Constrained Global Optimization by Constraint Partitioning and Simulated Annealing*

Benjamin W. Wah,† *Yixin Chen*, and *Andrew Wan*‡

Abstract

In this paper, we present *constraint-partitioned simulated annealing* (CPSA), an algorithm that extends our previous constrained simulated annealing (CSA) for constrained optimization. The algorithm is based on the theory of extended saddle points (ESPs). By decomposing the ESP condition into multiple necessary conditions, CPSA partitions a problem by its constraints into subproblems, solves each independently using CSA, and resolves those violated global constraints across the subproblems. Because each subproblem is exponentially simpler and the number of global constraints is very small, the complexity of solving the original problem is significantly reduced. We state without proof the asymptotic convergence of CPSA with probability one to a constrained global minimum in discrete space. Last, we evaluate CPSA on some continuous constrained benchmarks.

1 Problem Definition

A general *mixed-integer nonlinear programming problem* (MINLP) is formulated as follows:

$$(P_m): \quad \min_{z} \quad f(z) \qquad (1)$$
$$\text{subject to} \quad h(z)=0 \ \text{ and } \ g(z)\le 0,$$

where $z=(x,y)\in\mathcal{Z}$; $x\in\mathbb{R}^v$ and $y\in\mathbb{D}^w$ are, respectively, bounded continuous and discrete variables; $f(z)$ is a lower-bounded objective function; $g(z)=(g_1(z),\ldots,g_r(z))^T$ is a vector of r inequality constraint functions, and $h(z)=(h_1(z),\ldots,h_m(z))^T$ is a vector of m equality constraint functions. Functions $f(z)$, $g(z)$, and $h(z)$ can be discontinuous, non-differentiable, and not in closed form.

Our work is based on the theoretical conditions for finding locally optimal and feasible solutions to P_m. We first define the following basic terms.

Definition 1. A *mixed neighborhood* $\mathcal{N}_m(z)$ for $z=(x,y)$ in mixed space $\mathbb{R}^v\times\mathbb{D}^w$ is:

$$\{(x',y)\mid x'\in\mathcal{N}_c(x)\}\cup\{(x,y')\mid y'\in\mathcal{N}_d(y)\},$$

where $\mathcal{N}_c(x)=\{x':\|x'-x\|\le\epsilon \text{ and } \epsilon\to 0\}$ is the *continuous neighborhood* of x, and the *discrete neighborhood* $\mathcal{N}_d(y)$ is a *finite* user-defined set of points $\{y'\in\mathbb{D}^w\}$ such that $y'\in\mathcal{N}_d(y)\iff y\in\mathcal{N}_d(y')$. Here, $\epsilon\to 0$ means that ϵ is arbitrarily close to 0.

Definition 2. Point z^* is a CLM_m, a *constrained local minimum* of P_m with respect to points in $\mathcal{N}_m(z^*)$, if z^* is feasible ($h(z)=0$ and $g(z)\le 0$) and $f(z^*)\le f(z)$ for all feasible $z\in\mathcal{N}_m(z^*)$.

Definition 3. Point z^* is a *constrained global minimum* (CGM_m) of P_m iff z^* is feasible, and $f(z^*)\le f(z')$ for every feasible $z'\in\mathcal{Z}$.

A popular method for solving P_m is the penalty method that requires expensive global optimization, making it computationally intractable for large problems. To overcome this difficulty, we have recently proposed the theory of extended saddle points (ESPs) in mixed space [8] that shows the one-to-one correspondence between a CLM_m of P_m and an ESP of the corresponding penalty function. The ESP condition allows us to find a CLM_m of P_m by looking for the associated ESP of the penalty function.

We have previously proposed constrained simulated annealing (CSA) that looks for ESPs by performing stochastic descents in the original variable space and stochastic ascents in the penalty space. The algorithm is shown to asymptotically converge to a constraint global minimum with probability one for discrete constrained optimization problems.

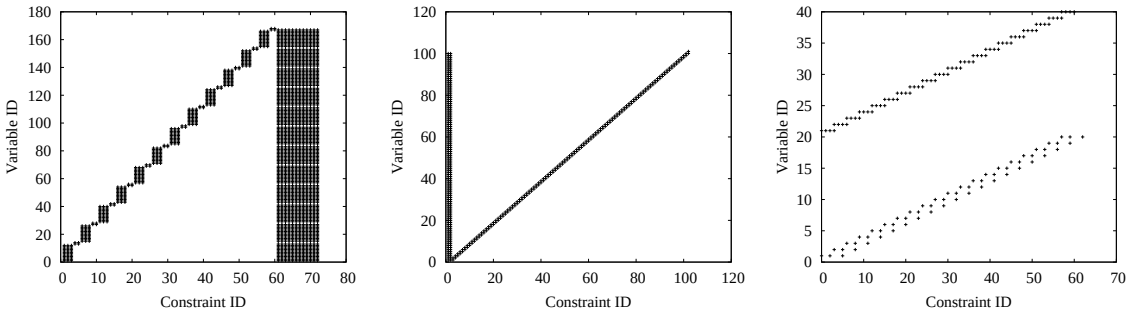
A major drawback of CSA is that its sampling-based search is too slow to be useful for solving large

*Research supported by National Science Foundation Grant ITR 03-12084 and a Department of Energy Early Career Principal Investigator grant.

†B. W. Wah is with the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801, http://manip.crhc.uiuc.edu, wah@uiuc.edu

‡Y. Chen and A. Wan are with the Department of Computer Science and Engineering, Washington University, St Louis, MO 63130, {chen, qw2}@cse.wustl.edu.

