Coupled Simulated Annealing for Continuous Global Optimization

Samuel Xavier-de-Souza[1], Johan A.K. Suykens[1], Joos Vandewalle[1], and Désiré Bollé[2]

[1] Katholieke Universiteit Leuven
Department of Electrical Engineering (ESAT-SCD/SISTA)
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
[samuel.xavierdesouza,johan.suykens,
joos.vandewalle]@esat.kuleuven.be

[2] Katholieke Universiteit Leuven
Institute for Theoretical Physics
Celestijnenlaan 200D, B-3001 Leuven, Belgium
desire.bolle@fys.kuleuven.be

Abstract

We propose a class of algorithms for continuous global optimization of multi-modal problems. The new class of Coupled Simulated Annealing (CSA) algorithms is characterized by an ensemble of parallel optimization processes which are coupled to each other. Each process is basically a Simulated Annealing (SA) process with novel acceptance probability functions, which are responsible for the coupling among the optimization processes. The new acceptance functions can be considered as a generalization of classical SA acceptance probabilities. We characterize the coupling by the form of the acceptance functions and a term that is defined by a function of the energies of all current states in the different processes. In order to illustrate the class of algorithms described by CSA, we present three example methods. Additionally, we demonstrate how coupling can be used to steer the overall performance of the optimization by adaptively adjusting the acceptance temperature in a very simple way. Results for the three proposed example algorithms lead to the conclusion that by choosing the right coupling, considerably better results can be achieved w.r.t. other CSA schemes and mainly w.r.t. classical Parallel SA.

Index Terms

global continuous optimization, simulated annealing, parallel simulated annealing, coupling.

I. Introduction

There exist a wide range of problems for which continuous optimization occur. These problems are spread among many fields and include both the academic and industrial world. Challenges are in multi-modal problems which are in fact often hard non-convex problems with many local optima. For such problems, many global optimization techniques have been developed, often with several heuristics. While being able to escape from multiple local minima, the strongest drawback of global optimization methods is the large number of cost function evaluations required to obtain a global optimum solution. This problem has been undertaken by the development of many fast global optimization techniques where local and global procedures are often combined in order to obtain a fast convergence [5]. Not surprisingly, fast convergence introduces a trade-off between speed and quality of solutions. Fast global techniques become more frequently trapped in poor local minima. A challenge of this field is to find a good trade-off for the given optimization problem.

Among several other global approaches, Simulated Annealing (SA) is a technique originally based on the thermodynamical annealing process. Physical annealing is modeled or simulated in software by Monte Carlo techniques resulting in an efficient computational algorithm that is widely used nowadays to optimize many different problems [6], [9], [17]. In past years, several SA versions were developed [5], [10], [12], [19]. Every version introduces a different level of trade-off between speed of convergence and quality of solution. Some methods achieve extra levels of speed of convergence by introducing parallelism into the originally strictly sequential SA algorithm [2], [3], [6], [8], [11], [16]. This of course at the cost of more physical resources. The two-side trade-off then becomes a three-side trade-off:

$$\begin{array}{c}\text{Quality of solution}\\ \swarrow\quad\searrow\\ \text{Physical resources} \leftrightarrow \text{Speed of Convergence}\end{array}$$

There are many classifications made for the different types of Parallel SA algorithms [2], [7], [13], [16]. It is not the purpose of this paper to classify the many existing versions, but the reader may refer to [16] for a recent discussion. However, as far as it is relevant for this scope, it can be said that there exists a more general classification concerning *non-sequential* algorithms which categorize Ensemble [11] and Parallel SA algorithms. In Ensemble SA, a group of SA processes move in the search space without interaction between solutions. The ensemble moves as a whole

and the decision to make a move is taken based on the average cost of the group. Conversely, in Parallel SA, the concurrent processes move independently with the objective of minimizing the cost of the best solution, which is shared in one way or another among the processes either synchronously or asynchronously. In this paper, we present a new class of algorithms, called Coupled Simulated Annealing (CSA), that can not be fully described by Ensemble SA neither by Parallel SA. In fact, CSA may be considered as a mixture of Ensemble and Parallel SA. The main principles are discussed below.

The focus of developing many accelerated global optimization methods seems to have been on increasing the optimization speed. In order to avoid undesirable local optima traps, we propose here to tackle the problem from a different angle. By turning the focus to the quality of the solution, we expect our method to escape from local optima without compromising too much the speed of convergence. To better understand the principles that are used to achieve our goal, consider the work of Suykens et al. [14]. This showed that coupling among local optimization processes can be used to help gradient optimization methods to escape from local optima in non-convex problems. Here, with the objective of increasing the quality of the final solution, we propose the use of coupling in a global optimization method like SA. Additionally, by designing a coupling mechanism with minimal communication, these coupling algorithms can be implemented very efficiently in parallel computer architectures.

It is then important to find ways to enhance the final solution of SA algorithms. Obviously this can be done by designing better acceptance probabilities; however, the question *How?* still persists. With CSA, we propose in this paper a new form for the acceptance probability functions. The idea is to allow several concurrent SA processes with acceptance probability defined by a function that depends on a coupling term. This term is defined by a function of all energies of the current states in each concurrent process. The form of the acceptance function and the coupling term attached to it define the type of coupling. Like in Ensemble SA, there exist no iterations between the solutions; furthermore, like in Parallel SA, the different optimization processes move independently, but under the influence of coupling in the acceptance probability function.

CSA does not define a single algorithm but rather a class of algorithms. An algorithm of this class is defined by the coupling applied to it. Moreover, it is seen here that this class appears to be an extended class of SA algorithms. Depending on the type of coupling, different search features may emerge from the optimization. Features like localized search in lower energy regions of the

search space can be combined with a more global search in higher energy regions.

In order to illustrate the effect of coupling in SA-like algorithms and to stress the importance of its design, we present examples of three different coupling schemes that yield three CSA algorithms. It is also presented here how coupling can contribute to a control rule for the variance of the acceptance probabilities in order to relax the importance of initial parameters and steer the optimization to a quasi-optimal run. Furthermore, the advantages of parallel implementations are also discussed here. Besides being intrinsically parallel, CSA needs little communication overhead and in contrary to Parallel SA, this overhead does not scale with the dimension of the problem.

This paper is organized as follows. In Section II, the general principles of our proposed approach are explained. In order to demonstrate the effect of coupling, we propose three CSA algorithms in Section III. In Section IV, one of the algorithms is used to demonstrate how the coupling can be used to steer the optimization to a quasi-optimal run. In Section V, we discuss parallel implementations. Section VI shows experimental results with the three proposed coupling methods and comparison with a parallel SA algorithm. We also illustrate the results of the variance control. Finally, conclusions are drawn in Section VII.

II. Coupled Simulated Annealing: General Principles

Importance Sampling, the main principle underlying classical SA, has been used in statistical physics to selectively, rather than randomly, choose sample states of a particle system model in order to efficiently estimate some physical quantity related to the system. Random sampling of these states is very inefficient once in these systems only a few low-energy states carry most of the relevant information. Importance sampling cares for a rejection/acceptance mechanism of sampled states in order to probabilistically favor states with lower energies. The well known Metropolis algorithm was the first to use the idea to estimate these quantities efficiently. Also, this algorithm complies with the principle of *Detailed Balance*[1], which gives a sufficient condition to test the validity of Monte Carlo schemes. It states that

$$\frac{P(x \to y)}{P(y \to x)} = \frac{\exp(-E(y)/T)}{\exp(-E(x)/T)}, \tag{1}$$

[1]Intuitively, Detailed Balance ensures that the balance between the probability of 'leaving' a given state and arriving in it from another state holds overall and individually for any pair of states.

where $P(x \to y)$ is the transition probability for the system to go from the current state x to a candidate state y, $\forall\ P(y \to x) \neq 0$, with T being a fixed temperature and the quantities $E(x)$ and $E(y)$ denoting the energy of the states x and y, respectively. Transition probabilities can be subdivided into the product of a generation or selection probability and an acceptance probability, i.e. $P(x \to y) = G(x \to y)A(x \to y)$ with G and A denoting generation and acceptance probabilities, respectively. If G is chosen to be equally alike for all states, i.e. $G = 1/n$, with n denoting the number of all possible states, (1) can be reduced to

$$\frac{A(x \to y)}{A(y \to x)} = \frac{\exp(-E(y)/T)}{\exp(-E(x)/T)}. \tag{2}$$

Many acceptance probability functions for SA were derived according to (2), including the Metropolis rule. Another very common function, which also satisfies (2), is

$$A(x \to y) = \frac{1}{1 + \exp\left(\frac{E(y)-E(x)}{T_k}\right)}, \tag{3}$$

where T_k is the temperature at iteration k. This function is used to exemplify the proposed methodology in the sequel of this paper. Observe that since the probing state y is randomly chosen, the only information about the status of the system that is taken into account, when deciding whether to accept or not the new state with (3), is the energy of the current state $E(x)$.

