Differential Evolution with Ensemble of Constraint Handling Techniques for solving CEC 2010 Benchmark Problems

Rammohan Mallipeddi and Ponnuthurai Nagaratnam Suganthan

Abstract— **Several constraint handling techniques have been proposed to be used with the evolutionary algorithms (EAs). According to the no free lunch theorem, it is impossible for a single constraint handling technique to outperform all other techniques on every problem. In other words, depending on several factors such as the ratio between feasible search space and the whole search space, multi-modality of the problem, the chosen EA and global exploration/local exploitation stages of the search process, different constraint handling techniques can be effective on different problems and during different stages of the search process. Motivated by these observations, we proposed an ensemble of constraint handling techniques (ECHT) to solve constrained real-parameter optimization problems. In ECHT, each constraint handling method has its own population and every function call is used effectively. Being a general concept, the ECHT can be realized with any existing EA. In this paper, we present ECHT with Differential Evolution (DE) as the basic search algorithm (ECHT-DE). The ECHT is formed using four different constraint handling techniques present in the literature. ECHT-DE is evaluated on the functions from CEC 2010 problem set.**

I. INTRODUCTION

Many optimization problems in science and engineering involve constraints, which complicate the search process by reducing the feasible region. Evolutionary algorithms (EAs) require additional mechanisms to handle constraints while solving constrained optimization problems. In the literature, several constraint handling techniques have been proposed to be used with the EAs [1].

In constraint optimization, the way in which the infeasible individuals are handled throughout the search process is crucial. EAs are probabilistic search methods and completely discarding the infeasible individuals may be ineffective as the potential information present in the infeasible individuals goes unutilized. In discontinuous search space, the EA may be trapped in one of the local minima due to the ineffective handling of the infeasible individuals. Therefore, different techniques have been developed to exploit the information in infeasible individuals. Michalewicz and Schoenauer [2] grouped the methods for handling constraints within EAs into four categories: preserving feasibility of solutions [3], penalty functions, make a separation between feasible and infeasible solutions, and hybrid methods.

According to the no free lunch theorem [4], no single state-of-the-art constraint handling technique can outperform all others on every problem. Hence, solving a particular constrained problem requires numerous trial-and-error runs to choose a suitable constraint handling technique and to fine tune the associated parameters. This approach clearly suffers from unrealistic computational requirements in particular if the objective function is computationally expensive or solutions are required in real-time. Motivated by these observations, we proposed [5] an ensemble of constraint handling techniques (ECHT) with four constraint handling techniques [1, 6-8], as an efficient alternative to the trial-and-error based search for the best constraint handling technique with its best parameters for a given problem. In ECHT, each constraint handling technique has its own population and each function call is efficiently utilized.

Differential Evolution (DE) [9] is a population based stochastic search technique which is a fast and simple technique. Since it was proposed [10], it has been successfully applied in diverse fields. DE has also been used to solve constrained optimization problems [6, 11-16].

The remainder of this paper is organized as follows. Section II presents the global constrained optimization problem and a brief overview of the constraint handling techniques used to form ECHT. Section III presents the generalized ECHT algorithm. The results and discussions are presented in Section IV. Section V concludes the paper.

II. CONSTRAINT HANDLING METHODS

A constrained optimization problem with n parameters to be optimized is usually written as a nonlinear programming problem of the following form [17]:

$$\text{Minimize: } f(X), \quad X=(x_1,x_2,...,x_n) \text{ and } X\in S \tag{1}$$

$$\text{subject to: } \begin{array}{ll} g_i(X)\le 0, & i=1,...,p \\ h_j(X)=0, & j=p+1,...,m \end{array}$$

Here f does not need to be continuous but it must be bounded. S is the whole search space. p is the number of inequality constraints. The number of equality constraints is $(m-p)$. The inequality constraints that satisfy $g_i(X)=0$ at the global optimum solution are called active constraints. All equality constraints are active constraints. The equality constraints can be transformed into inequality form and can be combined with other inequality constraints as

$$G_i(X)=\begin{cases}\max\{g_i(X),0\} & i=1,...p. \\ \max\{|h_i(X)|-\delta,0\} & i=p+1,...,m\end{cases} \tag{2}$$

R. Mallipeddi and P. N. Suganthan are with School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, 639798 (e-mail: mall0004@ntu.edu.sg, epnsugan@ntu.edu.sg).