Proc. IEEE International Conference on Tools with Artificial Intelligence, 2006.



a) TRIMLOSS b) ORTHREGC c) OPTCDEG3

Figure 1: Strongly regular constraint-variable structures in some continuous constrained optimization benchmarks. A dot in each graph represents a variable involved in a constraint.

problems. This paper proposes a constraint partitioning approach that improves the efficiency of CSA by exploiting the constraint structure of MINLPs.

We have observed that the constraints of many application problems do not involve variables randomly picked from their variable sets. Invariably, many constraints in existing benchmarks are highly structured because they model spatial or temporal relationships with strong locality, such as those in physical structures, optimal control, and staged processing. Figure 1 illustrates this point by depicting the regular constraint structure of three benchmarks. It shows a dot where a constraint (with a unique ID on the x axis) is related to a variable (with a unique ID on the y axis). When the order of the variables and the constraints are properly arranged, we see some strongly regular constraint-variable structures.

By exploiting the strong locality of constraints on variables in P_m, we develop in this paper constraint-partitioned simulated annealing (CPSA) that partitions P_m into multiple subproblems related by a few global constraints. CPSA independently solves each subproblem using CSA and iteratively resolves those inconsistent global constraints. We prove the convergence of CPSA to optimal ESPs in discrete constrained optimization and report significant efficiency improvements over CSA.

2 Previous Work

Direct and penalty methods are two general approaches for solving P_m. Since direct methods are only effective for solving some special cases of P_m, we focus on penalty methods in this paper.

2.1 Global Optimal Penalty Methods

Penalty methods belong to a general approach that can solve continuous, discrete, and mixed constrained optimization problems, with no continuity, differentiability, and convexity requirements. A penalty function of P_m is a summation of its objective and constraint functions weighted by penalties. The goal of a penalty method is to find suitable penalties such that z^* that minimizes the penalty function corresponds to either a CLM_m or a CGM_m of P_m.

A *static-penalty method* [6] typically minimizes the following penalty function with constant $\rho > 0$:

$$(P_1): \quad f(z) + \sum_{i=1}^{m} \alpha_i \, |h_i(z)|^\rho + \sum_{j=1}^{r} \beta_j \, (g_j(z)^+)^\rho,$$

where $g^+ = \max(0, g)$ for function g. Here, z^* is a CGM_m of P_m iff there exist finite $\alpha^* \geq 0$ and $\beta^* \geq 0$ such that z^* is a global minimum of P_1 for any $\alpha > \alpha^*$ and $\beta > \beta^*$. The approach is limited because the penalties have to be found by trial and error, where each trial involves an expensive global minimization.

A *dynamic-penalty method* [6] gradually increases the penalties, finds the global minimum z^* for the given combination of penalties, and stops when z^* is a feasible solution to P_m. It has the same limitation as static-penalty methods because it requires computationally expensive algorithms for finding the global minima of nonlinear functions.

There are many variations of dynamic penalty methods. A well-known one is the *non-stationary method* (NS) [5] that minimizes the following penalty function in iteration t:

$$(P_2): f(z) + \sum_{i=1}^{m} \alpha_i(t) \, |h_i(z)|^\rho + \sum_{j=1}^{r} \beta_j(t) (g_j(z)^+)^\rho$$
$$\text{where } \alpha_i(t+1) = \alpha_i(t) + C \cdot |h_i(z(t))|$$
$$\beta_j(t+1) = \beta_j(t) + C \cdot g_j(z(t))^+.$$

Here, C and ρ are constant parameters, with a reasonable setting of $C = 0.01$ and $\rho = 2$.

The *adaptive penalty method* (AP) [1] makes use

of a feedback from the search by minimizing the following penalty function in iteration t:

$$(P_3): \quad f(z) + \sum_{i=1}^{m} \alpha_i(t)\, h_i(z)^2 + \sum_{j=1}^{r} \beta_j(t)(g_j(z)^+)^2,$$

where $\alpha_i(t)$ is, respectively, increased to $\frac{\alpha_i(t)}{\lambda_1}$, decreased to $\lambda_2\alpha_i(t)$, or left unchanged when the constraint $h_i(z) = 0$ is respectively, infeasible, feasible, or neither in the last ℓ iterations. We use $\ell = 3$, $\lambda_1 = 1.5$, and $\lambda_2 = 1.25$ in our experiments.

The *threshold penalty method* estimates and dynamically adjusts a near-feasible threshold $q_i(t)$ (*resp.* $q'_j(t)$) for each constraint in iteration t. Each threshold indicates a reasonable amount of violation allowed for promising points. The penalty function used is:

$$(P_4): \quad f(z) + \alpha(t) \sum_{i=1}^{m} \left(\frac{h_i(z)}{q_i(t)}\right)^2 + \sum_{j=1}^{r} \left(\frac{g_j(z)^+}{q'_j(t)}\right)^2.$$

In summary, existing penalty methods are limited because they were designed for finding a CGM_m of P_m by looking for a global minimum of the penalty function. The process is computationally expensive for highly nonlinear functions.

2.2 ESP Condition for CLM_m

We summarize in this section the theory of ESPs and the partitioning of the ESP condition into multiple necessary conditions. Because the results have been published [8], we only summarize some high-level concepts without the precise formalism and their proofs.

Definition 4. For penalty vectors $\alpha \in \mathbb{R}^m$ and $\beta \in \mathbb{R}^r$, we define a *penalty function* of P_m as follows:

$$L_m(z, \alpha, \beta) = f(z) + \alpha^T |h(z)| + \beta^T g(z)^+. \qquad (2)$$

Next, we informally define a constraint qualification condition [8]. Consider a point $z' = (x', y')$ and a neighboring point $z'' = (x' + \epsilon\, \vec{p}, y')$ for an infinitely small perturbation in direction $\vec{p} \in X$. When the *constraint-qualification condition* is satisfied at z', it means that there is no $\vec{p}$ such that the rates of change for all equality and active inequality constraints between z'' and z' are all zeroes.

