Coupled Simulated Annealing

Samuel Xavier-de-Souza, *Member, IEEE,* Johan A.K. Suykens, *Senior Member, IEEE,*
Joos Vandewalle, *Fellow, IEEE,* and Désiré Bollé

***Abstract*—We present a new class of methods for global optimization of continuous variables based on Simulated Annealing (SA). The Coupled Simulated Annealing (CSA) class is characterized by a set of parallel SA processes coupled by their acceptance probabilities. The coupling is performed by a term in the acceptance probability function that is a function of the energies of the current states of all SA processes. A particular CSA instance-method is distinguished by the form of its coupling term and acceptance probability. In this paper we present three CSA instance methods and compare them with the uncoupled case—multi-start SA. The primary objective of the coupling in CSA is to create cooperative behavior via information exchange. This helps in the decision of whether or not to accept uphill moves. In combination with that, coupling can also serve to provide information that can be used on-line to steer the overall optimization process towards the global optimum. We present an example where we use the acceptance temperature to control the variance of the acceptance probabilities with a simple control scheme. This leads to a much better optimization efficiency because it reduces the sensitivity of the algorithm to initialization parameters while guiding the optimization process to quasi-optimal runs. We present the results of extensive experiments showing that the addition of the coupling and the variance control leads to considerable improvements w.r.t. the uncoupled case and a more recently proposed distributed version of SA.**

***Index Terms*—Cooperative behavior, simulated annealing, distributed optimization, global optimization.**

I. Introduction

IN this paper we define a class of optimization methods based on Simulated Annealing (SA) [1] that can be used to solve unconstrained non-convex problems in continuous variables. The existing methods used to solve such problems are vast. When a problem is differentiable, gradient descent can be applied, but in general, no guarantee can be given in this case about global optimality. Nevertheless, for some classes of non-convex problems, variations of these methods can perform surprisingly well. Coupled Local Minimizers (CLM), for example, use multiple gradient descent optimizers coupled by synchronization constraints to solve non-convex problems. CLM proved to be more effective than multi-start gradient descent optimization [2], [3]. For a large proportion of real-life non-convex problems, gradient based procedures cannot be applied due to the lack of differentiability of the cost function or to the huge computational cost of some large scale problems. For some challenging problems, gradient based techniques are also less suitable to be applied due to multimodality, multidimensionality and/or presence of many local minima.

A large number of global optimization techniques were developed in order to provide alternative solution-methods for multimodal and multidimensional problems. Examples include: simulated annealing, genetic algorithms, and particle swarm optimization. While having the advantage of escaping from multiple local minima, the strongest drawback of these methods is the large number of cost function evaluations that are often required to reach the global optimum. This problem has been tackled by the development of faster global optimization techniques where local and global procedures are often combined in order to obtain a faster convergence [4], [5]. Not surprisingly, faster convergence introduces a trade-off between the necessary number of cost function evaluations and the quality of the final solutions. Indeed, faster techniques are more often trapped in poor local minima. A challenge in this area is to find a good trade-off for a given optimization problem. Another disadvantage of several global optimization methods is the lack of robustness associated with initialization parameters. The right choice for such parameters affects the ability to find the global optimum. As a result, the effective number of cost-function evaluations after parameter tuning can still be much larger than the number of evaluations required in an optimization run with ideal initialization parameters. This is because in the tuning phase a cost function needs to be evaluated several times in order to find the values of the ideal initialization parameters for a given problem.

The class of global optimization methods that we define in this paper is targeted at multimodal and multidimensional problems with many local optima. The working principle of the methods in this class was inspired by the effect of coupling in CLM when compared to the uncoupled case—multi-start gradient descent optimization. It defines a set of distributed SA methods that have individual SA processes coupled to each other by a term in the acceptance probability function. Hereby, we aim at reducing the problems associated with tuning of initialization parameters, consequently reducing the overall number of cost function evaluations. The coupling term is a function of the current costs of all individual SA processes. Each individual SA process in a CSA method has the information about the status of the costs of all others. The composition of both coupling term and acceptance probability function defines the way in which this information is used to steer the optimization toward the global optimum. The information shared through coupling also allows for controlling general optimization indicators using optimization control parameters. We show this by a simple control on the variance of the

Samuel Xavier-de-Souza is with the Department of Computer Engineering and Automation, Universidade Federal do Rio G. do Norte, 59078-900 - Natal - RN - Brazil (email: samuel@dca.ufrn.br).

Johan A. K. Suykens and Joos Vandewalle are with the Department of Electrical Engineering (ESAT-SCD/SISTA), Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium (email [johan.suykens, joos.vandewalle]@esat.kuleuven.be).

Désiré Bollé is with the Institute for Theoretical Physics, Celestijnenlaan 200D, B-3001 Leuven, Belgium (email: desire.bolle@fys.kuleuven.be).

acceptance probabilities using the acceptance temperature. Such an approach is justified by the fact that it reduces the influence of the initialization and the choice of the schedule for this temperature, which are two well known critical issues in SA.

In order to demonstrate the potential of methods from the CSA class, we define three instance-methods of this class and compare them with the uncoupled case—multi-start classical SA. When no control is applied to general optimization indicators, this comparison shows that the performance of these three instances of the CSA class are compatible with the performance of the uncoupled case. CSA presents much better performance than SA when the variance control of acceptance probabilities is applied to one of the CSA instance-methods. We show that this method is in general much more robust than the uncoupled case w.r.t. initialization parameters, and considerably better when its results are compared with those of a recently proposed distributed SA method, namely the Sample-Sort SA algorithm [6].

This paper is organized as follows. In Section II we review the aspects of SA that are relevant to this paper. In Section III we describe the proposed class of CSA methods. We propose three instance-methods of this class in Section IV. In Section V we describe a simple control scheme applied to one of the CSA instance-methods. In Section VI we present our experiments and their results to demonstrate the performance of the methods discussed here. Finally, in Section VII, we draw conclusions about the main contributions.

II. Simulated annealing

Simulated annealing is a technique among several other global optimization approaches designed to solve difficult non-convex problems. It was originally based on the thermodynamic annealing process, which consists of heating up a metal, glass, or crystal, holding its temperature and then cooling it at a very slow rate. This physical-chemical process gives as a result high quality materials. The analogy with an optimization procedure in based upon the following relations [1]:

Physical material states	$\rightarrow$	Problem solutions
Energy of a state	$\rightarrow$	Cost of a solution
Temperature	$\rightarrow$	Control parameter.

Physical annealing is modelled or simulated with software using Monte Carlo techniques, resulting in an efficient computational algorithm that is widely used nowadays to optimize many different problems [7]–[9] (see reference [10] for an extensive list of applications). A SA algorithm is basically composed of two stochastic processes. One is responsible for the generation and the other for the acceptance of solutions. In numerical optimization, these processes are usually controlled by two temperature values[1]. As in physical annealing, temperatures must follow an annealing schedule.

The generation temperature is responsible for the correlation between generated probing solutions and the original solution. Generally, probing solutions are obtained by the addition of a random vector ε, of the same size as the solution vectors, to the vector representing the current solution. A variation in the generation temperature modifies the distribution from which ε is obtained. Namely, an increase in this temperature represents a widening of the distribution, whereas a decrease causes the distribution to become narrower. The correct distribution is the one which fits best with the generation temperature schedule. There exist many convergence proofs for classical SA which pair different temperature schedules with the correct distribution [4], [5], [11], [12].

[1]In combinatorics, the generation temperature is often omitted in exchange for the notion of neighborhood.

The acceptance temperature weighs the difference between a probing solution and the current one. Fixing this difference, the higher this temperature, the larger the probability that an uphill move is accepted. When this temperature becomes lower, the probability becomes smaller. Therefore, early in the optimization, many uphill moves are accepted, and in the evolution of the process, fewer and fewer uphill moves are allowed. Close to the end, almost no uphill moves are accepted. Such an approach permits an extensive exploration of the cost function at the beginning and a gradually more localized search towards the end.

In Fig. 1 a simplified flowchart of a classical SA algorithm depicting data and program flows is given. Observe that the temperatures are only updated once the *equilibrium criterion* is met. The objective of such a criterion is to wait for enough iterations until there is no or little variation in the energy of the accepted solutions, which means that the equilibrium was reached. An example of a straightforward criterion is to wait a fixed number of iterations N. Theoretically, every proof of convergence for SA assumes $N \rightarrow \infty$, which is unfeasible to reach in practice. Therefore, in practice a reasonable value of N is chosen by techniques which take into account the variance of the accepted solutions; or simply by setting a maximum N according to the available physical resources, like time and computational power.

Fig. 1 can be also characterized by the following algorithm:

Algorithm 1

1) *Initialization:* assign a random initial solution to x; assess its cost $E(x)$; set the 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 according to $y = x + \varepsilon$, where ε is a random variable sampled from a given distribution $g(\varepsilon, T_k)$; assess the cost for the new probing solution $E(y)$.
3) *Accept* solution y with probability 1 if $E(y) \leqslant E(x)$, otherwise with probability $A(x \rightarrow y)$, *i.e.* make $x := y$ only if $A > r$, where r is a random variable sampled from a uniform distribution [0,1]; *go to step 2* for N inner iterations (equilibrium criterion).
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*.

The variable T_k and T_k^{ac} are the generation and acceptance temperature parameters at time instant k, respectively. The function $g(\varepsilon, T_k)$ is the generation distribution and $A(x \rightarrow y)$ is the probability of accepting the solution y, given that the current solution of the system is x, with $x, y \in \Omega$, where Ω

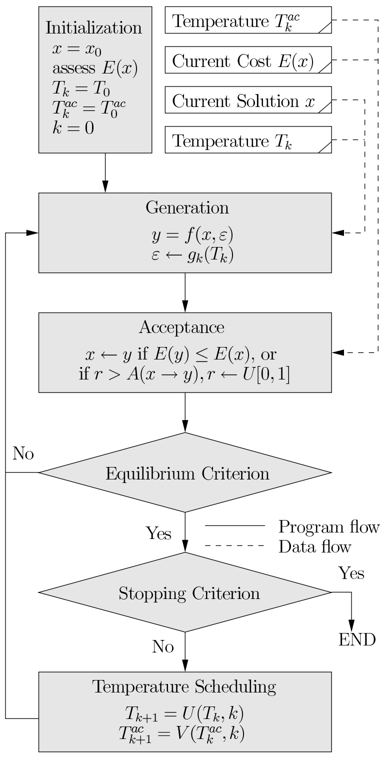

Fig. 1: Flowchart of a typical SA process. Full lines represent the program flow, whereas dashed lines represent the data flow, with x denoting the current solution, T_k and T_k^{ac} denoting the generation and acceptance temperatures at iteration k, and T_0 and T_0^{ac} denoting the respective initial temperatures. $E(\cdot)$ is the energy function, and $U(T_k, k)$ and $V(T_k^{ac}, k)$ are the generation and acceptance temperature schedules, respectively.

denotes the set of all possible solutions. Many SA convergence proofs [4], [5], [11], [12] were established by associating a given generating distribution $g(\varepsilon, T_k)$ with a generating temperature schedule $U(T_k, k)$. Therefore, the choice of the distribution $g(\varepsilon, T_k)$ depends on the schedule $U(T_k, k)$.

