2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

On the edge of feasibility: a case study of the particle swarm optimizer

Mohammad Reza Bonyadi
School of Computer Science
The University of Adelaide
mrbonyadi@cs.adelaide.edu.au

Zbigniew Michalewicz *
School of Computer Science
The University of Adelaide
zbyszek@cs.adelaide.edu.au

Abstract— **In many real-world constrained optimization problems (COPs) it is highly probable that some constraints are active at optimum points, i.e. some optimum points are boundary points between feasible and infeasible parts of the search space. A method is proposed which narrows the feasible area of a COP to its boundary. In the proposed method the thickness of the narrowed boundary is adjustable by a parameter. The method is extended in a way that it is able to limit the feasible regions to boundaries where *at least one* of the constraints in *a given subset* of all constraints is active and the remaining constraints might be active or not. Another extension is able to limit the search to cases where *all* constraints in *a given subset* are active and the rest might be active or not. The particle swarm optimization algorithm is used as a framework to compare the proposed methods. Results show that the proposed methods can limit the search to the requested boundary and they are effective in locating optimal solutions on the boundaries of the feasible and infeasible area.**

I INTRODUCTION

A constrained optimization problem (COP) is formulated as follows:

$$Find\ x \in \mathcal{F} \subseteq S \subseteq R^D\ such\ that \begin{cases} \forall y \in \mathcal{F}\ f(x) \le f(y), & (a) \\ g_i(x) \le 0, for\ i = 1\ to\ q & (b) \\ h_i(x) = 0, for\ i = q+1\ to\ m & (c) \end{cases} \quad 1$$

where f, g_i, and h_i are real-valued functions on the search space S, q is the number of inequalities, and $m - q$ is the number of equalities. The set of all feasible points which satisfy constraints (b) and (c) are denoted by $\mathcal{F}$ [1]. The equality constraints are usually replaced by $|h_i(x)| - \sigma \le 0$ where σ is a small value (normally set to 10^{-4}) [2]. Thus, a COP is formulated as

$$Find\ x \in \mathcal{F} \subseteq S \subseteq R^D\ such\ that \begin{cases} \forall y \in \mathcal{F}\ f(x) \le f(y), & (a) \\ g_i(x) \le 0, for\ i = 1\ to\ m & (b) \end{cases} \quad 2$$

where $g_i(x) = |h_i(x)| - \sigma$ for all $i \in \{q+1, \dots, m\}$. Hereafter, the term COP refers to this formulation.

The constraint $g_i(x)$ is called *active* at the point x if the value of $g_i(x)$ is zero. Also, if $g_i(x) < 0$ then $g_i(x)$ is called *inactive* at x. Obviously, if x is feasible and at least one of the constraints is active at x, then x is on the boundary of the feasible and infeasible areas of the search space.

In many real-world COPs it is highly probable that some constraints are active at optimum points [3], i.e. some optimum points are on the edge of feasibility. The reason is that constraints in real-world problems often represent some limitations of resources. Clearly, it is beneficial to make use of some resources as much as possible, which means constraints are active at quality solutions. Presence of active constraints at the optimum points causes difficulty for many optimization algorithms to locate optimal solution [4]. Thus, it might be beneficial if the algorithm is able to focus the search on the edge of feasibility for quality solutions.

In this paper it is assumed that there exists at least one active constraint at the optimum solution of COPs. A new function, called Subset Constraints Boundary Narrower (SCBN), is proposed which enables the search methods to focus on the boundary of feasibility with an adjustable thickness rather than the whole search space. SCBN is actually a function (with a parameter ε for thickness) that, for a point x, its value is smaller than zero if and only if x is feasible and the value of *at least one* of the constraints in *a given subset* of all constraint of the COP at the point x is within a predefined boundary with a specific thickness. By using SCBN in any COP, the feasible area of the COP is limited to the boundary of feasible area defined by SCBN, so that the search algorithms can only focus on the boundary. Some other extensions of SCBN are proposed that are useful in different situations. SCBN and its extensions are used in a particle swarm optimization (PSO) algorithm with a simple constraint handling method to assess if they are performing properly in narrowing the search on the boundaries.

The rest of this paper is organized as follows. Some background information on the constraints violation value, searching the edge of feasibility, and particle swarm optimizer are given in section II. Section III presents the proposed approach and its extensions, and in section IV the comparison results are reported and discussed. Section V concludes the paper.

II BACKGROUND

Some background on constraint violation value, searching the edge of feasibility, and particle swarm optimizer are discussed in this section.

A Constraint violation

A COP can be rewritten by combining all inequality constraints to form only one inequality constraint. In fact, any COP can be formulated as follows:

* Z. Michalewicz is with the School of Computer Science, the University of Adelaide, Adelaide, SA 5005, Australia, and also at the Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland, at the Polish-Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008 Warsaw, Poland.

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

$$Find\ x \in \mathcal{F} \subseteq S \subseteq R^D\ such\ that \begin{cases} \forall y \in \mathcal{F}\ f(x) \le f(y), & (a) \\ M(x) \le 0 & (b) \end{cases} \quad 3$$

where $M(x)$ is a function that combines all constraints $g_i(x)$ into one function. The function $M(x)$ can be defined in many different ways. The surfaces that are defined by different instances of $M(x)$ might be different. The inequality 3(b) should capture the feasible area of the search space. However, by using problem specific knowledge, one can also define $M(x)$ in a way that the area that is captured by $M(x) \le 0$ only refers to a sub-space of the whole feasible area where high quality solutions might be found. In this case, the search algorithm can focus only on the captured area which is smaller than the whole feasible area and make the search more effective. A frequently-used [5, 6] instance of $M(x)$ is a function $K(x)$

$$K(x) = \sum_{i=1}^{m} max\{g_i(x), 0\} \quad 4$$

Clearly, the value of $K(x)$ is non-negative. $K(x)$ is zero if and only if x is feasible. Also, if $K(x) > 0$, the value of $K(x)$ represents the maximum violation value (called the *constraint violation* value).