In this paper, we propose a new form for the acceptance probabilities for an ensemble of optimizers. This approach considers several current states which are coupled together by their energies in the acceptance probability function. The motivation to create coupled acceptance functions which comprise the energy of many current states, or solutions, is in fact to generate more information when deciding to accept less favorable solutions in a global optimization process.

Besides proposing a new class of acceptance probability functions which couples a set of SA processes, we also observe here that this class of functions is in fact a generalization of existing SA acceptance functions. Note that by multiplying both numerator and denominator of the right hand side in (3) with $\exp(\frac{-E(y)}{T_k})$, it results in the following equivalent equation

$$A(x \to y) = \frac{\exp\left(\frac{-E(y)}{T_k}\right)}{\exp\left(\frac{-E(y)}{T_k}\right) + \exp\left(\frac{-E(x)}{T_k}\right)}. \tag{4}$$

Consider now a physical system with only two states. The probability of this system being in each of its two states is given by the Boltzmann factor

$$P_i = \frac{\exp\left(\frac{-E_i}{k_b T}\right)}{Z},$$

where Z is called the partition function of the system and is given by

$$Z = \sum_{i=1}^{2} \exp\left(\frac{-E_i}{k_b T}\right),$$

where k_b is the Boltzmann constant, and E_i denotes the energy of the i-th state. Note that the acceptance probability function (4), for a given probing solution y and fixed temperature, can in fact be approximated by the Boltzmann probability for the two-states system. Therefore, a Simulated Annealing process can be approximated by the modeling of a two-state particle system with a variable probing state energy. In order to couple many SA processes, we propose to model the ensemble of process but a particle system with many states. Being x_i one of many states and y_i the corresponding probing state, we can achieve this modeling by inserting more current states to (4) by considering the sum over a set of current states $x \in \{x_1, x_2, \cdots, x_m\}$ within the term $\exp\left(\frac{-E(x)}{T_k}\right)$,

$$A(x_i \to y_i) = \frac{\exp\left(\frac{-E(y_i)}{T_k}\right)}{\exp\left(\frac{-E(y_i)}{T_k}\right) + \sum\limits_{x \in \{x_1, x_2, \cdots, x_m\}} \exp\left(\frac{-E(x)}{T_k}\right)}. \tag{5}$$

This is a typical example of an acceptance function for CSA, which is based on (4). Observe that if $m = 1$, this generalized equation is reduced again to (4). With this example we show that (5) is a generalization of (4). However, this is not a general proof that CSA is an extended class for SA since other SA acceptance functions exist.

More generally, each optimization process in CSA, i.e. the algorithmic steps involving generation and acceptance of a single current state, is performed separately. This process behaves for each current state as a single classical SA process. In fact, the only difference between such a process and a SA process is held on the acceptance probability. While in SA this probability, $0 \leq A(x, y) \leq 1$, is a scalar function for every $x, y \in \Omega$, with Ω denoting the set of all possible states, in CSA it is a scalar function $0 \leq A_\Theta(\gamma, x_i, y_i) \leq 1$ for every $x_i \in \Theta$, $y_i \in \Omega$, and $\Theta \subset \Omega$, with x_i and y_i being current and probing states, respectively. The set Θ, that we propose as the

set of current states, is denoted as $\Theta \equiv \{x_i, i = 1, \cdots, m\}$ throughout the paper. Given one of its elements, the decision about swapping the element with a probing state which is outside of the set Θ depends on the given element, on the probing state, and also on the coupling term γ, which is a function of the energy of the elements in Θ, $\gamma = f\left[E(x_1), E(x_2), \cdots, E(x_m)\right]$. The general difference between classical SA and CSA acceptance processes is illustrated in Fig. 1.

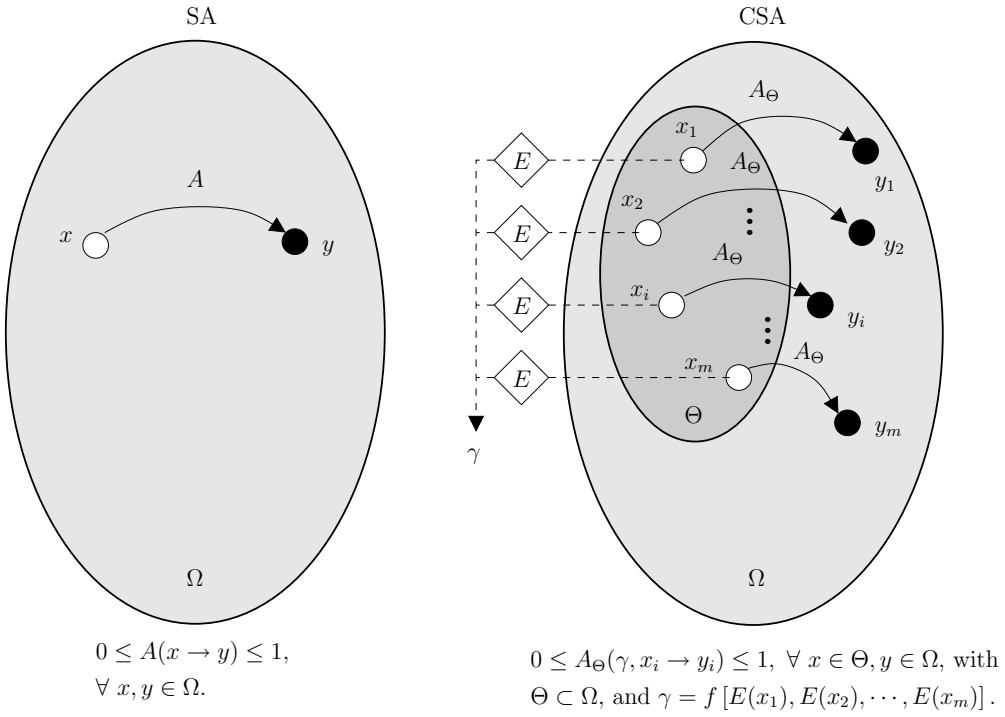


Fig. 1. The general difference between SA and CSA lies in the acceptance process. While SA only considers the current solution x for the acceptance decision of the probing state y, CSA considers many current states in the set Θ, which is a subset of all possible solutions Ω, and accepts each probing state y_i based not only on the corresponding current state x_i but by considering also the coupling term γ, which depends on the energy of all other elements of Θ.

Like in classical SA, in CSA an acceptance probability function can assume different forms. Besides what was mentioned above, these functions also need to inherit specific properties where the most desirable one is the steering of the states to low-energy regions. Compliance to detailed balance is also one of the most common desired features for SA acceptance functions, especially when the target of the simulation is the evaluation of statistical properties of particle systems. For optimization, however, this property may not be essential. In the three example methods that are given here, only one is satisfying the property of detailed balance.

