Simulated Annealing with Asymptotic Convergence for Nonlinear Constrained Global Optimization[*]

Benjamin W. Wah and Tao Wang

Department of Electrical and Computer Engineering
and the Coordinated Science Laboratory
University of Illinois, Urbana-Champaign
Urbana, IL 61801, USA
{wah, wangtao}@manip.crhc.uiuc.edu
http://www.manip.crhc.uiuc.edu

Abstract. In this paper, we present *constrained simulated annealing* (CSA), a global minimization algorithm that converges to constrained global minima with probability one, for solving nonlinear discrete non-convex constrained minimization problems. The algorithm is based on the necessary and sufficient condition for constrained local minima in the theory of discrete Lagrange multipliers we developed earlier. The condition states that the set of discrete saddle points is the same as the set of constrained local minima when all constraint functions are non-negative. To find the discrete saddle point with the minimum objective value, we model the search by a finite inhomogeneous Markov chain that carries out (in an annealing fashion) both probabilistic descents of the discrete Lagrangian function in the original-variable space and probabilistic ascents in the Lagrange-multiplier space. We then prove the asymptotic convergence of the algorithm to constrained global minima with probability one. Finally, we extend CSA to solve nonlinear constrained problems with continuous variables and those with mixed (both discrete and continuous) variables. Our results on a set of nonlinear benchmarks are much better than those reported by others. By achieving asymptotic convergence, CSA is one of the major developments in nonlinear constrained global optimization today.

1 Problem Definition

A general *discrete constrained minimization problem* is formulated as follows:

$$\begin{aligned} &minimize_x \;\; f(x) \\ &subject\ to \;\; g(x) \le 0; \qquad x = (x_1, \ldots, x_n) \\ &\qquad\qquad\quad h(x) = 0 \end{aligned} \tag{1}$$

[*] Research supported by National Science Foundation Grant NSF MIP 96-32316.

where $f(x)$ is a lower-bounded objective function, $g(x) = [g_1(x), \cdots, g_k(x)]^T$ is a set of k inequality constraints, $h(x) = [h_1(x), \cdots, h_m(x)]^T$ is a set of m equality constraints, and all the discrete variables x_i are finite. The functions $f(x)$, $g(x)$, and $h(x)$ can be either convex or non-convex, linear or nonlinear, continuous or discontinuous, and analytic (*i.e.* in closed-form formulae) or procedural. The search space X is the finite Cartesian product of discrete variables x_i, $i = 1, \cdots, n$.

Without loss of generality, we discuss our results with respect to minimization problems, knowing that maximization problems can be converted to minimization ones by negating their objectives. We first define the following basic terms.

Definition 1. *$\mathcal{N}(x)$, the neighborhood of point x in space X, is a user-defined set of points $\{x' \in X\}$ such that $x \notin \mathcal{N}(x)$ and that $x' \in \mathcal{N}(x) \iff x \in \mathcal{N}(x')$. Neighborhoods must be defined such that any point in the finite search space is reachable from any other point through traversals of neighboring points.*

Definition 2. *A point $x \in X$ is a feasible point if $h(x) = 0$ and $g(x) \leq 0$.*

Given $\mathcal{N}(x)$ to be the neighborhood of point $x \in X$ in search space X, we define local and global minima for (1) as follows:

Definition 3. *Point $x \in X$ is called a constrained local minimum iff a) x is a feasible point, and b) for every feasible point $x' \in \mathcal{N}(x)$, $f(x') \geq f(x)$.*

Note that point x may be a local minimum to one definition of $\mathcal{N}(x)$ but may not be for another definition of $\mathcal{N}'(x)$. The choice of neighborhood, however, does not affect the validity of a search as long as one definition is used consistently throughout. Normally, one may choose $\mathcal{N}(x)$ to include the nearest discrete points to x so that neighborhood carries its original meaning. The search will still be correct even if the neighborhood is chosen to include "far away" points.

Definition 4. *Point $x \in X$ is called a constrained global minimum iff a) x is a feasible point, and b) for every feasible point $x' \in X$, $f(x') \geq f(x)$. The set of all constrained global minima is X_{opt}.*

Finding constrained global minima of (1) is challenging as well as difficult. First, $f(x)$, $g(x)$, and $h(x)$ may be non-convex and highly nonlinear, making it difficult to even find a feasible point or a feasible region. Moreover, it is not useful to keep a search within a feasible region, as feasible regions may be disjoint and the search may need to visit multiple feasible regions before finding the global minimum. Second, $f(x)$, $g(x)$, and $h(x)$ may be discontinuous or may not have derivatives, rendering it impossible to apply existing theories and methods in continuous space. Third, there may be a large number of constrained local minima, trapping trajectories that only utilize local information.

As nonlinear constrained problems do not have closed-form solutions and cannot be solved analytically except in some trivial cases, they are generally solved by some iterative procedure ψ. In general, let Ω be a search space. Given starting point $\omega(k = 0) \in \Omega$, ψ generates iteratively a sequence of points, $\omega(k = 1), \omega(k = 2), \cdots, \omega(k), \cdots$ in Ω, until some stopping conditions hold. Here we are interested in *global optimization*, and let Ω_s be the set of all global minima.

Definition 5. *Procedure ψ is said to have* asymptotic convergence to global minimum, *or simply* asymptotic convergence *[2], if ψ converges with probability one to an element in Ω_s; that is,* $\lim_{k\to\infty} P(\omega(k) \in \Omega_s) = 1$, *independent of starting point $\omega(k=0)$.*

Definition 6. *Procedure ψ is said to have* reachability of global minimum *[2] if probability* $\lim_{k\to\infty} P(\omega(l) \in \Omega_s,\ \exists\, l,\ 0 \le l \le k) = 1$.

Reachability is much weaker than asymptotic convergence as it only requires $\omega(k)$ to hit a global solution sometime during the search. In practice, reachability can be achieved by keeping track of the best solution in the course of ψ, as done in a pure random search. Hence, reachability is also called *convergence in the best solution to the global minimum.* In contrast, asymptotic convergence requires ψ to converge to a global solution in Ω_s with probability one. Consequently, the probability of hitting a global solution increases as the search progresses, making it more likely to find the global solution than an algorithm with reachability alone if the search were stopped before it converges.

