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

Multi-objective Differential Evolution Algorithm Based on Fast Sorting and a Novel Constraints Handling Technique

J. J. Liang[1], B. Zheng[1], F. Y. Xu[1]
[1]School of Electrical Engineering
Zhengzhou University
Zhengzhou, China
liangjing@zzu.edu.cn

B. Y. Qu[2] and H. Song[1]
[2]School of Electric and Information Engineering
Zhongyuan University of Technology
Zhengzhou, China
qby1984@hotmail.com

***Abstract*—In this paper, an improved multi-objective differential evolution algorithm is proposed to solve constraints in multi-objective optimization. Research has shown that the information of infeasible solutions is also important and can help the algorithm improve the convergence and diversity of solutions. A novel constraint handling method is introduced to ensure that a certain number of good infeasible solutions will be kept in the procedure of evolution to guide the search of the individuals. The proposed method is compared with two other constrained multi-objective differential evolution algorithms and the results show that the proposed method is competitive.**

Keywords—differential evolution; fast sorting; constraint handling; multi-objective problems

I. INTRODUCTION

In recent years, more and more researchers pay their attention to multi-objective problems, for the reason that multi-objective optimization problems exist in many fields of science and engineering. Much work has been done to solve multi-objective optimization problems [1]-[5]. In real applications, how to handle constraints remains an important issue. There have been some methods to handle constraints in single-objective problems. In early years, penalty functions [6], decoders [7], special operators [8], separation of objective functions and constraints [9] are main approaches to solve single-objective problems. In recent years, some new techniques have been proposed to solve constrained problems such as feasibility rules [10], stochastic ranking [11] and ε-constrained method [12].

Most of techniques for constraint handling in solving multi-objective problems are inspired by feasibility rules proposed in [9]. In the literature [3], Deb introduced the concept of constrained-domination. Some excellent methods have been used successfully in multi-objective problems with constraints. Coello and Christiansen [13] applied death penalty in their approach to eliminate infeasible solutions. Since the information of infeasible solutions can also improve the performance of algorithms, Deb [14] used static penalty and dynamic penalty which are also used in single-objective problems to handle constraints in multi-objective optimization. Adaptive penalty function and a distance measure were developed by Woldesenbet *et al.* [15] to improve the performance of their algorithm.

In this paper, an effective technique, which considering the effect of infeasible solutions, is proposed to handle constraints. This technique is combined with a multi-objective differential evolution algorithm based on fast-sorting method proposed by Qu and Suganthan [16], [17] to solve constrained optimization problems.

The structure of this paper is organized as follows. Section II presents the concept of constrained multi-objective optimization problem. In section III, the multi-objective differential evolution based on fast-sorting method proposed by Qu and Suganthan is described. The proposed technique of handling constraints is introduced in detail in section IV. Experiments and results are discussed in section V. Section VI draws the conclusion and and discusses future research.

This work was supported in part by National Natural Science Foundation of China (61305080, U1304602), Postdoctoral Science Foundation of China (Grants 20100480859), Specialized Research Fund for the Doctoral Program of Higher Education (20114101110005), Scientific and Technological Project of Henan Province (132102210521, 122300410264), and Key Foundation of Henan Educational Committee (14A410001).

II. CONSTRAINED MULTI-OBJECTIVE OPTIMIZATION PROBLEM

For most cases, a constrained multi-objective optimization problem can be described as follows:

$$\begin{aligned} \textit{minimize} \quad & f(x) = [f_1(x), f_2(x), ..., f_m(x)] \\ & m = 1, 2, ..., m \\ \textit{subject to} \quad & h_k(x) = h_k(x_1, x_2, ..., x_n) = 0, \\ & k = 1, 2, ..., K \\ & g_j(x) = g_j(x_1, x_2, ..., x_n) \le 0, \\ & j = 1, 2, ..., J \\ & x_i^{\min} \le x_i \le x_i^{\max}, \quad i = 1, 2, ..., n \end{aligned} \tag{1}$$

where, $f(x) = [f_1(x), f_2(x), ..., f_m(x)]$ is the set of m objective functions to be minimized simultaneously, $h_k(x) = h_k(x_1, x_2, ..., x_n) = 0$ are equality constraints, the inequality constraints are $g_j(x) = g_j(x_1, x_2, ..., x_n) \le 0$. The equality constraints can be converted to inequality constraints by introducing a tolerance constant δ, then the form of equality constraints is indicated as $|h_k(x)| - \delta \le 0$.