III. Examples of Coupled Simulated Annealing Methods

In this section, three CSA example methods are shown. It is clarifying to mention that many others are possible and that these examples do not stand alone. What follows is a description of a general algorithm that is the basis for the three different coupling schemes illustrated next. Let Θ be a set containing m current solutions, let x_i be the ith element of this set, and y_i a corresponding probing solution. Let E be the cost, or energy to be minimized, associated to a given solution, and let γ be the coupling term as a function of the current states' energy. T_k and T_k^{ac} denote the temperatures of the generation and acceptance processes, respectively, at the iteration k. The following algorithm can now be formulated:

Algorithm 1

1) *Initialization:* assign random initial solutions to Θ; assess the costs $E(x_i)$, $\forall\ x_i \in \Theta$, and evaluate the coupling term γ; set initial temperatures $T_k = T_0$ and $T_k^{ac} = T_0^{ac}$; set the time index $k = 0$;
2) *Generate* a probing solution y_i for each element of Θ according to $y_i = x_i + \varepsilon_i$, $\forall\ x_i \in \Theta$, where ε_i is a random variable sampled from a given distribution $g(\varepsilon_i, T_k)$; assess the costs for all probing solutions: $E(y_i)$, $\forall\ i = 1, \cdots, m$;
3) *Accept* solution y_i with probability $A_\Theta(\gamma, x_i \rightarrow y_i)$, $\forall\ x_i \in \Theta$, i.e. make $x_i := y_i$ only if $A_\Theta > r$, where r is a random variable sampled from a uniform distribution [0,1]; evaluate γ again; and go to step 2 for N inner iterations;
4) *Decrease temperatures* according to schedules $U(T_k, k)$, and $V(T_k^{ac}, k)$. Increment k;
5) *Stop* if stopping criterion is met, *otherwise go to step 2*.

Many SA convergence proofs [4], [5], [15], [19] were established by associating a given generating distribution $g(\varepsilon, T_k)$ with a generating temperature schedule $U(T_k, k)]$. Therefore, the choice for the distribution $g(\varepsilon, T_k)$ depends on the schedule $U(T_k, k)$. We have not investigated which $[g(\varepsilon, T_k), U(T_k, k)]$ pair is the best for CSA. In Section VI, we present our choice of $[g(\varepsilon, T_k)$, $U(T_k, k)]$, and $V(T_k^{ac}, k)$ for the experiments performed in this paper.

A. Method 1: Multi-state Simulated Annealing

This method is a direct generalization of the classical SA with acceptance probability driven by (3), or (4). As mentioned before, these acceptance functions can be approximated by the

modeling of a particle system with only two states. In order to generate the acceptance function for this CSA method, we add more states to the original function. Hence the name Multi-state Simulated Annealing (CSA-MuSA). When accepting a probing solution, the whole collection of states or current solutions is taken into account. The following equation illustrates this example's acceptance function, which is essentially (5) with the actual CSA notation

$$A_\Theta(\gamma, x_i \to y_i'') = \frac{\exp\left(\frac{-E(y_i'')}{T_k^{ac}}\right)}{\exp\left(\frac{-E(y^\star)}{T_k^{ac}}\right) + \gamma}, \tag{6}$$

where γ is given by

$$\gamma = \sum_{x_j \in \Theta} \exp\left(\frac{-E(x_j)}{T_k^{ac}}\right), \tag{7}$$

for all $y^\star = y_j'', j = 1, 2, \cdots, m$. This acceptance function makes the probability of accepting a probing solution inversely proportional to its energy. Observe that the coupling term γ here is given by the only term in A_Θ that is shared among all current states.

The coupling here makes the probability equal to the Boltzmann factor in a system with many states, e.g. the sum of the probabilities for the current states Θ and the probing solution y_i'' equals 1. Observe that this probability is not valid for all y_i'' together but independently, i.e. for all $y^\star = y_j'', j = 1, 2, \cdots, m$. The condition of detailed balance can not be satisfied here since (6) does not satisfy (2). An overview of the formulas is shown in Table I, row 2.

B. Method 2: Blind Acceptance

In this CSA method, the Boltzmann factor describes the probability of a system to *stay* in the current state upon generation of a probing state with higher energy. All lower energy states are accepted with probability equal to 1. The probability of accepting a less favorable state is proportional to the energy of the current state, i.e. a higher energy probing state is more frequently accepted at high energy current states while low energy current states are more preserved. The acceptance function is chosen as follows

$$A_\Theta(\gamma, x_i \to y_i) = 1 - \frac{\exp\left(\frac{-E(x_i)}{T_k^{ac}}\right)}{\gamma}, \tag{8}$$

Method	Acceptance function	Coupling term γ	Normalization
CSA-MuSA	$A_\Theta(\gamma, x_i \to y_i'') = \dfrac{\exp\left(\dfrac{-E(y_i'')}{T_k^{ac}}\right)}{\exp\left(\dfrac{-E(y^\star)}{T_k^{ac}}\right) + \gamma}$	$\sum_{x_j \in \Theta} \exp\left(\dfrac{-E(x_j)}{T_k^{ac}}\right)$	$\forall y^\star = y_j'', j = 1, 2, \cdots, m :$ $\sum_{x_j'' \in \Theta} A_\Theta(\gamma, x_i \to x_j'')$ $+ A_\Theta(\gamma, x_i \to y^\star) = 1$
CSA-BA	$A_\Theta(\gamma, x_i \to y_i) = 1 - \dfrac{\exp\left(\dfrac{-E(x_i)}{T_k^{ac}}\right)}{\gamma}$	$\sum_{x_j \in \Theta} \exp\left(\dfrac{-E(x_j)}{T_k^{ac}}\right)$	$\sum_{x_i \in \Theta} A_\Theta(\gamma, x_i \to y_i) = 1$
CSAM	$A_\Theta(\gamma, x_i \to y_i) = \dfrac{\exp\left(\dfrac{E(x_i)}{T_k^{ac}}\right)}{\gamma}$	$\sum_{x_j \in \Theta} \exp\left(\dfrac{E(x_j)}{T_k^{ac}}\right)$	$\sum_{x_i \in \Theta} A_\Theta(\gamma, x_i \to y_i) = 1$
PSA	$A(x \to y) = \dfrac{1}{1 + \exp\left(\dfrac{E(y) - E(x)}{T_k^{ac}}\right)}$	–	–

TABLE I

OVERVIEW OF A PARALLEL SIMULATED ANNEALING (PSA) ALGORITHM AND THREE CSA ALGORITHMS: MULTI-STATE SA (CSA-MUSA); BLIND ACCEPTANCE (CSA-BA); AND COUPLED SIMULATED ANNEALING MODIFIED (CSAM).

where γ is given by (7). The probability of accepting state y_i and therefore leaving state x_i is the probability of *not staying* in state x_i, and it does not depend on y_i itself. Hence this method is called CSA Blind Acceptance (CSA-BA).

Low energy states are accepting fewer uphill moves than high energy states. This causes a localized search on low energy states and a more global exploration on high energy states. Observe that the acceptance probability for higher energy solutions is independent from the probing solution's energy; therefore the name of the method. Also note that although this method presents a considerably different approach to CSA than the previous one, its acceptance function has the same coupling term as in CSA-MuSA. The coupling here ensures that the sum of the probabilities of the system to *stay* in any of the current solutions equals 1. Observe that detailed balance is also not satisfied here. An overview of the formulas can be seen in Table I, row 3.

C. Method 3: Coupled Simulated Annealing Modified

Both previously described methods incorporate two distinct search features. The first one manages to hold more knowledge of low energy regions of the cost function, whereas the second explores better unknown regions. In Coupled Simulated Annealing Modified (CSAM),

we combine both search strategies. In both previous examples of CSA, the Boltzmann factor composes the acceptance function. Here a similar function is used, representing the probability of *leaving* the current states upon an uphill move:

$$A_\Theta(\gamma, x_i \to y_i) = \frac{\exp\left(\frac{E(x_i)}{T_k^{ac}}\right)}{\gamma}. \tag{9}$$

Therefore, the sum of probabilities of *leaving* any of the current states equals 1. Like in the second example method, this one also performs Blind Acceptance via an acceptance probability independent of the probing solution's energy.

The coupling term γ here is given by

$$\gamma = \sum_{x_j \in \Theta} \exp\left(\frac{E(x_j)}{T_k^{ac}}\right). \tag{10}$$

Observe that although a positive signal for the energy is considered, this method satisfies the condition of detailed balance. Refer to Table I, row 4, for an overview of the formulas.