Theorem 1. *Necessary and sufficient condition on CLM_m of P_m* [8]. Assuming $z^* \in \mathbb{R}^v \times \mathbb{D}^w$ satisfies the constraint-qualification condition, then z^* is a CLM_m of P_m iff there exist some finite $\alpha^* \geq 0$ and $\beta^* \geq 0$ that satisfies the following *extended saddle-point condition* (ESPC):

$$L_m(z^*, \alpha, \beta) \;\leq\; L_m(z^*, \alpha^{**}, \beta^{**}) \;\leq\; L_m(z, \alpha^{**}, \beta^{**})$$

for any $\alpha^{**} > \alpha^*$ and $\beta^{**} > \beta^*$ and for all $z \in \mathcal{N}_m(z^*)$, $\alpha \in \mathbb{R}^m$, and $\beta \in \mathbb{R}^r$.

An important feature of the ESPC in Theorem 1 is that it can be partitioned in such a way that each subproblem implementing a partitioned condition can be solved by looking for any α^{**} and β^{**} that are larger than α^* and β^*.

Consider P_t, a version of P_m whose constraints can be partitioned into N subsets, in addition to the global constraint functions $H = (H_1, \ldots, H_p)^T$ and $G = (G_1, \ldots, G_q)^T$:

$$\begin{array}{lll} (P_t): & \min\limits_{z} & f(z) \\ & \text{where} & h^{(t)}(z(t)) = 0, \quad g^{(t)}(z(t)) \leq 0 \\ & \text{and} & H(z) = 0, \qquad\quad G(z) \leq 0. \end{array}$$

Each subset of constraints can be treated as a subproblem, where Subproblem t, $t = 1, \ldots, N$, has local *state vector* $z(t) = (z_1(t), \ldots, z_{u_t}(t))^T$ of u_t mixed variables, and $\cup_{t=1}^{N} z(t) = z$. Here, $z(t)$ includes all the variables that appear in any of the m_t local equality constraint functions $h^{(t)} = (h_1^{(t)}, \ldots, h_{m_t}^{(t)})^T$ and the r_t local inequality constraint functions $g^{(t)} = (g_1^{(t)}, \ldots, g_{r_t}^{(t)})^T$. Since the partitioning is by constraints, $z(1), \ldots, z(N)$ may overlap with each other.

Definition 5. $\mathcal{N}_p(z)$, the *mixed neighborhood* of z for P_t when P_m is partitioned by its constraints, is:

$$\mathcal{N}_p(z) = \bigcup_{t=1}^{N} \mathcal{N}_p^{(t)}(z) \text{ and}$$

$$\mathcal{N}_p^{(t)}(z) = \left\{ z' \mid z'(t) \in \mathcal{N}_m(z(t)); z'_i = z_i \;\forall z_i \notin z(t) \right\}.$$

Here, $\mathcal{N}_m(z(t))$ is the mixed neighborhood of $z(t)$.

Definition 6. Let $\Phi(z, \gamma, \eta) = \gamma^T |H(z)| + \eta^T G(z)^+$ be the sum of the transformed global constraint functions weighted by their penalties, where $\gamma = (\gamma_1, \ldots, \gamma_p)^T \in \mathbb{R}^p$ and $\eta = (\eta_1, \ldots, \eta_q)^T \in \mathbb{R}^q$ are the penalty vectors for the global constraints. Then the penalty function for P_t and that for Subproblem t are defined as follows:

$$L_m(z, \alpha, \beta, \gamma, \eta) = f(z) + \sum_{t=1}^{N} \Big(\alpha(t) |h^{(t)}(z(t))| + \beta(t) g^{(t)}(z(t))^+ \Big) + \Phi(z, \gamma, \eta) \qquad (3)$$

$$\Gamma_m(z, \alpha(t), \beta(t), \gamma, \eta) = f(z) + \alpha(t) |h^{(t)}(z(t))| + \beta(t) g^{(t)}(z(t))^+ + \Phi(z, \gamma, \eta), \qquad (4)$$

1. **procedure CSA**
2. set starting point $\mathbf{z} = (z, \alpha)$;
3. set starting temperature $T = T_0$;
4. set cooling rate $0 < \kappa < 1$ and N_T;
5. **while** stopping condition is not satisfied **do**
6. **for** $k \leftarrow 1$ **to** N_T **do**
7. generate $\mathbf{z}' \in \mathcal{N}_m(\mathbf{z})$ using $G(\mathbf{z}, \mathbf{z}')$;
8. accept $\mathbf{z}'$ with probability $A_T(\mathbf{z}, \mathbf{z}')$;
9. **end_for**
10. reduce temperature by $T \leftarrow \kappa\, T$;
11. **end_while**
12. **end_procedure**

Figure 2: Constrained simulated annealing [10].

where $\alpha(t) = (\alpha_1(t), \ldots, \alpha_{m_t}(t))^T \in \mathbb{R}^{m_t}$ and $\beta(t) = (\beta_1(t), \ldots, \beta_{r_t}(t)) \in \mathbb{R}^{r_t}$ are the penalty vectors for the local constraints in Subproblem t.

