named it by Very Fast Simulated Annealing. Later, in 1993, Ingber [7] renamed it by adaptive simulated annealing (ASA) and it is the most used variant of the SA method today.

This variant is characterized by two functions: the generating probability density function, $f_{t^k y}(c_G)$, and the acceptance function, $A_{t^k y}(c_A)$. The first function determines how a new candidate point is generated and the second one establishes if a candidate point is accepted. Both functions depend on the current approximation, on the new candidate point and on the control parameters, $c_G \in \mathbb{R}^n$ and $c_A \in \mathbb{R}$, respectively.

Algorithm 3.1 *(ASA) Given a initial feasible approximation* t^0, $k_A = k_G = 0, \kappa = -\ln[\epsilon]\, e^{-\frac{\ln[N\epsilon]}{n}}$, *the control parameters* c_A^0, $c_{G_i}^0 = 1.0$ *and the number of iterations for reannealing* $N_{A\_\max}$

while *stopping criterion is not reached* ***do***

Generate a new candidate point y

Analyze the acceptance criterion

if $n_A \geq N_{A\_\max}$ ***then*** *redefine* k_A *and* k_G

Reduce the control parameter

end

Motivated by the fact that the objective function behaves differently along different directions, Ingber proposed different generating probability density functions for different variables. So, f_i represents the generating probability density function associated with the t_i variable, and it is given by

$$f_i\left(t_i^k, \lambda_i, c_{G_i}^k\right) = \frac{1}{2\left[|\lambda_i| + c_{G_i}^k\right] \ln\left(1 + \frac{1}{c_{G_i}^k}\right)} \text{ for } 1 \leq i \leq n.$$

A new candidate point, $y = [y_1, ..., y_n]$, is determined as follows

$$y_i = t_i^k + \lambda_i \left(b_i - a_i\right) \text{ for } 1 \leq i \leq n \tag{2}$$

where a_i and b_i are the lower and upper bounds for the t_i variable, respectively. The value $\lambda_i \in (-1, 1)$ is given by

$$\lambda_i = sgn\left(u - \frac{1}{2}\right)\left(\left(1 + \frac{1}{c_{G_i}^k}\right)^{|2u-1|} - 1\right) c_{G_i}^k \tag{3}$$

where u is a uniformly distributed random variable in $(0, 1)$.

When y is not a feasible point, then a new candidate point is computed using equations (2) and (3).

It is possible in this variant to redefine the control parameters $c_{G_i}^k$ and c_A^k in order to speed up the search process. After $N_{A\_\max}$ accepted points, the sensitivities given by

$$s_i = \left|\frac{g\left(t^* + \delta t_i^* e_i\right) - g^*}{\delta t_i^*}\right|$$

are computed, where t^* is the best point found so far, δ is a small real parameter and $e_i \in \mathbb{R}^n$ is the euclidian vector. Let

$$s_{\max} = \max_{1 \leq i \leq n} \left\{s_i\right\},$$

then the parameter k_{G_i} is updated by

$$k_{G_i} = \begin{cases} \left[-\frac{1}{\kappa} \ln\left(\frac{s_{\max}}{s_i} \frac{c_{G_i}^k}{c_{G_i}^0}\right)\right]^n & \text{if } \frac{s_{\max}}{s_i} \frac{c_{G_i}^k}{c_{G_i}^0} < 1 \\ 1 & \text{otherwise} \end{cases}$$

where $c_{G_i}^0$ is the initial value of the control parameter c_{G_i}.

Similarly, the parameters c_A^0 and k_A are redefined using

$$c_A^0 = \min\left\{c_A^0, \max\left\{\left|g\left(t^k\right)\right|, |g^*|, \left|g\left(t^k\right) - g^*\right|\right\}\right\}$$

and

$$k_A = \left[-\frac{1}{\kappa}\ln\left(\frac{\bar{c}_A}{c_A^0}\right)\right]^n$$

where $\bar{c}_A = \min\left\{c_A^0, \max\left\{\left|g\left(t^k\right) - g^*\right|, c_A^k\right\}\right\}$.

