A Multi-constraint Handling Technique-based Niching Evolutionary Algorithm for Constrained Multi-objective Optimization Problems

Zixu Wang  
School of Computer Science and Technology  
Xi’an, China  
cnwangzx@foxmail.com

Jingxuan Wei  
School of Computer Science and Technology  
Xi’an, China  
wjx@xidian.edu.cn

Yi Zhang  
School of Computer Science and Technology  
Xi’an, China  
auvyisen@gmail.com

Abstract—When solving constrained multi-objective optimization problems, the challenge is that how to deal with all kinds of constraints regardless of the shape of the feasible region. Especially when the feasible region is discrete or very small, some constraint handling techniques cannot solve it exactly. To address this issue, this paper proposes a new technique to handle constraints. First, all the constraints will be sorted to some grades from hard to easy according to their constrained violations. Second, a niching crowding distance mechanism is used to guarantee the diversity of the pareto front better. The experiments show that the proposed algorithm can generate a set uniformly distributed pareto optimal solutions under constraints.

Keywords—constraints, evolutionary algorithm, multi-objective, optimization

I. INTRODUCTION

The constrained optimization problems (COPs) exist widely in the real world. COPs usually contains multiple conflicting objectives and a series of constraints. According to the number of objectives, it can be divided into constrained multi-objective optimization problems (CMOPs) [1] and constrained many-objective optimization problems [2]. The number of objectives of the former is less than or equal to three, while the latter is more than three. Generally speaking, a CMOP can be formulated as:

\[
\begin{align*}
\text{minimize} & \quad F(x) = (f_1(x), \ldots, f_z(x), f_n(x))^T \\
\text{subject to} & \quad g_j(x) \geq 0, j = 1, \ldots, p \\
& \quad h_i(x) = 0, k = 1, \ldots, q \\
x & \in \Omega
\end{align*}
\]

Where \(x=(x_1, x_2, \ldots, x_n)^T\) is an n-dimensional decision vector. \(\Omega=[x_l, x_u]^n \subseteq \mathbb{R}\) represents the decision space. \(F: \Omega \rightarrow \mathbb{R}\) including m conflicting objective functions where \(\mathbb{R}\) is the objective space. \(g_j(x)\) is the j-th inequality constraint while \(h_i(x)\) is the k-th equality constraint. Next, some definitions about CMOP are introduced briefly.

The constraint violation (CV): first, the degree of violation of \(x\) in the j-th inequality constraint and k-th equality constraint can be defined as \(G_j(x) = \min \{0, g_j(x)\}, j = 1, \ldots, p\) and \(H_k(x) = \max \{0, h_i(x)\}, k = 1, \ldots, q\) respectively. Thus, the \(CV\) is defined as:

\[
CV(x) = \sum_{j=1}^{p} G_j(x) + \sum_{k=1}^{q} H_k(x)
\]

Feasible solution: \(x\) is a feasible solution if and only if it satisfies all constraints which means \(CV(x) = 0\).

Infeasible solution: \(x\) is an infeasible solution as long as it does not satisfy any constraints which means \(CV(x) > 0\).

Pareto dominance: \(x_1\) and \(x_2\) are two solutions that belong to \(\Omega\). If \(f_j(x_1) \leq f_j(x_2)\) \(\forall i \in \{1,2,\ldots,m\}\) and \(\exists i \in \{1,2,\ldots,m\}\) that make \(f_j(x_1) < f_j(x_2)\), we call \(x_1\) dominates \(x_2\), denoted as \(x_1 \prec x_2\). Moreover, if there is no solution dominating \(x_1\) (\(x_1\) is a feasible solution), then \(x_1\) is called a pareto optimal.

Pareto set (PS): Pareto set contains all pareto optimal solutions.

Pareto front (PF): PF is the image of all pareto optimal solutions.

Next, we will give a brief review of unconstrained multi-objective evolutionary algorithms.