There are some methods to handle constraints. One of these methods is penalty function [6]. However, the difficulty of penalty function is the requirement of specifying a number of parameters. Another popular method is feasibility rules proposed by Deb [3]. The rules are defined as follows:

Definition 1: A solution *i* is said to constrained-dominate a solution *j*, if any of the following conditions is true.

- Solution *i* is feasible and solution *j* is not;
- Solutions *i* and *j* are both infeasible, but solution *i* has a smaller overall constraint violation;
- Solutions *i* and *j* are feasible and solution *i* dominates solution *j*.

From the definition of the rules, it is obvious that, according to Deb's rules, feasible solutions have greater priority in the process of evolution than infeasible solutions. By researching the test functions for constrained problems [18], we found that, in some cases, information of infeasible solutions is also important in evolution process. To improve the performance of algorithm, a novel constraint handling technique which makes use of the information of both feasible and infeasible solutions for handling constraints is proposed and applied in the multi-objective differential evolution. The proposed method will be discussed in detail in section IV.

III. MULTI-OBJECTIVE DIFFERENTIAL EVOLUTION

In recently years, evolutionary algorithms (EAs) have been successfully used to solve optimization problems. Among them, differential evolution algorithm (DE) is an excellent approach for optimization problems. DE was first proposed by Storn and Price [19] in 1995, and has been used to solve single objective and multi-objective problems [20], and the performance has been proved to be very effective. Some researchers embedded constraints handling techniques into DE for constrained optimization. Vargas et al. applied adaptive penalty method in DE for constrained multi-objective optimization. Qu *et al.* [21] proposed a diversity enhanced constrained multi-objective differential evolution (DE-CMODE). DE-CMODE overcomes the pre-mature convergence problem which exists in constrained multi-objective differential evolution (CMODE). The procedure of typical multi-objective differential evolution (MODE) is given in Fig. 1.

In this paper, an improved multi-objective differential evolution proposed by Qu and Suganthan [16], [17] is used. The main difference between the improved MODE and basic MODE is that a new non-domination sorting method called fast-sorting is used in the improved MODE. It has been proved that the fast-sorting method is effectively in solving multi-objective problems [16], [17]. In addition, the complexity of the method is lower than the commonly used non-domination sorting method.

Fast-sorting consists of two parts. One is summation of normalized objective values (SNOV) and the other one is diversified selection (DS). There are some details need to be noticed in the procedure of DS:

1) Current population will be divided into 2 sets, preferential and backup set;
2) Solutions in the preferential set will be selected first for evolving;
3) If there are not enough solutions in preferential set, solutions in backup set will be selected based on SNOV;
4) If the number of solutions in preferential set is larger than needed, the required solutions will be selected randomly from the preferential set.

The procedures of SNOV and DS are given in Fig. 2 and Fig. 3 respectively.