Fig. 1 and Algorithm 1 represent a class of SA methods from which an instance-method is obtained by specifying $g(\varepsilon, T_k)$, $A(x \to y)$, $U(T_k, k)$, and $V(T_k^{ac}, k)$. Although there exist many other SA algorithms that may deviate from this classical representation, including sequential [13], [14] and distributed [6], [15]–[20] SA versions, algorithms from this class are still widely used today due to their simple structure and easy implementation.

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 quantities related to the system. Random sampling of these states turned out to be very inefficient because in these systems only a few low-energy states carry most of the relevant information. Importance sampling probabilistically favors states with lower energies, ensuring a more efficient rejection/acceptance mechanism. The well known Metropolis algorithm [21] was the first to use the idea to estimate these quantities efficiently. Also, this algorithm complies with the principle of *Detailed Balance*[2], which gives a sufficient condition to test the validity of Monte Carlo schemes. In terms of the master equation of a thermodynamic system, this principle states that

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

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 equal for all states, *i.e.* $G = 1/n$, with n denoting the number of 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). Among the most common there are the Metropolis rule:

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

and the rule:

$$A(x \to y) = \frac{1}{1 + \exp\left(\frac{E(y) - E(x)}{T_k^{ac}}\right)}. \tag{4}$$

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 the new state or not with (4) is the energy of the current state $E(x)$.

III. Coupled Simulated Annealing: general principles

The focus of developing accelerated global optimization methods mainly seems to have been on increasing the speed of convergence, rather than improving the quality of the final solution. For this reason, optimizing a certain cost function requires many attempts with a variety of different initialization conditions. Consequently, what *a priori* seemed to be an increase in convergence speed, might be counterbalanced by a possibly excessive number of different initialization attempts that are necessary in order to achieve a certain level in the quality of the final solution.

The class of CSA methods presented in this paper is designed to be able to easily escape from local optima and thus improve the quality of solutions without slowing down too much the speed of convergence. To better understand the underlying principles of the class of methods presented here, consider the work of Suykens *et al.* [3]. They have shown that coupling among local optimization processes can be used

[2]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.

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 present the use of coupling in a global optimization method. Additionally, by designing a coupling mechanism with minimal communication, these coupled methods can be implemented very efficiently in parallel computer architectures, making them very appealing to the multi-core trend in the new generation of computer architectures [22].

CSA features a new form of acceptance probability functions that can be applied to an ensemble of optimizers. This approach considers several current states which are coupled together by their energies in their acceptance probability function. Also, as distinct from classical SA techniques, parallelism is an inherent characteristic of this class of methods. The objective of creating coupled acceptance probability functions that comprise the energy of many current states, or solutions, is to generate more information when deciding to accept less favorable solutions. It can also be observed that this class of acceptance probability functions can be a generalization of existing SA acceptance probability functions (see Section III-B).

A. A formal definition of CSA

In CSA, each optimization process, *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. The only difference between such a process and a SA process concerns the acceptance probability. While in SA this probability is a scalar function, $0 \leq A(x \to y) \leq 1$, for every $x, y \in \Omega$, with Ω denoting the set of all possible states, in CSA it is a scalar function according to

$$0 \leq A_\Theta(\gamma, x_i \to y_i) \leq 1, \tag{5}$$

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, for every $i = 1, \cdots, m$, with m being the number of elements in Θ. The set Θ is presented as the set of current states and is defined as $\Theta \equiv \{x_i\}_{i=1}^m$ throughout the paper. Given one of its elements, the decision about swapping the element with a probing state which is outside 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,

$$\gamma = f\left[E(x_1), E(x_2), \cdots, E(x_m)\right]. \tag{6}$$

In summary, in order to identify a method belonging to the CSA class, this method needs to comply with both (5) and (6). The general difference between classical SA and CSA acceptance processes is illustrated in Fig. 2.

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. 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 desirable features for Monte Carlo methods, especially when the target of the simulation is the evaluation of statistical properties of particle systems. For optimization, however, this feature may not be essential.

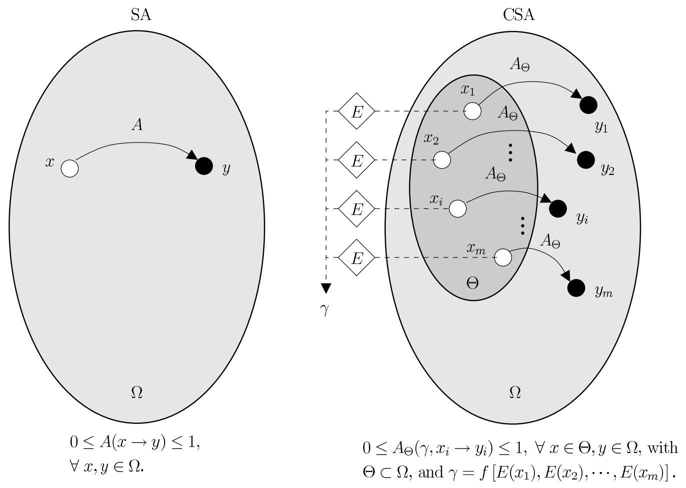
SA
CSA
A
x
y
Ω
A_Θ
x_1
y_1
x_2
y_2
x_i
y_i
x_m
y_m
E
E
E
E
γ
Θ
Ω
$0 \leq A(x \to y) \leq 1$, $\forall\, x, y \in \Omega$.
$0 \leq A_\Theta(\gamma, x_i \to y_i) \leq 1$, $\forall\, x \in \Theta, y \in \Omega$, with $\Theta \subset \Omega$, and $\gamma = f\left[E(x_1), E(x_2), \cdots, E(x_m)\right]$.

Fig. 2: 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 a 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.

B. A CSA generalization of SA

In this Section we show how a SA algorithm with acceptance probability function defined by (4) can be generalized as a CSA algorithm. We omit here any reference to parts of the algorithms other than the acceptance probability function, which is the only part in which the two algorithms differ.

When we multiply both numerator and denominator of the right hand side in (4) with $\exp(\frac{-E(y)}{T_k^{ac}})$, the following equivalent equation results:

$$A(x \to y) = \frac{\exp\left(\frac{-E(y)}{T_k^{ac}}\right)}{\exp\left(\frac{-E(y)}{T_k^{ac}}\right) + \exp\left(\frac{-E(x)}{T_k^{ac}}\right)}. \tag{7}$$

Consider now a heat-bath thermodynamic particle system with only two states. The probability of this system being in each state 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, given in this case 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. It can be observed that the acceptance probability function (7), for a given probing solution y and fixed temperature, can in fact be approximated by the Boltzmann probability for the two-state system. Therefore, a SA process can be approximated by the modelling of a two-state particle system with a variable probing state energy.

In CSA, in order to couple many SA processes, we can model the ensemble of processes by a particle system with many states. With x_i being one of the many states and y_i the corresponding probing state, we can achieve this by inserting more current states in (7) by considering the sum over a set of current states $x \in \{x_1, x_2, \cdots, x_m\}$

$$A(x_i \to y_i) = \frac{\exp\left(\frac{-E(y_i)}{T_k^{ac}}\right)}{\exp\left(\frac{-E(y_i)}{T_k^{ac}}\right) + \sum\limits_{x \in \{x_1, x_2, \cdots, x_m\}} \exp\left(\frac{-E(x)}{T_k^{ac}}\right)}. \tag{8}$$

This is a typical example of an acceptance probability function for CSA since it satisfies (5) and (6) with

$$\gamma = \sum_{x \in \{x_1, x_2, \cdots, x_m\}} \exp\left(\frac{-E(x)}{T_k^{ac}}\right).$$

Instead of a system with two states, x_i and y_i, in (8) we have a system with many current states $\{x_1, \cdots, x_i, \cdots, x_m\}$ and many respective probe states $\{y_1, \cdots, y_i, \cdots, y_m\}$. Each of these pairs have their own acceptance probabilities $\{A(x_1 \to y_1), \cdots, A(x_i \to y_i), \cdots, A(x_m \to y_m)\}$. The difference between this system with many current states and many systems with two states (one current state and one probe state) is the existence of coupling. While in the latter (the uncoupled case) all acceptance probability functions are independent, in the former they all depend not only on the respective current state x_i but also on all others x_j, $j = 1, \cdots, m$. We presented (8) as a natural step between an existing SA acceptance (7) and a CSA instance. It is a natural step because (8) reduces to (7) if we make $m = 1$. With this example we show that (8) is a generalization of (7). This is not a general proof that CSA is an extended class for SA since other SA acceptance probability functions exist.

IV. Three instances of the CSA class of methods

In this Section, three CSA example methods are presented. 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 described 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 $E(\cdot)$ be the cost function, or energy function to be minimized, associated with a given solution, and let γ be the coupling term as a function of the energy of the current states. 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 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 $k = 0$.
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) For each $i \in 1, \cdots, m$, *accept* solution y_i with probability 1 if $E(y_i) \leqslant E(x_i)$, otherwise with probability $A_\Theta(\gamma, x_i \to y_i)$, *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; and go to step 2 for N inner iterations (equilibrium criterion).
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*.

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. Multi-state Simulated Annealing (CSA-MuSA)

This method is a direct generalization of the classical SA with acceptance probability driven by (4), or (7). As mentioned before in Section III-B, these acceptance probability functions can be approximated by the modelling of a particle system with two states. In order to generate the acceptance probability 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 current states is taken into account. The following equation illustrates this acceptance probability function, which is essentially (8) with the actual CSA notation:

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

where the coupling term γ is given by

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

This acceptance probability 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. An overview of the formulas is shown in Table I, row 1.