In this paper, we present a new global minimization algorithm, called *constrained simulated annealing* (CSA), for finding constrained global minima with asymptotic convergence. To achieve asymptotic convergence, we have built CSA based on simulated annealing (SA) because SA is the general algorithm that can guarantee asymptotic convergence in unconstrained optimization.

The paper has six sections. Section 2.2 reviews the theory of discrete Lagrange multipliers [8, 11, 12], that states that finding a discrete saddle point with the minimum objective value is necessary and sufficient for global minimization. Hence, CSA described in Section 3 aims to find such a saddle point by performing probabilistic descents of the Lagrangian function in the original-variable space and probabilistic ascents in the Lagrange-multiplier space. To prove its asymptotic convergence, we model the search by a strongly ergodic Markov chain, and show that CSA minimizes an implicit virtual energy at any constrained global minimum with probability one [10]. Section 4 sketches the proof and illustrates the search behavior. Finally, Section 5 shows improvements in applying CSA to solve some discrete, continuous, and mixed nonlinear optimization problems.

2 Previous Work

There are two approaches to solve (1): direct solution or transformation into an unconstrained problem before solving it. Examples of direct methods include reject/discard methods, repair methods, feasible-direction methods, and interval methods. These methods may be unable to cope with nonlinear constraints, or very problem-specific, or computationally expensive. Hence, the majority of methods are based on transformations and are discussed in this section.

2.1 Penalty Formulations

This approach first transforms (1) into an unconstrained optimization problem or a sequence of unconstrained problems, and then solves it by using existing unconstrained minimization methods. Many heuristics developed to handle constraints [7] are normally problem-dependent, have difficulties in finding feasible regions or in maintaining feasibility, and get stuck easily in local minima.

Static-penalty formulations [3, 6] transform (1) into an unconstrained problem,

$$min_x \quad L_\rho(x, \gamma) = f(x) + \sum_{i=1}^{m} \gamma_i |h_i(x)|^\rho + \sum_{j=1}^{k} \gamma_{m+j} max^\rho(0, g_j(x)) \tag{2}$$

where $\rho > 0$, and penalty $\gamma = \{\gamma_1, \gamma_2, \cdots, \gamma_{m+k}\}$ is *fixed* and chosen to be large enough so that

$$L_\rho(x^*, \gamma) < L_\rho(x, \gamma) \quad \forall x \in X - X_{opt} \text{ and } x^* \in X_{opt}. \tag{3}$$

Based on (3), an unconstrained global minimum of (2) over x is a constrained global minimum to (1), and thus it is sufficient to minimize (2). Because both $f(x)$ and $|h_i(x)|$ are lower bounded and because x takes finite discrete values, γ always exists and is finite. This ensures the correctness of the approach. Note that other forms of penalty formulations are also available in the literature.

The major problem of static-penalty methods is the ruggedness of $L_\rho(x, \gamma)$ and the depth of its unconstrained local minima due to the large γ used. Unless starting points are close to one of the unconstrained global minima, it becomes very unlikely to traverse the search space X and find the global solution. Selecting a suitable γ also proves to be difficult. If it is much larger than necessary, the terrain will become too rugged to be searched. If it is too small, the solution to (2) may be a constrained local minimum or even not be a feasible solution.

Dynamic-penalty formulations address these difficulties by increasing penalties gradually. They transform (1) into a sequence of unconstrained problems:

$$min_x \quad L_\rho(x, \lambda(\kappa)) = f(x) + \sum_{i=1}^{m} \lambda_i(\kappa) |h_i(x)|^\rho + \sum_{j=1}^{k} \lambda_{m+j}(\kappa) max^\rho(0, g_j(x)) \tag{4}$$

for an increasing sequence $\lambda(\kappa), \kappa = 1, 2, \cdots, \mathcal{K}$, where $0 < \lambda(\kappa) < \lambda(\kappa + 1)$, and $\lambda(\mathcal{K}) = \gamma$. Here $\lambda \geq \lambda'$ iff $\lambda_i \geq \lambda'_i$ for every $i = 1, 2, \cdots, m$, and $\lambda > \lambda'$ iff $\lambda \geq \lambda'$ and there exists at least one i such that $\lambda_i > \lambda'_i$.

Dynamic-penalty methods are asymptotically convergent if, for every $\lambda(\kappa)$, (4) is solved optimally [3, 6]. The requirement of obtaining an unconstrained global minimum of (4) in every stage is, however, difficult to achieve in practice, given only finite amount of time in each stage. If the result in one stage is not a global minimum, then the process cannot be guaranteed to find a constrained global minimum. Approximations to the process that sacrifice global optimality of solutions have been developed, such as two-phase evolutionary programming and two-phase neural networks.

2.2 Lagrangian Formulations

Lagrangian methods are based on the theory of Lagrange multipliers that augment the original search space X by a Lagrange-multiplier space Λ, and gradually resolve constraints through iterative updates. We summarize our extensions and relevant theory of Lagrangian formulations that work in discrete space [8, 11, 12]. Let us first consider a discrete equality-constrained minimization problem,

$$\begin{aligned} &\mathit{minimize}_x \quad f(x) \\ &\mathit{subject\ to} \quad h(x) = 0; \qquad x = (x_1, \ldots, x_n) \end{aligned} \tag{5}$$

where x is a vector of finite discrete variables. A ***generalized discrete Lagrangian function*** [8] of (5) is defined to be:

$$L_d(x, \lambda) = f(x) + \lambda^T H(h(x)). \tag{6}$$

where H is a continuous function, and $\lambda = \{\lambda_1, \cdots, \lambda_m\}$, is a set of Lagrange multipliers. Based on (6), we define point (x^*, λ^*) to be a ***saddle point*** if:

$$L_d(x^*, \lambda) \leq L_d(x^*, \lambda^*) \leq L_d(x, \lambda^*), \tag{7}$$

for all $x \in \mathcal{N}(x^*)$ and all possible λ. The first inequality in (7) only holds when all the constraints are satisfied, which implies that it must be true for all λ. The following theorem [11, 12] states the first-oder necessary and sufficient conditions for all constrained local minima.