The effect of the coupling in this method has clear advantages w.r.t. the other two. Probabilistically at least one state is likely to change, ensuring global search even at very low temperatures. Additionally many states are allowed to have probabilities very close to zero, which generates knowledge for the decision of accepting or not uphill moves.

IV. Coupling, Temperature, and Variance

In thermodynamical systems, the temperature plays a significant role of control in determining the system's macro-state. The same is true for CSA modeling; however, here the temperature is not a unique factor. Since the number of current solutions m in an optimization problem can assume virtually any positive integer value, this number also plays a significant role. To illustrate this, consider two instances of CSAM, one with 2 current states and another with 10 current states subject to the same temperature. If all the current solutions have the same energy in both systems, the acceptance probability in the two-state system is $\frac{1}{2}$ while for the ten-states system it would be $\frac{1}{10}$. It is easy to see that this probability approaches zero very fast with an increasing value of m. Depending on the type of coupling, the number of elements of Θ can force the system to either have slow changes or have rapid changes, i.e. frequently, uphill moves are not allowed, or low energy states are not preserved. Considering a fixed temperature, a large value for

m would mean probabilities approaching $\frac{1}{m}$, while a small value would mean all but one of the probabilities approaching zero. Both cases are clearly undesirable when optimizing multi-modal problems. However, the right balance between temperature and the number of coupled optimizers can generate valuable knowledge for decisions on when to take uphill moves. Consider now a CSAM algorithm. Knowing that

$$\sum_{x_i \in \Theta} A_\Theta(\gamma, x_i \rightarrow y_i) \equiv \sum_{x_i \in \Theta} A_\Theta = 1,$$

the variance for A_Θ assumes the following form:

$$\begin{aligned} \sigma^2 &= \frac{1}{m} \sum_{x_i \in \Theta} A_\Theta^2 - \left(\frac{1}{m} \sum_{x_i \in \Theta} A_\Theta \right)^2 \\ &= \frac{1}{m} \sum_{x_i \in \Theta} A_\Theta^2 - \frac{1}{m^2}. \end{aligned} \tag{11}$$

By using the fact that

$$\frac{1}{m} \leq \sum_{x_i \in \Theta} A_\Theta^2 \leq 1,$$

it can be concluded that

$$0 \leq \sigma^2 \leq \frac{m-1}{m^2}.$$

The upper and lower bound of σ^2 correspond to two undesirable cases for the probabilities, i.e. probabilities approaching $\frac{1}{m}$, or all but one of the probabilities approaching zero. It remains to know which variance value is ideal for an optimal process, and whether it depends on the problem at hand. Once a reasonable desired value for σ^2 can be established, either analytically or experimentally, we can attempt to control the variance by modifying one of the external parameters, i.e. T_k^{ac} or m. Although the tuning of m is allowed, besides the physical incoherence, it would make parallelism unpractical. Moreover, temperature is inherently a good choice since it is traditionally the control parameter for SA optimization processes.

A. Variance Control

Classically, temperature is used as a control parameter for the SA algorithms. Often it is employed in a fixed schedule but adaptive schedules are not rare. In this approach the temperature is stated as an active control parameter for the variance among the probabilities of the different current solutions in a CSA algorithm, more precisely for CSAM.

The first step is to determine the desired value for the variance. It is clear that the two extreme values are not desired, thus the best value must lie somewhere in between. Although it was not possible to devise this value analytically, experiments indicated that values in the neighborhood of the maximum variance give good results. The control of the temperature can be done in the following manner:

$$\text{if } \sigma^2 < \ \sigma_D^2, \ T_k^{ac} = T_{k-1}^{ac}\,(1-\alpha)\,,$$
$$\text{if } \sigma^2 > \ \sigma_D^2, \ T_k^{ac} = T_{k-1}^{ac}\,(1+\alpha)\,,$$

where σ_D^2 is the desired variance value and α is the rate for increase or decrease of the temperature, typically in the range of $|0, 0.1]$. The complete algorithm with the variance control can now be stated as follows:

Algorithm 2

1) *Initialization:* assign random initial solutions to Θ; assess the costs $E(x_i)$, $\forall\ x_i \in \Theta$, and evaluate the coupling term γ; set initial temperatures $T_k = T_0$ and $T_k^{ac} = T_0^{ac}$; set the time index $k = 0$; set σ_D^2, e.g. $\sigma_D^2 = 0.99\left(\frac{m-1}{m^2}\right)$, and e.g. $\alpha = 0.05$;
2) *Generate* a probing solution y_i for each element of Θ according to $y_i = x_i + \varepsilon_i$, $\forall\ x_i \in \Theta$, where ε is a random variable sampled from a given distribution $g(\varepsilon_i, T_k)$; assess the costs for all probing solutions: $E(y_i)$, $\forall\ i = 1, \cdots, m$;
3) *Accept* solution y_i with probability $A_\Theta(\gamma, x_i \to y_i)$, $\forall\ x_i \in \Theta$, as in (9), i.e. make $x_i := y_i$ only if $A_\Theta > r$, where r is a random variable sampled from a uniform distribution [0,1]; evaluate γ again; and go to step 2 for N inner iterations;
4) *Adjust* acceptance temperature T_k^{ac} according to the following rules: if $\sigma^2 < \ \sigma_D^2$, $T_k^{ac} = T_{k-1}^{ac}(1-\alpha)$; if $\sigma^2 > \ \sigma_D^2$, $T_k^{ac} = T_{k-1}^{ac}(1+\alpha)$;
5) *Decrease generation temperature* according to a schedule $U(T_k, k)$, increment k;
6) *Stop* if stopping criterion is met, *otherwise go to step 2*.

V. Parallel Implementation

One of the most commonly used arguments by engineers and researchers to choose Genetic Algorithms (GA) rather than SA is the limited parallelization capabilities of SA algorithms. In fact, there exist many parallel SA versions [2], [3], [6], [8]. However, their effective gain with parallelism can only be seen in low and very low temperature regions of the annealing schedules. This is mostly due to the fact that SA is inherently a sequential algorithm. Moreover,

many parallel versions of SA seem to focus on accelerating convergence, rather than distributing processing while ensuring global search. There exists a trade-off between faster convergence and quality of solution, i.e. a trade-off between speed and global optimality. In CSA, we mainly aim at improving the quality of the final solution. This approach also happens to be highly suitable for parallelization.

While SA is inherently sequential, CSA is inherently parallel in every aspect. In CSA, every generation and acceptance process can be run separately in different CPUs. The effective necessary communication is due to the coupling term. This involves very limited communication when compared with other parallel approaches which communicate entire parameter vectors. Besides, transferring only the cost, or energy, does not scale with the dimension of the optimization problem. Finally, since each acceptance process accepts new current solutions not necessarily at the same time, the type of communication can be asynchronous. However, synchronous communication is clearly also not discarded.

The actual parallel implementation can be elaborated in many different ways, from distributed systems to multi-processor machines. Fig. 2 shows two typical examples of possible CSA implementations.

VI. Experiments and Results

The three CSA example methods are compared in this section. Additionally, a parallel version of the classical SA (PSA) [1] is included in the comparison for the purpose of reference. PSA features several sequential SA algorithms running in parallel and sharing the best current solution. As soon as one of the parallel instances finds a better solution, all the others are informed about the new current solution and proceed to the next generation step. Table I presents an overview of the 4 algorithms.

A. Test problems

A collection of multi-dimensional and multi-modal continuous functions were chosen from the literature to be used as test cases. These functions feature many local minima and therefore are regarded as being difficult to optimize [18], [20]. Table II gives a short description of all functions in the collection. Fig. 3 pictures the shape of the first problem in one and two dimensions.