B Searching the edge of feasibility

As in many real-world COPs, there is at least one active constraint near the global best solution of COPs [3], some researchers developed operators to enable search methods to focus the search on the edges of feasibility. GENOCOP (GEnetic algorithm for Numerical Optimization for COnstrained Optimization) [7] was probably the first genetic algorithm variant that applied boundary search operators for dealing with COPs. Indeed, GENOCOP had three mutations and three crossovers operators and one of these mutation operators was a boundary mutation which could generate a random point on the boundary of the feasible area. Experiments showed that the presence of this operator caused significant improvement in GENOCOP for finding optimum for problems which their optimum solution is on the boundary of feasible and infeasible area [7].

A specific COP was investigated in [8] and a specific crossover operator, called *geometric crossover*, was proposed to deal with that COP. The COP was defined as follows:

$$f(x) = \left| \frac{\sum_{i=1}^{D} cos^4(x_i) - 2\prod_{i=1}^{D} cos^2(x_i)}{\sqrt{\sum_{i=1}^{D} ix_i^2}} \right|$$
$$g_1(x) = 0.75 - \prod_{i=1}^{D} x_i \le 0 \quad 5$$
$$g_2(x) = \sum_{i=1}^{D} x_i - 0.75D \le 0$$

where $0 \le x_i \le 10$ for all i. Earlier experiments [9] shown that the value of the first constraint ($g_1(x)$) is very close to zero at the best known feasible solution for this COP. The geometric crossover was designed as $x_{new,j} = \sqrt{x_{1,i}x_{2,j}}$, where $x_{i,j}$ is the value of the j^{th} dimension of the i^{th} parent, and $x_{new,j}$ is the value of the j^{th} dimension of the new individual. By using this crossover, if $g_1(\vec{x}_1) = g_1(\vec{x}_2) = 0$, then $g_1(\vec{x}_{new}) = 0$ (the crossover is *closed* under $g_1(x)$). It was shown that an evolutionary algorithm that uses this crossover is much more effective than an evolutionary algorithm which uses other crossover operators in dealing with this COP. In addition, another crossover operator was also designed [8], called *sphere crossover*, that was closed under the constraint $g(x) = \sum_{i=1}^{D} x_i^2 - 1$. In the sphere crossover, the value of the new offspring was generated by $x_{new,j} = \sqrt{\alpha x_{1,j}^2 + (1-\alpha)x_{2,j}^2}$, where $x_{i,j}$ is the value of the j^{th} dimension of the i^{th} parent, and both parents $\vec{x}_1$ and $\vec{x}_2$ are on $g(x)$. This operator could be used if $g(x)$ is the constraint in a COP and it is active on the optimal solution.

In [4], several different crossover operators closed under $g(x) = \sum_{i=1}^{D} x_i^2 - 1$ were discussed. These crossovers operators included *repair*, *sphere* (explained above), *curve*, and *plane* operators. In the repair operator, each generated solution was normalized and then moved to the surface of $g(x)$. In this case, any crossover and mutation could be used to generate offspring; however, the resulted offspring is moved (repaired) to the surface of $g(x)$. The curve operator was designed in a way that it could generate points on the *geodesic curves*, curves with minimum length on a surface, on $g(x)$. The plane operator was based on the selection of a plane which contains both parents and crosses the surface of $g(x)$. Any point on this intersection is actually on the surface of the $g(x)$ as well. These operators were incorporated into several optimization methods such as GA and Evolutionary Strategy (ES) and the results of applying these methods to two COPs were compared.

A variant of evolutionary algorithm for optimization of a water distribution system was proposed [10]. The main argument was that the method should be able to make use of information on the edge between infeasible and feasible area to be effective in solving the water distribution system problem. The proposed approach was based on an adapting penalty factor in order to guide the search towards the boundary of the feasible search space. The penalty factor was changed according to the percentage of the feasibility of the individuals in the population in such a way that there are always some infeasible solutions in the population. In this case, crossover can make use of these infeasible and feasible individuals to generate solutions on the boundary of feasible region.

In [11], a boundary search operator was adopted from [7] and added to an ant colony optimization (ACO) method. The boundary search was based on the fact that the line segment that connects two points x and y, where one of these points are infeasible and the other one is feasible, crosses the boundary of feasibility. A binary search can be used to search along this line segment to find a point on the boundary of feasibility. Thus, any pair of points (x, y), where one of them is infeasible and the other is feasible, represents a point on the boundary of feasibility. These points were moved by an ACO during the run. Experiments showed that the algorithm is effective in locating optimal solutions that are on the boundary of feasibility.

C Particle swarm optimization

Particle swarm optimization (PSO) algorithm is an iterative stochastic optimization method proposed in 1995 [12]. In PSO, there is a population (aka swarm) of individuals (aka particles), each individual i at iteration t contains three vectors: position ($\vec{x}_t^i$), velocity ($\vec{v}_t^i$), and personal best ($\vec{p}_t^i$). The position of each particle is used to evaluate the particle and it replaces $\vec{p}_t^i$ if it was better than that. The velocity vector is used to move the particles to new positions. In each iteration, the velocity and position of each particle is updated by:

$$\vec{v}_{t+1}^i = \vec{v}_t^i + \varphi_1 R_1(\vec{p}_t^i - \vec{x}_t^i) + \varphi_2 R_2(\vec{g}_t - \vec{x}_t^i) \quad 6$$

$$\vec{x}_{t+1}^i = \vec{x}_t^i + \vec{v}_t^i \quad 7$$

where $\vec{g}_t$ is the best personal best over the whole swarm, φ_1 and φ_2 are two constants, known as *acceleration coefficients*, and R_1 and R_2 are two D by D diagonal matrices. The values of the diagonal of R_1 and R_2 are set by uniform random numbers in the interval [0, 1] in each iteration for each particle independently. The velocity updating rule was modified in [13] where $\vec{v}_t^i$ was replaced by $\omega\vec{v}_t^i$. The coefficient ω is known as inertia weight and controls the influence of the previous velocity vector on the new velocity. From now on, the term PSO in this paper refers to this model with inertia weight. The algorithm has been used in many different areas of constraint optimization, such as dealing with linear equality constraints in COPs [14], dealing with COPs in general case [2, 15], locating feasible regions in COPs [16], among others.