Theorem 1 (Necessary & Sufficient Condition for Discrete Constrained Local Minima). *In discrete space, if function H is a non-negative (or non-positive) continuous function satisfying $H(x) = 0$ iff $x = 0$, then the set of constrained local minima is the same as the set of discrete saddle points.*

The condition in Theorem 1 is stronger than its continuous counterpart. In continuous space, points that satisfy the first-order necessary and second-order sufficient conditions [6] are a *subset* of all the constrained local minima. Hence, the global minima of points satisfying these conditions are not necessarily the constrained global minima of the original problem. Further, these conditions require the existence of derivatives of the objective and constraint functions, and are not applicable when any one of these functions is discontinuous. In contrast, finding saddle points in discrete space always leads to a constrained local minimum. Further, if one can find the saddle point with the minimum objective value, then it is the constrained global minimum (according to Theorem 1). This observation provides the basis for the global minimization procedure studied here.

Theorem 1 also provides a way to handle inequality constraints. We first transform inequality constraint $g_j(x) \leq 0$ into an equivalent equality constraint $\tilde{g}_j(x) = max(0, g_j(x)) = 0$, resulting in an optimization problem with equality constraints only. We then use the absolute function H and define the Lagrangian function of (1) as:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i |h_i(x)| + \sum_{j=1}^{k} \lambda_{m+j} \tilde{g}_j(x) \quad \text{where } \lambda = [\lambda_1, \cdots, \lambda_{m+k}]. \tag{8}$$

```
1. procedure CSA
2.     set starting point $\mathbf{x} = (x, \lambda)$;
3.     set starting temperature $T = T^0$ and cooling rate $0 < \alpha < 1$;
4.     set $N_T$ (number of trials per temperature);
5.     while stopping condition is not satisfied do
6.         for $k \leftarrow 1$ to $N_T$ do
7.             generate a trial point $\mathbf{x}'$ from $\mathcal{N}(\mathbf{x})$ using $G(\mathbf{x}, \mathbf{x}')$;
8.             accept $\mathbf{x}'$ with probability $A_T(\mathbf{x}, \mathbf{x}')$
9.         end_for
10.        reduce temperature by $T \longleftarrow \alpha \times T$;
11.    end_while
12. end_procedure
```

Fig. 1. CSA: the constrained simulated annealing algorithm (see text for the initial values of parameters).

3 Constrained Simulated Annealing (CSA)

Figure 1 presents our global minimization algorithm called CSA for solving (1) using Lagrangian function (8). The algorithm is based on SA that normally does probabilistic descents in one space, with probabilities of acceptance governed by a temperature that is reduced in an exponentially decreasing fashion. CSA, in contrast, does probabilistic ascents in the Lagrange-multiplier space and probabilistic descents in the original-variable space. Due to space limitation, we do not present the basic steps of SA, but only discuss the steps of CSA here.

Line 2 sets a starting point $\mathbf{x} = (x, \lambda)$, where x can be either user-provided or randomly generated (*e.g.* based on a fixed seed 123 in our experiments), and λ is initialized to be zero.

Line 3 initializes control parameter T, called *temperature*, to be large enough in order to allow almost all trial points $\mathbf{x}'$ to be accepted. In our experiments, we generate the initial temperature by first randomly generating 100 points of x and their corresponding neighboring points x' where each component $|x'_i - x_i| \leq 0.001$, and then setting $T = max_{x,x',i}\{|L(x', 1) - L(x, 1)|, |h_i(x)|\}$. Our rationale is based on the initial amount of violations observed in a problem. We also set α (to be discussed later) to be 0.8 in our experiments.

Line 4 sets the number of iterations at each temperature. In our experiments, we set $N_T = \zeta(20n + m)$ where $\zeta = 10(n + m)$, n is the number of variables, and m is the number of equality constraints. This setting is based on the heuristic rule in [4] using $n + m$ instead of n.

Line 5 stops CSA when the current point $\mathbf{x}$ is not changed, *i.e.*, no other new point $\mathbf{x}'$ is accepted, for a couple of successive temperatures, or the current temperature T is small enough (*e.g.* $T < 10^{-6}$).

Line 7 generates a random trial point $\mathbf{x}'$ in neighborhood $\mathcal{N}(\mathbf{x})$ of current point $\mathbf{x} = (x, \lambda)$ in search space $S = X \times \Lambda$ using generation probability $G(\mathbf{x}, \mathbf{x}')$, where $\mathcal{N}(\mathbf{x})$ and $\mathcal{N}_2(\lambda)$, neighborhood of λ at $\mathbf{x}$, are defined as follows:

$$\mathcal{N}(\mathbf{x}) = \{(x', \lambda) \in S \text{ where } x' \in \mathcal{N}_1(x)\} \cup \{(x, \lambda') \in S \text{ where } \lambda' \in \mathcal{N}_2(\lambda)\} \quad (9)$$

$$\mathcal{N}_2(\lambda) = \{\mu \in \Lambda \mid \mu < \lambda \text{ and } \mu_i = \lambda_i \text{ if } h_i(x) = 0\} \\ \bigcup \{\mu \in \Lambda \mid \mu > \lambda \text{ and } \mu_i = \lambda_i \text{ if } h_i(x) = 0\} \tag{10}$$

where relation "<" on two vectors has been defined earlier. Neighborhood $\mathcal{N}_2(\lambda)$ prevents λ_i from being changed when the corresponding constraint is satisfied, *i.e.*, $h_i(x) = 0$. An example of $\mathcal{N}_2(\lambda)$ is that μ differs from λ in one variable (*e.g.* $\mu_i \neq \lambda_i$, and $\mu_j = \lambda_j$ for $j \neq i$), and $\{\mu_i | \; i \neq j\}$ is a set of values, some of which are larger than λ_i and some are smaller. In short, a trial point (x', λ) is a neighboring point to (x, λ) if x' is a neighboring point to x in variable space X, and (x, λ') is a neighboring point to (x, λ) if λ' is a neighboring point to λ in Lagrange multiplier space Λ and $h(x) \neq 0$.