#	Energy function E	source	Dim.	Range
1	$1 - \prod_{i=1}^{D} \text{sign}\left(\frac{\sin x_i}{x_i}\right) \left\lvert \left(\frac{\sin x_i}{x_i}\right)^{\frac{1}{4}} \right\rvert$	unknown	D	$[-50, 50]$
2	$\sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	[18]	D	$[-5.12, 5.12]$
3	$-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=i}^{D} x_i^2}\right) - \exp\left(\frac{1}{D} * \sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$	[18]	D	$[-32, 32]$
4	$\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[18]	D	$[-600, 600]$

TABLE II

COLLECTION OF MULTI-DIMENSIONAL AND MULTI-MODAL FUNCTIONS USED AS TEST CASES AND THEIR RESPECTIVE PARAMETER RANGE. THE GLOBAL OPTIMUM OF ALL PROBLEMS IS LOCATED AT THE ORIGIN WITH A CORRESPONDING ENERGY EQUAL TO ZERO.

B. Experimental Results

Fig. 4 shows statistical box-plots over 50 runs of the function #1 for each algorithm with dimension $D = 8$, $m = 8$ parallel processes, maximum number of function evaluation set to 20,000 per processor, and 500 steps per temperature. All four algorithms tested were subjected to the same temperature schedules ([15] for generation, and [4] for acceptance):

$$T_{k+1} = \frac{T_0}{k+1},$$

and

$$T_{k+1}^{ac} = \frac{T_0^{ac}}{\ln(k+1)};$$

hence, according to [15], the generation distribution is a Cauchy distribution

$$g(\varepsilon, T_k) = \frac{T_k}{(\varepsilon^2 + T_k^2)^{(D+1)/2}}.$$

Initial temperatures were chosen after an exhaustive search. For the purpose of this comparison, we used the maximum number of iterations as the stopping criterion. The same plot is shown in Fig. 5 to 7 for the functions #2 to #4, respectively.

It can be observed from Fig. 4 to 7 that the coupling algorithms performed most of the time better than PSA. There are also differences in performance among the coupling algorithms. While

the first two CSA methods show some performance inconsistency for different problems, CSAM performed always best. Therefore it is right to say that the coupling alone does not yield the desired results but the performance of coupling algorithms is strongly influenced by the coupling term and the type of coupling.

C. Results for CSA with Variance Control

The effects of the variance control of Section IV were analyzed using function #2. To better demonstrate this effect, results were generated for different dimensions and number of optimizers. The number of optimizers were chosen to be equal to the problem dimension with values ranging from 5 to 30. Three different configurations were used, two of them using CSAM without the variance control and one with it. These results can be seen in Fig. 8.

Observe in Fig. 8 that configuration (c) follows very closely (a). The initial parameters T_0 and T_0^{ac} for (a) were established after an exhaustive search. Results for (b) and (c) were generated with T_0 the same as in (a) but T_0^{ac} was set to the average value of the cost function for 100 random solutions times a random value sampled from the interval $[0, 2]$ uniformly distributed. Although (c) was initialized with random values for T_0^{ac}, the variance control was able to steer the optimization process to quasi-optimal runs. In fact, configurations (a) and (c), unlike (b), were able to find at the end of the optimization the basin of the global optimum, i.e. a location in the problem space from where the global optimum can easily be found with a simple local optimization method. In conclusion, the coupling can also be used to approximate the ideal annealing temperature, thus reducing the influence of this initial parameter on the final search result, independently of the number of optimizers m.

In Fig. 9, the reader can find a plot of a typical run of the CSAM algorithm with variance control for the test function #2. For this plot, $D = 5$ and also the number of optimization processes $m = 5$. The values for the initial parameters are the same as for configuration (c) in Fig. 8. Although it might not be clear in the figure, the overall best solution is not retained by only a single optimizer, but switches many times from one to another. This is a clear effect of the coupling on the balance of the acceptance probabilities, which also provides an equilibrium between global and local search by avoiding concentration of the optimizers in both low and high energy regions.

VII. Conclusions

Coupling of SA processes has been introduced in this work as a new approach to solve hard continuous global optimization problems. The coupling is applied within the acceptance probability function. Coupled Simulated Annealing is not considered as a single algorithm but rather as a class of algorithms defined by the form of the acceptance function and the coupling term. Supposedly, this class is also an extended class of SA algorithms. By analyzing the results of the methods presented here, we conclude that the coupling of SA processes increases their exploration capabilities. We can also conclude that although there might exist many possible coupling schemes, the right design can be essential to boost such capabilities. In addition, CSA processes can run on parallel systems with little communication overhead, since only the coupling term needs to be distributed or evaluated among the processing nodes. Besides improving convergence rates, coupling is also useful to reduce the influence of the initial parameters and steer the overall optimization process to quasi-optimal runs. This is possible due to a simple but efficient variance control that uses the acceptance temperature as the control parameter to keep the variance of the acceptance probabilities away from undesired values. Finally, the algorithms described here are based on classical SA, the addition of many features developed along the past years to improve SA, like individual parameter *sensitivity* as in [5], can possibly improve even further the results shown here.

Acknowledgments Research supported by: • Research Council KUL: GOA-Mefisto 666, GOA-AMBioRICS, BOF OT/03/12, several PhD/postdoc & fellow grants, Center of Excellence Optimization in Engineering; • Flemish Government: ○ FWO: PhD/postdoc grants, G.0407.02, G.0080.01, G.0211.05, G.0499.04, G.0226.06, research communities (ICCoS, ANMMM); ○ Tournesol 2005 • Belgian Federal Science Policy Office IUAP P5/22 (‘Dynamical Systems and Control: Computation, Identification and Modelling’). J. Suykens is an associate professor, D. Bollé and J. Vandewalle are full professors, all with K.U. Leuven Belgium. DB thanks Marcus Müller for useful correspondence, and SXS thanks João Ramos for insightful discussions.

REFERENCES

[1] E. H. L. Aarts and J. H. M. Korst, *Simulated Annealing and Boltzmann Machines*. New York: Wiley (Interscience), 1989.

[2] A. Bevilacqua, "A Methodological Approach to Parallel Simulated Annealing on an SMP System," *J. Parallel and Distributed Computing*, vol. 62, pp. 1548–1570, 2002.

[3] H. Chen, N. S. Flann, and D. W. Watson, "Parallel Genetic Simulated Annealing: A Massively Parallel SIMD Algorithm," *IEEE Trans. Parallel and Distributed Systems*, vol. 9, no. 2, pp. 126–136, Feb 1998.

[4] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, Nov 1984.

[5] L. Ingber, "Very Fast Simulated Re-Annealing," *Journal of Mathematical Computer Modelling*, vol. 12, pp. 967–973, 1989.

[6] K. Kurbel, B. Schneider, and K. Singh, "Solving Optimization Problems by Parallel Recombinative Simulated Annealing on a Parallel Computer - An Application to Standard Cell Placement in VLSI Design," *IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 28, no. 3, pp. 454–461, Jun 1998.

[7] S. Lee and K. Lee, "Synchronous and asynchronous parallel simulated annealing with multiple Markov chains," *IEEE Trans. on Parallel and Distributed Systems*, vol. 7, no. 10, pp. 993–1008, Oct 1996.

[8] S. W. Mahfoud and D. Goldberg, "Parallel Recombinative Simulated Annealing: A Genetic Algorithm," *Parallel Computing*, vol. 21, pp. 1–28, 1995.

[9] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "VLSI module placement based on rectangle-packing by the sequence-pair," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 12, pp. 1518–1524, Dec. 1996.

[10] S. Rajasekaran, "On Simulated Annealing and Nested Annealing," *J. Global Optimization*, vol. 16, pp. 43–56, 2000.

[11] G. Ruppeiner, J. M. Pedersen, and P. Salamon, "Ensemble approach to simulated annealing," *J. Phys. I*, vol. 1, pp. 455–470, 1991.

[12] P. Siarry, G. Berthiau, F. Durbin, and J. Haussy, "Enhanced Simulated Annealing for Globally Minimizing Functions of Many-Continuous Variables," *ACM Trans. Mathematical Software*, vol. 23, no. 2, pp. 209–228, Jun 1997.

[13] E. O. soğlu and L. Özdamar, "Parallel Simulated Annealing Algorithms in Global Optimization," *J. Global Optimization*, vol. 19, no. 1, pp. 27–50, Jan 2001.