III PROPOSED APPROACH

In this section, three alternative instances for *M(x)* (Eq. 3) are proposed to reduce the feasibility area of any COP to only the boundary of that COP. In fact, by using any of the proposed instances for *M(x)*, a new COP is constructed in which $M(x) \le 0$ if and only if the point x is on the boundary of the COP.

A Definition of a boundary point

Let's define a δ-active constraint in a COP as follows:

Definition 1: $g_i(x)$ is a *δ-active constraint at the point x* if $-\delta \le g_i(x) \le \delta$, where δ is a small positive value.

Accordingly, a δ-boundary point is defined as:

Definition 2: x is a *δ-boundary point* of a COP if x is feasible and at least one of the constraints of the COP is δ-active at x, where δ is a small positive value. Also, the set of all δ-boundary points in a COP is called *δ-boundary of the COP*.

Definition 2 allows us to define different levels of boundaries. The edge of feasibility is defined as follows:

Definition 3: x is said to be *on the edge of feasibility* if it is a 0-boundary point.

Definition 3 refers also to the fact that for any point x on the edge of feasibility, $g_i(x) = 0$ for *at least* one i.

B Proposed instances for M(x)

Consider a COP with constraints $g_i(x)$. The value of *K(x)* in Eq. 4 is zero if and only if x is feasible i.e. *K(x) vs.* x is a flat area for the set of all feasible points. Because *K(x)* is zero for all feasible solutions, it is impossible to distinguish whether a feasible point x is on the edge of feasibility or not. Of course there are many alternative instances for *M(x)*. One possible instance for *M(x)* is a function *G(x)*

$$G(x) = \max_{1\le i\le m}\{g_i(x)\} \quad 8$$

The function *G(x)* is called *Maximum Constraints Violation*, MCV, function. Clearly, $G(x) \le 0$ if and only if $x \in \mathcal{F}$. Note that, according to definition 1 and Eq. 3, if x is a δ-boundary point of a COP then $-\delta \le G(x) \le 0$. By using this instance for *M(x)*, one can recognize whether x is a δ-boundary of the COP by simply testing whether $-\delta \le G(x) \le 0$. Note that, although *G(x)* allows determination whether a point is a δ-boundary, $G(x)$ maintains the whole feasible area for a COP and it is not able to *restrict* the search on the boundary of feasibility by its own.

Assume that for a given COP, it is known that *at least one* of the constraints in the set $\{g_{i\in\Omega}(x)\}$ is δ-active at the optimum solution and the remaining constraints are satisfied at x, where $\Omega \subseteq \{1,2,\dots,m\}$. Let's define $H_{\Omega,\varepsilon}(x)$ as follows:

$$H_{\Omega,\varepsilon}(x) = \max\left\{\left|\max_{i\in\Omega}\{g_i(x)\} + \varepsilon\right| - \varepsilon, \max_{i\notin\Omega}\{g_i(x)\}\right\} \quad 9$$

where ε is a positive value. This function is called *Subset Constraint Boundary Narrower*, SCBN. Obviously, $H_{\Omega,\varepsilon}(x) \le 0$ if and only if at least one of the constraints in the subset Ω is 2ε-active and the others are satisfied. The reason is that, the component $|max_{i\in\Omega}\{g_i(x)\} + \varepsilon| - \varepsilon$ is negative if x is feasible and at least one of $g_{i\in\Omega}(x)$ is 2ε-active. Also, the component $max_{i\notin\Omega}\{g_i(x)\}$ ensures that the rest of constraints are satisfied.

Let's investigate a special case of SCBN where $\Omega = \{1,2,\dots,m\}$ in more detail. We define the function $H_\varepsilon(x)$ as follows (note that we drop the subscript Ω when it refers to *all* existing constraints):

$$H_\varepsilon(x) = |G(x) + \varepsilon| - \varepsilon \quad 10$$

where ε is a positive value and *G(x)* is defined by Eq. 8. This function is called *Constraint Boundary Narrower* (CBN) throughout the paper.

Figure 1 presents $H_\varepsilon(x)$ *vs. G(x)* and illustrates that $-2\varepsilon \le G(x) \le 0$ if and only if $H_\varepsilon(x) \le 0$. This in fact implies that the areas in where x satisfies $H_\varepsilon(x) \le 0$ correspond with the 2ε-boundary of the COP. Thus, if we set ε to a value in the interval $[0, \frac{\delta}{2}]$ then $H_\varepsilon(x) \le 0$ if x is a δ-boundary point of the COP. The value of ε determines the desired thickness of the boundary. Note that if $\varepsilon = 0$ then $H_\varepsilon(x) \le 0$ corresponds to the points where *G(x)=0*, i.e. edge of feasibility of the COP. Also, if ε is set to a value larger than $|\min(G(x))|$ over all x then $H_\varepsilon(x)$ is equal to $G(x)$ for all x.

To summarize,

- The area where $H_\varepsilon(x) \leq 0$ corresponds to the 2ε-boundary of the COP
- If $\varepsilon \in \left[0, \frac{\delta}{2}\right]$ then $H_\varepsilon(x) \leq 0$ corresponds to δ-boundary of the COP
- If $\varepsilon = 0$ then $H_\varepsilon(x) \leq 0$ corresponds to the edge of feasibility of the COP
- If $\varepsilon > \left|\max_{\forall x \in S}\left(G(x)\right)\right|$ then $H_\varepsilon(x) = G(x)$ for all x, which refers to the whole feasible area of the COP

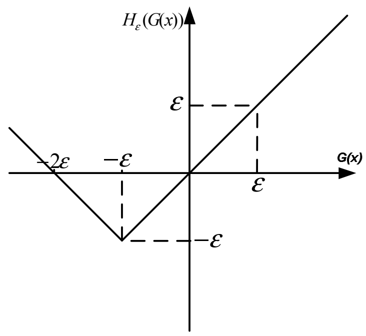


Figure 1. The graph of $H_\varepsilon\left(G(x)\right)$. The value of $H_\varepsilon\left(G(x)\right)$ is negative if and only if $-2\varepsilon \leq G(x) \leq 0$ for any value of x. Also, if $G(x) > 0$ then $H_\varepsilon\left(G(x)\right) = G(x)$ for any value of x.