$G(\mathbf{x}, \mathbf{x}')$, the *generation probability* from $\mathbf{x}$ to $\mathbf{x}' \in \mathcal{N}(\mathbf{x})$ satisfies:

$$G(\mathbf{x}, \mathbf{x}') > 0 \quad \text{and} \quad \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} G(\mathbf{x}, \mathbf{x}') = 1 \tag{11}$$

The choice of $G(\mathbf{x}, \mathbf{x}')$ is arbitrary as long as it satisfies (11). In our illustrative example, we use a uniform probability over $\mathcal{N}(\mathbf{x})$, independent of T.

$$G(\mathbf{x}, \mathbf{x}') = 1/|\mathcal{N}(\mathbf{x})| \tag{12}$$

Line 8 accepts $\mathbf{x}'$ with acceptance probability $A_T(\mathbf{x}, \mathbf{x}')$ that consists of two components, depending on whether x or λ is changed in $\mathbf{x}'$.

$$A_T(\mathbf{x}, \mathbf{x}') = \begin{cases} exp\left(-\frac{(L(\mathbf{x}') - L(\mathbf{x}))^+}{T}\right) & \text{if } \mathbf{x}' = (x', \lambda) \\ exp\left(-\frac{(L(\mathbf{x}) - L(\mathbf{x}'))^+}{T}\right) & \text{if } \mathbf{x}' = (x, \lambda') \end{cases} \tag{13}$$

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

The acceptance probabilities in (13) differ from the acceptance probabilities used in conventional SA, which only has the first part of (13) and whose goal is to look for global minima in the x space. Without the λ space, only probabilistic descents in the x space need to be done.

Our goal here is to look for saddle points in the joint space $X \times \Lambda$ of x and λ, which exist at local minima in x space and at local maxima in λ space. To this end, the first part of (13) carries out *probabilistic descents* of $L(x, \lambda)$ with respect to x for fixed λ. That is, when we generate a new point x' while λ is fixed, we accept it with probability one when $\delta_x = L(x', \lambda) - L(x, \lambda)$ is negative; otherwise we accept it with probability $e^{-\delta_x/T}$. This is performing exactly descents while allowing occasional ascents in x space as done in conventional SA.

However, descents in x space alone only lead to local/global minima of the Lagrangian function without satisfying the constraints. To this end, the second part of (13) carries out *probabilistic ascents* of $L(x, \lambda)$ with respect to λ for fixed x in order to increase the penalties of violated constraints and to force them into satisfaction. Hence, when we generate a new point λ' while x is fixed, we accept it with probability one when $\delta_\lambda = L(x, \lambda') - L(x, \lambda)$ is positive; otherwise we accept it with probability $e^{-\delta_\lambda/T}$. This is performing exactly ascents in λ space while allowing occasional descents (and reducing the ruggedness of the terrains and deepening the local minima) as done in conventional SA. Note that when a constraint is satisfied, the corresponding Lagrange multiplier will not be changed according to (10).

Although our algorithm only changes one variable in x or λ at a time, it is possible to derive $A_T(\mathbf{x}, \mathbf{x}')$ that allows multiple variables in x and λ to be changed together. In that case, we can decompose the aggregate change into a sequence of one-variable changes.

Finally, Line 10 reduces T using the following cooling schedule after looping N_T times at a given T:

$$T \longleftarrow \alpha \times T \tag{14}$$

where α is a constant smaller than 1 (typically between 0.8 and 0.99). Theoretically, if T is reduced slow enough, then CSA will converge to a constrained global minimum of (1) with probability one as T approaches 0.

Note that (13) enables any trial point to be accepted with high probabilities at high T, allowing the search to traverse a large space and overcome infeasible regions. As T is gradually reduced, the acceptance probability decreases, and at very low temperatures the algorithm behaves like a local search.

4 Asymptotic Convergence of CSA

In this section, we prove the asymptotic convergence of CSA to constrained global minima by modeling the process by an inhomogeneous Markov chain, showing that the Markov chain is strongly ergodic, proving that the Markov chain minimizes an implicit virtual energy based on the framework of generalized SA (GSA) [10], and showing that the virtual energy is at its minimum at any constrained global minimum. Due to space limitations, we only sketch the proofs and illustrate the results by an example.

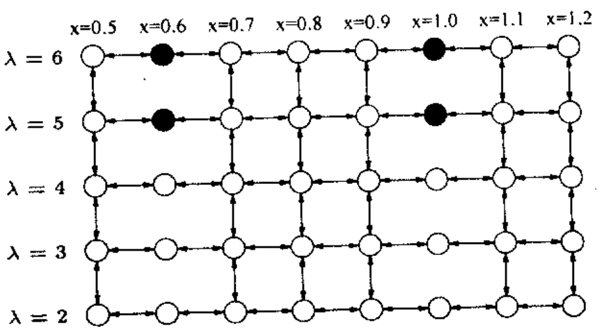
CSA can be modeled by an inhomogeneous Markov chain consisting of a sequence of homogeneous Markov chains of finite length, each at a specific temperature in a given temperature schedule. According to the generation and acceptance probabilities, $G(\mathbf{x}, \mathbf{x}')$ and $A_T(\mathbf{x}, \mathbf{x}')$, the ***one-step transition probability*** of the Markov chain is:

$$P_T(\mathbf{x}, \mathbf{x}') = \begin{cases} G(\mathbf{x}, \mathbf{x}')A_T(\mathbf{x}, \mathbf{x}') & \text{if } \mathbf{x}' \in \mathcal{N}(\mathbf{x}) \\ 1 - \sum_{\mathbf{y} \in \mathcal{N}(\mathbf{x})} P_T(\mathbf{x}, \mathbf{y}) & \text{if } \mathbf{x}' = \mathbf{x} \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

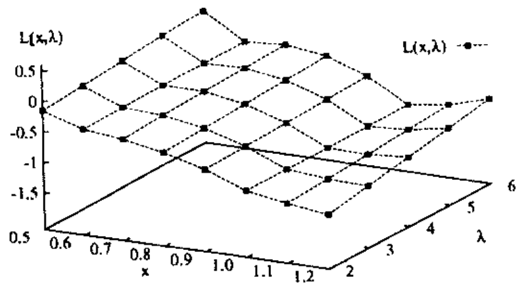
and the corresponding ***transition matrix*** is $P_T = [P_T(\mathbf{x}, \mathbf{x}')]$.