Theorem 2. *Partitioned necessary and sufficient ESPC on CLM_m of P_t* [8]. Given $\mathcal{N}_p(z)$, the ESPC in Theorem 1 can be rewritten into $N+1$ necessary conditions that, collectively, are sufficient:

$$\begin{aligned} &\Gamma_m(z^*, \alpha(t), \beta(t), \gamma^{**}, \eta^{**}) \qquad (5)\\ &\quad \le \Gamma_m(z^*, \alpha(t)^{**}, \beta(t)^{**}, \gamma^{**}, \eta^{**})\\ &\quad \le \Gamma_m(z, \alpha(t)^{**}, \beta(t)^{**}, \gamma^{**}, \eta^{**}), \quad t = 1..N\\ &L_m(z^*, \alpha^{**}, \beta^{**}, \gamma, \eta) \qquad (6)\\ &\quad \le L_m(z^*, \alpha^{**}, \beta^{**}, \gamma^{**}, \eta^{**}), \end{aligned}$$

for any $\alpha(t)^{**} > \alpha(t)^* \ge 0$, $\beta(t)^{**} > \beta(t)^* \ge 0$, $\gamma^{**} > \gamma^* \ge 0$, and $\eta^{**} > \eta^* \ge 0$, and for all $z \in \mathcal{N}_p^{(t)}(z^*)$, $\alpha(t) \in \mathbb{R}^{m_t}$, $\beta(t) \in \mathbb{R}^{r_t}$, $\gamma \in \mathbb{R}^p$, $\eta \in \mathbb{R}^q$.

Theorem 2 shows that the original ESPC in Theorem 1 can be partitioned into N necessary conditions in (5) and an overall necessary condition in (6) on the global constraints across the subproblems.

The major benefit of this decomposition is that each subproblem involves only a fraction of the original constraints and is a significant relaxation of the original problem with exponentially lower complexity. This leads to a large reduction in the complexity of the original problem if the number of global constraints is small and can be resolved efficiently.

2.3 Constrained Simulated Annealing

Figure 2 outlines CSA [10], an algorithm for finding z^* and the corresponding α^{**} and β^{**} that satisfy the ESPC. Without loss of generality, we only consider P_m with equality constraints, since inequality constraints can be handled in a similar fashion. We summarize in this section the steps of CSA [10].

Line 3 initializes control parameter *temperature* T to be so large that almost any trial point $\mathbf{z}'$ will be accepted. In our experiments on continuous problems, we set the initial T by first randomly generating 100 points of x and their corresponding neighbors $x' \in \mathcal{N}_c(x)$ in close proximity, where $|x'_i - x_i| \le 0.001$, and then setting $T = \max_{x,x',i}\{|L_m(x',1) - L_m(x,1)|, |h_i(x)|\}$.

Line 4 sets the initial κ to 0.8 and the number of iterations at each temperature. In our experiments, we choose $N_T = \zeta(20(v+w)+m)$ where $\zeta = 10(v+w+m)$, $v+w$ is the number of mixed variables, and m is the number of equality constraints.

Line 7 generates a random trial point $\mathbf{z}' \in \mathcal{N}_m(\mathbf{z})$ from the current point $\mathbf{z} \in \mathcal{S} = \mathcal{Z} \times \Lambda$. In our implementation, $\mathcal{N}_m(\mathbf{z})$ consists of (z', α) and (z, α'), where $z' \in \mathcal{N}'_m(z)$ is a point neighboring to z in the $\mathcal{Z}$ subspace (Definition 1), and $\alpha' \in \mathcal{N}''_m(\alpha)$ is a point neighboring to α in the Λ subspace when $h(z) \neq 0$.

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

$$A_T(\mathbf{z}, \mathbf{z}') = \begin{cases} exp\left(-\frac{(L_m(\mathbf{z}') - L_m(\mathbf{z}))^+}{T}\right) & \text{if } \mathbf{z}' = (z', \alpha) \\ exp\left(-\frac{(L_m(\mathbf{z}) - L_m(\mathbf{z}'))^+}{T}\right) & \text{if } \mathbf{z}' = (z, \alpha'). \end{cases}$$

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

$$T \longleftarrow \kappa \cdot T \quad \text{where } \kappa < 1. \qquad (7)$$

CSA provides a systematic way for finding an optimal ESP. For discrete problems, the process can be modeled by a non-homogeneous Markov chain, which can be proved to converge to a constrained global minimum CGM_d with probability one.

Theorem 3. [10, 12] The Markov chain modeling CSA converges to a CGM_d with probability one as $k \to \infty$ when a logarithmic cooling schedule is used.

3 Constraint-Partitioned SA

We now present constraint-partitioned simulated annealing (CPSA), an extension of CSA that decomposes the constraints of P_m into $N+1$ subsets as defined by P_t. Figure 3 illustrates the idea in CPSA. Unlike the original CSA that solve the problem as a whole, CPSA solves each subproblem independently. In Subproblem t, $t = 1, ..., N$, CSA is performed in the smaller space of those variables $z(t)$ and $\alpha(t)$ related to the local constraints $h^{(t)}(z(t)) = 0$. In addition, there is a global search that probes in the space of the variables z and γ on the global constraints

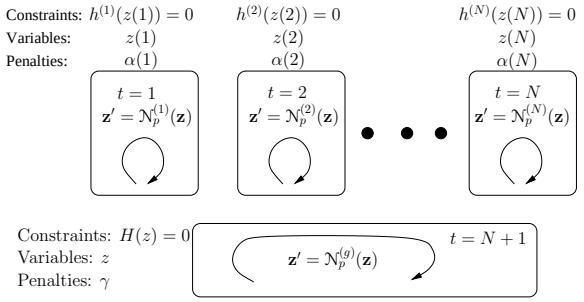


Figure 3: Illustration of the CPSA procedure.

```
1.  procedure CPSA
2.      set starting point z = (z, α, γ);
3.      set starting temperature T = T^0;
4.      set cooling rate 0 < κ < 1 and N_T;
5.      while stopping condition is not satisfied do
6.          for k ← 1 to N_T do
7.              set t to be random between 1 and N + 1;
8.              if 1 ≤ t ≤ N then
9.                  generate z' ∈ N_p^(t)(z) using G^(t)(z, z');
10.                 accept z' with probability A_T(z, z');
11.             else /* t = N + 1 */
12.                 generate z' ∈ N_p^(g)(z) using G^(g)(z, z');
13.                 accept z' with probability A_T(z, z');
14.             end_if
15.         end_for
16.         reduce temperature by T ← κ T;
17.     end_while
18. end_procedure
```

Figure 4: The CPSA search procedure.