Let's consider some examples to see how $H_\varepsilon(x) \leq 0$ corresponds to the 2ε-boundary of feasible area.

C *Examples for CBN*

Three examples are provided to show how $H_\varepsilon(x) \leq 0$ represents the boundary of feasibility of a COP.

Example 1:

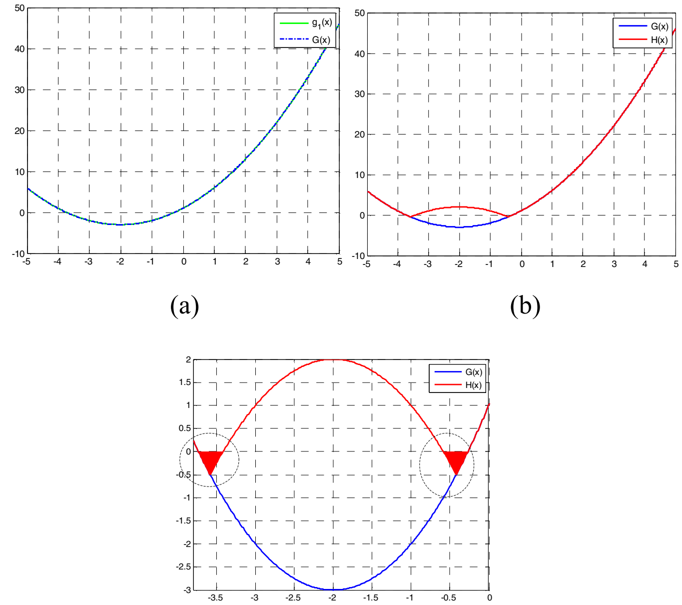
Consider a single variable COP with the following single constraint:

$$g_1(x) = (x+2)^2 - 3 \leq 0$$

$H_\varepsilon(x)$ is calculated as:

$$H_\varepsilon(x) = \left|\max_{1 \leq i \leq m}\{g_i(x)\} + \varepsilon\right| - \varepsilon = \left|g_1(x) + \varepsilon\right| - \varepsilon$$

Figure 2 shows the constraint $g_1(x)$, $G(x)$, and $H_\varepsilon(x)$ and Figure 2(a) shows the curves of the constraint $g_1(x)$ and $G(x) = \max_{\forall i}\left(g_i(x)\right)$ where $-5 \leq x \leq 5$. Figure 2(b) shows $G(x)$ and $H_\varepsilon(x)$ in the same interval of x. The value of ε was set to 0.5. The red shaded areas in Figure 2(c) show the areas where x satisfies $H_\varepsilon(x) \leq 0$. It is obvious that if x satisfies $H_\varepsilon(x) \leq 0$ then it satisfies $G(x) \leq 0$ and also x is a δ-boundary point. Note that the value of $H_\varepsilon(x)$ reduces only to $-\varepsilon$, however, for any x where $-2\varepsilon \leq G(x) \leq 0$, we have $H_\varepsilon(x) \leq 0$.



(a) (b)

(c)

Figure 2. Effects of the function CBN on a COP with one constraint (a) the constraint $g_1(x)$ and the constraint violation MCV, (b) MCV and CBN ($\boldsymbol{H_{0.5}(x)}$), (c) zoomed in of MCV and CBN ($\boldsymbol{H_{0.5}(x)}$).

Example 2:

Consider a single variable COP with the following two constraints:

$$g_1(x) = e^x - 1$$
$$g_2(x) = x^2 - 3$$

$H_\varepsilon(x)$ is calculated as:

$$H_\varepsilon(x) = \left|\max_{1 \leq i \leq m}\{g_i(x)\} + \varepsilon\right| - \varepsilon$$

Figure 3 shows the constraints, $G(x)$, and $H_\varepsilon(x)$ and Figure 3(a) shows the curves of two constraints ($g_1(x)$ and $g_2(x)$) and $G(x) = \max_{1 \leq i \leq m}\{g_i(x)\}$ where $-3 \leq x \leq 3$. Figure 3(b) shows $G(x)$ and $H_\varepsilon(x)$ in the same interval of x. The value of ε was set to 0.1. Figure 3(c) shows a zoomed in version of Figure 3(b). The regions between the red curve, $H_\varepsilon(x)$, and the horizontal axes are the regions where $H_\varepsilon(x) \leq 0$ (red shaded). It is clear that these two regions correspond to the δ-boundary in the original COP.

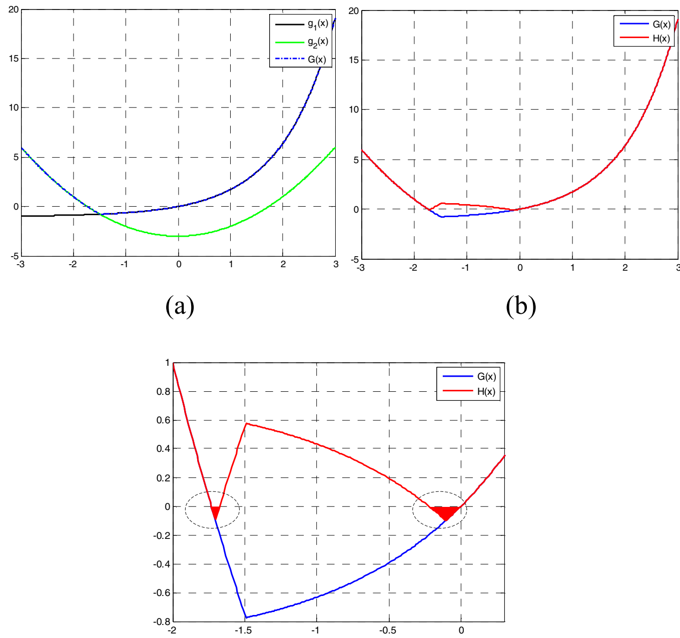
Example 3:

Let's consider another example to see how $H_\varepsilon(x) \leq 0$ represents the δ-boundary of a COP in a two dimensional space (a COP with two variables) with two constraints:

$$g_1(\vec{x}) = e^{x_1 + x_2} - 1$$
$$g_2(\vec{x}) = \sin(x_1) + 1.9\cos(x_2) + 1$$

The value of $H_\varepsilon(x)$ for this COP can be expressed by the following formula:

$$H_\varepsilon(x) = \left|\max_{1 \leq i \leq m}\{g_i(x)\} + \varepsilon\right| - \varepsilon$$



(a) (b)

(c)

Figure 3. Effects of CBN on a COP with two constraints (a) two constraints *g1(x)* and *g2(x)*, and the constraint violation MCV (*G(x)*), (b) MCV and $H_{0.1}(x)$, (c) the red areas are the areas where $H_{0.1}(x) \leq 0$.

Figure 4 shows the original feasible space as well as the space where $H_\varepsilon(x) \leq 0$ $(-5 < x_1, x_2 < 5)$ for different values of ε.

Clearly, $H_\varepsilon(x) \leq 0$ indicates the boundaries of feasible regions of the COP with some thickness. This thickness is adjustable by changing the value of ε, the smaller the value of ε, the thinner the boundary. Note that if ε set to zero, $H_\varepsilon(x)$ represents the boundary of feasibility with the thickness equal to zero. Thus, it shows the lines where $G(x) = 0$, i.e. at least one of the constraints is 0-active.

There is an important advantage for the proposed CBN (and SCBN) in comparison to other methods discussed in section II.B: it is possible to control how close to the boundaries should be sought by the algorithm, which was not adjustable in other methods. In fact, other methods considered two types of feasible points: boundary and non-boundary. However, in CBN, one can define a spectrum from non-boundary to boundary points. As an example, a feasible point x for which $G(x) = -2$ is considered as a point in the 1-boundary of the COP, while if $G(x') = 0$, it is in fact in the 0-boundary of the COP. This is beneficial as one can start with a larger value of ε and reduce it to smaller value with the aim of searching all feasible areas (explorations) at the beginning of the search and then focusing on the boundaries (exploitation).

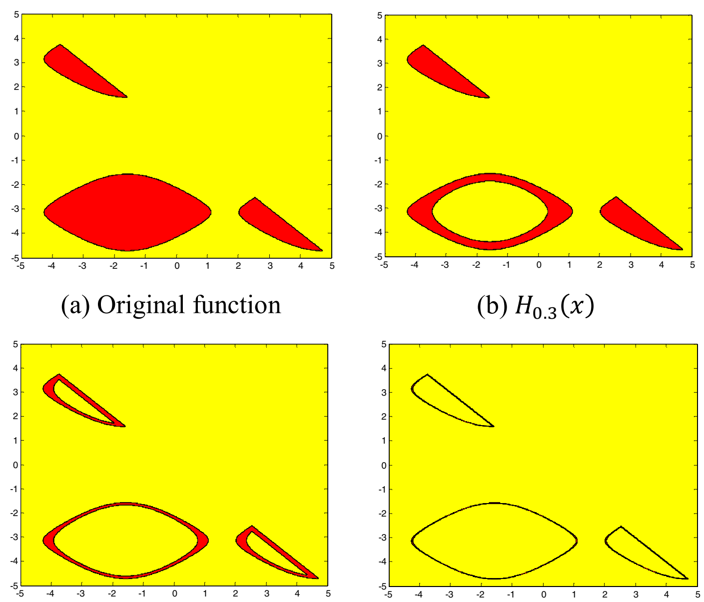
Note that if $H_\varepsilon(x) \leq 0$ then *at least* one of the constraints of the COP is 2ε-active.

D Generalization and other derivatives of CBN

It might be the case that the active constraints are known in a problem based on some prior knowledge. Let's assume that all constraints in a subset of constraints are 2ε-active at x (shown by $\Phi \subseteq \{1, 2, \dots, m\}$). We define $C_{\Phi,\varepsilon}(x)$ as follows:

$$C_{\Phi,\varepsilon}(x) = \max\left\{\max_{\forall i \in \Phi}\{|g_i(x) + \varepsilon| - \varepsilon\}, \max_{\forall i \notin \Phi}\{g_i(x)\}\right\} \quad 11$$

This function is called *All in a subset CBN*, ACBN. According to this definition, $C_{\Phi,\varepsilon}(x) \leq 0$ if and only if *all* constraints in Φ are 2ε-active at *x* and the other constraints are satisfied. Note that, it does not mean that the remaining constraints cannot be 2ε-active.