B. Blind Acceptance (CSA-BA)

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 larger energy than the current one. All lower-energy probing 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 larger energy probing state is more frequently accepted at

Method	Alg.	Acceptance function	Coupling term γ
CSA-MuSA	2	$\frac{\exp\left(\frac{-E(y_i)}{T_k^{ac}}\right)}{\exp\left(\frac{-E(y_i)}{T_k^{ac}}\right)+\gamma}$	$\sum_{x_j\in\Theta}\exp\left(\frac{-E(x_j)}{T_k^{ac}}\right)$
CSA-BA	2	$1-\frac{\exp\left(\frac{-E(x_i)}{T_k^{ac}}\right)}{\gamma}$	$\sum_{x_j\in\Theta}\exp\left(\frac{-E(x_j)}{T_k^{ac}}\right)$
CSA-M	2	$\frac{\exp\left(\frac{E(x_i)}{T_k^{ac}}\right)}{\gamma}$	$\sum_{x_j\in\Theta}\exp\left(\frac{E(x_j)}{T_k^{ac}}\right)$
CSA-MwVC	3	$\frac{\exp\left(\frac{E(x_i)}{T_k^{ac}}\right)}{\gamma}$	$\sum_{x_j\in\Theta}\exp\left(\frac{E(x_j)}{T_k^{ac}}\right)$
SA	1	$\frac{1}{1+\exp\left(\frac{E(y)-E(x)}{T_k^{ac}}\right)}$	$-$

TABLE I: Overview of the studied methods: Multi-state SA (CSA-MuSA), Blind Acceptance (CSA-BA), Coupled Simulated Annealing-Modified (CSA-M), CSA-M with Variance Control (CSA-MwVC), and the classical SA algorithm. The algorithms used for each method are indicated by the second column.

high energy current states while low energy current states are more preserved. The acceptance probability function is chosen as follows

$$A_\Theta(\gamma, x_i \to y_i) = 1 - \frac{\exp\left(\frac{-E(x_i)}{T_k^{ac}}\right)}{\gamma}, \qquad (11)$$

where γ is given by (10). The probability of accepting state y_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 accept fewer uphill moves than high-energy states. This causes a localized search at low-energy states and a more global exploration at high-energy states. Notice that although this method presents a considerably different approach to CSA when compared to the previous one, its acceptance probability function has the same coupling term. An overview of the formulas can be seen in Table I, row 2.

C. CSA Modified (CSA-M)

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 unknown regions better. In Coupled Simulated Annealing Modified (CSA-M), we combine both search strategies. In both previous examples of CSA, the Boltzmann factor composes the acceptance probability function. Here a similar function is used, representing the probability of *leaving* the current states upon an uphill move:

$$A_\Theta(\gamma, x_i \to y_i) = \frac{\exp\left(\frac{E(x_i)}{T_k^{ac}}\right)}{\gamma}. \qquad (12)$$

Like in the second example method, this one also performs *blind acceptance* because its acceptance probability is independent of the energy of the probing solution. The coupling term γ here is given by

$$\gamma = \sum_{x_j \in \Theta} \exp\left(\frac{E(x_j)}{T_k^{ac}}\right). \qquad (13)$$

Observe that the energy of the states has a positive sign. This may cause numerical instabilities in the evaluation of the acceptance probability functions. Fortunately, for many cost functions this problem can be easily solved by a simple cost function normalization. Pre-scaling the output of the cost function may be sufficient to suppress the instability. However, with unknown cost function bounds, it is necessary to use other approaches. In this case, we suggest that all energies in (12) and (13) are subtracted by the maximum current energy, as follows

$$A_\Theta^\star(\gamma^\star, x_i \to y_i) = \frac{\exp\left(\frac{E(x_i) - \max\limits_{x_i\in\Theta}(E(x_i))}{T_k^{ac}}\right)}{\gamma^\star}, \qquad (14)$$

and

$$\gamma^\star = \sum_{\forall x \in \Theta} \exp\left(\frac{E(x) - \max\limits_{x_i\in\Theta}(E(x_i))}{T_k^{ac}}\right). \qquad (15)$$

The resulting equation is numerically more attractive because all exponents are negative.

At low temperatures at each iteration, the coupling gives the current state with the highest cost high chances to change. This emphasizes global search during the whole optimization process. Additionally, many current states are allowed to have acceptance probabilities very close to zero, which generates knowledge via the coupling term in order to better decide if it is worth accepting uphill moves or not. The coupling here is given by equating to 1 the sum of the probabilities of *leaving* any current state.

Refer to Table I, row 3, for an overview of the formulas related to this method.

V. Controlling optimization performance indicators

The benefits that coupling can offer to SA go beyond pure exchange of information. Coupling can also be useful for controlling general statistical measures that may have crucial influence on the performance of the optimization. Depending on the actual coupling design, these statistical indicators can be driven by modifying specific control parameters. As an example, the variance of the current acceptance probabilities of the current solutions in CSA-M can be controlled by modifying the acceptance temperature. Because of the way that the coupling is established in CSA-M, the variance of the acceptance probabilities is always a bounded value. We observed that generally significant improvement in the best current solution occurred when this variance value was in the neighborhood of its upper bound. However, when the variance value was too close to its maximum, the optimization performance decreased. In the following subsection we describe how we used these observations to design a simple scheme to control variance using temperature.

A. Using temperature to control the variance of the acceptance probabilities

In CSA-M, the acceptance temperature is responsible for weighing the proportion that each acceptance probability has to their sum, which equals 1. The contribution of each process to this sum is given by (12). The value of this contribution depends on the acceptance temperature and on the energy of all current solutions, but individually, it is exponentially proportional to the energy of its own current solution. The higher this energy, the larger the probability that the process accepts a probing solution. On the other hand, the acceptance temperature has a mixed role in this contribution. It appears in the numerator as well as in the denominator of (12). Because of that, a change in this temperature affects the acceptance probability of individual processes differently. A temperature increase (or decrease) causes the probability of the process with the lowest energy to increase (decrease), while it causes the probability to decrease (increase) for the process with the highest energy. This effect spreads gradually across the energies in intermediate ranges, reducing (raising) the variance of the probabilities for a temperature increase (decrease).

The bounds for the variance of the probabilities can be found as follows. Knowing that for CSA-M

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

the variance for A_Θ assumes the following form:

$$\begin{aligned} \sigma^2 &= \frac{1}{m} \sum_{\forall x_i \in \Theta} A_\Theta^2 - \left(\frac{1}{m} \sum_{\forall x_i \in \Theta} A_\Theta \right)^2 \\ &= \frac{1}{m} \sum_{\forall x_i \in \Theta} A_\Theta^2 - \frac{1}{m^2}. \end{aligned} \tag{16}$$

By using the fact that

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

it can be concluded that

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

This variance plays a significant role in optimization. A good variance value is the one which gives the correct balance between global exploration and localized search. From (12) it is easy to see that at a high enough temperature, regardless of the energy of the current solutions, all the acceptance probabilities approach $1/m$, whereas for a low enough temperature, the probability of the highest energy approximates 1 and all the others approach 0. These two cases correspond to the two bounds for the variance. In short, the acceptance temperature can be used to control the variance of the probabilities regardless of the current energies. Our experiments with different cost functions show that the optimization performance improves with the variance around 99% of the maximum value (see Section VI-D). A very simple control rule can be used to steer this variance to the desired value. It can be done in the following manner, as presented in [23]:

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

where σ_D^2 is the desired variance value and α is the rate for the increase or decrease of the temperature, typically in the range of $(0, 0.1]$. When the value of the acceptance variance is below its desired value, the acceptance temperature is decreased by a factor of $1 - \alpha$, otherwise, it is increased by a factor of $1 + \alpha$.

Such variance control can be applied only due to the coupling in the acceptance probability function. It substitutes a schedule for the acceptance temperature and more importantly, it works for any initial acceptance temperature. This is important because the setup of initial parameters in SA is most of the time a very tentative process. With this approach, we reduce two initialization issues at once, which are the choices for an acceptance schedule and an initial acceptance temperature. In return, two other parameters are introduced, α and σ_D^2, but these have a well defined operating range and are much less dependent on the optimization problem at hand. The complete algorithm with the variance control can now be stated as follows

Algorithm 3

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_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) For each $i \in 1, \cdots, m$, *accept* solution y_i with probability 1 if $E(y_i) \leqslant E(x_i)$, otherwise with probability $A_\Theta(\gamma, x_i \to y_i)$, as in (12), *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; and go to step 2 for N inner iterations (equilibrium criterion);
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.*

VI. Experiments and Results

We performed several experiments in order to assess the optimization potential of the class of CSA algorithms. More specifically we tested the three CSA example algorithms described in Section IV with 14 functions from 3 different problem groups. We used a multi-start version of the classical SA algorithm described in Section II as reference for most of the experiments. We have used the same generation and acceptance schedules for the three CSA and the classical SA algorithms, with exception of the experiments where the

No.	Function $f(\mathbf{x})$	Input range
1	$f_1 = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$
2	$f_2 = \sum_{i=1}^{D-1} \left((1-x_i)^2 + 100(x_{i+1} - x_i^2)^2\right)$	$[-2.048, 2.048]$

TABLE II: Unimodal and simple multimodal functions: test problems, group 1. These are simple functions that are easy to minimize. Zero is the minimum of both functions. The dimensionality of these problems can be adjusted with the term D.

variance control is evaluated. In this case, the acceptance temperature schedule for the CSA algorithm does not exist because this temperature is used as the manipulated variable for the control problem. In addition, in order to level the algorithms w.r.t. parallelism and number of cost function evaluations, we compared the results of multiple runs of the classical SA algorithm in such a way that the number of independent SA runs is equivalent to the number of parallel CSA processes, with the same total number of cost function evaluations. Table I presents an overview of the algorithms used in the experiments. The source codes of these methods and of the test functions are available for download in [24]. The majority of the experiments presented here were performed on the K.U.Leuven high performance computing cluster [25].

A. *Test problems*

According to the *no free lunch theorem* [26], no global optimization method can be universally better than all the others. Therefore, although the CSA methods presented here were developed with hard multimodal cost functions as the main target problem, we have tested these methods with three different problem groups, including one group with a unimodal and a simple multimodal function. The remaining functions are of moderate to hard difficulty. In total, we have tested the algorithms with 14 functions with very different characteristics. We have chosen these 14 functions because they often appear in global optimization research papers [27]–[30].

1) Unimodal and simple multimodal functions—group 1: The first problem of this group, function no. 1, is an easy unimodal sphere function. The second problem is the ubiquitous Rosenbrock's function, very often used for testing optimization algorithms. The reader may refer to Table II for the equations.

2) Multi-modal functions—group 2: A collection of multidimensional and multimodal 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 [27], [28]. The six multimodal test functions belonging to this group are presented in Table III.