The value κ depends on ϵ and N_ϵ. This parameter is defined by

$$\kappa = -\ln\left[\epsilon\right] e^{-\frac{\ln[N\epsilon]}{n}}$$

where the values ϵ and N_ϵ should be such that

$$\begin{cases} c_{G_i}^f = c_{G_i}^0 \epsilon \\ k^f = N_\epsilon \end{cases}$$

with $c_{G_i}^f$, the final value of the control parameter c_{G_i}, and k^f represents the maximum number of iterations allowed. The influence of the values ϵ and N_ϵ in the algorithm can be analyzed in Niu [13]. In this algorithm, the size of the chain is one, meaning that the control parameter is always updated as follows

$$\begin{cases} k_{G_i} = k_{G_i} + 1 \\ c_{G_i}^k = c_{G_i}^0 e^{-\kappa\left(k_{G_i}\right)^{\frac{1}{n}}} \end{cases} \quad \text{for } 1 \leq i \leq n.$$

Similarly, the control parameter associated with the acceptance function is updated by

$$\begin{cases} k_A = k_A + 1 \\ c_A^k = c_A^0 e^{-\kappa(k_A)^{\frac{1}{n}}} \end{cases}$$

where c_A^0 represents the initial value of the control parameter c_A. Ingber ensures that, statistically, the algorithm determines a global maximum of the initial problem. We refer to [6, 8, 9] for more details.

3.4. SALO variant

Desai and Patil [4] suggested the variant SALO that combines the ASA algorithm with a local search procedure. SALO variant is similar to the ASA variant except in the generation of the new candidate point, where a local search algorithm, named hill climber, is used.

So, given a current approximation t^k, a slight perturbation is carried out on this point to obtain $\bar{y}^{k+1}$. Then, a local search algorithm is implemented

based on the initial approximation $\bar{y}^{k+1}$ and the resulting point is the new candidate point for SA method, y.

This procedure generates a sequence of local maxima of the initial problem, $\{t^k\}$. Desai and Patil guarantee that SALO variant converges, with probability one, to a global maximum of the optimization problem.

3.5. ASALO variant

In practice, some variants of the SA algorithm converge to an approximation that might not be sufficiently close to the global maximum. Our purpose is then to improve the precision of the approximation to a global maximum as well as to reduce the execution time. We propose herein the ASALO variant that is based on ASA and SALO algorithms and contains some strategies suggested by Romeijn and Smith [15] to guarantee that the generated points are feasible.

The determination of infeasible points causes an expense of execution time. For this reason, our ASALO variant incorporates a reflection technique proposed by Romeijn and Smith which can be summarized as follows.

Given a point $\bar{y} = [\bar{y}_1, ..., \bar{y}_n]$, the new candidate point is obtained by applying the following function to each coordinate of the point $\bar{y}$

$$r(\bar{y}_i) = \begin{cases} a_i + (a_i - \bar{y}_i) & \text{if} \quad \bar{y}_i < a_i \\ \bar{y}_i & \text{if} \quad a_i \leq \bar{y}_i \leq b_i \\ b_i - (\bar{y}_i - b_i) & \text{if} \quad \bar{y}_i > b_i \end{cases} .$$

The new candidate, in ASALO variant, is then the point $y = [r(\bar{y}_1), ..., r(\bar{y}_n)]$.

If this point is accepted, a local search procedure is implemented with y as the initial approximation. The resulting point of the local search procedure is the new approximation to the global maximum.

4. Computational Results

The five previously presented variants were implemented in C on a Pentium II, Celeron 466 Mhz with 64Mb of RAM. For the computational experiences we considered eight test functions (Branin (B), Goldstein and Price (GP), Shubert (S), Rosenbrock (R_2 e R_4), sphere model (Me_3), Hartmann (H_3) and Rastrigin (Ra_4)). Each variant was run four times for each test function with different random initial approximations. The following results are the average of the obtained numerical results in the successful runs.

We choose to use the following values. In the stopping criterion: $N^* = 5$, $\varepsilon = 10^{-6}$, $N_{FE\_MAX} = 100000$ and $N_{FE\_MIN} = 1000$. For the CSA variant, we considered $\overline{V_i} = 2.0$ and $\lambda_i^0 = 1.0$ for $i = 1, ..., n$ (as suggested by Corana *et al.*). The reduce factor and the length of the chains have the same values for

the SSA and CSA variants: $\mu = 0.95$ and N_c^k=21. In ASA, SALO and ASALO variants, we used $\epsilon = 10^{-5}$ and $N_\epsilon = 100$.

To determine the initial control parameter value, c^0, (or c_A^0 in ASA, SALO and ASALO variants) a preliminary analysis for each test function was carried out. For that, we considered a sample of $10 \times n$ feasible points (where n represents the dimension of the problem), and tested Dekkers and Aarts [3], Laarhoven and Aarts [12] and Ingber [8] proposals. Dekkers and Aarts proposal provided the best results.

4.1. Characterization of the presented variants

Some tests were done to characterize the presented variants. This study aims to identify the parameters that most influence the behavior of the presented variants.