Where δ is a tolerance parameter for the equality constraints. Therefore, the objective is to minimize the fitness function $f(X)$ such that the optimal solution obtained satisfies all the inequality constraints $G_i(X)$. The overall constraint violation for an infeasible individual is a weighted mean of all the constraints, which is expressed as:

$$\upsilon(X) = \frac{\sum_{i=1}^{m} w_i(G_i(X))}{\sum_{i=1}^{m} w_i} \tag{3}$$

where $w_i \ (= 1/G_{\max_i})$ is a weight parameter, $G_{\max_i}$ is the maximum violation of constraint $G_i(X)$ obtained so far. Here, w_i is set as $1/G_{\max_i}$ which varies during the evolution in order to balance the contribution of every constraint in the problem irrespective of their differing numerical ranges.

A brief overview of the four constraint handling techniques used to form the ECHT is presented below:

A. Superiority of Feasible Solutions (SF)

In SF [18], when comparing two solutions X_i and X_j, X_i is regarded superior to X_j when

- X_i is feasible and X_j is not.
- X_i and X_j are both feasible and X_i has a smaller objective value (in a minimization problem) than X_j.
- X_i and X_j are both infeasible, but X_i has a smaller overall constraint violation $\upsilon(X_i)$ as computed by using Eqn (3).

Therefore, in SF, feasible ones are always considered better than the infeasible ones. Two infeasible solutions are compared based on their overall constraint violations only, while two feasible solutions are compared based on their objective function values only. Comparison of infeasible solutions based on the overall constraint violation aims to push the infeasible solutions to feasible region, while comparison of two feasible solutions on the objective value improves the overall solution.

B. Self adaptive Penalty (SP)

The simplest and the earliest method of involving infeasible individuals in the search process are by employing a static penalty approach. Static penalty functions are popular due to their simplicity. However, it involves problem-dependent parameters to be set by the user when multiple constraints are violated. To overcome this difficulty, adaptive penalty functions [19] are suggested where information gathered from the search process will be used to control the amount of penalty added to infeasible individuals. Adaptive penalty functions are easy to implement and they do not require users to define parameters.

In [8], a self adaptive penalty function method is proposed to solve constrained optimization problems. Two types of penalties are added to each infeasible individual to identify the best infeasible individuals in the current population. The two penalties will allow the algorithm to switch between finding more feasible solutions and searching for the optimum solution at anytime during the search process. This algorithm required no parameter tuning. The final fitness value based on which the population members are ranked is given as $F(X) = d(X) + p(X)$, where $d(X)$ is the distance value and $p(X)$ is the penalty value. The distance value is computed as follows:

$$d(X) = \begin{cases} \upsilon(X), & \text{if } r_f = 0 \\ \sqrt{f''(X)^2 + \upsilon(X)^2}, & \text{otherwise} \end{cases} \tag{4}$$

where $r_f = \dfrac{\text{Number of feasible individuals}}{\text{population size}}$,

$\upsilon(X)$ is the overall constrain violation as defined in Eq (3), $f''(X) = \dfrac{f(X) - f_{\min}}{f_{\max} - f_{\min}}$. $f_{\max}$ and $f_{\min}$ are the maximum and minimum values of the objective function $f(X)$ in the current combined population. The penalty value is defined as $p(X) = (1 - r_f)M(X) + r_f N(X)$ where

$$M(X) = \begin{cases} 0, & \text{if } r_f = 0 \\ \upsilon(X), & \text{otherwise} \end{cases} \tag{5}$$

$$N(X) = \begin{cases} 0, & \text{if } X \text{ is a feasible individual} \\ f''(X), & \text{if } X \text{ is an infeasible individual} \end{cases} \tag{6}$$

C. $\varepsilon -$Constraint (EC)

The ε-constraint handling method was proposed in [15] in which the relaxation of the constraints is controlled by using the ε parameter. As solving a constrained optimization problem becomes tedious when active constraints are present, proper control of the ε parameter is essential [15] to obtain high quality solutions for problems with equality constraints. The ε level is updated until the generation counter k reaches the control generation T_c. After the generation counter exceeds T_c, the ε level is set to zero to obtain solutions with no constraint violation.

$$\varepsilon(0) = \upsilon(X_\theta) \tag{7}$$

$$\varepsilon(k) = \begin{cases} \varepsilon(0)\left(1 - \dfrac{k}{T_c}\right)^{cp}, & 0 < k < T_c \\ 0, & k \geq T_c \end{cases} \tag{8}$$

where X_θ is the top θ-th individual and $\theta = (0.05 * NP)$. The recommended parameter ranges are [15]: $T_c \in [0.1T_{\max}, 0.8T_{\max}]$ and $cp \in [2,10]$.