Function no. 3 is called the Ackley's function and is probably the easiest in the group with one narrow global optimum basin and many minor shallow local optima. Function no. 4 is the Griewank's function. Its cosine term causes linkages among variables, making this function difficult to optimize. However, interestingly enough, this function is more difficult to optimize in lower than higher dimensions [31]. Function no. 5, the Weierstrass' function, is continuous but non-differentiable at several points. Function no. 6 is the Rastrigin's function. It is a complex multimodal problem with many persistent local optima. Function no. 7 is a non-continuous version of the Rastrigin's function with the same number of local optima. Schwefel's function is function no. 8 and the last of this group. Its complexity is due to deep local optima that are far from the global optimum. A common characteristic of most of the functions in this group is that they can also be minimized by multiple unidimensional searches, which does not reflect well the common characteristics of real-life problems.

3) Rotated multimodal functions—group 3: Although the functions in group 2 are considered to be hard multimodal problems, they can possibly be separable. This means that the minimization problem can be solved using D unidimensional searches, where D is the dimension of the problem. A large variety of real-life optimization problems are non separable. Therefore, in order to approximate these problems, in this group of test problems, we use rotated versions of the functions in group 2. The rotated functions preserve the same shape characteristics as those of the original functions but cannot be solved by D unidimensional searches.

In order to rotate a function, we multiply the argument $\mathbf{x}$ of the original function by a orthogonal rotation matrix $\mathbf{M}$ to obtain the new argument $\mathbf{z}$ for the rotated function. This rotation matrix is obtained using Salomon's method [32].

Finally, we can define the functions in this group as described by the following equations:

$$f_n(\mathbf{x}) = f_{n-6}(\mathbf{z}), \forall n = 9, \cdots, 13;$$

with

$$\mathbf{z} = \mathbf{M}\mathbf{x},$$

for the first 5 test problems of this group. The last function is defined as follows

$$f_{14}(\mathbf{x}) = f_8(\mathbf{z}),$$

with

$$\begin{aligned} z_i &= \begin{cases} y_i \sin\left(|y_i|^{\frac{1}{2}}\right) & |y_i| \leqslant 500, \\ 0.001\left(|y_i| - 500\right)^2 & |y_i| > 500, \end{cases} \\ & \forall i = 1, \cdots, D, \\ \mathbf{y} &= \mathbf{y}' + 420.96, \\ \mathbf{y}' &= \mathbf{M}(\mathbf{x} - 420,96). \end{aligned}$$

This is necessary in order to keep the global optimum of the original Schwefel's function, located at $[420.96, 420.96, \cdots, 420.96]$, within the search range after rotation.

B. *Initialization and temperature schedules*

SA algorithms tend to be very sensitive to different initial parameters such as temperature. Therefore, in order to avoid too much tuning of these parameters, we have predefined the

No.	Function $f(\mathbf{x})$	Input range
3	$f_3 = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right] + 20 + e$	$[-32.768, 32.768]$
4	$f_4 = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_i^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$
5	$f_5 = \sum_{i=1}^{D}\left\{\sum_{k=0}^{20}\left[(0.5)^k \cos\left(2\pi 3^k (x_i + 0.5)\right)\right]\right\} - D\sum_{k=0}^{20}\left[(0.5)^k\cos\left(\pi 3^k\right)\right]$	$[-0.5, 0.5]$
6	$f_6 = \sum_{i=1}^{D}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]$
7	$f_7 = \sum_{i=1}^{D}\left[y_i^2 - 10\cos(2\pi y_i) + 10\right], \quad y_i = \begin{cases} x_i & \lvert x_i\rvert < \frac{1}{2} \\ \frac{\mathrm{round}(2x_i)}{2} & \lvert x_i\rvert \geqslant \frac{1}{2} \end{cases}, \quad \forall i = 1, \cdots, D$	$[-5.12, 5.12]$
8	$f_8 = 419 \times D + \sum_{i=1}^{D} x_i \sin\left(\lvert x_i\rvert^{\frac{1}{2}}\right)$	$[-500, 500]$

TABLE III: Multi-modal functions: test problems, group 2. These functions present many local minima and are considered to be hard problems to optimize, especially in large dimensions. Zero is the minimum of all these functions. The dimensionality of these functions can be adjusted with the term D.

following set of initial test values for the acceptance and generation temperatures for all algorithms:

$$\begin{aligned} T_0 &\in \{0.001, 0.01, 0.1, 1, 10, 100\}, \\ T_0^{ac} &\in \{0.0001, 0.001, 0.01, 0.1, 1, 10, 100\}. \end{aligned} \tag{17}$$

For each of the experiments presented here, all algorithms involved were first tested with all possible pair combinations of these temperatures values, with about 5 to 10 runs per combination and with 40,000 and 200,000 evalutions per optimizer for $D = 10$ and $D = 30$, respectively. The pair with best average results for each method was then used as initial temperatures in the specific experiment. The values for $D = 10$ are presented in Table IV. What is remarkable here is that CSA-M presents affinity with much larger values of T_0^{ac} than the other methods. For T_0 in general, all methods agree with the best values, except SA, which due to the longer run ($10\times$), requires considerably larger values.

The number of steps per fixed temperature, N, was kept fixed because its effect on the performance of an SA algorithm can be related to the value of the initial temperatures. In this way we hoped that our chosen initial temperatures would fit the value we fixed for N. The number of steps per temperature is often chosen to be polynomially proportional to D. We chose these values conveniently to be $N = D^2$, resulting in $N = 100$ and $N = 900$ for $D = 10$ and $D = 30$, respectively.

All algorithms tested were subjected to the same generation and acceptance schedules [12], namely

$$U(T_k, k) \Rightarrow T_{k+1} = \frac{T_0}{k+1},$$

and

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

hence, according to Szu *et al.* [12], we used the Cauchy distribution to sample ε:

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

By choosing the same schedules for all algorithms, including classical SA, we hope to better demonstrate the effect that the different coupling schemes have on the optimization. There was no study to establish which schedule would suit best each algorithm. Therefore, most probably the performance of the algorithms proposed here is not yet optimal.

C. Results for CSA versus multi-start SA

In order to establish a reference for comparison with the CSA algorithms proposed here, we performed experiments with all 14 test functions and compared the results of these algorithms with the results of the best performance of multiple runs of the classical SA algorithm described in Section II. Details about the choice of the initialization conditions and temperature schedules were given in Section VI-B.

For each function, we performed 100 optimization runs in 10 dimensions. Each optimization run was composed of 10 parallel processes with a budget of 40,000 iterations per process. For CSA methods, the parallel processes are coupled, whereas for the classical SA method, each process consists of an independent optimization run—multi-start SA (MSA). In this case, at the end of all 10 runs, only the best result was used as output. We also present the average results for sequential SA with the same number of overall cost-function evaluations.

Table V presents mean and variance summaries of the final results from the five algorithms analyzed in this experiment. The best results of the parallel methods for each function are shown in bold. Underlined SA results are better than the bold results. Analyzing these results for each group, we made the following observations.

1) Unimodal and simple multimodal functions: group 1: We could see that SA and Multi-start SA showed the worst performance for f_1. Although the performance of the other three methods are similar, CSA-BA and CSA-M can be distinguished for being slightly better. For the Rosenbrock's function, multi-start SA performed best while the other methods presented comparable results.

2) Multi-modal functions: group 2: We consider the general performance of all 4 parallel algorithms in this group to be very similar to each other. The long runs of SA on the other hand presented a reasonably better performace achieving the best results for 4 out of the 6 test problems from this group.

3) Rotated multimodal functions: group 3: We see that all parallel methods showed a reasonably similar performance. Multi-start SA and CSA-BA achieved each one of the best results, while CSA-MuSA and CSA-M equally shared the other 4 best results. SA did not repeat the performance for the rotated versions of the test problems.

Sequential SA presents good results for about a third of

	Method	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
T_0^{ac}	SA	0.0001	0.001	0.0001	0.1	0.001	0.0001	0.0001	1	1	0.01	1	10	10	0.001
	MSA	0.0001	0.1	0.0001	0.001	0.0001	0.001	0.0001	1	0.001	0.001	0.001	10	10	100
	MuSA	0.0001	0.001	0.001	0.0001	0.0001	0.001	0.0001	0.1	0.01	0.1	1	0.0001	100	0.001
	CSA-BA	0.001	0.01	0.0001	0.0001	0.0001	0.0001	0.001	10	0.0001	0.001	0.001	0.1	0.01	0.0001
	CSA-M	0.001	0.1	1	100	10	1	0.1	10	0.1	100	10	1	1	0.01
T_0	SA	0.001	1	0.01	0.1	0.1	0.1	0.1	1	0.1	1	0.1	10	10	100
	MSA	0.001	0.1	0.01	0.01	0.01	0.1	0.1	1	0.1	0.1	1	1	1	1
	MuSA	0.001	0.01	0.01	0.01	0.01	0.1	0.1	1	0.1	0.1	0.1	1	1	1
	CSA-BA	0.001	0.01	0.01	0.01	0.01	0.1	0.1	1	0.1	0.1	1	1	10	10
	CSA-M	0.001	0.01	0.01	0.01	0.01	0.1	0.1	1	0.1	0.1	1	1	10	1

TABLE IV: Best values of T_0^{ac} and T_0 obtained from experiments with all possible pair combinations from (17).