(a) Original function (b) $H_{0.3}(x)$

(c) $H_{0.1}(x)$ (d) $H_{0.01}(x)$

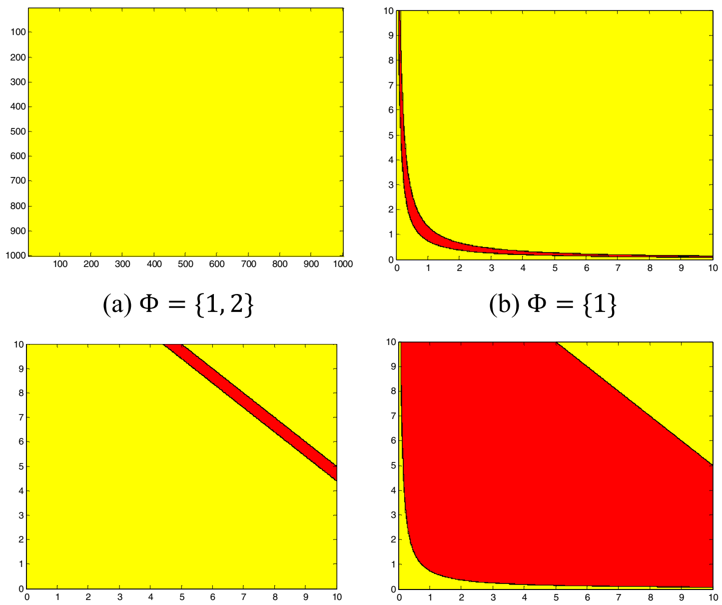
Figure 4. Effects of CBN on a COP with two constraint (a) shows the infeasible region (dark red) and feasible regions (dark blue), (b, c, d) show $H_\varepsilon(x) > 0$ (dark red) and $H_\varepsilon(x) \leq 0$ (dark blue) with different ε values (0.3, 0.1, and 0.01, respectively).

Figure 5 shows the areas where $C_{\Phi,0.3}(x) \leq 0$ for the COP defined in Eq. 5. When $\Phi = \{1, 2\}$ (Figure 5(a)), we are after areas where both constraints are 0.6-active. The figure shows that there is no feasible point in this situation. This was actually expected because there is no point in the search space where both constraints are active. Figure 5(b) shows that, for $\Phi = \{1\}$, referring to constraint *g1(x)* is 2ε-active while constraint *g2(x)* might be active or not, there is an arc-shape narrow area where $C_{\Phi,0.3}(x) \leq 0$. This is in fact the shape of the boundaries of the first constraint in Eq. 5. This is also similar (with a different shape) in the case $\Phi = \{2\}$. Finally, Φ might be empty ($\Phi = \{\}$) that refers to any constraint might be active or non-active, which is actually the whole feasible space.

All of the aforementioned formulations can be useful in different situations. For a particular COP:

- If nothing is known about the active constraints at the optimal solution, then use any of the four proposed formulations (Eq. 8, 9, 10 or 11) with ε set to the largest possible number if applicable,

- If it is known that *at least one* of the constraints is 2ε-active then use $H_{\varepsilon}(x)$, CBN,
- If it is known that *at least one* of the constraint of *a known subset* of constraints is 2ε-active (and of course the rest are satisfied) at the optimal solution then use $H_{\Omega,\varepsilon}(x)$, SCBN,
- If it is known that *a subset* of constraints are 2ε-active and the remaining constraints are satisfied (might be 2ε-active or not), then use $C_{\Phi,\varepsilon}(x)$, ACBN.



(a) $\Phi = \{1, 2\}$ (b) $\Phi = \{1\}$

(c) $\Phi = \{2\}$ (d) $\Phi = \{\}$

Figure 5. The shape of the feasible (dark blue) and infeasible (dark red) areas according to the function $C_{\Phi,0.3}(x)$ when different knowledge is available about the active constraints (different Φ) (a) both constraints are active, (b) first constraint is active while the second might be active or not, (c) second constraint is active while the first might be active or not, and (d) there is no knowledge about active constraints.

These instances of *M(x)* (Eq. 3) are compared with each other to see if they are really effective according to different knowledge about the constraints (whether they are active or not). We use PSO as the optimizer to compare these different instances. PSO is combined with a simple constraint handling method (used also in [2]) for COPs where the optimal solution is on the edge of feasibility. A simple constraint handling method for comparing two points, *x* and *y*, in the search space is as follows:

- for $max\{M(x), M(y) < 0$, x is better than y iff $f(x) < f(y)$
- for $M(x) = M(y)$, x is better than y iff $f(x) < f(y)$
- for all other cases, i.e. $max\{M(x), M(y)\} > 0$ and $M(x) \neq M(y)$, *x* is better than *y* iff $M(x) < M(y)$

IV EXPERIMENTS AND COMPARISONS

In this section, the proposed instances of *M(x)* (Eq. 3), MCV, CBN, SCBN, and ACBN are tested when they are added to the PSO to solve COPs. The aim is to find out if the proposed constraint boundary narrower approaches can improve the performance of the algorithm in finding optimal solution. The COP test cases used for the comparisons were taken from a benchmark known as CEC06 [17]. This benchmark contains 24 COPs, however, we only consider the first 7 of them in our comparisons (shown by G01, G02, …, G07)[2]. The optimal solutions for these functions as well as the active constraints at the optimal solution are known [11]. In the case of SCBN and ACBN the known active constraints were specified for the algorithm. To make comparisons easier, the *gap* between the found solutions by the algorithm and the optimal solution is calculated and reported. This gap is calculated as follows:

$$gap = \left|\frac{z^* - z}{z^*}\right| \quad 12$$

where *z** is the best known solution and *z* is the found solution by the algorithm. The maximum number of function evaluations was set to 100,000 and the number of dimensions for each test case was set according to the specifications recommended in [17]. The parameters for PSO were set to: $\omega = 0.729$, $\varphi_1 = \varphi_2 = 1.49$ (these parameters are frequently used in other PSO studies), population size = 30. The tests are done with two different values for *ε* (1 and 0.01).

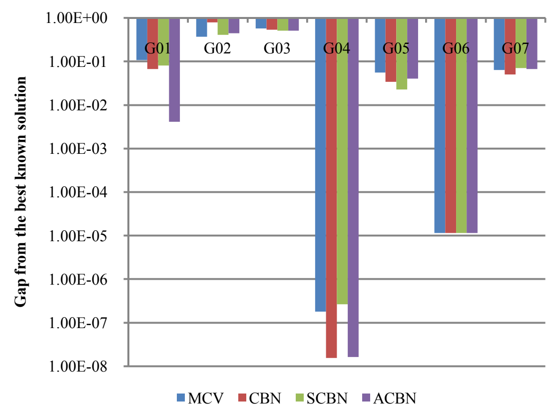


Figure 6. Results of using MCV, CBN, SCBN, and ACBN for the constraint violation. *ε = 1* for all tests.