[14] J. A. K. Suykens, J. Vandewalle, and B. De Moor, "Intelligence and cooperative search by coupled local minimizers," *Int. J. of Bifurcation and Chaos*, vol. 11, no. 8, pp. 2133–2144, Aug 2001.

[15] H. H. Szu and R. L. Hartley, "Fast Simulated Annealing," *Physics Letters A*, vol. 122, pp. 157–162, 1987.

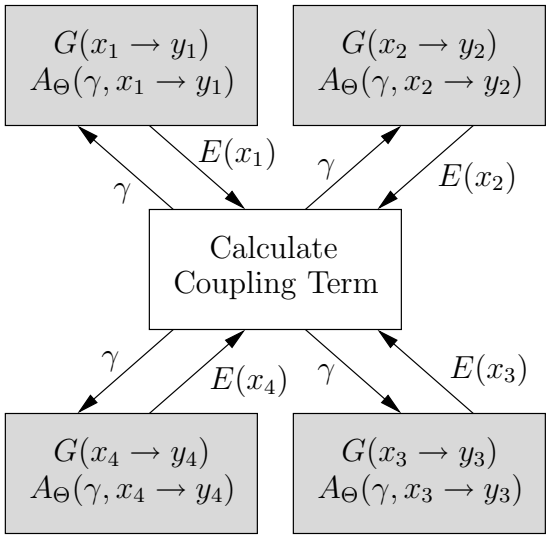
[16] D. Thompson and G. Bilbro, "Sample-sort simulated annealing," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 35, no. 3, pp. 625–632, Jun 2005.

[17] S. Xavier-de-Souza, M. E. Yalcin, J. A. K. Suykens, and J. Vandewalle, "Toward CNN Chip-specific robustness," *IEEE Trans. on Circuits and Systems-I*, vol. 51, no. 5, pp. 892–902, May 2004.

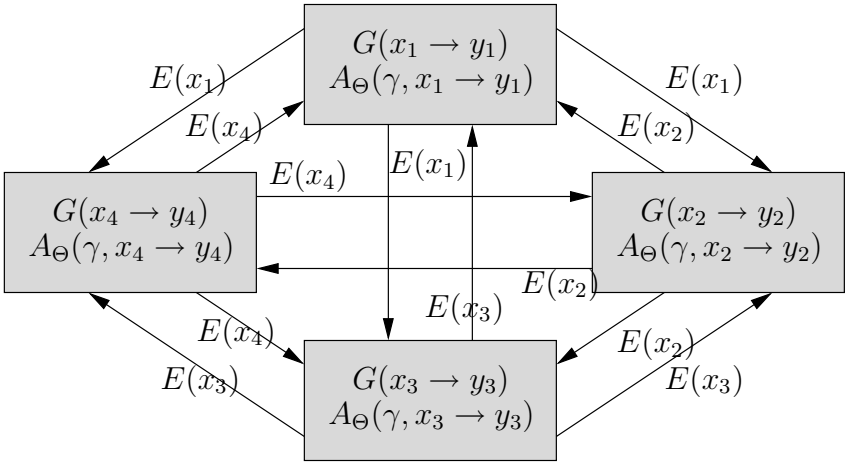
[18] Y. Xin, L. Yong, and L. Guangming, "Evolutionary Programming Made Faster," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, Jul 1999.

[19] X. Yao, "A New Simulated Annealing Algorithm," *Intern. J. Computer Math.*, vol. 56, pp. 161–168, 1995.

[20] T. Zhenguo and L. Yong, "A Robust Stochastic Genetic Algorithm (StGA) for Global Numerical Optimization," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 5, pp. 456–470, Oct 2004.



(a)



(b)

Fig. 2. Two typical examples of CSA parallel implementations: (a) Master-slave architecture; (b) Full connectivity. Each box in the figure can be implemented in an individual processing node.

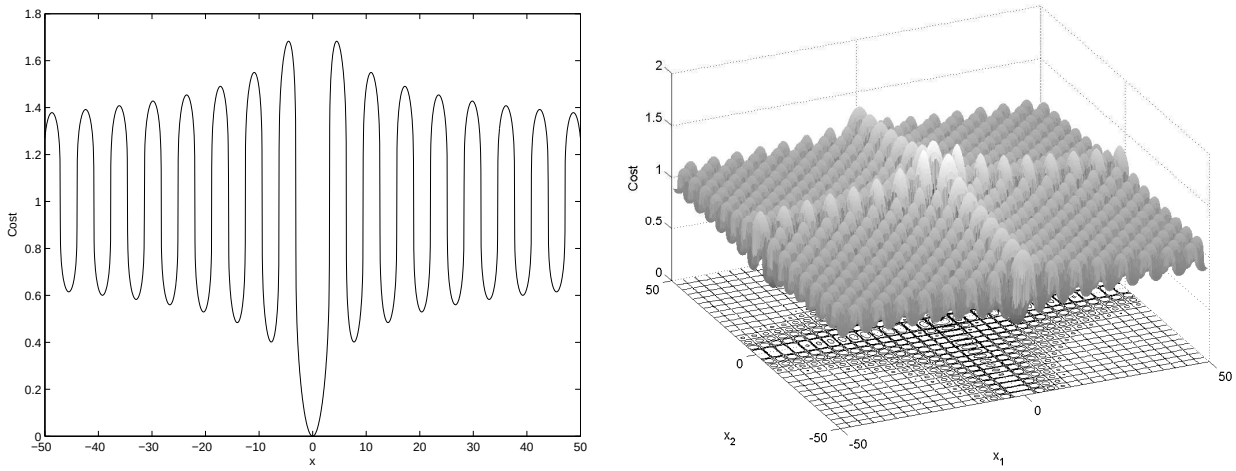


Fig. 3. Illustration of test problem #1 in 1 and 2 dimensions. A unique global optimum exists together with many local optima, which makes this type of functions difficult to optimize.

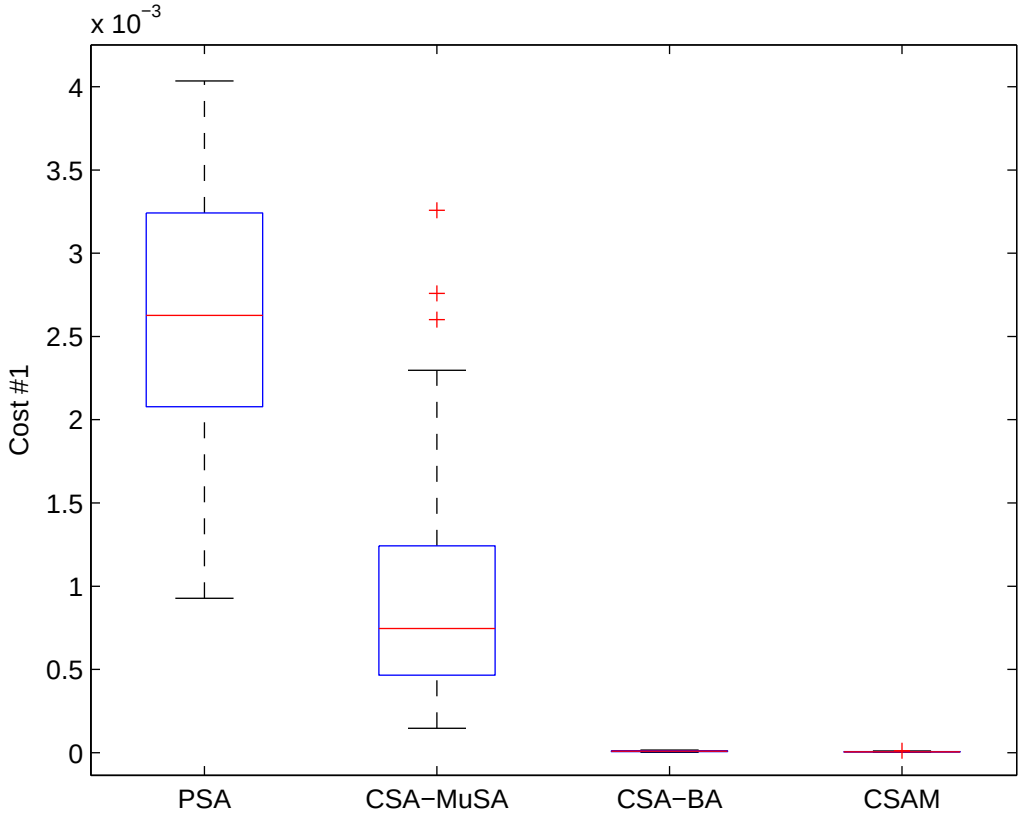


Fig. 4. Comparison among the 4 algorithms for function #1. For all algorithms, $m = 8$ parallel processes were used with $D = 8$, 500 steps per fixed temperature, the same cooling schedules and initial temperatures, and maximum number of cost-function evaluations equal to 20,000 per processing node.