	SA	MSA	MuSA	CSA-BA	CSA-M
f_1	2.23e-06	3.27e-06	5.80e-07	1.65e-07	**1.02e-07**
	3.10e-13	6.91e-13	1.95e-14	2.61e-15	**6.91e-16**
f_2	1.21e-02	**9.78e-03**	1.93e-02	2.54e-02	2.20e-02
	3.51e-05	**2.83e-05**	1.20e-04	2.14e-04	1.27e-04
f_3	2.17e-01	1.31e-03	1.32e-03	1.66e-03	**1.30e-03**
	2.34e+00	4.23e-08	**3.73e-08**	5.89e-08	3.81e-08
f_4	5.60e-02	3.13e-02	3.14e-02	**2.95e-02**	3.25e-02
	5.24e-04	**1.58e-04**	1.71e-04	1.81e-04	1.68e-04
f_5	2.03e-02	2.24e-02	2.22e-02	2.20e-02	**2.19e-02**
	3.85e-06	**4.33e-06**	4.49e-06	6.01e-06	5.04e-06
f_6	7.55e-06	5.65e-04	**5.40e-04**	8.51e-04	5.88e-04
	3.72e-12	2.14e-08	**2.10e-08**	5.81e-08	2.10e-08
f_7	7.79e-06	**5.08e-04**	5.45e-04	8.60e-04	5.53e-04
	4.16e-12	1.66e-08	**1.54e-08**	4.88e-08	2.06e-08
f_8	2.57e+00	2.58e+01	2.28e+01	**7.96e+00**	2.10e+01
	2.75e+02	2.87e+03	2.36e+03	**7.87e+02**	1.97e+03
f_9	8.07e-02	9.62e-02	**3.43e-02**	1.01e-01	7.99e-02
	2.16e-04	9.81e-02	**2.57e-02**	1.11e-01	7.39e-02
f_{10}	2.34e-01	9.98e-02	1.00e-01	9.81e-02	**7.80e-02**
	1.03e-02	1.24e-03	9.52e-04	8.36e-04	**7.08e-04**
f_{11}	9.12e-01	6.07e-01	6.36e-01	7.18e-01	**5.74e-01**
	6.70e-01	1.30e-01	2.41e-01	2.09e-01	**1.15e-01**
f_{12}	1.23e+01	**7.71e+00**	1.10e+01	1.09e+01	1.07e+01
	3.41e+01	**6.85e+00**	1.07e+01	9.80e+00	1.05e+01
f_{13}	6.09e+00	6.80e+00	**4.97e+00**	8.26e+00	8.38e+00
	3.00e+00	2.66e+00	**1.54e+00**	1.96e+00	1.85e+00
f_{14}	3.69e+02	7.47e+01	1.16e+02	**2.89e+01**	6.08e+01
	2.34e+05	2.83e+04	4.07e+04	**4.31e+03**	1.85e+04

TABLE V: Mean and variance for 100 optimization runs of all 14 test problems in $D = 10$. Each run was composed of 10 parallel processes with a maximum of 40,000 cost function evaluations per process except for SA, which was composed of one sequential process of 400,000 cost function evaluations. Table IV shows the values used for T_0 and T_0^{ac}. The best results for the parallel algorithms are presented in bold font. The underlined SA results are better than the bold font results. The general performance of all 4 parallel algorithms is very comparable.

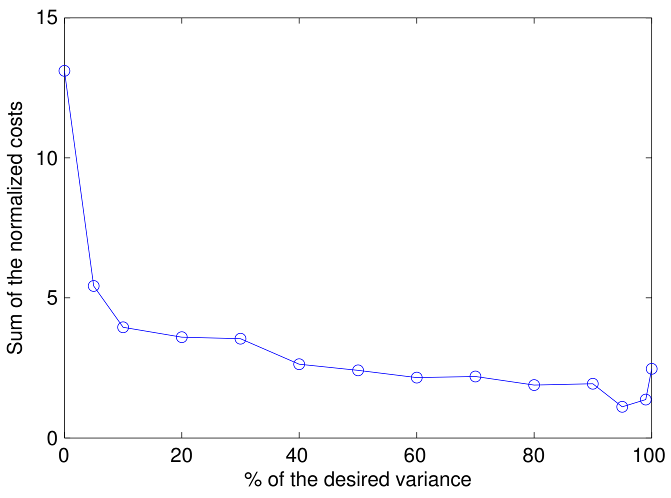

Fig. 3: Plot of the sum of the normalized average costs for a number of values of σ_D^2 (expressed in % of the maximum value). The average cost were calculated over 50 runs, for each of the 14 test problems. The best values for σ_D^2 are in the neighborhood of the maximum.

the test functions. This shows that when parallelism is not an option, a long SA run can sometimes outperform multiple shot runs. We can see that all 4 algorithms presented a reasonably similar performance although CSA-M performed generally better than the other three algorithms. While CSA-M has 5 out of 14 best means, the other algorithms have equally achieved 3 out of 14 best results. Intentionally, these results show that the proposed algorithms are not overall better than the uncoupled case. The influence of the coupling in the optimization process will depend on the design of this coupling. CSA-BA and MuSA as they are described here are merely two demonstrative instances of the class of CSA methods, with no claim to outperform other methods. CSA-M, on the other hand, demonstrates (see next Section) that the coupling has a dramatic advantage w.r.t. the uncoupled case when used to steer performance indicators as described in Section V.

Although we only presented three examples of CSA methods here, many other possibly more efficient instances can be designed. Considering that the coupling is an exclusive feature of CSA methods, it opens interesting paths to improvement as demonstrated in the next Section.

D. Analysis of the variance control

In this Section we analyze the effects of the variance control described in Section V. Fig. 3 presents an analysis of the effect of σ_D^2 on the optimization process. We have used 50 optimization runs of each test problem. The resulting costs for each function were normalized in the range $[0, 1]$. The curve seen in Fig. 3 is the sum of the normalized costs for each analyzed value of σ_D^2. Observe that indeed the average best values for σ_D^2 are in the neighborhood of the maximum but not at the maximum itself. The reason for that is possibly related to the fact that at the maximum σ_D^2 value the rate $(1 + \alpha)$ is never or very seldom used to update the temperature. This is equivalent to using a geometric cooling schedule with rate $(1 - \alpha)$ instead of the proposed variance control.

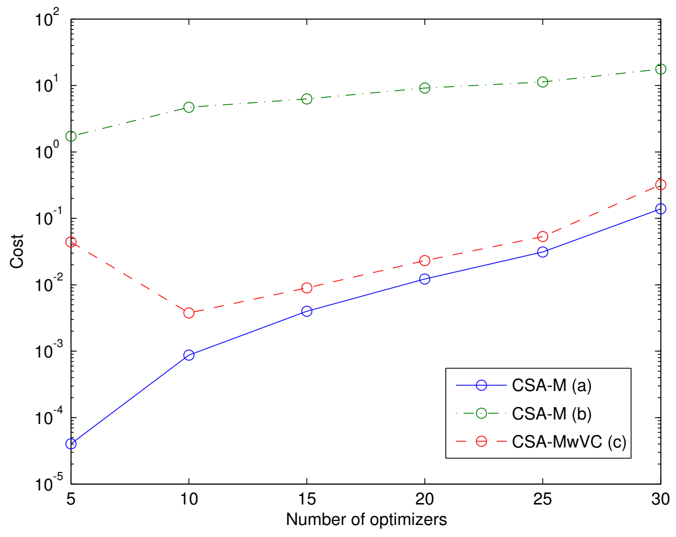

Fig. 4: Plot of 50 average optimization runs for f_6 with $D = \Theta$ for 3 different configurations: (a) CSA-M using the best found initial parameters; (b) CSA-M using T_0^{ac} equal to random values uniformly distributed between 0 and twice the mean value of the cost function at 100 random points; (c) CSA-M using variance control with the same T_0^{ac} values 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 $\alpha = 0.05$.

It is worth mentioning that due to the porposed variace control, one of the optimizers assumes the role of a quasi-random walk. This role may switch quite often from optimizer to optimizer during a run. We also have noticed that although the variance control steers the variance value to σ_D^2, the actual average value of σ^2 is typically 2-3% lower, for $\sigma_D^2 = 0.99$.

We also performed experiments for different dimensions and number of optimizers using test problem f_6. The number of optimizers was chosen to be equal to the problem dimension with values ranging from 5 to 30. Three different configurations were used, two of them using CSA-M without the variance control and one with it. The first configuration is a CSA-M algorithm using the best initialization parameters obtained after exhaustive search. The second one uses the same algorithm but with random values for T_0^{ac}, uniformly distributed between 0 and twice the mean value of f_6 evaluated at 100 random points. In the third configuration we have CSA-M with variance control. We used the same random values for T_0^{ac} as those used in the previous configuration. All configurations featured a maximum number of function evaluations per process equal to 100,000, with 500 steps per temperature and $T_0 = 0.075$. For the variance control configuration, the desired variance was set to $\sigma_D^2 = 0.99\left(\frac{m-1}{m^2}\right)$ and the changing rate was set to $\alpha = 0.05$. The results of these experiments can be seen in Fig. 4.

Notice in Fig. 4 that the CSA-M configuration with variance control follows very closely the best-run configuration without the control. Although T_0^{ac} in CSA-M with variance control was initialized with random values, it was able to deliver quasi-optimal runs. Here the coupling combined with the variace control served to approximate the ideal annealing temperature, thus reducing the influence of the initialization of this parameter and its schedule on the final result. It is also important to notice that the last observation is also valid when the dimensions of the problem and the number of optimizers, represented on the horizontal axis, increase.

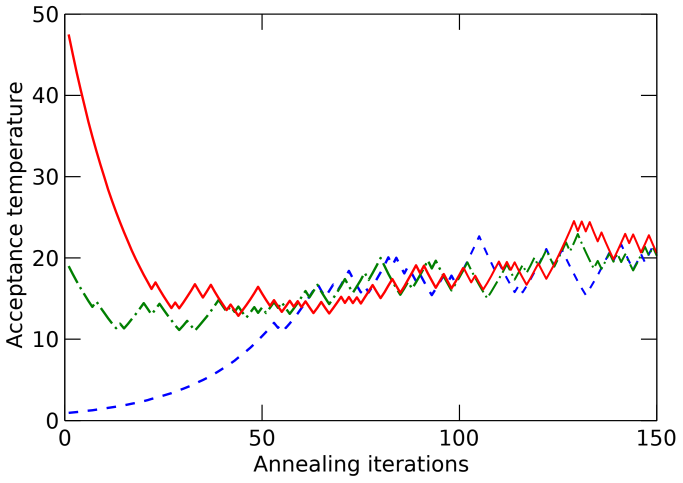

Fig. 5: Plot of three typical runs of CSA-MwVC on test problem f_6 with $D = 10$, $\alpha = 0.05$ and $\sigma_D^2 = 99\%$ for three different values of $T_0^{ac} \in 1, 20, 50$. After a short transient, all three runs converge to the same temperature range.

Finally, to conclude the analysis of the proposed variance control, we present a plot of three typical CSA-MwVC optimization runs on the test problem f_6 with $D = 10$, $\alpha = 0.05$ and $\sigma_D^2 = 99\%$ for three different values of $T_0^{ac} \in 1, 20, 50$. Fig. 5 shows that after a short transient, all three runs converge to the same temperature range. This explains the reduced sensitivity for T_0^{ac}, if we assume that this temperature range is close to the ideal one. This assumption can be inferred from Fig. 4.

E. Results for CSA-M with variance control versus Multi-start SA

We performed experiments with all 14 test problems in 10 and 30 dimensions using CSA-M with variance control, described in Section V. We compared the results of these experiments with the best results of multi-start SA. The initialization parameters were defined according to Section VI-B, except for the initial acceptance temperatures. These temperatures were chosen randomly from the predefined set in (17) for each optimization run, for both CSA-M and multi-start SA. This way, we expect to demonstrate the advantages of CSA-M w.r.t. its sensitivity to the initial acceptance temperature. The parameters of the variance control were fixed to $\sigma_D^2 = 0.99\left(\frac{m-1}{m^2}\right)$ and $\alpha = 0.05$ for all experiments.