IV. NOVEL CONSTRAINTS HANDLING METHOD

As mentioned in section II, the weakness of the method proposed by Deb in NSGA-II is that the method pefers feasible solutions to infeasible solutions and does not make full use of the information of infeasible solutions. To overcome this problem and enhance the diversity, the proposed method divides the set *R* which is combined by parents and offspring in every generation into two subsets. *R1* is the first subset which is composed of all feasible solutions. All infeasible solutions consist of the second

subset *R2*. *R1* is sorted by fast-sorting method as used in Fig. 2 and Fig. 3. Then *R'1* is obtained. For *R2*, the fast-sorting rule is modified. The original fast-sorting sorts the solutions based on objective functions. Different from the original one, the modified fast-sorting method is based on constraints. In this method, objective functions are replaced by constraints. The goal of this method is to minimize the constraints. Then *R'2* is obtained. At last *R'1* and *R'2* make up a new set *R'* which is used as the parents for next generation. The advantage of the proposed method

1. Randomly generate a number of initial trial solutions and initialize the external archive;
2. Create N_p number of offspring solutions from the parents;
3. Combining the parents and offspring into a solutions pool;
4. Identify the non-dominated solutions and assign the front number to each of the solutions;
5. Sort the total solution in the ascending order with respect to the front number;
6. Select the first N_p solutions and update the external archive;
7. Stop if the criterion is satisfied, otherwise go to 2.

Fig. 1. Procedure of the basic MODE

Step 1: For m = 1 to M // M is the number of objectives

Normalize the objective value of each member by (2)

$$f_m'(x) = \frac{f_m(x) - f_{\min}}{f_{\max} - f_{\min}} \quad (2)$$

Where $f_{\max}$ and $f_{\min}$ are the maximum and minimum objective values of the m^{th} objective. $f_m'(x)$ is the normalized m^{th} objective value.

End For

Step 2: For i = 1 to NP //NP is the size of population

Sum all normalized objective values of the member to obtain a single value.

End For

Fig. 2. Procedure of SNOV

step 1: For m = 1 to M //M is the number of objectives

a) Divide the range of the objective space into 100 bins equally

b) Scan P percentage of the 100 bins (*i.e.* from bin 1 to P, P may be chosen as 80 or 90, and 90 is used in our experiment)

c) For each scanned bin (if this is empty, otherwise just continue to next bin), the solution with the smallest summation of normalized objective values will be chosen to enter preferential set.

End For

Step 2: Accumulate the solutions excluded from the preferential set and store them in backup

Fig. 3. Procedure of DS

is that the information of infeasible solutions is kept for next generation. The diversity of solutions is improved. Since *R2* which consists of all infeasible solutions is sorted based on constraints, bad infeasible solutions (far away from the feasible space) are also removed, and the good infeasible solutions are kept. The flow chart of the algorithm in this paper is given in Fig. 4.

V. EXPERIMENT AND RESULTS

A. Experiments and Rsults

Three algorithms are tested in the experiments, including CMODE, DE-CMODE and the improved DE proposed in this paper. The population size of each algorithm used in this paper is set as 100. The parameters of DE used in this paper are set as follows:

F=0.3; CR=0.3; FES=200000;

B. Test Functions

Eight commonly used functions [18] are employed as benchmarks in this paper.

1) TNK:

$$\textit{Minimize}: f_1(x) = x_1$$
$$f_2(x) = x_2$$

Constraints:

$$C_1(x) = x_1^2 + x_2^2 - 1 - 0.1\cos\left(16\arctan\left(x_1 / x_2\right)\right) \ge 0$$
$$C_2(x) = \left(x_1 - 0.5\right)^2 + \left(x_2 - 0.5\right)^2 \le 0.5$$

Where, $0 \le x_1 \le \pi, 0 \le x_2 \le \pi$

2) SRN:

$$\textit{Minimize}: f_1(x) = 2 + \left(x_1 - 2\right)^2 + \left(x_2 - 1\right)^2$$
$$f_2(x) = 9x_1 - \left(x_2 - 1\right)^2$$

Constraints: $C_1(x) = x_1^2 + x_2^2 \le 225$
$$C_2(x) = x_1 - 3x_2 + 10 \le 0$$

Where, $-20 \le x_1 \le 20, -20 \le x_2 \le 20$

3) OSY:

Minimize:

$$f_1(x) = -\left[25\left(x_1 - 2\right)^2 + \left(x_2 - 2\right)^2 + \left(x_3 - 1\right)^2 + \left(x_4 - 4\right)^2 + \left(x_5 - 1\right)^2\right]$$
$$f_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$$

Constraints: $C_1(x) = x_1 + x_2 - 2 \geq 0$

$C_2(x) = 6 - x_1 - x_2 \geq 0$

$C_3(x) = 2 - x_2 + x_1 \geq 0$

$C_4(x) = 4 - (x_3 - 3)^2 - x_4 \geq 0$

$C_5(x) = 2 - x_1 + 3x_2 \geq 0$

$C_6(x) = (x_5 - 3)^2 + x_6 - 4 \geq 0$

Where, $x_1 \geq 0, x_2, x_6 \leq 10$

$1 \leq x_3, x_5 \leq 5, 0 \leq x_4 \leq 6$

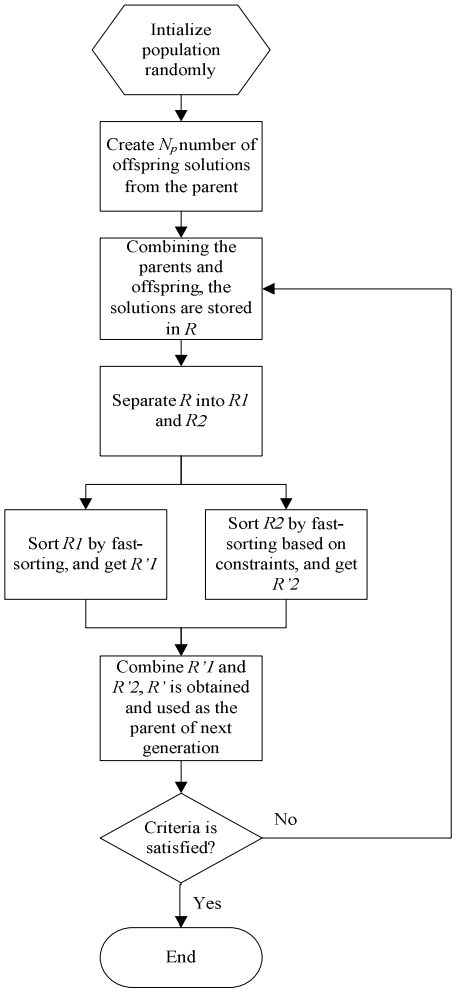

Fig. 4. Flow Chart of Algorithm

4) CTP1:

Minimize: $f_1(x) = x_1$

$$f_2(x) = g(x)\exp\left(-\frac{f_1(x)}{g(x)}\right)$$

Constraints: $f_2(x) - a_j \exp(-b_j f_1(x)) \geq 0$

$j = 1, \ldots, J$

Where,

$0 \leq x_1 \leq 1, -5 \leq x_2, x_3, x_4 \leq 5$ *and*

$g(x) = 31 + \sum_{i=2}^{4}\left(x_i^2 - 10\cos(4\pi x_i)\right)$

$J = 2, a_1 = 0.858, a_2 = 0.541, \mathrm{b}_2 = 0.728$ *and* $b_2 = 0.295$

5) CTP2-5:

Minimize: $f_1(x) = x_1$

$$f_2(x) = g(x)\left(1 - \sqrt{\frac{f_1}{g(x)}}\right)$$

Constraints:

$\cos(\theta)(f_2(x) - e) - \sin(\theta) f_1(x) \geq$

$a\left|\sin\left(b\pi\left(\sin(\theta)(f_2(x) - e) + \cos(\theta) f_1(x)\right)^c\right)\right|^d$

Where, $0 \leq x_1 \leq 1, -5 \leq x_2, x_3, x_4 \leq 5$ *and*

$g(x) = 31 + \sum_{i=2}^{4}\left(x_i^2 - 10\cos(4\pi x_i)\right)$

The parameters of CTP2-5 are given in Table I.