$H(z) = 0$. This additional search is needed for resolving any violated global constraint and for ensuring the asymptotic convergence of CPSA.

The CPSA procedure. Figure 4 describes the CPSA procedure. The first six lines are similar to those in CSA [10] in Figure 2.

To facilitate the convergence analysis of CPSA in a Markov-chain model, the algorithm is designed to randomly pick a subproblem in each step, instead of deterministically enumerating the subproblems in a round-robin fashion. The approach leads to a memoryless Markovian process in CPSA. This is done in Line 7, where we randomly pick a subproblem between 1 and $N + 1$ with probability $p_t(i)$. Here, the probability that $i = 1, \ldots, N + 1$ can be arbitrarily chosen as long as $\sum_{i=1}^{N+1} p_t(i) = 1$ and $p_t(i) > 0$.

When t is between 1 and N (Line 8), it represents a local exploration step. In this case, Line 9 generates a trial point $\mathbf{z}'$ in the neighborhood $\mathcal{N}_p^{(t)}(\mathbf{z})$ of the current point $\mathbf{z} = (z, \alpha, \gamma) \in \mathcal{S}$ by perturbing $z(t)$ and $\alpha(t)$:

$$\begin{aligned}
\mathcal{N}_p^{(t)}(\mathbf{z}) &= \{(z', \alpha, \gamma) \in \mathcal{S} \mid z' \in \mathcal{N}_{p_1}^{(t)}(z)\} \cup \\
&\quad \{(z, \alpha', \gamma) \in \mathcal{S} \mid \alpha' \in \mathcal{N}_{p_2}^{(t)}(\alpha) \text{ and } h^{(t)}(z(t)) \neq 0\} \\
\mathcal{N}_{p_1}^{(t)}(z) &= \{z' \mid z'(t) \in \mathcal{N}_m(z(t)); \forall z_i \notin z(t), z_i' = z_i\} \\
\mathcal{N}_{p_2}^{(t)}(\alpha) &= \{\alpha' \mid \alpha'(t) \in \mathbb{R}^{m_t}; \forall \alpha_i \notin \alpha(t), \alpha_i' = \alpha_i\}.
\end{aligned}$$

The trial point is accepted with the Metropolis probability $A_T(\mathbf{z}, \mathbf{z}')$ as in CSA (Line 10). Intuitively, $\mathcal{N}_p^{(t)}(\mathbf{z})$ is the neighborhood of the original variable vector $z(t)$ and the penalty vector $\alpha(t)$.

In solving Subproblem t, we generate $\mathbf{z}' \in \mathcal{N}_p^{(t)}(\mathbf{z})$ with a generation probability $G^{(t)}(\mathbf{z}, \mathbf{z}')$ that can be arbitrarily defined as long as the following is satisfied:

$$0 \leq G^{(t)}(\mathbf{z}, \mathbf{z}') \leq 1 \text{ and } \sum_{\mathbf{z}' \in \mathcal{N}_p^{(t)}(\mathbf{z})} G^{(t)}(\mathbf{z}, \mathbf{z}') = 1. \quad (8)$$

When $t = N + 1$ (Line 11), it represents a global exploration step. In this case, Line 12 generates a random trial point $\mathbf{z}' \in \mathcal{N}_p^{(g)}(\mathbf{z})$ where:

$$\begin{aligned}
\mathcal{N}_p^{(g)}(\mathbf{z}) &= \{(z, \alpha, \gamma) \in \mathcal{S} \mid z' \in \mathcal{N}_{p_1}^{(g)}(z)\} \\
&\quad \cup \{(z, \alpha, \gamma') \in \mathcal{S} \mid \gamma' \in \mathcal{N}_{p_2}^{(g)}(\gamma) \text{ and } H(z) \neq 0\} \\
\mathcal{N}_{p_1}^{(g)}(z) &= \bigcup_{t=1}^{N} \{z' | z'(t) \in \mathcal{Z}; \forall z_i \notin z(t), z_i' = z_i\} \\
\mathcal{N}_{p_2}^{(g)}(\gamma) &= \left\{\gamma' \;\middle|\; \gamma' \in \mathbb{R}^q\right\}.
\end{aligned}$$

Intuitively, new probes in $\mathcal{N}_p^{(g)}(\mathbf{z})$ achieves a global exploration by updating γ and by perturbing z in the $\mathcal{Z}$ subspace. $\mathcal{N}_p^{(g)}(\mathbf{z})$ ensures the ergodicity of the underlying Markov chain, which is required for achieving asymptotic convergence.

Again, the trial point is accepted with probability $A_T(\mathbf{z}, \mathbf{z}')$ as in CSA (Line 13).

When compared to CSA, CPSA reduces the search complexity through constraint partitioning. Since both CSA and CPSA need to converge to an equilibrium distribution of variables at a given temperature before the temperature is reduced, the total search time depends on the convergence time at each temperature. By partitioning the constraints into subsets, each subproblem only involves an exponentially smaller subspace with a small number of variables and penalties. Thus, each subproblem takes significantly less time to converge to an equilibrium state at a given temperature, and the total time for all the subproblems to converge is also significantly reduced. This reduction in complexity is experimentally validated in Section 4.