CBN outperforms MCV in all cases except the case G02. This good performance was actually expected as CBN applies to the COPs with some prior knowledge, i.e. the optimal solution is on the boundaries. SCBN performs better than MCV in G01, G02, G03, G05, and G06. However, it is just slightly worse than MCV in G04 and G07. ACBN performs also better than MCV in G01, G03, G04, G05, G06, and it is slightly worse than MCV in G02 and G07. Let's take a closer look into the test case G02. The formulation of G02 is exactly the same as in Eq. 5 (note that the first constraint is active at the optimal point). Figure 7 shows *M(x)* (for two dimensional *x's* in the interval 0 and 10) when it is equal to CBN, ACBN, and MCV (note that, as G02 has only two constraints and its first constraint is active, the contour for ACBN and SCBN is the same as each other).

[2] More comprehensive experiments are planned for the extended version of this paper.

When MCV is used, most of the area in the center (color spectrum from dark blue to light green in Figure 7(a)) is the feasible area. When the function CBN is used (Figure 7(b)), the area in the middle becomes infeasible and actually two disjoint feasible regions (two dark blue areas in Figure 7(b)) appear. The optimal solution is in the dark blue area in the bottom left of the figure. Moving from the feasible area on the top right of the figure to the feasible area on the bottom left is not that easy for optimization algorithms. The reason is that the area between these two regions is infeasible with very high value of *M(x)* (in this case, CBN), which traps the individuals in one of these two feasible areas. Thus, individuals that have converged to the solutions on the top right area are not able to move to the bottom left area easily, which results in missing optimal solution. However, when ACBN is used, there is a high chance for individuals to move towards the area where it contains the optimal solution (the dark blue area at the bottom left of Figure 7(c)). Note that, this area is in fact the only valley in the search space that is easy to converge to by most of optimization methods (PSO in this case).

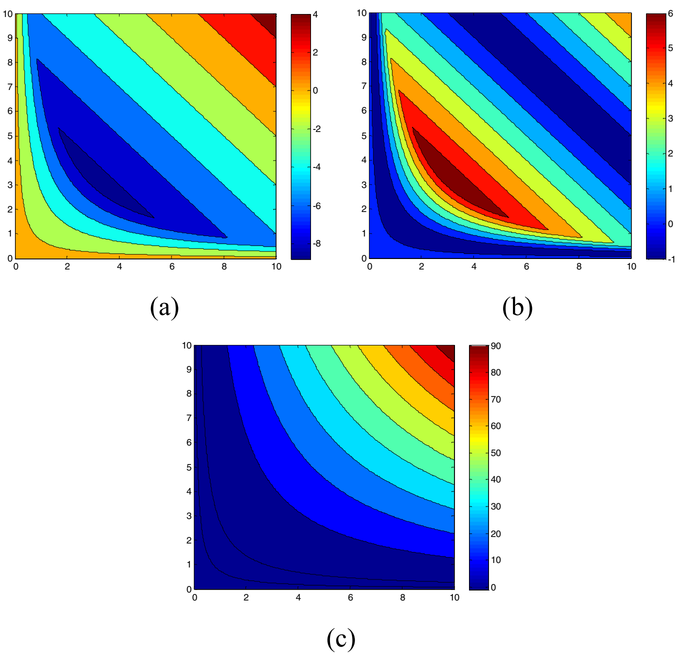


Figure 7. The constraint violation space for G02 when *M(x)* is (a) MCV, (b) CBN with $\varepsilon = 1$, and (c) ACBN with $\varepsilon = 1$ and $\Phi = \{1\}$.

To study the effect of ε on the performance of CBN functions, we applied the same test as mentioned above, but this time with $\varepsilon=0.01$. Figure 8 shows the results for $\varepsilon=0.01$ when *M(x)* is set to different functions (MCV, CBN, SCBN, and ACBN).

MCV outperforms CBN in G02, G05 (slightly), G06 (slightly), and G07. The reason for the worse performance of CBN was the same as the one explained in the previous test when $\varepsilon=1$.

In the test case G07, it is clear that the result has been affected by the value of ε. In fact, smaller value for ε results in worse performance of CBN in dealing with G07. A potential reason would be the smaller values for ε causes smaller feasible areas that might be far from each other. Hence, finding the feasible area that contains global optimum might be harder when ε is smaller.

SCBN performs better than MCV in G03, G05, G06, and G07 and slightly worse than that in G01, G02, and G04. Also, ACBN performs better than MCV in G01, G03, and G07 and slightly worse than that in G02 and G05. However, ACBN performs substantially worse than MCV in G04 and G06. In G04 test case, it is obvious that the performance of ACBN has dropped substantially when the value of ε has decreased. This means a potential reason behind worse performance would be generating smaller feasible areas when ε is small, which makes finding feasible areas harder.

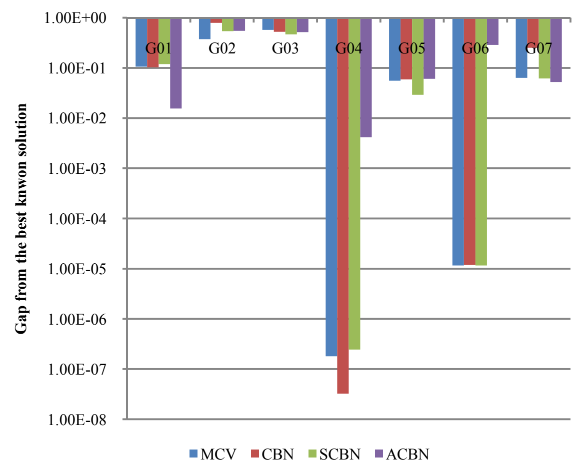


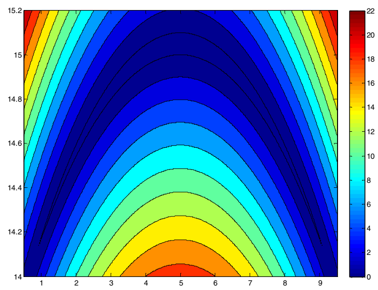
Figure 8. Results of using MCV, CBN, SCBN, and ACBN for the constraint violation. $\varepsilon = 0.01$ for all tests.