Example 1. The following simple example illustrates the Markov chain for a problem that minimizes a quadratic objective with one quadratic constraint.

$$\begin{aligned} &\textit{minimize} \quad f(x) = -x^2 \\ &\textit{subject to} \;\; h(x) = |(x-0.6)(x-1.0)| = 0 \end{aligned} \tag{16}$$



(a) State-space with non-zero transition probabilities



(b) Lagrangian function value at each state

Fig. 2. The Markov chain modeling the Lagrangian space of (16) and the corresponding Lagrangian function value. The four saddle points are shaded in (a).

where $x \in X = \{0.5, 0.6, \cdots, 1.2\}$ and $\lambda \in \Lambda = \{2, 3, 4, 5, 6\}$ are both discrete, and γ, the maximum Lagrange multiplier, is 6. The state space is, therefore, $S = \{(x, \lambda) | \; x \in X, \lambda \in \Lambda\}$, and the total number of states is $|S| = 8 \times 5 = 40$.

In the Markov chain, we define the neighborhoods for x and λ as follows:

$$\begin{aligned}\mathcal{N}_1(x) &= \{x-1, x+1 | \; 0.6 \le x \le 1.1\} \cup \{x+1 | \; x = 0.5\} \cup \{x-1 | \; x = 1.2\} \\ \mathcal{N}_2(x) &= \{\lambda-1, \lambda+1 | \; 3 \le \lambda \le 5, x \ne 0.6, \text{ and } x \ne 1.0\} \cup \{\lambda - 1 | \; \lambda = 6, \\ &\qquad x \ne 0.6, \text{ and } x \ne 1.0\} \cup \{\lambda + 1 | \; \lambda = 2, x \ne 0.6, \text{ and } x \ne 1.0\}\end{aligned} \tag{17}$$

Given $\mathcal{N}_1(x)$ and $\mathcal{N}_2(\lambda)$, $\mathcal{N}(\mathbf{x})$ is defined as in (9).

Figure 2 shows the Markov chain constructed. In the chain, a node $\mathbf{x} = (x, \lambda)$ represents a point in S, and an arrow from $\mathbf{x}$ to $\mathbf{x}' \in \mathcal{N}(\mathbf{x})$ (where $\mathbf{x}' = (x', \lambda)$ or (x, λ')) means that there is a one-step transition from $\mathbf{x}$ to $\mathbf{x}'$ with $P_T(\mathbf{x}, \mathbf{x}') > 0$. For $x = 0.6$ and $x = 1.0$, there is no transition among the λ's because the constraints are satisfied at these points (according to (10)).

There are two saddle points in this Markov chain at $(0.6, 5)$ and $(0.6, 6)$, corresponding to the local minima $x = 0.6$, and two saddle points at $(1.0, 5)$ and $(1.0, 6)$, corresponding to the local minima $x = 1.0$. Since $h(x)$ is non-negative, each saddle point has an associated constrained local minimum (according to Theorem 1). Hence, the solution space is the set of four saddle points or the set of two local minima. CSA is designed to locate the saddle points corresponding to the constrained global minimum $x^* = 1.0$ and $\lambda = \gamma$ at $(1.0, 6)$. ■

Let $\mathbf{x}_{opt} = \{(x^*, \gamma) | \; x^* \in X_{opt}\}$, and N_L be the maximum of the minimum number of transitions required to reach $\mathbf{x}_{opt}$ from all $\mathbf{x} \in S$. By properly constructing $\mathcal{N}(\mathbf{x})$, we state without proof that P_T is irreducible, and that N_L can always be found. This property is illustrated in Figure 2 in which any two nodes can reach each other.

Consider the sequence of temperatures $\{T_k, k = 0, 1, 2, \cdots\}$, where $T_k > T_{k+1}$ and $\lim_{k\to\infty} T_k = 0$, and set N_T, the number of trials per temperature, to be N_L. The following theorem proves the strong ergodicity of the Markov chain.

Theorem 2. *The inhomogeneous Markov chain is* strongly ergodic *if the sequence of temperatures $\{T_k\}$ satisfies:*

$$T_k \geq \frac{N_L \Delta_L}{log_e(k+1)} \tag{18}$$

where $\Delta_L = 2\max_{\mathbf{x}}\{|L(\mathbf{x}') - L(\mathbf{x})|, \mathbf{x}' \in \mathcal{N}(\mathbf{x})\}$.

This theorem can be proved by following the steps used to show weak ergodicity of SA [1] and by using the strong ergodicity conclusions [2]. Strongly ergodicity implies that the Markov chain has a unique stationary distribution π_T, where $\pi_T(\mathbf{x})$ is the probability of hitting point $\mathbf{x}$ during the search of CSA. The Markov chain in Figure 2 is strongly ergodic.

Our Markov chain also fits into the framework of generalized simulated annealing (GSA) [10] if we define an irreducible Markov kernel $P_T(\mathbf{x}, \mathbf{x}')$ and its associated communication cost $V(\mathbf{x}, \mathbf{x}')$:

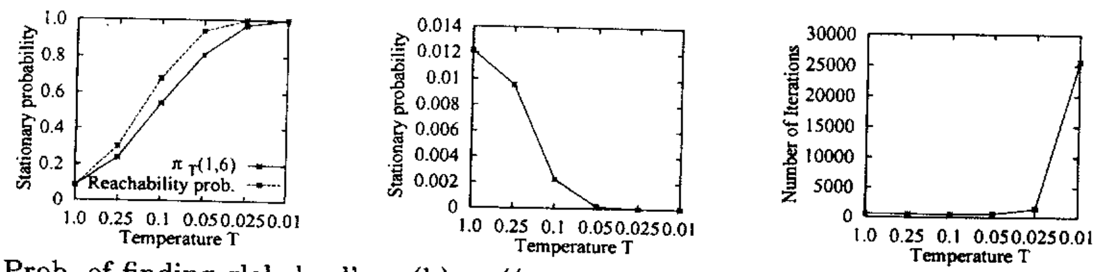
$$V(\mathbf{x}, \mathbf{x}') = \begin{cases} (L(\mathbf{x}') - L(\mathbf{x}))^+ \text{ if } \mathbf{x}' = (x', \lambda) \\ (L(\mathbf{x}) - L(\mathbf{x}'))^+ \text{ if } \mathbf{x}' = (x, \lambda') \end{cases} \tag{19}$$

Obviously $V(\mathbf{x}, \mathbf{x}') \geq 0$ and function V: $S \times S \to [0, +\infty]$.