In the results presented below, we used exponentially growing budgets for the number of cost function evaluations, namely 1,000, 10,000, and 100,000 evaluations per optimizer.

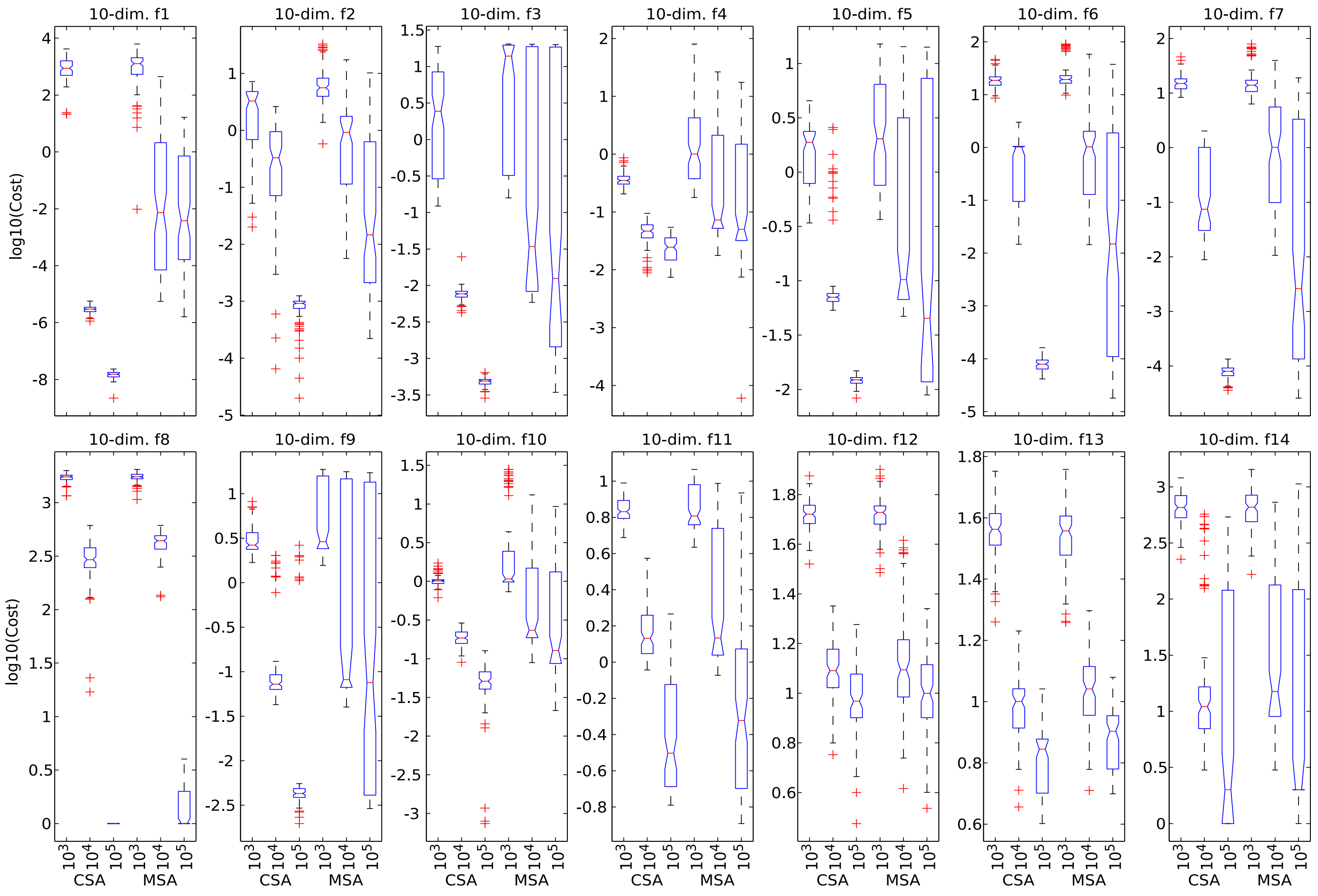

Fig. 6: Box plots of the $\ln_{10}$ of the final costs of 100 runs of CSA-M with variance control (CSA) and multi-start SA (MSA) for all 14 test problems in $D = 10$ with 3 different budgets for cost function evaluations, namely 10^3, 10^4, and 10^5 (horiz. axis). CSA-M with variance control generally has smaller variance and considerably better median.

Box plots[3] with the results for all test problems are presented in Fig. 6, for $D = 10$, and in Fig. 7, for $D = 30$. It is worth mentioning that the data points used to generate the box plots for an increasing number of cost-function evaluations are the results of independent experiments and not snapshot results of single experiments, e.g. in order to generate the box plots for multi-start SA for test problem 1, we performed 100 runs with 1,000 evaluations, then 100 runs with 10,000 evaluations, and finally more 100 runs with 100,000 evaluations of f_1. In Table VI we give the mean and variance for the same results.

In Fig. 8 we show the box plots of the results for another experiment with $D = 30$. For each test problem, these figures show the statistical evolution in box plots of the results for a linear increase in the maximum number of cost function evaluations, namely from 30,000 to 210,000 evaluations with increments of 30,000 evaluations.

The initial acceptance temperature as well as its annealing schedule are very sensitive initialization issues for classical SA. The result of such sensitivity can be confirmed by the large variance in the results of MSA in Fig. 6 and Fig. 8. For CSA-M with variance control, the acceptance temperature is an autonomously adaptive parameter and so is its schedule, which explains its general low variance.

Almost all optimization experiments showed better, or much better mean performance of CSA-M with variance control w.r.t. multi-start SA. Without the expense of tuning, we can say that thanks to coupling and a simple control scheme, CSA-M with variance control is consistently better than multi-start SA.

[3]This graphical representation shows lines at the median (notch) and lower and upper quartile values of the data. The whiskers go from each quartile to the extent of the data. Outliers are represented by crosses beyond the ends of the whiskers.

F. *Comparison: CSA-MwVC and Sample-Sort SA*

We present a comparison between CSA-MwVC and the results for the Sample-Sort SA (SS SA) method—another distributed SA algorithm which was recently published in [6]. The objective of this comparison is to project the potential of CSA methods on the state-of-the-art of distributed SA methods. Yet, we have no intention to demonstrate the efficiency of CSA against these methods. For that, a more rigorous analysis of aspects that are relevant to parallelism (e.g. synchronous versus asynchronous mode, process communication overhead, scalability issues) should be encompassed. However, from the

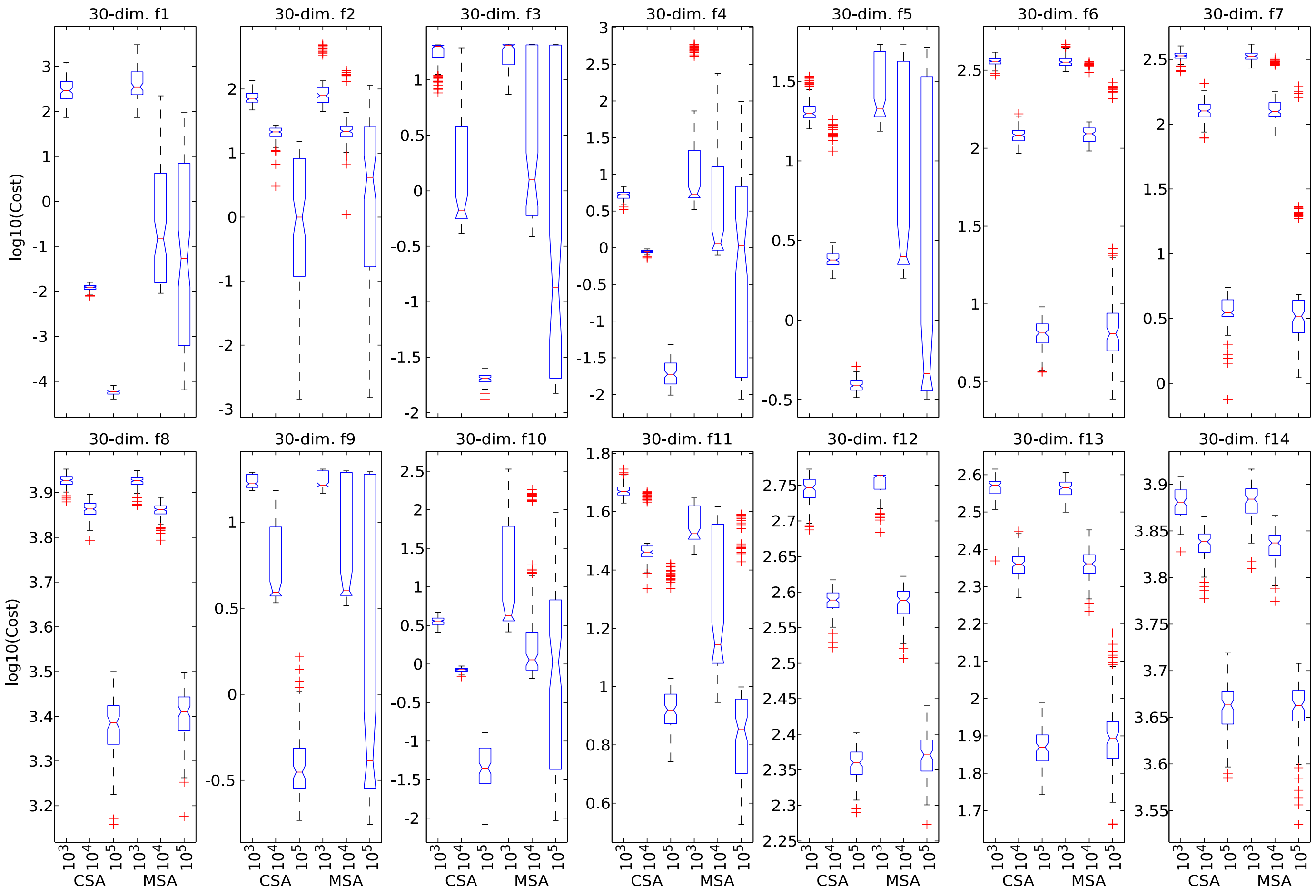

Fig. 7: Box plots of the $\ln_{10}$ of the final costs of 100 runs of CSA-M with variance control (CSA) and multi-start SA (MSA) for all 14 test problems in $D = 30$ with 3 different budgets for cost function evaluations, namely 10^3, 10^4, and 10^5 (horiz. axis). CSA-M with variance control generally has smaller variance and considerably better median, especially towards the end of the optimization.

results of this simplified analysis, it is no overstatement to say that CSA-MwVC is a very competitive method.