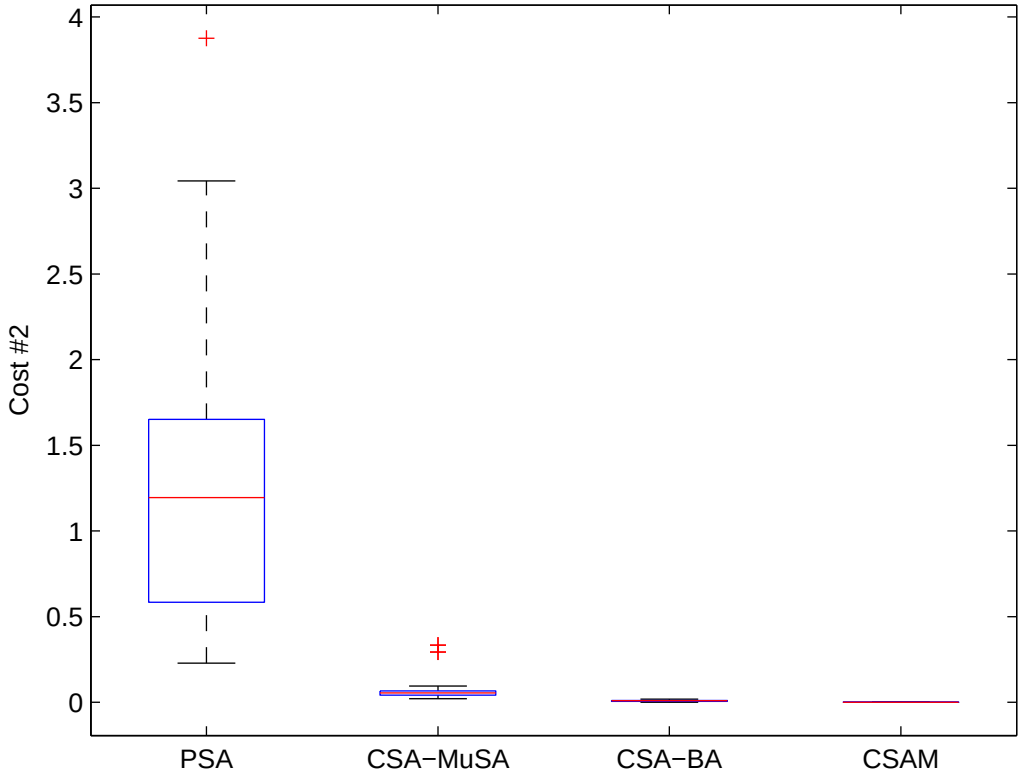


Fig. 5. Comparison among the 4 algorithms for function #2. For all algorithms, $m = 8$ parallel processes were used with D = 8, 500 steps per fixed temperature, the same cooling schedules and initial temperatures, and maximum number of cost-function evaluations equal to 20,000 per processing node.

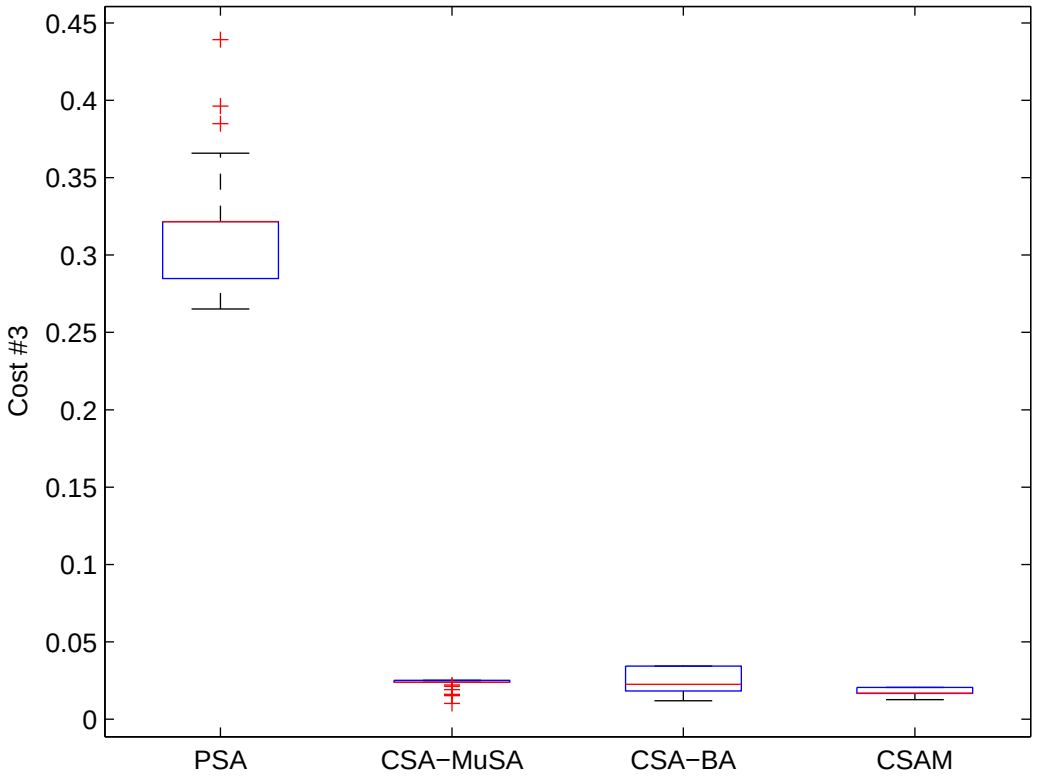


Fig. 6. Comparison among the 4 algorithms for function #3. For all algorithms, $m = 8$ parallel processes were used with D = 8, 500 steps per fixed temperature, the same cooling schedules and initial temperatures, and maximum number of cost-function evaluations equal to 20,000 per processing node.