Note that CSA does not minimize $L(x, \lambda)$ due to its probabilistic descents in x space and probabilistic ascents in λ space. This property is illustrated in Figure 2b in which the four saddle points are not at the global minimum in the Lagrangian space. It is quite different from SA for unconstrained problems, whose explicit objective is to minimize a single objective by performing probabilistic descents. In fact, CSA aims at minimizing an implicit virtual energy $W(\mathbf{x})$ according to GSA [10], and converges to the global minimum of $W(\mathbf{x})$ with probability one. Here, the virtual energy $W(\mathbf{x})$ is the cost of the minimum spanning tree rooted at point $\mathbf{x}$ of the digraph governed by $\mathcal{N}(\mathbf{x})$. Hence, to prove that our Markov chain converges asymptotically to the constrained global minimum of the original problem (1), we need to show that $W(\mathbf{x})$ is minimized at (x^*, γ) for $x^* \in X_{opt}$, and for all $x \in X - X_{opt}$ and $\lambda \in \Lambda$, $W((x^*, \gamma)) < W((x, \lambda))$. This is stated in the following theorem.

Theorem 3. *The Markov chain modeling CSA converges to the constrained global minimum $x^* \in X_{opt}$ with probability one.*

We sketch the proof of the theorem that consists of two steps. First, we show that for a given x, the virtual energy satisfies $W((x, \lambda')) \leq W((x, \lambda))$ for any $\lambda' > \lambda$. Hence, $W((0.6, 4)) \leq W((0.6, 3))$ and $W((0.8, 6)) \leq W((0.8, 2))$ in Figure 2. Second, we show that $W((x^*, \gamma)) < W((x, \gamma))$, where $x^* \in X_{opt}$ and $x \in X - X_{opt}$ at the maximum value γ of the Lagrange multipliers. Hence, $W((1.0, 6)) < W((0.6, 6))$ and $W((1.0, 6)) < W((0.8, 6))$ in Figure 2. Finally, we show that $W(\mathbf{x})$ of $\mathbf{x} = (x, \lambda)$ is minimized at (x^*, γ), and the Markov chain converges to the constrained global minimum $x^* \in X_{opt}$ with probability one.



(a) Prob. of finding global sol'n (b) $\pi_T((x = 1.2, \lambda = 2))$ (c) K_T to arrive at π_T

Fig. 3. Example showing the convergence probabilities of two states and the number of iterations to arrive at convergence. Part (a) also shows the reachability probability of reaching the global solution if the search were stopped.

Example 1 (cont'd). Since multiple runs of CSA do not illustrate the asymptotic convergence to the global minimum, we evaluate the stationary probabilities π_T numerically at a given T by first computing acceptance probability $A_T(\mathbf{x}, \mathbf{x}')$ using (13) and the one-step transition probability $P_T(\mathbf{x}, \mathbf{x}')$ using (15). The stationary distribution π_T of the Markov chain with transition matrix P_T is:

$$p(k+1) = p(k)P_T \qquad \text{for any given initial vector } p(k=0) \tag{20}$$

until $||p(k+1) - p(k)|| \leq \varepsilon$. Here we select $\varepsilon = 10^{-16}$ as the convergence precision, and denote the number of iterations by K_T. As $\pi_T = \lim_{k\to\infty} p(k)$, independent of starting vector $p(k=0)$, we set $p_i(k=0) = 1/|S|$ for all $i = 1, 2, \cdots, |S|$.

In this simple example, the virtual energy $W(\mathbf{x})$ is minimal at $\mathbf{x}^* = (1.0, 6)$. Figure 3a shows the stationary probability $\pi_T(\mathbf{x}^*)$ at this point, which increases monotonically as T is reduced, and approaches one as T is small ($T = 0.01$). Thus the Markov chain converges asymptotically to the constrained global solution. The figure also shows the reachability probability at each temperature, which is the probability of hitting the constrained global minimum in any of the previous iterations. The stationary probabilities $\pi_T(\mathbf{x})$ for other states $\mathbf{x} \neq \mathbf{x}^*$ decrease monotonically to zero as T is reduced. Figure 3b illustrates this property at $\tilde{\mathbf{x}} = (1.2, 2)$. Note that the process does not minimize $L(x, \lambda)$, whose minimum exists at $\tilde{\mathbf{x}}$; instead, it minimizes virtual energy $W(\mathbf{x})$ implicitly defined over communication cost $V(\mathbf{x}, \mathbf{x}')$ in (19). Finally, Figure 3c shows K_T required to reach stationary distribution π_T with precision ε. The process arrives at π_T quickly at high T, but needs a large number of iterations at low T. The reason is that at low T, the probability of escaping from a local minimum is small, and hence it takes a long time to arrive at the global solution.

5 Experimental Results on Constrained Problems

In this section, we apply CSA to solve general discrete, continuous, and mixed nonlinear constrained problems. Due to a lack of discrete/mixed benchmarks, we derive them from some existing continuous benchmarks [7, 5] as follows. In generating a mixed problem, we assume that variables with odd indices are continuous and those with even indices are discrete. In discretizing continuous variable x_i in the range $[a_i, b_i]$, if $b_i - a_i < 1$, we force the variable to take values from the set $A_i = \{a_i + \frac{b_i - a_i}{s} j\}, j = 0, 1, \cdots, s$; otherwise, we force it to take values from the set $A_i = \{a_i + \frac{1}{s} j\}, j = 0, 1, \cdots, (b_i - a_i)s$. Here, $s = 10^4$ if the number of variables n is less than 5, and $s = 10^5$ otherwise. The discrete/mixed search space produced is, therefore, very huge. We also shift every discretized variable x_i by a very small constant value in such a way that the set A_i contains the value of the best solution.