Asymptotic convergence. We show the asymptotic convergence of CPSA to a CGM_d y^* of P_d by modeling the process as an inhomogeneous Markov chain and by following a similar approach as in the original proof for the asymptotic convergence of CSA [12]. The main result is stated as follows:

Theorem 4. The Markov chain modeling CPSA in discrete space asymptotically converges to a CGM_d with probability one when the following cooling schedule is used:

$$T_k \geq \frac{N_L \triangle_L}{ln(k+1)}, \tag{9}$$

where $\triangle_L = \max_{\mathbf{y}}\{|L(\mathbf{y}') - L(\mathbf{y})|, \mathbf{y}' \in \mathcal{N}(\mathbf{y})\}$.

Due to space limitation, we only outline the proof strategy here [3]. We model CPSA by a Markov chain that consists of multiple chains, each for a subproblem at a specific temperature. We show that the overall Markov chain is strongly ergodic, that it minimizes an implicit virtual energy based on the framework of GSA [7], and that the virtual energy is at the its minimum at any CGM_d. The key difference between the proof for CPSA and that for CSA is that the latter is done under a non-partitioned neighborhood, whereas the former is done under a partitioned neighborhood.

Partitioning strategy. In CPSA, we follow a previously proposed automated partitioning strategy [9] for analyzing the constraint structure and for determining the partitioning scheme. Based on P_m with continuous variables and represented in AMPL [4], our partitioning strategy consists of two steps.

In the first step, we enumerate all the indexing vectors in the AMPL model and select one that leads to the minimum R_{global}, which is the ratio of the number of global constraints to that of all constraints. We choose R_{global} as a heuristic metric for measuring the partitioning quality.

In the second step, after fixing the partitioning index, we decide on a suitable number of partitions. We have found a convex relationship between the number of partitions (N) and the overall complexity. When N is small, there are very few subproblems to be solved but each is expensive to evaluate; in contrast, when N is large, there are many subproblems to be solved although each is simple to evaluate. Hence, there is an optimal N that leads to the minimum time for solving the original problem. To find the optimal N, we have developed an iterative algorithm that starts from a large N, that evaluates one subproblem under this partitioning to estimate the complexity of solving the original problem, and that reduces N by half until the estimated complexity starts to increase. We leave the details of the algorithm to the reference [9].

4 Experimental Results

In this section, we apply CPSA to solve some nonlinear continuous optimization benchmarks and compare its performance to that of CSA and other dynamic penalty methods, including P_2, P_3, and the greedy ESP method (GEM) [11]. GEM is a greedy neighborhood search algorithm for finding ESPs by performing greedy descents in the original-variable space and greedy ascents in the penalty space. Like other dynamic penalty methods, GEM is not guaranteed to find ESPs because greedy descents and ascents may get stuck at infeasible local minima. We have tested selected problems from the CUTE benchmark suite [2], a constrained and unconstrained test environment. We have selected those problems based on the criterion that at least the objective or one of the constraints is nonlinear.

Table 1 compares the results of using CPSA, CSA, GEM, P_2, and P_3 on the CUTE benchmark problems studied. In our experiments, we have used the parameter settings in the GEM package developed by Zhe Wu and dated 08/13/2000 [11], and the parameters of P_2 and P_3 presented in Section 2.1. For each solver and each problem instance, we tried 100 runs from random starting points and report the average solution found (Q_{avg}), the average CPU time per run for those successful runs (T_{avg}), the best solution found (Q_{best}), and the fraction of runs that were successful (p_s). We do not list the best solutions of P_2 and P_3 because they are always worse than other methods.

When compared to the other three dynamic penalty methods, the results show that CPSA and CSA have much better average solution quality and the best quality on most of the instances. CPSA and CSA also have a higher success probability in finding a solution for all the instances tested.

The experimental results show the effectiveness of integrating constraint partitioning with CSA. When compared to CSA, CPSA is much faster in terms of T_{avg} for all the instances tested. The reduction in time can be more than an order of magnitude for large problems, such as ZAMB2-8 and READING6. CPSA can also achieve the same or better quality and success ratio than CSA for most of the instances tested. For example, for LAUNCH, CPSA achieves an average quality of 21.85, best quality of 9.01, and a success ratio of 100%, whereas CSA achieves, respectively, 26.94, 9.13, and 90%.

Table 1: Experimental results comparing CPSA, CSA, GEM, P_2, and P_3 in solving selected nonlinear continuous problems from CUTE. Each instance was solved by a solver 100 times from random starting points. The best Q_{avg} (*resp.* Q_{best}) among the five solvers are underlined. '$-$' means that no feasible solution was found in a time limit of 36000 sec. All runs were done on an AMD Athlon MP2800 PC with RH Linux AS4.