MOEAs can be classified into three categories: dominance-based, decomposition-based and indicator-based. One common feature of these algorithms is that they all focus on the convergence and diversity of population. 1) dominance-based EAs ranking individuals by non-dominant relations such as NSGAII [3] which use nondominated sorting to represent convergence and then use crowding distance assignment to represent diversity. 2) decomposition-based EAs such as MOEA/D [4] decompose a MOP into a series of single-objective subproblems according to uniformly distributed weight vectors in space. The convergence can be guaranteed by optimizing each single objective, and the diversity can be guaranteed by uniformly distributed vectors. There are many variants of MOEA/D such as MOEA/DD [5], MOEA/D-M2M [6] and MOEA/D-AWA [7]. 3) indicator-based EAs use performance indicators to evaluate individuals of each generation. The performance of the algorithms will be affected by the different emphasis of the indicator. For example, IBEA [8] used \(I_{X^*}\) as indicator.
which is a convergence indicator, so IBEA performs poorly in diversity [9]. Hypervolume indicator concern both convergence and diversity but it has high computational complexity. As a result, many indicator-based EAs that use hypervolume as their indicator such as HypE [10] are computationally heavy and very slow.

However, compared with the prosperity of unconstrained multi-objective evolutionary algorithms, the research of constrained multi-objective evolutionary algorithms is few. Most of the existing algorithms use the combination of a well performed unconstrained algorithm and a constraint handling technique. In this case, the performance of the constraint handling technique will affect the performance of the algorithm. We will give details of existing constraint handling techniques in Section II.

In order to address the limitations, the goal of this paper is to develop an evolutionary algorithm to find the pareto front under constraints. The pareto optimal solutions obtained under constraints are expected to have good performance in terms of convergence and diversity. To achieve the goal, we propose two new techniques:

1) A new constraint handling technique that deal with multi-constraints is proposed, denoted as MC-CHT. This technique can enlarge the feasible region during the early stage of evolution and then gradually reduce to true size. Moreover, we hybrid this method with \( \varepsilon \)-constraint method.

2) A new niching method is used to maintain the diversity of population. This method can avoid two extreme situations in original crowding distance [3].

The rest of this paper is organized as follows. Section II overviews existing constrained handling techniques. Then explain the motivations of new constraint handling technique. Section III shows the details of the proposed constrained handling technique and the whole algorithm. Section IV introduces the general experimental setting and shows specific experiment results that compared with four popular CMOEAs and analyzes the experiment results. Section V gives the conclusion and future direction.

II. PRELIMINARIES

A. Existing constraint handling techniques

Constraint handling is a crucial part of CMOPs. All these constraint handling techniques can be divided into three categories.

1) The first category always give priority to feasible solutions when choosing promising solutions into next generation. The typical methods include penalty function methods [11], constrained-domination principle [12], and \( \varepsilon \)-constraint method [13]. The main idea of penalty function is adding a penalty term that connect with CV to original objectives. A solution with high CV will have high penalty than others with low CV, of course, feasible solutions have no penalty. Constrained -domination principle transforms normal dominance relation that defined before to constrained dominance relation. Specifically, there are two solutions \( x_1 \) and \( x_2 \), we call \( x_1 \) constrained-dominate \( x_2 \) if any conditions are satisfied: 1) \( x_1 \) is feasible solution while \( x_2 \) is infeasible solution; 2) \( x_1 \) and \( x_2 \) are both feasible solutions but \( x_1 \prec \varepsilon \); 3) \( x_1 \) and \( x_2 \) are both infeasible solutions but \( CV(x_1) < CV(x_2) - \varepsilon \). \( \varepsilon \)-constraint method is very similar to CDP except for one parameter \( \varepsilon \). Specifically, if the CV of an infeasible solution is less than \( \varepsilon \), then we think of it as a feasible solution. Otherwise, it is still infeasible solution. After redefining the feasibility of the solutions, CDP is then used to determine the constraint-dominated relation.

2) The second category tries to balance the trade-off between feasibility and convergence. In [14], stochastic ranking is proposed. When comparing two solutions, stochastic ranking will decide the winner randomly according to its value of objectives or constraint violation. [15] proposes a multi-objective-based method that transform constraint into objective and so a CMOP is transformed into a MOP. Recently, a Tri-Goal framework [16] is proposed to balance convergence, diversity and feasibility. In this framework, convergence, diversity and feasibility of a solution are regarded as three objectives for non-dominated sorting.