5.1 Implementation Details

In theory, any neighborhood $\mathcal{N}_1(x)$ and $\mathcal{N}_2(\lambda)$ that satisfy Condition (10) and Definition 1 will guarantee asymptotic convergence. In practice, however, it is important to choose appropriate neighborhoods and generate proper trial points in x and λ in order to solve constrained problems efficiently.

In our implementation, we choose a simple neighborhood $\mathcal{N}_1(x)$ as the set of points x' that differ from x in one variable x_i. We characterize $\mathcal{N}_1(x)$ by vector σ, where σ_i is the scale parameter in the Cauchy distribution along x_i. Similarly, we choose $\lambda' \in \mathcal{N}_2(\lambda)$ to differ from λ in one variable and characterize $\mathcal{N}_2(\lambda)$ by vector ϕ, where ϕ_i is the maximum possible perturbation along λ_i.

In generating trial point $\mathbf{x}' = (x', \lambda)$ from $\mathbf{x} = (x, \lambda)$, we consider two cases. To generate a continuous trial point, we set:

$$x' = x + r_0\, \theta_i\, \mathbf{e}_i \tag{21}$$

where r_0 is a random variable uniformly generated in the range $[-1, +1]$, $\mathbf{e}_i$ is a vector with its i^{th} component being 1 and the other components being 0, θ_i is generated from Cauchy distribution of density $f_d(x) = \frac{1}{\pi} \frac{\sigma_i}{\sigma_i^2 + x^2}$, and i is randomly generated from $\{1, 2, \cdots, n\}$. To generate a discrete trial point, we obtain x' by rounding the point generated by (21) to its closest discrete grid point. If it happens that $x' = x$, then we set $x' = x + s \times j$, where j has equal probability to take value $+1$ or -1.

In generating a trial point $\mathbf{x}' = (x, \lambda')$ from $\mathbf{x} = (x, \lambda)$, we apply the following:

$$\lambda' = \lambda + r_1\, \phi_j\, \mathbf{e}_j \tag{22}$$

where r_1 is randomly generated from $[-1, +1]$, and j is uniformly generated from $\{1, 2, \cdots, m\}$.

The trial point $\mathbf{x}' = (x', \lambda)$ or $\mathbf{x}' = (x, \lambda')$ generated is accepted according to probability (13). We set the ratio of generating (x', λ) and (x, λ') from the current point (x, λ) to be $20n$ to m, meaning that x is updated more frequently than λ.

During the course of CSA, we dynamically adjust the neighborhood $\mathcal{N}_1(x)$ by updating scale vector σ for x using a modified $1:1$ rate rule [4] in order to balance the ratio between accepted and rejected configurations.

$$\sigma_i = \begin{cases} \sigma_i \left[1 + \beta_0 (p_i - p_u)/(1 - p_u)\right] & \text{if } p_i > p_u \\ \sigma_i / \left[1 + \beta_1 (p_v - p_i)/p_v\right] & \text{if } p_i < p_v \end{cases} \tag{23}$$

where p_i is the ratio of accepting x' in which x'_i differs from x_i. We chose the parameters experimentally: $\beta_0 = 7$, $\beta_1 = 2$, $p_u = 0.3$, and $p_v = 0.2$. If p_i is low, then too many trials of (x', λ) are rejected, and σ_i is reduced. In contrast, if p_i is high, then trial points (x', λ) are too close to (x, λ), and σ_i is increased.

We adjust ϕ according to the degree of constraint violations, where

$$\phi = w \otimes h(x) = [w_1 h_1(x), w_2 h_2(x), \cdots, w_m h_m(x)] \tag{24}$$

and $\otimes$ represents vector product. When $h_i(x)$ is satisfied, λ_i does not need to be updated; hence, $\phi_i = 0$. In contrast, when a constraint is not satisfied, we adjust ϕ_i by modifying w_i according to how fast $h_i(x)$ is changing:

$$w_i = \begin{cases} \eta_0 \, w_i & \text{if } \; h_i(x) > \tau_0 T \\ \eta_1 \, w_i & \text{if } \; h_i(x) < \tau_1 T \end{cases} \tag{25}$$

where $\eta_0 = 1.25$, η_1=0.8, $\tau_0 = 1.0$, and $\tau_1 = 0.01$ were chosen experimentally. When $h_i(x)$ is reduced too quickly (*i.e.*, $h_i(x) < \tau_1 T$), $h_i(x)$ is over-weighted, leading to possibly poor objective values or difficulty in satisfying other under-weighted constraints. Hence, we reduce λ_i's neighborhood. In contrast, if $h_i(x)$ is reduced too slowly (*i.e.*, $h_i(x) > \tau_0 T$), we enlarge λ_i's neighborhood in order to improve its chance of satisfaction. Note that w_i is adjusted using T as a reference because constraint violations are expected to decrease when T decreases.

5.2 Evaluation Results

In this section, we show the results of applying CSA on 10 constrained optimization problems G1-G10 [7, 5] with objective functions of various types (linear, quadratic, cubic, polynomial, and nonlinear) and constraints of linear inequalities, nonlinear equalities, and nonlinear inequalities. The number of variables is up to 20, and that of constraints, including simple bounds, is up to 42.

These problems were originally invented to be solved by evolutionary algorithms (EAs) using well-tuned constraint handling techniques for each problem in order to get good results. Examples of these techniques include keeping the search within feasible regions with some specific genetic operators, and dynamic and adaptive penalty methods.

We also solved the continuous problems using DONLP2 [9], a popular sequential quadratic programming (SQP) package that uses derivatives. SQP is an efficient local-search method widely used for solving constrained optimization problems, whose quality depends heavily on starting points. To be fair, we ran DONLP2 from multiple starting points using the same amount of average CPU time as one run of CSA for continuous problems.

Table 1 shows the comparison results for continuous problems. The first two columns show the problem identifiers and the constrained global minima (or maxima), if known. The third and fourth columns show the best solutions obtained by EAs and the specific constraint handling techniques used to generate the solutions. The fifth thru sixth columns show the best solutions of SQP, and

Table 1. Comparison results of DONLP2, GA, and CSA for 10 continuous problems. (S.T. stands for strategic oscillation, H.M. for homomorphous mappings, and D.P. for dynamic penalty. All runs were done on a Sun SparcStation Ultra 60 computer. Numbers in bold represent the best solution.)