TABLE I PARAMETERS of CTP2-CTP5

	θ	a	b	c	d	e
CTP2	-0.2 π	0.2	10	1	6	1
CTP3	-0.2 π	0.1	10	1	0.5	1
CTP4	-0.2 π	0.75	10	2	0.5	1
CTP5	–0.05π	40	5	1	6	0

C. Performance Measure

In order to compare the performance of the algorithms, two indicators, I_{R2} and $I_{\bar{H}}$, are picked as performance metric. They are summarized in [22].

R indicator (I_{R2}) is used to effectively measure the difference in the mean distance of the attainment surfaces from the reference point. I_{R2} defined as:

$$I_{R2} = \frac{\sum_{\lambda \in \Lambda} u^*(\lambda, A) - u^*(\lambda, R)}{|\Lambda|}$$

Where R is a reference set, u* is the maximum value of the utility function u with weight λ on an approximation set A. In our experiment, the augmented Tchebycheff function is chosen as the utility function.

Hypervolume difference to a reference set ($I_{\bar{H}}$) values represent the diversity of the algorithm. I_H indicates the hypervolume of the objective space which is dominated by set A. Considering R a reference set. $I_{\bar{H}}$ is defined as $I_{\bar{H}} = I_H(R) - I_H(A)$. Obviously, a smaller $I_{\bar{H}}$ means a better

diversity of the approximation set.

.TABLE II R-INDICATOR FOR 30 RUNS

		proposed	CMODE	DE-CMODE
TNK	best	-9.75e-06	1.02e-04	1.01e-04
	worst	8.26e-05	5.53e-04	6.21e-04
	mean	**3.24e-05**	2,86e-04	3.17e-04
	std	2.56e-05	1.25e-04	1.55e-04
OSY	best	-4.57e-05	4.51e-05	3.26e-06
	worst	0.10e-03	3.85e-02	4.91e-04
	mean	**1.17e-04**	1.18e-02	2.21e-04
	std	2.83e-04	1.35e-02	1.37e-04
SRN	best	-5.14e-06	3.33e-05	2.39e-05
	worst	9.40e-05	8.16e-04	5.96e-04
	mean	**3.43e-05**	2.31e-04	1.85e-04
	std	2.06e-05	1.97e-04	1.47e-04
CTP1	best	-5.86e-10	2.76e-08	4,32e-08
	worst	3.71e-07	3.77e-04	1.63e-05
	mean	**6.80e-08**	1.21e-04	3.13e-06
	std	1.24e-07	1.79e-04	4.27e-06
CTP2	best	7.07e-08	9.97e-07	4.33e-07
	worst	6.74e-06	2.30e-05	1.07e-05
	mean	**1.41e-06**	9.09e-06	6.08e-06
	std	1.43e-06	7.05e-06	4.61e-06
CTP3	best	-3.50e-06	1.66e-05	2.48e-05
	worst	1.25e-04	2.36e-04	2.29e-04
	mean	**6.20e-05**	1.31e-04	1.30e-04
	std	3.15e-05	5.68e-05	5.09e-05
CTP4	best	1.07e-04	7.69e-05	7.69e-05
	worst	9.82e-04	1.31e-02	2.30e-03
	mean	**4.98e-04**	3.00e-03	1.10e-03
	std	2.53e-04	4.10e-03	6.04e-04
CTP5	best	0	1.31e-08	1.90e-08
	worst	1.55e-07	6.40e-03	4.48e-07
	mean	**1.15e-08**	7.24e-04	1.32e-07
	std	2.97e-08	1.80e-03	1.30e-07

D. Results and Comparison

For each algorithm, the 8 test functions are run 30 times, and the average of I_{R2} and $I_{\bar{H}}$ are shown in Table II and Table III. The best of the results are denoted in bold.