3) The third category is to hybrid several constraint handling techniques properly in order to get a better performance. [18] proposes an ensemble of constraint handling techniques to solve COPs, where each constraint handling technique has its own population and can learn from other populations to deal with complicate problems. [19] proposes \((\mu + \lambda)\)-constrained differential evolution that hybrids differential evolution with three mutation strategies and improved adaptive trade-off model which uses three CHTs to handle three types of situation. [20] introduces CDP and \( \varepsilon \)-constrained method in a DE framework.

B. Motivation

We can intuitively conclude that the more constraints, the more difficult it is to deal with. Under this consideration, we can deal with constraints one by one. Before formal explanation, we propose a new definition called constraint capability:

Constraint capability: for example, there are two constraints \( C_s \) and \( C_b \). Each constraint determines part of the infeasible region. Let us say \( R_s \) is the infeasible region determined by \( C_s \) and \( R_b \) is the infeasible region determined by \( C_b \). If the area or hypervolume of \( R_s \) is larger than \( R_b \), then we say that \( C_s \) has stronger constraint capability than \( C_b \).

When the constraints are many and complex, the real feasible region may be very narrow. It is hard for the population to find the right direction of evolution. However, when the algorithm only deals with one constraint, the feasible region will be much larger. After dealing with one constraint, then we add new constraints one by one, in doing so, the population can gradually approximate the real pareto front in the real feasible region. Considering the convergence rate, our proposed technique (MC-CHT) will deal with constraints from difficult to easy according to the constraint capability.
III. PROPOSED ALGORITHM

The algorithm proposed in this paper is called MC/MOA. The details of the whole algorithm are described in the following.

A. General framework

The main procedure is described in Algorithm 1. First, constrained sorting is used to get the sorted constraints according to constraint capability. Then, an initial population $P_0$ is generated randomly. Before the main loop, a procedure named environmentSelection is used to get $f_c$, $f_d$ and $f_f$ of each solution. $f_c$, $f_d$ and $f_f$ are values that can reflect the convergence, diversity and feasibility of solutions in population. In the main loop, we get $P'$ by selecting promising parents according to their $f_c$, $f_d$ and $f_f$. Then, after variation operators, we get $P'$. Finally, we combine $P$ and $P'$, and choose the next generation by environmentSelection.

Algorithm 1: general framework

Input: population size $N$, parameter $N_{MC}$, maximum iteration
Output: population $P$

1. sorted constraints $\leftarrow$ constrained sorting ($N_{MC}$) (see III-B, this procedure sort constraints according to constraint capability)
2. $P_0 \leftarrow$ initialization ($N$)
3. $t \leftarrow 0$
4. $[c_r, f_c, f_d, f_f] \leftarrow$ environmentSelection ($P$, sorted constraints, $N$) (see III-C)
5. while not terminated do
6. $P \leftarrow$ matingSelection ($f_f$, $f_c$, $f_d$) (see III-F)
7. $P' \leftarrow$ variation ($P'$)
8. $S \leftarrow P \cup P'$
9. $[P_{sel}, f_c, f_d, f_f] \leftarrow$ environmentSelection ($S$, sorted constraints, $N$)
10. $t \leftarrow t+1$
11. end

B. Constrained sorting

This procedure only needs one parameter, which is the number of generated solutions, denoted as $N_{MC}$. In this paper, we set it as 1000 by experiments. After generating these solutions, we calculate feasible proportion ($fp$) and constraint violation (CV) of all solutions under each constraint. Finally, we sort the constraints by feasible proportion and CV. The smaller the feasible proportion, the greater the constraint capability. If two constraints have same feasible proportion, then we will compare constraint violation of them. The rules are like feasible proportion. The details are showed in Algorithm 2.

Noted that constraint violation $CV_i$ means constraint violation under constraint $C_i$, where $C_i$ is one constraint.