D. Stochastic Ranking (SR)

Runarsson and Yao [7] introduced a stochastic ranking (SR) method to achieve a balance between objective and the overall constraint violation stochastically. A probability factor p_f is used to determine whether the objective function value or the constraint violation value determines the rank of each individual. Basic form of the SR [7] is:

```
If (no constraint violation or rand < $p_f$)
    Rank based on the objective value only
else
    Rank based on the constraint violation only
End
```

In [20], an improved version of the SR (ISR) was proposed using evolution strategies and differential variation. In SR, comparison between two individuals may be based on objective value alone or overall constraint violation alone as randomly determined. Thus, infeasible solutions with better objective value have a chance to be selected in all three phases of evolution. In our work, a modified version of the SR presented in [7] is used. Here, the value of p_f is not maintained a constant instead, decreased linearly from $p_f = 0.475$ in the initial generation to $p_f = 0.025$ in the final generation.

III. ECHT

Each constrained optimization problem would be unique in terms of the ratio between feasible search space and the whole search space, multi-modality and the nature of constraint functions. As evolutionary algorithms are stochastic in nature, the evolution paths can be different in every run even when the same problem is solved by using the same algorithm. In other words, the search process passes through different phases at different points during the search process. Therefore, depending on several factors such as the ratio between feasible search space and the whole search space, multi-modality of the problem, nature of equality / inequality constraints, the chosen EA and global exploration/local exploitation stages of the search algorithm, different constraint handling methods can be effective during different stages of the search process. Due to the strong interactions between these diverse factors and the stochastic nature of the evolutionary algorithms, it is not straightforward to determine which constraint handling method is the best during a particular stage of the evolution to solve a given problem using a given EA. Motivated by these observations, we proposed the ECHT to implicitly benefit from the match between constraint handling methods, characteristics of the problem being solved, chosen EA and the exploration-exploitation stages of the search process.

A real-world problem can take several minutes to several hours to compute the objective function value. Therefore, finding a better constraint handling method for such problem by trial-and-error may become difficult. The computation time wasted in searching for a better constraint handling method can be saved by using the proposed ensemble method.

In this paper, ECHT is formed with the four constraint handling techniques discussed in Sections II-A-D. Each constraint handling technique has its own population and parameters. Each population corresponding to a constraint handling method produces its offspring and evaluates them. The parent population corresponding to a particular constraint handling method not only competes with its own offspring population but also with offspring population of the other three constraint handling methods. Due to this, an offspring produced by a particular constraint handling method may be rejected by its own population, but could be accepted by the populations of other constraint handling methods. Hence, in ECHT every function call is utilized effectively. If the evaluation of objective / constraint functions is computationally expensive, more constraint handling methods can be included in the ensemble to benefit more from each function call. And if a particular constraint handling technique is best suited for the search method and the problem during a point in the search process, the offspring population produced by the population of that constraint handling method will dominate the other and enter other populations too. In the subsequent generations, these superior offspring will become parents in other populations too. Therefore, ECHT transforms the burden of choosing the best constraint handling technique and tuning the associated parameter values for a particular problem into an advantage. If the constraint handling methods selected to form an ensemble are similar in nature then the populations associated with each of them may lose diversity and the search ability of ECHT may deteriorate. Thus, the performance of ECHT can be improved by selecting constraint handling methods with diverse and competitive nature. The general framework of the ensemble algorithm is illustrated in the flowchart shown in Fig. 1.

The effectiveness of conventional DE in solving a numerical optimization problem depends on the selected mutation strategy and its associated parameter values. However, different optimization problems require different mutation strategies [21, 22] with different parameter values depending on the nature of problem (uni-modal and multi-modal) and available computation resources. In addition, to solve a specific problem, different mutation strategies with different parameter settings may be better during different stages of the evolution than a single mutation strategy with unique parameter settings as in the conventional DE. Motivated by these observations, we form an ensemble of mutation strategies and parameter values for DE [23] in which a pool of mutation strategies, along with a pool of values corresponding to each associated parameter competes to produce successful offspring population. The candidate pool of mutation strategies and parameters should be restrictive to avoid the unfavorable influences of less effective mutation strategies and parameters. The mutation strategies or the parameters present in a pool should have diverse characteristics, so that they can exhibit distinct performance characteristics during different stages of the evolution, when dealing with a particular problem.