The SSA and CSA variants have similar behavior and the crucial parameters in these variants are μ and c^0 values. If μ has a value near 1 then the sequence $\{c^k\}$ slowly decreases to zero and consequently the initial control parameter must be small. Two cases were analyzed: $\mu = 0.995$ and $c^0 = min\{1.5, \bar{c}\}$ where $\bar{c}$ is the value obtained by a preliminary analysis; $\mu = 0.95$ and the initial control parameter is determined by a preliminary analysis.

Figure 1 shows the accepted points provided by the SSA variant on the Branin test function for both cases. In the first, the initial control parameter at the beginning of the process has a high value and SSA variant behaves like a random method.

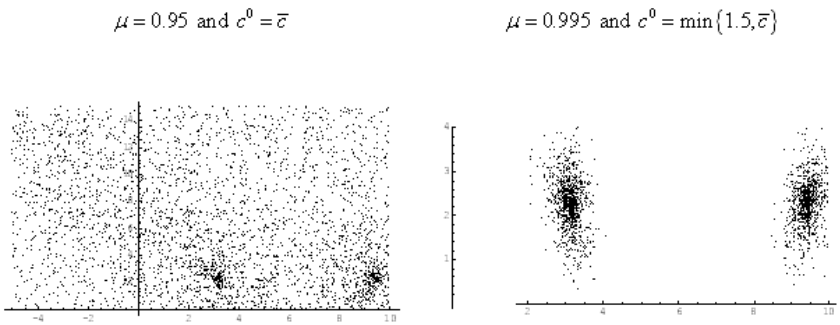

Figure 1: Accepted points in SSA algorithm

In all performed tests we verified that when $\mu = 0.995$ then $c^0 = 1.5$. This occurs because the value determined by the preliminary analysis is very

high. In this case, the initial control parameter is a small value and the SSA variant only accepts points close to the global maxima, as shown in Figure 1.

When the test function has more than one global maximum, the SSA variant was able to identify them.

Based on the same type of tests, identical conclusions can be drawn for the CSA variant.

For ASA, SALO and ASALO variants, we verified that c^0, ϵ and N_ϵ parameters are the ones that most influence the performance of the algorithms. In all tests, these variants have similar behavior: they rapidly converge to a point and provide a good approximation to an optimum.

Figure 2 presents the accepted points obtained by ASA and ASALO variants for the Branin test function.

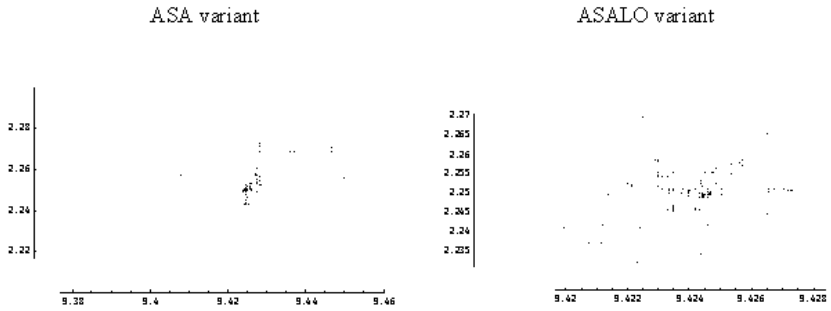

Figure 2: Accepted points in ASA and ASALO algorithms

We can see that these variants concentrate the search on the promising region of a global maximum. We also found that the ASA, SALO and ASALO variants quickly identify one promising region and that most of the accepted points are used to improve the precision of the approximation to a global maximum. However, we point out that these variants usually find only one global maximum.

4.2. Comparison of the presented variants

The computational experiences were mainly carried out to identify robustness, efficiency and the precision of the approximations to the global maximum.

The efficiency of the variants has been measured according to the number

of function evaluations N_{FE}, number of the accepted points N_{AP}, the final function average value g_m^* and the best final function value found g^*. When $^{(\#)}$ appears before the number of function evaluations, it means that # is the number of runs (out of 4) that did not converge to a global maximum. In these cases, the variant provided a local maximum.

In Tables 1 and 2 we present the obtained numerical results for SSA and CSA variants.