We have performed the same type of experiments presented in [6]. We have applied CSA-MwVC to the same functions as in the SS SA case, respecting the maximum number of cost-function evaluations per dimension, *i.e.* $1,000$, $10,000$, and $100,000$ evaluations[4]. Just as in SS SA in [6], we generated results for CSA-MwVC with 10 and 100 parallel optimizers. While in [6] the acceptance temperature was initialized using an automatic procedure, for CSA-MwVC we have used random values sampled from (17). We used $T_0 = 1$ for all experiments in this Section. The results are given in terms of the probability to reach a solution that was within 5% of the known global optimum. Table VII presents the results of our experiments next to the results for Sample-Sort SA obtained from [6].

Observe in Table VII that both methods improved performance considerably with the increasing number of cost-function evaluations. However, CSA-MwVC presents a performance that was often much superior to SS SA. In fact, the CSA method presented inferior performance in only one of the experiments. It is remarkable that CSA-MwVC did not use any *a priori* assessment for the initialization temperatures. Even more remarkable is that the CSA method used random values for T_0^{ac} for such a performance.

[4]Observe that the maximum number of cost-function evaluations in this Section is no longer given per optimizer as in the rest of the paper. We have used a maximum per dimension in order to match the experiments in [6].

VII. Conclusions

We have presented the design of a new class of global optimization methods called Coupled Simulated Annealing (CSA). Methods from this class are composed by several distributed Simulated Annealing processes with coupled acceptance probabilities. A particular CSA method is characterized by its acceptance probability function and a coupling term, which is part of this function. The coupling term is a function of the current energies of all SA processes. Thanks to the coupling, general performance indicators can be controlled on-line with the objective to improve the status of the optimization. As an example of such, we presented the case where one of the three proposed CSA methods uses the acceptance temperature to control the variance of the acceptance probabilities. We presented several experiments with the proposed methods

	$D = 10$						$D = 30$					
	MFE = 1000		MFE = 10000		MFE = 100000		MFE = 1000		MFE = 10000		MFE = 100000	
	MSA	CSA	MSA	CSA	MSA	CSA	MSA	CSA	MSA	CSA	MSA	CSA
f_1	1.45e+03	**1.14e+03**	2.80e+01	**2.97e-06**	1.65e+00	**1.49e-08**	7.23e+02	**3.56e+02**	2.61e+01	**1.22e-02**	1.67e+01	**5.84e-05**
	1.45e+06	**7.91e+05**	6.02e+03	**6.91e-13**	1.71e+01	**1.56e-17**	5.95e+05	**5.12e+04**	3.62e+03	**2.90e-06**	1.04e+03	**7.07e-11**
f_2	8.51e+00	**2.99e+00**	2.50e+00	**6.07e-01**	1.61e+00	**8.41e-04**	1.24e+02	**7.55e+01**	3.51e+01	**2.08e+01**	2.35e+01	**4.23e+00**
	5.93e+01	**4.88e+00**	1.78e+01	**5.32e-01**	1.05e+01	**6.97e-08**	1.48e+04	**3.42e+02**	1.79e+03	**2.27e+01**	1.38e+03	**2.60e+01**
f_3	1.03e+01	**5.44e+00**	6.05e+00	**7.79e-03**	5.05e+00	**4.79e-04**	**1.76e+01**	1.77e+01	1.00e+01	**4.05e+00**	9.66e+00	**2.03e-02**
	8.03e+01	**4.00e+01**	7.81e+01	**4.30e-06**	7.20e+01	**3.54e-09**	1.92e+01	**1.58e+01**	9.61e+01	**3.87e+01**	1.04e+02	**4.54e-06**
f_4	9.62e+00	**3.70e-01**	3.19e+00	**4.81e-02**	1.75e+00	**2.60e-02**	7.78e+01	**5.19e+00**	2.68e+01	**8.90e-01**	1.41e+01	**2.18e-02**
	4.37e+02	**1.10e-02**	4.02e+01	**3.07e-04**	1.17e+01	**1.61e-04**	2.85e+04	**4.48e-01**	3.69e+03	**2.17e-03**	8.96e+02	**8.90e-05**
f_5	4.46e+00	**1.81e+00**	3.07e+00	**2.00e-01**	2.91e+00	**1.21e-02**	3.02e+01	**2.17e+01**	1.57e+01	**4.39e+00**	1.26e+01	**3.91e-01**
	2.50e+01	**1.11e+00**	2.53e+01	**1.75e-01**	1.95e+01	**1.27e-06**	1.96e+02	**2.24e+01**	4.03e+02	**2.26e+01**	3.64e+02	**1.24e-03**
f_6	2.55e+01	**1.95e+01**	5.13e+00	**9.71e-01**	6.00e+00	**8.01e-05**	3.65e+02	**3.60e+02**	1.53e+02	**1.21e+02**	3.72e+01	**6.46e+00**
	3.87e+02	**4.93e+01**	1.71e+02	**5.01e-01**	1.33e+02	**4.63e-10**	1.40e+03	**5.68e+02**	6.52e+03	**1.65e+02**	6.11e+03	**1.58e+00**
f_7	1.81e+01	**1.62e+01**	6.21e+00	**4.97e-01**	2.96e+00	**7.93e-05**	3.37e+02	**3.36e+02**	1.47e+02	**1.29e+02**	1.28e+01	**3.66e+00**
	2.25e+02	**3.90e+01**	1.11e+02	**2.93e-01**	3.30e+01	**3.80e-10**	9.26e+02	**6.56e+02**	3.94e+03	**4.84e+02**	1.18e+03	**9.45e-01**
f_8	1.73e+03	**1.71e+03**	4.16e+02	**3.02e+02**	6.50e-01	**0.00e+00**	**8.38e+03**	8.44e+03	**7.23e+03**	7.27e+03	2.53e+03	**2.40e+03**
	2.36e+04	**2.13e+04**	**1.14e+04**	1.60e+04	1.17e+00	**0.00e+00**	8.58e+04	**6.35e+04**	8.92e+04	**8.44e+04**	**1.18e+05**	1.25e+05
f_9	6.91e+00	**3.25e+00**	4.74e+00	**2.74e-01**	5.41e+00	**1.87e-01**	1.74e+01	**1.73e+01**	9.94e+00	**6.14e+00**	5.86e+00	**4.52e-01**
	4.41e+01	**2.05e+00**	5.26e+01	**2.48e-01**	4.93e+01	**2.76e-01**	3.56e+00	**2.14e+00**	5.69e+01	**1.62e+01**	6.77e+01	**7.18e-02**
f_{10}	4.14e+00	**1.02e+00**	1.61e+00	**1.90e-01**	1.24e+00	**5.52e-02**	6.36e+01	**3.59e+00**	2.18e+01	**8.42e-01**	1.07e+01	**5.34e-02**
	5.13e+01	**2.54e-02**	9.37e+00	**1.91e-03**	4.60e+00	**5.67e-04**	1.17e+04	**2.28e-01**	2.45e+03	**2.63e-03**	6.40e+02	**1.05e-03**
f_{11}	7.22e+00	**7.01e+00**	2.95e+00	**1.53e+00**	1.82e+00	**5.47e-01**	**3.58e+01**	4.75e+01	**2.16e+01**	3.14e+01	1.22e+01	**1.15e+01**
	3.85e+00	**1.39e+00**	8.09e+00	**3.01e-01**	7.72e+00	**2.07e-01**	2.36e+01	**8.94e+00**	1.47e+02	**4.52e+01**	1.30e+02	**4.84e+01**
f_{12}	5.31e+01	**5.25e+01**	1.55e+01	**1.28e+01**	1.11e+01	**9.74e+00**	5.67e+02	**5.55e+02**	**3.84e+02**	3.87e+02	2.36e+02	**2.28e+02**
	7.91e+01	**5.05e+01**	8.36e+01	**1.13e+01**	1.78e+01	**8.14e+00**	**5.66e+02**	5.74e+02	4.01e+02	**2.24e+02**	3.39e+02	**1.51e+02**
f_{13}	**3.61e+01**	3.66e+01	1.10e+01	**9.92e+00**	7.77e+00	**6.56e+00**	**3.66e+02**	3.67e+02	2.30e+02	**2.29e+02**	8.24e+01	**7.40e+01**
	7.00e+01	**5.27e+01**	8.81e+00	**5.53e+00**	2.75e+00	**2.19e+00**	**4.15e+02**	5.05e+02	3.96e+02	**3.63e+02**	4.48e+02	**8.62e+01**
f_{14}	6.89e+02	**6.87e+02**	1.04e+02	**5.83e+01**	1.10e+02	**6.36e+01**	7.60e+03	**7.59e+03**	**6.82e+03**	6.85e+03	**4.55e+03**	4.58e+03
	7.85e+04	**4.68e+04**	3.11e+04	**1.69e+04**	4.07e+04	**1.69e+04**	1.13e+05	**8.45e+04**	7.66e+04	**6.47e+04**	1.11e+05	**8.42e+04**

TABLE VI: Mean and variance for 100 runs of CSA-M with variance control and Multi-start SA after 1,000, 10,000 and 100,000 cost function evaluations with $D = 10$ and $D = 30$. Initial temperatures for both algorithms where chosen randomly for each run from (17). The best values are presented in bold font. CSA with variance control showed a clearly superior efficiency in both $D = 10$ and $D = 30$.