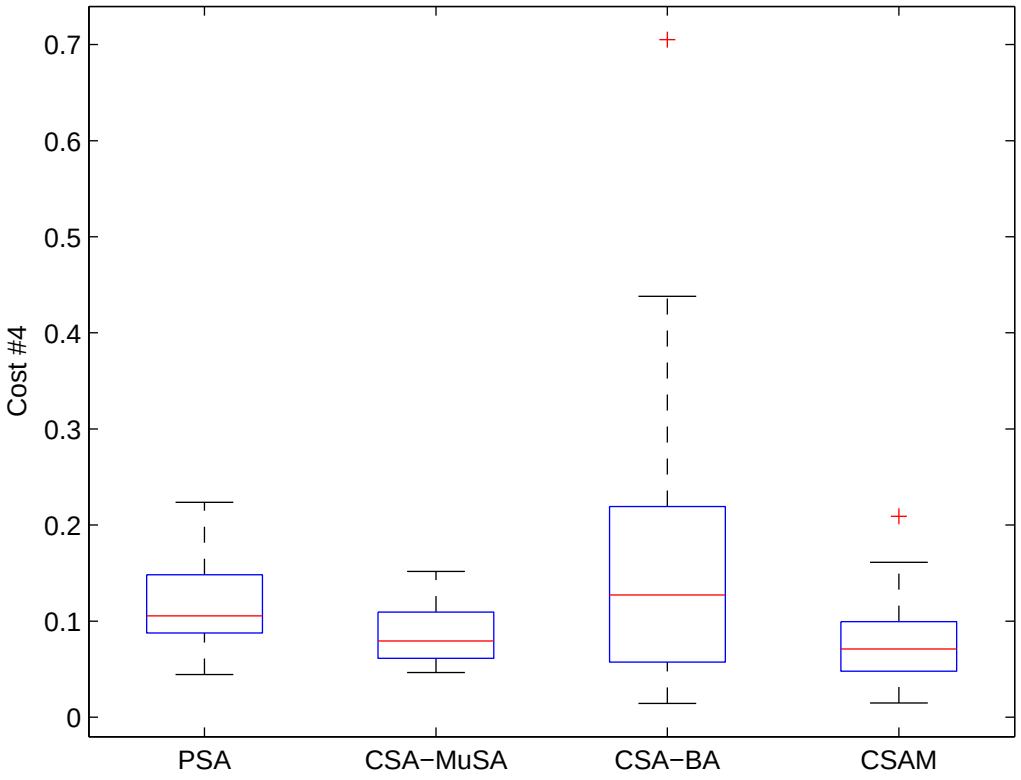


Fig. 7. Comparison among the 4 algorithms for function #3. For all algorithms, $m = 8$ parallel processes were used with D = 8, 500 steps per fixed temperature, the same cooling schedules and initial temperatures, and maximum number of cost-function evaluations equal to 20,000 per processing node.

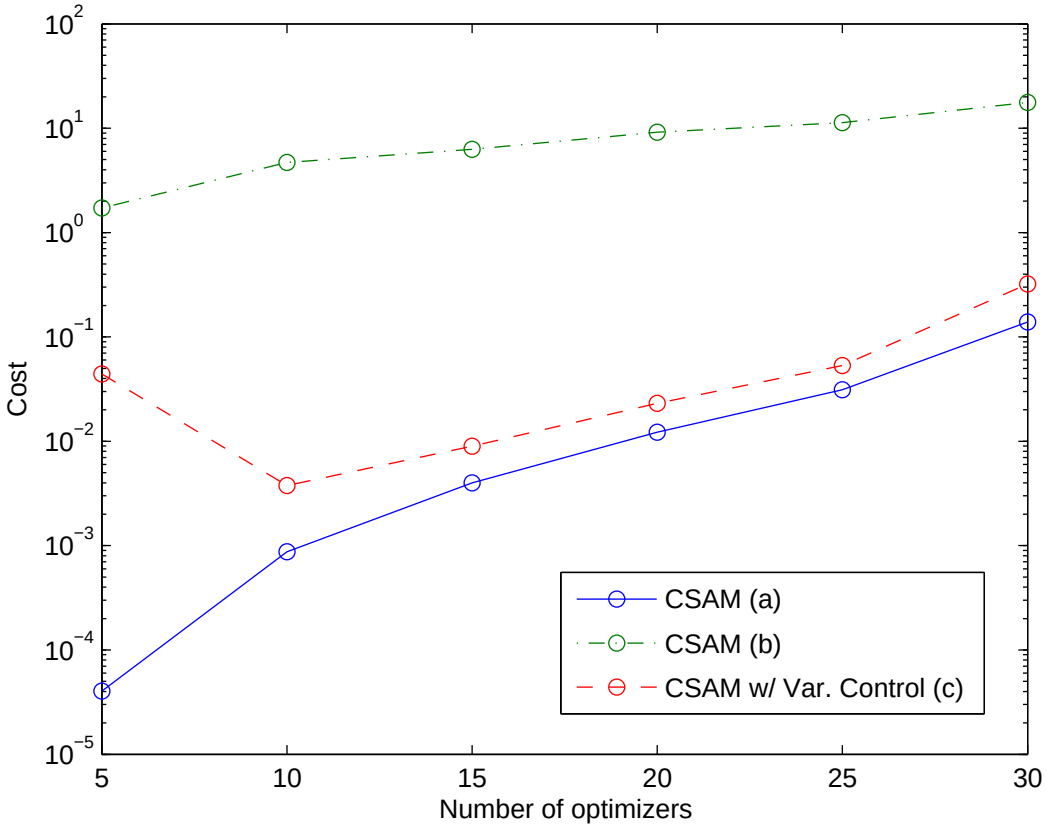


Fig. 8. Plot of 50 average runs for function #2 with $D = m$ chosen for 3 different configurations: (a) CSAM using best found initial parameters; (b) CSAM using T_0^{ac} equal to random values uniformly distributed between 0 and twice the mean value of the cost function; (c) CSAM using variance control with the same values for T_0^{ac} as used in (b). All configurations featured a maximum number of iterations per optimizer equal to 100,000, with 500 steps per temperature and $T_0 = 0.075$. For configuration (c), the value for $\sigma_D^2 = 0.99\left(\frac{m-1}{m^2}\right)$ and $\alpha = 0.05$.

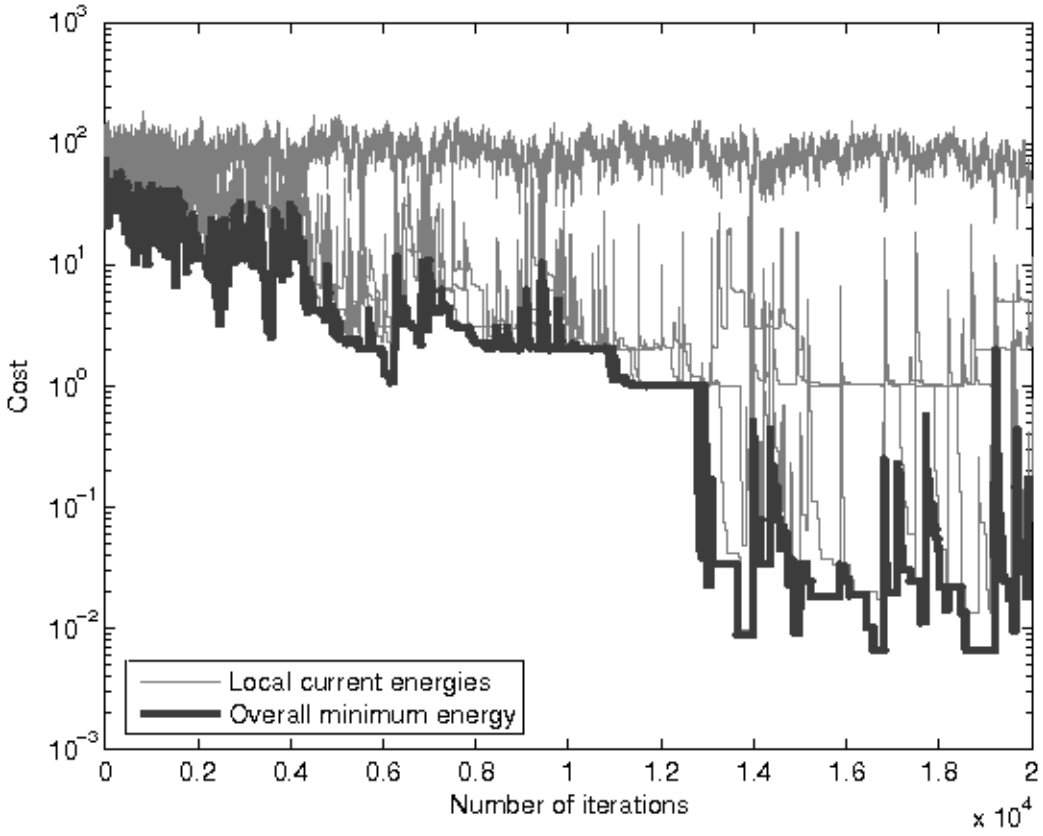


Fig. 9. Plot of a typical CSAM run with variance control for function #2 with $D \equiv m = 5$. Initial parameters are: $T_0^{ac} = 64$; 500 steps per temperature; $T_0 = 0.075$; $\sigma_D^2 = 0.99\left(\frac{m-1}{m^2}\right)$; and $\alpha = 0.05$.