Test functions	n	N_{FE}	N_{AP}	g_m^*	g^*
B	2	100000	6493	$-0,3978880$	-0.3978874
GP	2	100000	10844	-3.0000513	-3.0000076
S	2	100000	2693	186.73047	186.730853
R_2	2	100000	9215	-0.0153402	-7.4×10^{-4}
R_4	4	$^{(2)}100000$	19072	-0.0142707	-0.0114511
Me_3	3	100000	9088	-1.4×10^{-6}	-6.6×10^{-7}
H_3	3	100000	7109	3.8627790	3.8627790
Ra_4	4	100000	10755	-9.7×10^{-4}	-9.5×10^{-4}

Table 1: Results of four runs of SSA algorithm

Test functions	n	N_{FE}	N_{AP}	g_m^*	g^*
B	2	24402	9435	-0.3978874	-0.3978874
GP	2	33769	15489	-3	-3
S	2	23562	5694	186.730909	186.730909
R_2	2	100000	23240	$-0,0241800$	-8.4×10^{-4}
R_4	4	$^{(1)}72762$	28109	$-0,0217699$	$-0,0011088$
Me_3	3	34587	13603	-6.3×10^{-8}	-1.5×10^{-8}
H_3	3	26565	10812	$3,86277820$	$3,8627821$
Ra_4	4	46452	16209	-1.4×10^{-7}	-2.6×10^{-8}

Table 2: Results of four runs of CSA algorithm

The SSA variant requires a high number of function evaluations always reaching the maximum value allowed. This fact indicates that the variant intensely searches on the feasible set. In problem R_4, two runs converge to a local maximum. The CSA variant requires a smaller number of function evaluations and provides better approximations to the global maximum. However, as expected, it accepts more points than the SSA variant. This increase in the number of accepted points improves the precision of the approximations to a global optimum.

Tables 3 and 4 present the numerical results provided by ASA and SALO variants.

Test functions	n	N_{FE}	N_{AP}	g_m^*	g^*
B	2	1000	311	-0.3978875	$-0,3978874$
GP	2	$^{(1)}1000$	306	$-3,0000004$	$-3,0000001$
S	2	1101	320	$186,730909$	$186,730909$
R_2	2	26015	20745	$-0,0394671$	$-0,0049240$
R_4	4	$^{(2)}64262$	4131	$-0,0362470$	$-0,0230065$
Me_3	3	1000	122	-2.8×10^{-8}	-5.3×10^{-10}
H_3	3	$^{(1)}2068$	374	$3,8627819$	$3,8627821$
Ra_4	4	2680	174	-9.4×10^{-7}	-8.3×10^{-8}

Table 3: Results of four runs of ASA algorithm

Test functions	n	N_{FE}	N_{AP}	g_m^*	g^*
B	2	43271	322	$-0,3978874$	$-0,3978874$
GP	2	50004	281	$-3,0000002$	-3
S	2	53016	318	$186,730908$	$186,730909$
R_2	2	43363	203	-2.8×10^{-6}	-1.3×10^{-7}
R_4	4	$^{(2)}100000$	188	-0.0397128	-0.0199250
Me_3	3	40221	87	-1.7×10^{-8}	-4.7×10^{-11}
H_3	3	67699	296	$3,8627814$	$3,8627820$
Ra_4	4	79033	96	-2.4×10^{-6}	-2.9×10^{-7}

Table 4: Results of four runs of SALO algorithm

When we compare these results with the CSA variant results, we may conclude that ASA variant does not improve the precision of the approximations. However, the ASA variant drastically reduces the number of function evaluations and of accepted points.

In some problems, SALO variant provides better approximations to a global maximum than the previous variants. This was already expected since this variant incorporates a local search procedure. In terms of accepted points, ASA and SALO have similar behavior, except for the problems R_2 and R_4 where SALO variant has the better results. Due to the local search procedure, SALO variant requires a high number of function evaluations. ASA variant is very fast to identify the region where a maximum is, so needing a fewer number of function evaluations. This is probably the reason why the ASA variant is not able to identify the global maximum and converges to a local one.

Finally, Table 5 presents the numerical results obtained by the ASALO variant.

The ASALO variant improves the precision of the approximations to a global maximum for all test functions, except in the R_4 and Ra_4 problems.

Test functions	n	N_{FE}	N_{PA}	g_m^*	g^*
B	2	15531	284	$-0,3978874$	$-0,3978874$
GP	2	15944	285	-3	-3
S	2	21527	374	$186,730907$	$186,730909$
R_2	2	16671	294	-2.8×10^{-6}	-8.8×10^{-8}
R_4	4	$^{(1)}40923$	577	$-0,0265422$	$-0,0049288$
Me_3	3	5717	98	-4.0×10^{-9}	-2.6×10^{-11}
H_3	3	15237	260	$3,8627815$	$3,8627821$
Ra_4	4	10293	143	-1.6×10^{-6}	-5.5×10^{-8}

Table 5: Results of four runs of ASALO algorithm

When compared with ASA, the ASALO variant needs more function evaluations although fewer than the remaining variants. In terms of accepted points, ASA, SALO and ASALO variants have similar behavior. Of all the presented variants, SALO and ASALO were the best ones as far as the number of accepted points is concerned.