Let's take a closer look at G06 to find out what is the reason that ACBN does not perform well on that; the function *M(x)* for G06 has been shown in Figure 9(a) when *M(x)* is MCV (the dark blue arc shaped area is the feasible area). The optimal solution is at the bottom right of the feasible area. By using ACBN (the feasible areas are two dark blue narrow areas at left and right of Figure 9(b)), this feasible area is divided into two areas, corresponding to the edges of feasibility (the optimal solution is in the right one).

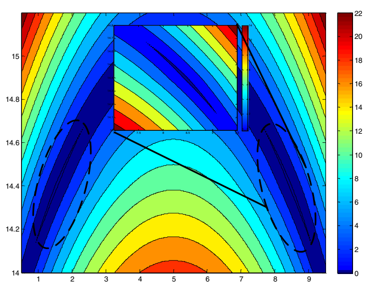
This splitting of the feasible area in fact causes the algorithm to sometimes converge to the feasible area where the optimal solution is not in (the left one in Figure 9(b)). This causes poor average performance of the algorithm.

V CONCLUSIONS AND FUTURE WORK

There has been some experimental evidence that showed the importance of searching the boundaries of feasible and infeasible areas in a constraint optimization problem (COP) [3, 4, 8]. This boundary is defined as: the points that are feasible and *at least one* of the constraints is zero for them. In this paper, three new instances (called Constraint Boundary Narrower, CBN, Subset CBN, SCBN, and All in a subset CBN, ACBN) for the constraint violation function were proposed which were able to reduce the feasible area to only boundaries of the feasible area. In the SCBN (ACBN), it is possible to select a subset of constraints and limit the boundaries where *at least one* of these constraints (*all* of these constraints) is (are) active. The thickness of the boundaries was adjustable in the proposed method by a parameter (ε). Experiments showed that changing the value of ε influences the performance of the algorithm. In fact, a smaller value of ε causes limiting the feasible area to narrower boundaries, which makes finding the feasible areas harder. However, although it is harder to find the feasible areas (narrower boundaries), improving the final solutions is easier once the correct boundary was found. Thus, as a potential future work, one can design an adaptive method so that the search is started to explore the feasible area and then it is concentrated on the boundaries.



(a)



(b)

Figure 9. The contour of *M(x)* for the function G06 when (a) MCV is used, (b) ACBN is used ($\varepsilon = 0.01$ and $\Phi = \{1, 2\}$).

VI ACKNOWLEDGEMENTS

This work was partially funded by the ARC Discovery Grant DP130104395 and by grant N N519 5788038 from the Polish Ministry of Science and Higher Education (MNiSW).

VII REFERENCES ATGF

[1] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evolutionary Computation,* vol. 4, pp. 1-32, 1996.

[2] M. R. Bonyadi, X. Li, and Z. Michalewicz, "A hybrid particle swarm with velocity mutation for constraint optimization problems," in *Genetic and Evolutionary Computation Conference*, ACM, 2013, pp. 1-8.

[3] M. Schoenauer and Z. Michalewicz, "Evolutionary computation at the edge of feasibility," *PPSN IV,* pp. 245-254, 1996.

[4] M. Schoenauer and Z. Michalewicz, "Boundary Operators for Constrained Parameter Optimization Problems," in *ICGA*, 1997, pp. 322-329.

[5] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Tran. on Evolu. Comp.,* vol. 4, pp. 284-294, 2000.

[6] X. Li, M. R. Bonyadi, Z. Michalewicz, and L. Barone, "Solving a real-world wheat blending problem using a hybrid evolutionary algorithm," in IEEE CEC, 2013, pp. 2665-2671.

[7] Z. Michalewicz, *Genetic algorithms + data structures = evolution programs*: Springer, 1992.

[8] Z. Michalewicz, G. Nazhiyath, and M. Michalewicz, "A note on usefulness of geometrical crossover for numerical optimization problems," in *Fifth Ann. Conf. on Evolu. Progr.*, 1996, pp. 305–312.

[9] A. Keane. (1994, Thursday, May 19). *Genetic algoritm digest*. Available: ftp://ftp.cse.msu.edu/pub/GA/gadigest/v8n16.txt

[10] Z. Y. Wu and A. R. Simpson, "A self-adaptive boundary search genetic algorithm and its application to water distribution systems," *Journal of Hydraulic Research,* vol. 40, pp. 191-203, 2002.

[11] G. Leguizamon and C. A. C. Coello, "Boundary Search for Constrained Numerical Optimization Problems With an Algorithm Inspired by the Ant Colony Metaphor," *IEEE Tran. Evolu. Comp.,* vol. 13, pp. 350-368, 2009.

[12] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Symposium on Micro Machine and Human Science*, IEEE, 1995, pp. 39-43.

[13] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *World Congress on Computational Intelligence*, IEEE, 1998, pp. 69-73.

[14] U. Paquet and A. P. Engelbrecht, "Particle swarms for linearly constrained optimisation," *Fundamenta Informaticae,* vol. 76, pp. 147-170, 2007.

[15] J. Liang, S. Zhigang, and L. Zhihui, "Coevolutionary comprehensive learning particle swarm optimizer," in *Congress on Evolutionary Computation*, IEEE, 2010, pp. 1-8.

[16] M. R. Bonyadi and Z. Michalewicz, "Locating potentially disjoint feasible regions of a search space with a particle swarm optimizer," in *Evolutionary Constrained Optimization*, K. Deb, Ed., ed: Springer-Verlag, 2014.

[17] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. C. Coello*, et al.*, "Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization," Nangyang Technological University, Singapore, Technichal report, 2006.