Problem ID	Global Solution	EAs		SQP: DONLP2		CSA (average of 20 runs)		
		best sol'n	specific method	best sol'n	% with best sol'n	best sol'n	% with best sol'n	time (sec.) per run
G1 (min)	-15	**-15**	Genocop	**-15**	14.8%	**-15**	**100%**	17.04
G2 (max)	unknown	0.803553	S.T.	0.640329	0.4%	**0.803619**	**100%**	94.03
G3 (max)	1.0	0.999866	S.T.	**1.0**	93.1%	**1.0**	**100%**	69.62
G4 (min)	-30665.5	-30664.5	H.M.	**-30665.5**	61.4%	**-30665.5**	**100%**	2.70
G5 (min)	unknown	5126.498	D.P.	**4221.956**	94.0%	**4221.956**	**100%**	3.81
G6 (min)	-6961.81	**-6961.81**	Genocop	**-6961.81**	87.2%	**-6961.81**	**100%**	0.978
G7 (min)	24.3062	24.62	H.M.	**24.3062**	99.3%	**24.3062**	**100%**	14.71
G8 (max)	unknown	**0.095825**	H.M.	**0.095825**	44.9%	**0.095825**	**100%**	1.22
G9 (min)	680.63	680.64	Genocop	**680.63**	99.7%	**680.63**	**100%**	5.61
G10 (min)	7049.33	7147.9	H.M.	**7049.33**	29.1%	**7049.33**	**100%**	10.81

Table 2. CSA results on 10 derived discrete and mixed problems with 20 runs per problem. (All runs are done on a Sun SparcStation Ultra 60 computer.)

Problem ID	CSA for Discrete Problems			CSA for Mixed Problems		
	best sol'n	% with best sol'n	time (sec.) per run	best sol'n	% with best sol'n	time (sec.) per run
G1 (min)	**-15**	100%	17.97	**-15**	100%	21.56
G2 (max)	**0.803619**	90%	99.03	**0.803619**	100%	100.62
G3 (max)	**1.0**	100%	70.15	**1.0**	100%	74.78
G4 (min)	**-30665.5**	100%	2.43	**-30665.5**	100%	3.04
G5 (min)	**4221.956**	90%	3.32	**4221.956**	100%	4.16
G6 (min)	**-6961.81**	90%	0.962	**-6961.81**	100%	1.00
G7 (min)	**24.3062**	100%	17.48	**24.3062**	95%	17.39
G8 (max)	**0.095825**	100%	1.37	**0.095825**	100%	1.33
G9 (min)	**680.63**	100%	6.83	**680.63**	100%	6.76
G10 (min)	**7049.33**	100%	13.31	**7049.33**	100%	12.57

the percentage of runs reaching these solutions. The last three columns give the results of CSA and the average CPU time per run.

The results show that CSA is the best in terms of both solution quality and chance of reaching the best solutions. For problems with known global optima, CSA always found these optima, independent of starting points. For problems with unknown global optima, CSA always found the best solutions in every run. DONLP2, in contrast, worked well for problems with a small number of local optima if enough starting points were used. For these problems, DONLP2 was generally very fast and was able to complete within one second in many runs. However, DONLP2 has difficulty with G2, a maximization problem with a huge number of local maxima. In 243 runs of DONLP2, only one was able to find the best solution of 0.640329, which is much worse than those obtained by EAs (0.803553) and CSA (0.803619). Even with 10,000 runs, DONLP2 was only able to find the best solution of 0.736554. Finally, EA was only able to find the best solutions in three of the ten problems despite extensive tuning.

Table 2 shows the results of applying CSA to solve derived discrete and mixed problems. It shows that CSA can find the best solutions with high success ratios (larger than or equal to 90% for every problem). Overall, the experimental results indicate the robustness and wide applicability of CSA to solve discrete, continuous, and mixed nonlinear constrained problems.

6 Conclusions

We have reported in this paper a new algorithm called constrained simulated annealing that can achieve asymptotic convergence for solving nonlinear discrete constrained optimization problems.

1 It is based on a strong mathematical foundation of the theory of discrete Lagrange multipliers. By looking for saddle points in the discrete Lagrangian space, it can find the saddle point with the best objective value with asymptotic convergence. This amounts to finding the constrained global optimum.
2 It does not require derivatives of the objective and constraint functions, and can be applied to solve discrete, continuous, and mixed problems. To our knowledge, this is the first algorithm that can solve efficiently discrete, mixed, and continuous constrained optimization problems.
3 Even though CSA requires exponential time to have asymptotic convergence, it has higher reachability probability than algorithms using random restarts because its probability of hitting the global optimum increases with time rather than constant.

References

1. E. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines.* J. Wiley and Sons, 1989.
2. S. Anily and A Federgruen. Simulated annealing methods with general acceptance probabilities. *Journal of Appl. Prob.*, 24:657–667, 1987.
3. D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, 1982.
4. A. Corana, M. Marchesi, C. Martini, and S. Ridella. Minimizing multimodal functions of continuous variables with the simulated annealing algorithm. *ACM Trans. on Mathematical Software*, 13(3):262–280, 1987.
5. S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7(1):19–44, 1999.
6. D. G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley Publishing Company, 1984.
7. Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32, 1996.
8. Y. Shang and B. W. Wah. A discrete Lagrangian based global search method for solving satisfiability problems. *J. Global Optimization*, 12(1):61–99, January 1998.
9. P. Spellucci. An SQP method for general nonlinear programs using only equality constrained subproblems. *Mathematical Programming*, 82:413–448, 1998.
10. A. Trouve. Cycle decomposition and simulated annealing. *SIAM Journal on Control and Optimization*, 34(3):966–986, 1996.
11. B. W. Wah and Z. Wu. The theory of discrete lagrange multipliers for nonlinear discrete optimization. *Principles and Practice of Constraint Programming*, (accepted to appear) October 1999.
12. Zhe Wu. *Discrete Lagrangian Methods for Solving Nonlinear Discrete Constrained Optimization Problems.* M.Sc. Thesis, Dept. of Computer Science, Univ. of Illinois, Urbana, IL, May 1998.