Problem ID	CPSA (with $\kappa = 0.8$ and 100 runs)				CSA (with $\kappa = 0.8$ and 100 runs)				GEM Penalty Method				P_2 Penalty Method			P_3 Penalty Method		
	Q_{avg}	Q_{best}	T_{avg}	p_s	Q_{avg}	Q_{best}	T_{avg}	p_s	Q_{avg}	Q_{best}	T_{avg}	p_s	Q_{avg}	T_{avg}	p_s	Q_{avg}	T_{avg}	p_s
AVION2	$9.47 \cdot 10^7$	$9.47 \cdot 10^7$	7.93	100%	$9.47 \cdot 10^7$	$9.47 \cdot 10^7$	204.74	100%	$9.47 \cdot 10^7$	$9.47 \cdot 10^7$	3213.49	100%	$9.47 \cdot 10^7$	2798	100%	$9.47 \cdot 10^7$	2578	100%
BATCH	$2.14 \cdot 10^5$	$2.00 \cdot 10^5$	35.03	20%	-	-	-	-	$2.42 \cdot 10^5$	$1.94 \cdot 10^5$	13242.43	5%	–	–	–	–	–	–
CRESC4	29.23	1.78	3.48	100%	34.24	1.98	7.37	100%	82.84	2.17	31.82	100%	82.84	31.82	100%	82.84	31.82	100%
CSFI1	−38.87	-49.08	0.06	100%	−28.65	-49.08	1.24	100%	−17.34	-49.05	2.95	98%	−23.26	0.98	91%	−20.57	1.54	93%
DEMBO7	174.80	174.79	2.58	100%	174.80	174.80	65.86	83%	174.80	174.80	232.16	57%	174.81	198.23	42%	174.81	203.64	40%
DIPIGRI	680.63	680.63	0.17	100%	680.65	680.63	1.79	100%	680.64	680.63	11.18	100%	680.64	9.45	100%	680.64	14.38	100%
DNIEPER	$1.87 \cdot 10^4$	$1.87 \cdot 10^4$	80.36	25%	-	-	-	-	$1.87 \cdot 10^4$	$1.87 \cdot 10^4$	3071.67	3%	–	–	–	–	–	–
EXPFITA	1.20	0.06	8.15	100%	2.35	0.10	18.45	100%	1.24	1.12	60.18	100%	1.24	63.94	100%	1.26	76.18	100%
FLETCHER	14.65	11.65	0.07	100%	4227.11	11.65	0.7	100%	20.28	11.72	3.75	100%	23.97	4.76	100%	23.76	6.42	100%
GIGOMEZ2	1.95	1.95	0.02	50%	1.95	1.95	0.55	48%	1.95	1.95	9.67	50%	–	–	–	1.95	12.04	22%
HIMMELBI	−1734.83	-1735.39	195.28	100%	−1735.55	−1735.57	*5091.47	100%	–	–	–	–	–	–	–	–	–	–
HIMMELBJ	−1910.45	−1910.56	17.83	100%	−1909.98	-1910.33	*619.19	100%	−1909.39	-1910.05	3514.92	99%	−1904	2304.04	94%	−1908	1953.94	57%
HIMMELP2	−62.05	-62.05	0.02	100%	−62.05	-62.05	0.44	100%	−62.05	−62.05	0.95	97%	−62.05	0.95	89%	−62.05	0.95	19%
HIMMELP6	−59.01	-59.01	0.04	100%	−59.01	-59.01	0.62	100%	−59.01	−59.01	1.58	100%	−59.01	2.34	100%	−59.01	1.77	100%
HONG	22.53	22.53	0.06	100%	22.53	22.53	0.82	100%	22.57	22.57	8.68	100%	22.57	8.9	100%	22.57	10.3	100%
HUBFIT	0.017	$1.69 \cdot 10^{-2}$	0.02	100%	0.017	$1.69 \cdot 10^{-2}$	0.36	100%	0.017	$1.69 \cdot 10^{-2}$	14.73	100%	0.017	15.62	100%	0.017	16.07	100%
LAUNCH	21.85	9.01	12.33	100%	26.94	9.13	*495.89	90%	22.80	9.01	1205.03	87%	24.583	1403.40	40%	24.54	1543.04	34%
LIN	−0.02	-0.02	0.1	100%	−0.02	-0.02	1.21	100%	−0.019	−0.02	3.02	100%	−0.019	3.01	100%	−0.019	3.21	100%
LOADBAL	76.35	0.78	6.34	100%	100.71	33.53	*226.07	100%	13.40	2.66	2350.60	100%	13.45	2454.65	100%	14.54	1934.34	100%
LOOTSMA	1.41	1.41	0.04	100%	1.41	1.41	0.52	100%	1.41	1.41	1.56	100%	1.41	2.75	100%	1.41	2.43	100%
MESH	-10^5	-10^5	17.37	100%	-10^5	-10^5	*625.03	100%	-10^5	-10^5	14453.76	100%	-10^5	14560	100%	-10^5	12139	100%
MISTAKE	−1.00	-1.00	0.44	100%	−1.00	-1.00	11.18	100%	−1.00	−1.00	63.84	90%	−1.00	70.48	45%	−1.00	77.74	98%
MRIBASIS	30.11	21.52	31.27	100%	31.04	29.32	1031.34	100%	31.56	31.22	24345.34	100%	34.03	20434	90%	–	–	–
MWRIGHT	2564.35	1.53	0.04	100%	12029.65	1.53	0.6	100%	$1.3 \cdot 10^5$	35.83	9.83	100%	$1.4 \cdot 10^5$	11.84	100%	$2 \cdot 10^5$	10.34	100%
ODFITS	−2225.33	-2379.4	5.56	100%	−1442.48	-2379.4	6.95	100%	9393.45	2699.04	21.15	100%	9497.43	20.48	100%	10320.3	24.32	100%
OPTCNTRL	549.61	549.49	12.36	100%	549.61	549.49	103.34	100%	550.00	550.00	1376.45	100%	550.00	1487.73	100%	550.00	1432.54	100%
OPTPRLOC	−16.42	-16.42	6.23	100%	−16.42	-16.42	296.49	100%	−16.42	−16.42	1381.46	100%	−16.42	872.43	100%	−16.42	787.42	100%
PENTAGON	0.00	0.00	0.4	100%	0.00	0.00	6.92	100%	0.00	0.00	47.32	93%	0.00	49.38	75%	0.00	36.34	94%
POLAK5	50.00	50.00	0.03	100%	50.00	50.00	0.43	100%	50.00	50.00	1.68	100%	50.00	1.83	100%	50.00	1.98	100%
QC	−998.64	−1018.09	0.33	100%	−970.19	-1007.35	5.77	100%	−763.80	−956.14	29.56	100%	−743.65	30.56	100%	−743.22	23.49	100%
READING6	−58.45	−105.45	202.86	100%	−54.71	-97.3	3059.25	100%	−66.12	−94.72	26027*	100%	−66.33	29820*	100%	−68.12	30223*	100%
RK23	25906.32	13016.09	0.88	39%	29614.39	16928.29	7.45	13%	–	–	–	–	–	–	–	–	–	–
ROBOT	5.51	5.46	0.21	100%	5.47	5.46	4.86	100%	5.46	5.46	38.85	100%	5.46	40.66	100%	5.46	38.95	100%
S316-322	334.13	334.13	0.01	100%	334.13	334.13	0.25	100%	334.30	334.30	1.20	100%	334.30	1.24	100%	334.30	1.50	100%
SINROSNB	0.00	0.00	0.02	100%	0.00	0.00	0.4	100%	–	–	–	–	–	–	–	–	–	–
SNAKE	0.00	0.00	0.02	100%	79.12	0.00	0.38	100%	267.08	0.00	2.45	100%	268.54	2.56	100%	269.18	2.48	100%
SPIRAL	360.21	0.00	0.05	100%	360.86	0.00	0.88	100%	505.70	0.00	2.70	100%	512.56	2.72	100%	505.80	2.67	100%
STANCMIN	4.29	4.25	0.03	100%	4.25	4.25	0.63	100%	4.25	4.25	2.53	100%	4.25	2.57	100%	4.25	2.54	100%
SVANBERG	15.73	15.73	0.78	100%	15.73	15.73	16.2	100%	–	–	–	–	–	–	–	–	–	–
SYNTHES1	2.76	0.76	0.13	100%	2.33	0.76	2.2	100%	2.61	0.76	18.93	100%	2.98	19.23	100%	2.86	13.43	100%
SYNTHES2	−0.56	-0.56	0.77	100%	−0.56	-0.56	17.63	100%	−0.55	-0.55	95.36	100%	−0.55	92.98	100%	−0.55	92.21	100%
SYNTHES3	15.08	15.08	4.85	100%	15.09	15.08	48.54	100%	15.08	15.08	336.45	100%	15.08	458.92	100%	15.08	492.3	100%
TENBARS4	1586.97	1586.97	11.54	77%	2566.82	509.5	15.55	9%	–	–	–	–	–	–	–	–	–	–
ZAMB2-8	−0.13	-0.15	364.06	100%	1.35	-0.15	6247.57	83%	–	–	–	–	–	–	–	–	–	–