Algorithm 2: constrained sorting

Input: parameter $N_{MC}$, the number of constraints $N_c$
Output: sorted constraints (SC)

1. $P_{MC} \leftarrow$ initialization ($N_{MC}$) (randomly generate solutions)
2. $fp_i \leftarrow$ feasibleProportion ($f_p_i$) $\leftarrow N_{MC}$, $CV_i \leftarrow 0$ for $i = 1, \ldots, N_c$
3. for each constraint $i$ ($C_i$) do
4. for each solution $x$ do
5. if $C_i(x) > 0$ then
6. $fp_i \leftarrow 1$
7. $CV_i \leftarrow C_i(x)$ ($C_i(x)$ means constraint violation of $x$ under constraint $i$)
8. end
9. end
10. end
11. sorted constraints $\leftarrow$ sort ($fp_i, CV_i$)

C. EnvironmentSelection

This procedure can calculate $f_c$, $f_d$ and $f_f$ of solutions and then choose promising solutions into next generation according to the values. The details are showed in Algorithm III. First, MC-CHT is used to calculate the current constraint violation (denoted as $f_f$) of each solution. Second, non-dominated sorting with constrained-dominance principle is used to calculate level (denoted as $f_d$) of each solution. $f_l$ includes solutions that have same level $l$. Third, a niching method is used to calculate $f_d$. Finally, we choose $N$ solutions according to $f_c$ and $f_d$.

D. MC-CHT

This operation is to enlarge the feasible region in the stage of evolution. And $\varepsilon$-constraint can also achieve this goal. So, this paper hybrid $\varepsilon$-constraint method to enhance the ability of MC-CHT. First, the number of constraints that need to be used in the current generation is calculated as follows:

$$d = \left\lceil \frac{N_c}{T_{max}} \right\rceil \times N_c$$

where $t$ is the current generation, $T_{max}$ is the maximum of generation and $N_c$ is the total number of constraints. $\varepsilon$ is the maximum of generation that use MC-CHT. Second, the constraint violation of solutions will be calculated as follows:

$$CV(x) \leftarrow \sum_{i=1}^{d} CV_i(x)$$

where $CV_i(x)$ is the constraint violation of $x$ under ith constraint in the sorted constraints. Third, $\varepsilon$-constraint will be used to calculate $f_f$ of solutions. If solution $x$ are
feasible under $\varepsilon$-constraint, then the $f_{i}$ equals to zero, otherwise, the $f_{i}$ equals to $CV(x)$.

**Algorithm 3: environmentSelection**

Input: population $P$, population size $N$, $\phi$, sorted constraints (SC)

Output: population $Q$, $f_e$, $f_d$, $f_f$

1. $Q \leftarrow \emptyset$, $l \leftarrow 1$
2. $f_f \leftarrow$ MC-CHT ($P, \phi, SC, N$) (see III-D)
3. $f_e \leftarrow$ constrained non-dominated sorting ($P, f_f$)
4. $f_d \leftarrow$ niching ($P$) (see III-E)
5. while $|Q| \leq N$ do
6. $Q \leftarrow Q \cup F_l$
7. $l \leftarrow l + 1$
8. end
9. if $|Q| > N$ then
10. $Q \leftarrow Q - F_l$
11. end
12. $P_{next} = \text{sort}(f_d(F_i))$
13. $Q \leftarrow P_{next}$

**E. Niching method**

In NSGAIII, when two solutions are close together in the same level, two extreme situations may occur. Fig 1 and Fig 2 show the two extreme situations. In Fig 1, solution E and F have bigger crowding distance than solution B and C ($f_i$ and $f_f$ are objectives). In this case, neither B or C can be chosen to be parents or into next generation. In Fig 2, when solution D and G are closer with E and F, situation are different. In this case, neither E or F can be chosen to be parents or into next generation. In either case, it will lead to a serious loss of population diversity. Under this consideration, we use a niching method to improve original crowding distance. In this paper, we use $\frac{1}{\sqrt[4]{N}}$ to determine whether the two solutions are close, where $N$ is the number of solutions and $M$ is the number of objectives. When some solutions are too close, we set $f_d$ of solutions to zero except only one. The details are showed in Algorithm 4.

**F. MatingSelection**

The matingSelection is used to choose promising parents from population. When two solutions are compared, solution with better feasibility will be selected. If they have same feasibility, then we will choose solution that have better convergence.

**IV. EXPERIMENT RESULTS**

A. Experimental setup

1) Test problems: DAS-CMOPs [21] are chosen as the
For IGD or HV. Especially on DASCMOPs-6, MC/MOEAs performs particularly well on HV indicator. CMOEAs/D performs best on DASCMOP2 whether on IGD and HV. It should be noted that the performance of MC/MOEAs is a little poor on this problem compared with CMOEA/D. KnEA performs best on DASCMOP8 whether on IGD or HV, while MC/MOEAs performs second worst on HV. But MC/MOEAs performs second best on IGD. ARMOEAs performs best on DASCMOP7 whether on IGD or HV, while MC/MOEAs performs second best on IGD. Overall, MC/MOEAs performs significantly better than other algorithms on DASCMOP1-9.