TNK and OSY are two commonly used test problems. The constraints of the two problems are easier, compared with other test problems. All of the three algorithms used in this paper perform well on these two problems. However,

TABLE III H-INDICATOR FOR 30 RUNS

		proposed	CMODE	DE-CMODE
TNK	best	-6.55e-06	2.94e-04	2.95e-04
	worst	1.26e-04	1.50e-03	1.70e-03
	mean	**5.52e-05**	7.82e-04	8.65e-04
	std	3.71e-05	3.28e-04	4.06e-04
OSY	best	2.20e-04	1.90e-03	1.80e-03
	worst	1.40e-03	6.31e-02	7.70e-03
	mean	**4.00e-04**	1.90e-02	2.80e-03
	std	2.67e-04	2.03e-02	1.10e-03
SRN	best	9.90e-04	2.90e-03	2.80e-03
	worst	1.20e-03	3.60e-03	3.50e-03
	mean	**1.10e-03**	3.20e-03	3.10e-03
	std	5.64e-05	2.04e-04	2.00e-04
CTP1	best	5.44e-06	5.89e-06	6.10e-06
	worst	7.92e-06	9.64e-04	3.01e-05
	mean	**5.97e-06**	3.13e-04	1.11e-05
	std	5.84e-07	4.55e-04	6.39e-06
CTP2	best	1.19e-06	3.96e-06	4.39e-06
	worst	1.16e-05	3.25e-05	2.05e-05
	mean	**3.88e-06**	1.36e-05	1.13e-05
	std	2.19e-06	8.27e-06	4.35e-06
CTP3	best	2.41e-05	4.85e-05	6.88e-05
	worst	2.29e-04	3.74e-04	3.64e-04
	mean	**1.21e-04**	2.20e-04	2.24e-04
	std	4.70e-05	8.29e-05	7.57e-05
CTP4	best	2.09e-04	1.77e-04	1.88e-04
	worst	1.60e-03	2.03e-02	3.40e-03
	mean	**8.14e-04**	4.70e-03	1.80e-03
	std	3.92e-04	6.60e-03	8.92e-04
CTP5	best	2.25e-06	1.03e-06	1.04e-06
	worst	2.65e-06	1.02e-02	2.28e-06
	mean	2.48e-06	1.10e-03	**1.29e-06**
	std	9.80e-08	2.80e-03	2.75e-07

we can see that the proposed method is much better than the other algorithms both in convergence and diversity of solutions.

For SRN, solutions obtained by the proposed method are much closer to the reference front than the others. As for diversity of solutions, the three algorithms are almost the

same. The advantage of proposed method is not significant.

CTP1 – CTP5 are relatively difficult test problems. The properties of these five problems can be tuned by adjusting parameters. Compared with the other two algorithms, the performance of the proposed method stands out on convergence and diversity of solutions when tested on CTP1 – CTP4. Although the proposed method performs better in convergence to the true Pareto-optimal, it's a little worse than DE-CMODE considering the diversity of solutions.

After all, some conclusions can be drawn. Compared with the other algorithms, the results of the proposed method are competitive considering the convergence to Pareto-optimal. As for the diversity of solutions, the proposed method performs better than the other algorithms except for CTP5. Generally, the proposed method performs well on all the 8 benchmark test functions. The main reason is that the information of infeasible is kept well in each generation, and contributes to the search procedure.

VI. CONCLUSION

In this paper, an improved differential evolution is modified to solve the constrained multi-objective optimization problems. The proposed method sorts the feasible solutions and infeasible solutions separately, hence feasible solutions and infeasible solutions all have chance to survive in the next generation. In this way, the information of both feasible and infeasible solutions is maintained in the search procedure. The performance of the proposed method is proved competitive on eight commonly used benchmark functions through comparing with two other state-or-art constrained MODEs. For future work, an adaptive function will be applied to adjust the rate of infeasible solutions in the population.