* Only 10 runs were made for these problems due to the extensive CPU time required for each run.

5 Conclusions

In this paper, we have studied constraint-partitioned simulated annealing (CPSA) that partitions a mixed-integer nonlinear programming problem by its constraints into subproblems, solves each subproblem independently using constrained simulated annealing (CSA), and resolves those violated global constraints across the subproblems. The approach is based on the observation that the constraints of many application problems are highly structured because they model spatial or temporal relationships with strong locality. Because each subproblem is exponentially simpler and the number of global constraints is very small, our approach leads to significant reduction in the complexity of solving the original problem. The algorithm is theoretically important because it converges asymptotically with probability one to a constrained global minimum in discrete space. Our experimental results demonstrate significant efficiency improvements over CSA on some continuous constrained optimization benchmarks.

References

[1] J. C. Bean and A. B. Hadj-Alouane. A dual genetic algorithm for bounded integer programs. In *Technical Report TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan*, 1992.

[2] I. Bongartz, A. R. Conn, N. Gould, and P. L. Toint. CUTE: Constrained and unconstrained testing environment. *ACM Trans. on Mathematical Software*, 21(1):123–160, 1995.

[3] Y. X. Chen. *Solving Nonlinear Constrained Optimization Problems through Constraint Partitioning*. Ph.D. Thesis, Dept. of Computer Science, Univ. of Illinois, Urbana, IL, September 2005.

[4] R. Fourer, D. M. Gay, and B. W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Brooks Cole Publishing Company, 2002.

[5] J. Joines and C. Houck. On the use of nonstationary penalty functions to solve nonlinear constrained optimization problems with gas. In *Proc. of the First IEEE Int'l Conf. on Evolutionary Computation*, pages 579–584, 1994.

[6] R. L. Rardin. *Optimization in Operations Research*. Prentice Hall, 1998.

[7] A. Trouve. Rough large deviation estimates for the optimal convergence speed exponent of generalized simulated annealing algorithms. Technical report, LMENS-94-8, Ecole Normale Superieure, France, 1994.

[8] B. Wah and Y. X. Chen. Constraint partitioning in penalty formulations for solving temporal planning problems. *Artificial Intelligence*, 170(3):187–231, 2006.

[9] B. W. Wah and Y. X. Chen. Solving large-scale nonlinear programming problems by constraint partitioning. In *Proc. Principles and Practice of Constraint Programming, LCNS-3709*, pages 697–711. Springer-Verlag, October 2005.

[10] B. W. Wah and T. Wang. Simulated annealing with asymptotic convergence for nonlinear constrained global optimization. In *Proc. Principles and Practice of Constraint Programming*, pages 461–475. Springer-Verlag, October 1999.

[11] B. W. Wah and Z. Wu. The theory of discrete Lagrange multipliers for nonlinear discrete optimization. In *Proc. Principles and Practice of Constraint Programming*, pages 28–42. Springer-Verlag, October 1999.

[12] T. Wang. *Global Optimization for Constrained Nonlinear Programming*. Ph.D. Thesis, Dept. of Computer Science, Univ. of Illinois, Urbana, IL, December 2000.