The ASALO and CSA variants obtained so far the best approximations to the global maximum. In particular, the CSA variant gives better results than ASALO variant in two test function, R_2 and R_4. In the other test functions, ASALO variant produces better or equal approximations than the remaining variants. ASALO and CSA variants reached the best final function average value and both have only one run that does not converge to a global maximum. Besides ASA, the ASALO variant also requires a small number of function evaluations.

5. Conclusions

We propose a new variant of the SA algorithm, herein denoted by ASALO, combining the adaptive simulated annealing and a local search procedure with a reflection technique which aims to generate feasible points. The new algorithm, together with other four well-known variants of the SA algorithm were tested with a set of standard test functions in order to analyze their performances.

The numerical results indicate that the variants that are more effective, in terms of number of function evaluations, are ASA and ASALO. When we compare the number of accepted points, ASA, SALO and ASALO variants have similar behavior. The ASALO variant provides the best maximum function value. When the problem has more than one global optimum, SSA and CSA were able to recognize more than one solution. However, ASA, SALO and ASALO variants usually identify only one solution.

We propose that either the CSA or the SSA should be used when more

than one global maximum have to be identified. If only one global maximum is requested, than ASA is more efficient as far as the number of function evaluations is concerned. To obtain the best function value and a high percentage of solved problems, the ASALO variant seems slightly superior.

6. Bibliography

[1] Bohachevsky, I. O., Johnson, M. E. and Stein, M. L. (1986). *Generalized Simulated Annealing for Function Optimization.* Technometrics 28, no. 3, 209-217.

[2] Corana, A., Marchesi, M., Martini, C. and Ridella, S. (1987). *Minimizing Multimodal Functions of Continuous Variables with the simulated Annealing Algorithm.* ACM Transactions on Mathematical Software 13, no 3, 262-280.

[3] Dekkers, A. and Aarts, E. (1991). *Global Optimization and Simulated Annealing.* Mathematical Programming 50, 367-393.

[4] Desai, R. and Patil, R. (1996). *SALO: Combining Simulated Annealing and Local Optimization for Efficient Global Optimization.* Proceedings of the 9th Florida AI Research Symposium FLAIRS - 96, 233-237.

[5] Goffe, W. L., Ferrier, G. D. and Rogers, J. (1994). *Global Optimization of Statistical Functions with Simulated Annealing.* Journal of Econometrics 60, 65-99.

[6] Ingber, L. (1989). *Very Fast Simulated Re-Annealing.* Mathematical and Computer Modelling 12, no. 8, 967-973.

[7] Ingber, L. (1993). *Simulated Annealing: Practice versus Theory.* Mathematical Computer Modelling 18, no. 11, 29-57.

[8] Ingber, L. (1996). *Adaptive Simulated Annealing (ASA): Lessons Learned.* Control and Cybernetics 25, no. 1, 33-54.

[9] Ingber, L. and Rosen, B. (1992). *Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison.* Mathematical Computer Modelling 16, no. 11, 87-100.

[10] Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C. (1991). *Optimization by Simulated Annealing: An Experimental Evaluation; part II, graph coloring and number partitioning.* Operations Research 39, no. 3, 378-406.

[11] Kirkpatrick, S., Gelatt, C. D. Jr. and Vecchi, M. P. (1983). *Optimization by Simulated Annealing.* Science 220, no. 4598, 671-680.

[12] Van Laarhoven, P. J. M. and Aarts, E. H. L. (1987). *Simulated Annealing: Theory and Applications.* Mathematics and Its Applications. Kluwer Academic Publishers.

[13] Niu, X. (1999). *An Integrated System of Optical Metrology for Deep Sub-Micron Lithography.* Ph.D Thesis. University of California.

[14] Romeijn, H. E. and Smith, R. L. (1994). *Simulated Annealing for Constrained Global Optimization.* Journal of Global Optimization 5, 101-126.

[15] Romeijn, H. E., Zabinsky, Z. B., Graesser, D. L. and Neogi, S. (1999). *New Reflection Generator for Simulated Annealing in Mixed-Integer/Continuous Global Optimization.* Journal of Optimization Theory and Applications 101, no. 2, 403-427.

[16] Szu, H. and Hartley, R. (1987). *Fast Simulated Annealing.* Phys Lett A 122, no. 3-4, 157-162.

[17] Tsallis, C. and Stariolo, D. A. (1996). *Generalized Simulated Annealing.* Phys. Lett A 233.