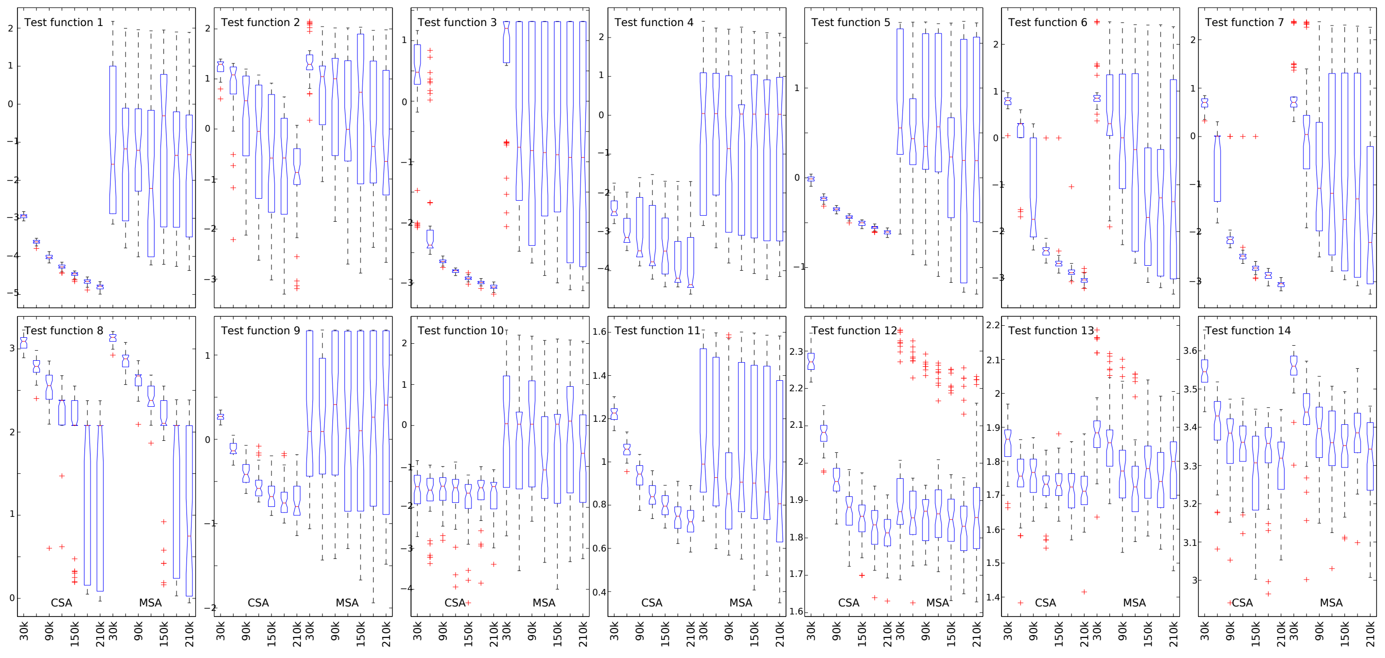

Fig. 8: Box plots of the $\ln_{10}$ of the final costs of f_{1-14} for 50 runs of Multi-start SA (MSA) and CSA-M with variance control (CSA) with initial acceptance temperatures chosen randomly from (17). The horizontal axis indicates the number of cost function evaluations ranging from 30,000 to 210,000 for both algorithms.

	10 parallel optimizers						100 parallel optimizers					
MFEPD =	1000		10000		100000		1000		10000		100000	
Function [6]	CSA	SS SA	CSA	SS SA	CSA	SS SA	CSA	SS SA	CSA	SS SA	CSA	SS SA
Branin	**1.00**	0.20	**1.00**	0.24	**1.00**	0.44	**1.00**	0.13	**1.00**	0.72	**1.00**	0.79
GoldPrice	**1.00**	0.34	**1.00**	0.82	**1.00**	0.75	**1.00**	0.08	**1.00**	0.92	**1.00**	0.88
Shekel5	**0.76**	0.03	**0.92**	0.91	**0.96**	0.95	**0.03**	0.00	**1.00**	0.33	**1.00**	**1.00**
Shekel7	**0.81**	0.05	**0.94**	0.92	**0.97**	**0.97**	**0.03**	0.00	**1.00**	0.35	**1.00**	**1.00**
Hartman3	**0.97**	0.54	**1.00**	**1.00**	**1.00**	**1.00**	0.22	**0.62**	**1.00**	**1.00**	**1.00**	**1.00**
Hartman6	**0.70**	0.01	**1.00**	**1.00**	**1.00**	**1.00**	**0.00**	**0.00**	**1.00**	0.37	**1.00**	**1.00**
Schubert3	**1.00**	0.20	**1.00**	0.93	**1.00**	0.93	**0.22**	0.02	**1.00**	0.78	**1.00**	0.90
Griewank2	**1.00**	0.00	**1.00**	0.14	**1.00**	0.23	**1.00**	0.02	**1.00**	0.02	**1.00**	0.34
Schubert5	**0.87**	0.00	**1.00**	0.69	**1.00**	**1.00**	**0.00**	**0.00**	**1.00**	0.07	**1.00**	**1.00**

TABLE VII: Comparison between CSA-MwVC and Sample-Sort SA (SS SA) [6]. The values correspond to the probability to reach a solution that was within 5% of the known global optimum of the respective functions (described in [6]). For each probability calculation there was $1,000$, $10,000$, and $100,000$ cost-function evaluations per dimension, for 10 and 100 parallel optimizers. Over 450 runs were used to calculate each probability for CSA-MwVC.

comparing them to the uncoupled case and to a more recently proposed distributed version of SA: the Sample-Sort SA algorithm. The first results show that the coupling does affect the performance of the optimization and that this performance varies with the type of the coupling, indicating the possible design of more efficient coupling schemes. The results for CSA with variance control show an excellent reduction in the sensitivity to initialization parameters when compared to the uncoupled case. When compared to Sample-Sort SA, CSA with variance control presented superior performance even with random initialization parameters.

The source codes in *C* of the methods and functions used in this paper are available at `http://www.esat.kuleuven.be/sista/-chaoslab/CSA/CSAv1_1.tar`. Future research could consider extending the coupling to the generation process. This might improve the efficiency by providing faster convergence while preserving the global search features. Finally, it is worth mentioning that the CSA algorithms presented here are based on classical SA. Extensions to other SA versions are straightforward and may improve performance.

ACKNOWLEDGMENT

Research supported by: • Research Council KUL: GOA AMBioRICS, CoE EF/05/006 Optimization in Engineering(OPTEC), IOF-SCORES4CHEM, several PhD/postdoc & fellow grants; • Flemish Government: ○ FWO: PhD/postdoc grants, projects G.0452.04 (new quantum algorithms), G.0499.04 (Statistics), G.0211.05 (Nonlinear), G.0226.06 (cooperative systems and optimization), G.0321.06 (Tensors), G.0302.07 (SVM/Kernel), research communities (ICCoS, ANMMM, MLDM); ○ IWT: PhD Grants, McKnow-E, Eureka-Flite+; • Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011); • EU: ERNSI. Joos Vandewalle and Désiré Bollé are full professors, and Johan Suykens is professor at the Katholieke Universiteit Leuven, Belgium. The authors would like to thank the reviewers for many valuable comments that helped to improve an earlier version of the paper.

REFERENCES

[1] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.
[2] J. A. K. Suykens, M. E. Yalcin, and J. Vandewalle, "Coupled Chaotic Simulated Annealing Processes," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Bangkok, Thailand, May 2003, pp. 582–585.
[3] 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.
[4] L. Ingber, "Very Fast Simulated Re-Annealing," *Journal of Mathematical Computer Modelling*, vol. 12, pp. 967–973, 1989.
[5] X. Yao, "A New Simulated Annealing Algorithm," *Intern. J. Computer Math.*, vol. 56, pp. 161–168, 1995.
[6] D. R. Thompson and G. L. Bilbro, "Sample-sort simulated annealing," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 35, no. 3, pp. 625–632, Jun 2005.
[7] 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.
[8] S. Xavier-de-Souza, M. E. Yalcin, J. A. K. Suykens, and J. Vandewalle, "Toward CNN Chip-specific robustness," *IEEE Transactions on Circuits and Systems-I: Regular Papers*, vol. 51, no. 5, pp. 892–902, May 2004.
[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] J. Kalivas, Ed., *Adaption of Simulated Annealing to Chemical Optimization Problems*. Elsevier Science, 1995.
[11] 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.
[12] H. H. Szu and R. L. Hartley, "Fast Simulated Annealing," *Physics Letters A*, vol. 122, pp. 157–162, 1987.
[13] S. Rajasekaran, "On Simulated Annealing and Nested Annealing," *J. Global Optimization*, vol. 16, pp. 43–56, 2000.
[14] 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.
[15] S. W. Mahfoud and D. Goldberg, "Parallel Recombinative Simulated Annealing: A Genetic Algorithm," *Parallel Computing*, vol. 21, pp. 1–28, 1995.
[16] A. Bevilacqua, "A Methodological Approach to Parallel Simulated Annealing on an SMP System," *J. Parallel and Distributed Computing*, vol. 62, pp. 1548–1570, 2002.
[17] 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.
[18] G. Ruppeiner, J. M. Pedersen, and P. Salamon, "Ensemble approach to simulated annealing," *J. Phys. I*, vol. 1, pp. 455–470, 1991.

[19] S. Lee and K. G. 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.

[20] E. Onbaşoğlu and L. Özdamar, “Parallel Simulated Annealing Algorithms in Global Optimization,” *J. Global Optimization*, vol. 19, no. 1, pp. 27–50, Jan 2001.

[21] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, “Equation of state calculations by fast computing machines,” *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953. [Online]. Available: http://link.aip.org/link/?JCP/21/1087/1

[22] G. Koch, “Discovering Multi-Core: Extending the Benefits of Moore’s Law. In Technology@intel Magazine,” Intel Corporation, Tech. Rep., Jul 2005.

[23] S. Xavier-de-Souza, J. A. K. Suykens, J. Vandewalle, and D. Bollé, “Cooperative behavior in coupled simulated annealing processes with variance control,” in *The 2006 International Symposium on Nonlinear Theory and its Applications (NOLTA2006)*, Bologna, Italy, Sep 2006.

[24] Coupled Simulated Annealing Software, http://www.esat.kuleuven.be/-sista/chaoslab/CSA/CSAv1_1.tar, Oct 2007, Source code in C for classical SA and CSA methods and test problems.

[25] “VIC - High-performance Linux Cluster,” K.U.Leuven, Belgium, http://ludit.kuleuven.be/hpc/.

[26] D. Wolpert and W. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, Apr 1997.

[27] Z. Tu and Y. Lu, “A Robust Stochastic Genetic Algorithm (StGA) for Global Numerical Optimization,” *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 5, pp. 456–470, Oct 2004.

[28] X. Yao, Y. Liu, and G. Lin, “Evolutionary Programming Made Faster,” *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, Jul 1999.

[29] L. Özdamar and M. Demirhan, “Experiments with new stochastic global optimization search techniques,” *Computers & Operations Research*, vol. 27, pp. 841–865, 2000.

[30] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” *IEEE Trans. on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, Jun 2006.

[31] D. Whitley, S. B. Rana, J. Dzubera, and K. E. Mathias, “Evaluating evolutionary algorithms,” *Artificial Intelligence*, vol. 85, no. 1-2, pp. 245–276, 1996.

[32] R. Salomon, “Reevaluating Genetic Algorithm Performance Under Coordinate Rotation of Benchmark Functions: A Survey of Some Theoretical and Practical Aspects of Genetic Algorithms,” *BioSystems*, vol. 39, no. 3, pp. 263–278, 1996.