V. CONCLUSION

The goal of this paper was to propose an evolutionary algorithm that can generate a set of well distributed optimal solutions under constraints. And the goal was successfully achieved by two new techniques: one is a new constraint handling technique that can enlarge the feasible region in the early stage of evolution. The other is a new niching method that can maintain diversity better. The results showed that the proposed algorithm has high competitiveness compared with four popular algorithms. In the future, we still have a lot to expand in our work. For example, we can improve MC-CHT to accommodate different number of constraints. Moreover, we can apply MC-CHT to constrained many-objective optimization problems.

### TABLE I. STATISTICAL RESULTS (MEAN AND STANDARD DEVIATION) OF THE HV VALUES OBTAINED BY PEER ALGORITHMS FOR DASCMOP PROBLEMS.

<table>
<thead>
<tr>
<th>Problem</th>
<th>CMOEAD</th>
<th>ANSGAIII</th>
<th>KnEA</th>
<th>ARMOEA</th>
<th>MC/MOEAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DASCMOP1</td>
<td>4.0592e-3 (2.61e-3) =</td>
<td>5.6373e-3 (5.82e-3) =</td>
<td>5.7262e-3 (5.44e-3) =</td>
<td>2.9035e-3 (3.81e-3) -</td>
<td>9.0480e-3 (1.56e-2) -</td>
</tr>
<tr>
<td>DASCMOP2</td>
<td>2.4340e-1 (1.03e-2) +</td>
<td>2.3568e-1 (9.73e-3) =</td>
<td>2.2618e-1 (6.89e-3) -</td>
<td>2.3202e-1 (8.33e-3) -</td>
<td>2.3571e-1 (9.24e-3) -</td>
</tr>
<tr>
<td>DASCMOP3</td>
<td>1.9679e-1 (4.97e-2) -</td>
<td>2.0837e-1 (1.10e-2) -</td>
<td>1.9516e-1 (4.62e-2) -</td>
<td>1.9564e-1 (4.60e-2) -</td>
<td>2.0054e-1 (1.58e-2) -</td>
</tr>
<tr>
<td>DASCMOP4</td>
<td>2.0370e-2 (2.62e-2) -</td>
<td>2.4308e-2 (1.50e-2) -</td>
<td>3.0582e-2 (1.37e-2) -</td>
<td>2.4228e-2 (1.26e-2) -</td>
<td>5.1233e-2 (3.58e-2) -</td>
</tr>
<tr>
<td>DASCMOP5</td>
<td>6.8806e-2 (7.33e-2) -</td>
<td>5.1422e-2 (5.31e-2) -</td>
<td>5.4822e-2 (4.73e-2) -</td>
<td>5.7267e-2 (7.67e-2) -</td>
<td>1.4700e-1 (1.13e-1) -</td>
</tr>
<tr>
<td>DASCMOP6</td>
<td>4.2500e-2 (6.06e-2) -</td>
<td>2.5596e-2 (2.65e-2) -</td>
<td>1.9456e-2 (1.49e-2) -</td>
<td>1.9106e-2 (1.86e-2) -</td>
<td>1.0370e-1 (9.46e-2) -</td>
</tr>
<tr>
<td>DASCMOP7</td>
<td>1.7121e-1 (5.96e-2) -</td>
<td>2.1748e-1 (2.65e-2) =</td>
<td>2.3005e-1 (2.79e-2) -</td>
<td>2.3391e-1 (1.33e-2) =</td>
<td>2.6906e-1 (3.86e-2) -</td>
</tr>
<tr>
<td>DASCMOP8</td>
<td>8.8443e-2 (4.74e-2) -</td>
<td>1.3728e-1 (2.92e-2) =</td>
<td>1.5756e-1 (2.47e-2) +</td>
<td>1.4372e-1 (3.60e-2) =</td>
<td>1.2995e-1 (3.26e-2) -</td>
</tr>
<tr>
<td>DASCMOP9</td>
<td>9.6557e-2 (2.55e-2) -</td>
<td>9.7584e-2 (1.31e-2) -</td>
<td>9.3521e-2 (1.18e-2) -</td>
<td>9.5820e-2 (1.37e-2) -</td>
<td>1.0587e-1 (1.66e-2) -</td>
</tr>
</tbody>
</table>

+/-/= 1/6/2 0/4/5 1/5/3 0/5/4

### TABLE II. STATISTICAL RESULTS (MEAN AND STANDARD DEVIATION) OF THE IGD VALUES OBTAINED BY PEER ALGORITHMS FOR DASCMOP PROBLEMS.