REFERENCES

[1] E. Mezura-Montes and C. A. C. "Cello, Constraint-handling in nature inspired numerical optimization: Past, present and future," Swarm and Evolutionary Computation, vol. 1, 2011, pp. 173-194.

[2] A. Zhou, B. Y. Qu, H. Li, S. Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," Swarm and Evolutionary Computation, vol. 1, 2011, pp. 32-49.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-□," IEEE Transactions on Evolutionary Computation, vol. 6, 2002, pp. 182-197.

[4] L. Tang and X. Wang, "A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems," IEEE Transactions on Evolutionary Computation, vol. 17, 2003, pp. 20–45.

[5] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," IEEE Transactions on Evolutionary Computation, vol. 8, 2004, pp. 256–279.

[6] A. E. Smith and D. W. Coit, "Penalty functions," Handbook of Evolutionary Computation, 1997, pp. C5: 1-6.

[7] S. Koziel and Z. Michalewicz, "A decoder-based evolutionary algorithm for constrained parameter optimization problems," in Parallel Problem Solving from Nature – PPSN V. Springer Berlin Heidelberg, 1998, pp. 231–240.

[8] Z. Michalewicz, "Genetic algorithms + data structures = evolution programs," springer, 1996.

[9] K. Deb, "An efficient constraint handling method for genetic algorithms," Computer methods in applied mechanics and engineering, vol. 186, 2000, pp. 311–338.

[10] E. Mezura-Montes and C. A. Coello Coello, "A simple multimembered evolution strategy to solve constrained optimization problems," IEEE Transactions on Evolutionary Computation, vol. 9, 2005, pp. 1–17.

[11] T. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," IEEE Transactions on Evolutionary Computation, vol. 4, 2000, pp. 284–294.

[12] T. Takahama, S. Sakai, and N. Iwane, "Constrained optimization by the constrained hybrid algorithm of particle swarm optimization and genetic algorithm," in AI 2005: Advances in Artificial Intelligence, ser. Lecture Notes in Computer Science, S. Zhang and R. Jarvis, Eds. Springer Berlin Heidelberg, 2005, vol. 3809, pp. 389-400.

[13] C. A. Coello Coello and A. D. Christiansen, "Moses: A multiobjective optimization tool for engineering design," Engineering Optimization, vol. 31, 1999, pp. 337-368.

[14] K. Deb and D. Kalyanmoy, "Multi-Objective Optimization Using Evolutionary Algorithms," USA: John Wiley & Sons, 2001.

[15] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema, "Constraint handling in multiobjective evolutionary optimization," IEEE Transactions on Evolutionary Computation, vol. 13, 2009, pp. 514–525.

[16] B. Y. Qu and P. N. Suganthan, "Multi- objective differential evolution based on the summation of normalized objectives and diversified selection," Information sciences, vol. 180, 2010, pp 3170 – 3181.

[17] B. Y. Qu and P. N. Suganthan, "Multi - objective Differential Evolution based on the Summation of Normalized Objectives and Improved Selection Method," IEEE Symposium on Differential Evolution, 2011.

[18] K. Deb, A. Pratap, and T. Meyarivan, Constrained Test Problems for Multi-objective Evolutionary Optimization," Computer Science, vol. 1993, 2001, pp 284-298.

[19] R. Storn and K. V. Price, "Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol. 11, 1995, pp. 341-359.

[20] Y. Zhao, S. Xiong, and M. Li, "Constrained single- and multiple-objective optimization with differential evolution," Third International Conference on Natural Computation, ICNC 2007,vol. 4, 2007, pp. 451-455. Third International Conference on Natural Computation, ICNC 2007.

[21] B. Y. Qu and P. Suganthan, "Constrained multi-objective optimization algorithm with diversity enhanced differential evolution," 2010 IEEE Congress on Evolutionary Computation (CEC), 2010, pp. 1–5.

[22] J. Knowles, L. Thiele, and E. Zitzler, "A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers," Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland, Feb 2005.