<table>
<thead>
<tr>
<th>Problem</th>
<th>CMOEAD</th>
<th>ANSGAIII</th>
<th>KnEA</th>
<th>ARMOEA</th>
<th>MC/MOEAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DASCMOP1</td>
<td>7.2504e-1 (1.46e-2) -</td>
<td>7.2283e-1 (2.99e-2) =</td>
<td>7.2493e-1 (2.98e-2) =</td>
<td>7.3820e-1 (2.26e-2) -</td>
<td>7.1328e-1 (4.01e-2) -</td>
</tr>
<tr>
<td>DASCMOP2</td>
<td>2.7671e-1 (3.78e-2) -</td>
<td>2.9307e-1 (2.73e-2) -</td>
<td>3.3831e-1 (2.12e-2) -</td>
<td>3.0839e-1 (2.73e-2) -</td>
<td>2.9928e-1 (2.73e-2) -</td>
</tr>
<tr>
<td>DASCMOP3</td>
<td>3.7632e-1 (1.07e-1) -</td>
<td>3.4672e-1 (3.20e-2) -</td>
<td>3.8748e-1 (9.14e-2) -</td>
<td>3.7997e-1 (9.12e-2) -</td>
<td>2.9251e-1 (5.11e-2) -</td>
</tr>
<tr>
<td>DASCMOP4</td>
<td>6.7564e-1 (1.87e-1) -</td>
<td>5.0047e-1 (1.32e-1) -</td>
<td>5.5467e-1 (1.96e-2) -</td>
<td>5.6909e-1 (7.22e-2) -</td>
<td>4.7648e-1 (9.38e-2) -</td>
</tr>
<tr>
<td>DASCMOP5</td>
<td>6.1370e-1 (2.24e-1) -</td>
<td>6.7304e-1 (1.81e-1) -</td>
<td>6.1087e-1 (1.34e-1) =</td>
<td>6.5909e-1 (9.09e-1) -</td>
<td>4.1289e-1 (2.76e-1)</td>
</tr>
<tr>
<td>DASCMOP6</td>
<td>9.9696e-2 (2.06e-1) -</td>
<td>7.1496e-1 (1.32e-1) -</td>
<td>7.6614e-1 (1.83e-1) -</td>
<td>7.5042e-1 (1.13e-1) -</td>
<td>5.0575e-1 (9.02e-1) -</td>
</tr>
<tr>
<td>DASCMOP7</td>
<td>2.7333e-1 (1.63e-1) -</td>
<td>1.6588e-1 (6.64e-2) =</td>
<td>1.5142e-1 (8.06e-2) =</td>
<td>1.4247e-1 (8.27e-2) =</td>
<td>1.4933e-1 (9.12e-2) -</td>
</tr>
<tr>
<td>DASCMOP8</td>
<td>4.3384e-1 (2.58e-1) -</td>
<td>2.6775e-1 (1.47e-1) -</td>
<td>1.8022e-1 (9.28e-2) +</td>
<td>2.6501e-1 (1.74e-1) -</td>
<td>2.4411e-1 (1.10e-1) -</td>
</tr>
<tr>
<td>DASCMOP9</td>
<td>4.7719e-1 (1.48e-1) =</td>
<td>4.7389e-1 (6.31e-2) =</td>
<td>4.8661e-1 (6.23e-2) -</td>
<td>4.9047e-1 (7.67e-2) =</td>
<td>4.5258e-1 (8.60e-2) -</td>
</tr>
</tbody>
</table>

+/-/= 1/6/2 0/5/4 1/5/3 0/5/4

a, b indicate that the results are significantly better, worse or similar to that obtained by MC/MOEAs under rank sum test, respectively.