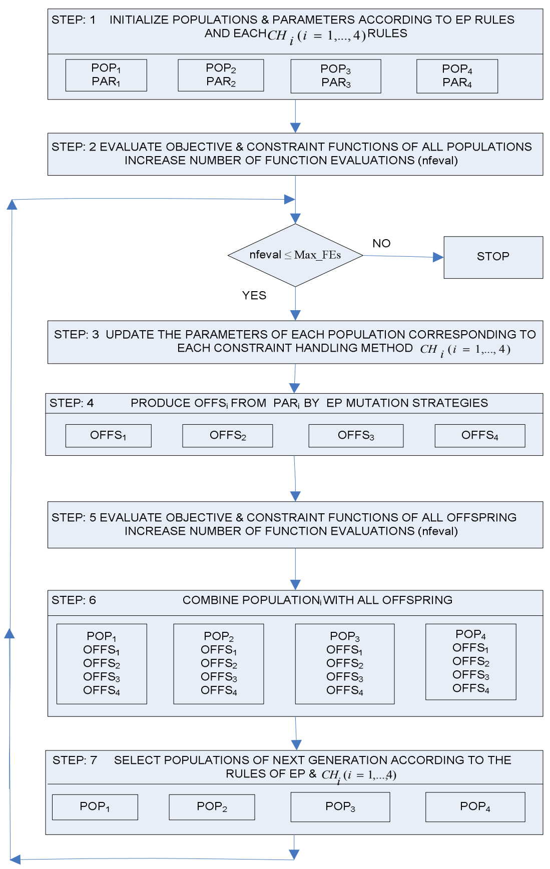
STEP: 1 INITIALIZE POPULATIONS & PARAMETERS ACCORDING TO EP RULES AND EACH CH_i $(i = 1,...,4)$ RULES
POP_1 PAR_1
POP_2 PAR_2
POP_3 PAR_3
POP_4 PAR_4
STEP: 2 EVALUATE OBJECTIVE & CONSTRAINT FUNCTIONS OF ALL POPULATIONS INCREASE NUMBER OF FUNCTION EVALUATIONS (nfeval)
nfeval $\leq$ Max_FEs
NO
STOP
YES
STEP: 3 UPDATE THE PARAMETERS OF EACH POPULATION CORRESPONDING TO EACH CONSTRAINT HANDLING METHOD CH_i $(i = 1,...,4)$
STEP: 4 PRODUCE $OFFS_i$ FROM PAR_i BY EP MUTATION STRATEGIES
$OFFS_1$ $OFFS_2$ $OFFS_3$ $OFFS_4$
STEP: 5 EVALUATE OBJECTIVE & CONSTRAINT FUNCTIONS OF ALL OFFSPRING INCREASE NUMBER OF FUNCTION EVALUATIONS (nfeval)
STEP: 6 COMBINE $POPULATION_i$ WITH ALL OFFSPRING
POP_1 $OFFS_1$ $OFFS_2$ $OFFS_3$ $OFFS_4$
POP_2 $OFFS_1$ $OFFS_2$ $OFFS_3$ $OFFS_4$
POP_3 $OFFS_1$ $OFFS_2$ $OFFS_3$ $OFFS_4$
POP_4 $OFFS_1$ $OFFS_2$ $OFFS_3$ $OFFS_4$
STEP: 7 SELECT POPULATIONS OF NEXT GENERATION ACCORDING TO THE RULES OF EP & CH_i $(i = 1,...,4)$
POP_1 POP_2 POP_3 POP_4

Fig. 1 Flowchart of ECHT (CH: Constraint Handling Method, POP: Population, PAR: Parameters, OFF: Offspring, Max_FEs: Maximum number of function evaluations)

The strategy pool contains mutation strategies "DE/rand/2/bin" and "DE/current-to-rand/1/bin". In order to balance the speed and efficiency while solving problems with different characteristics, the pool of *CR* values is taken in the range 0.1 to 0.9 in steps of 0.1, while the pool of *F* values is taken in the range 0.4 to 0.9 in steps of 0.1.

IV. Experimental Results

In this paper, the performance of ECHT-DE is evaluated on 10D and 30D versions of 18 test functions presented in [24]. For each problem, the ECHT-DE algorithm is run 25 times and the results are tabulated in Tables 1-7. In each table, *c* is the number of violated constraints at the median solution: the sequence of three numbers indicates the number of violations (including inequality and equalities) by more than 1.0, more than 0.01 and more than 0.0001 respectively. $\bar{v}$ is the mean value of the violations of all constraints at the median solution. The numbers in the parenthesis after the fitness value of the best, median, worst solution are the number of constraints which cannot satisfy feasible condition at the best, median and worst solutions respectively. Figures 2-5 present the convergence plots for some of the selected problems.

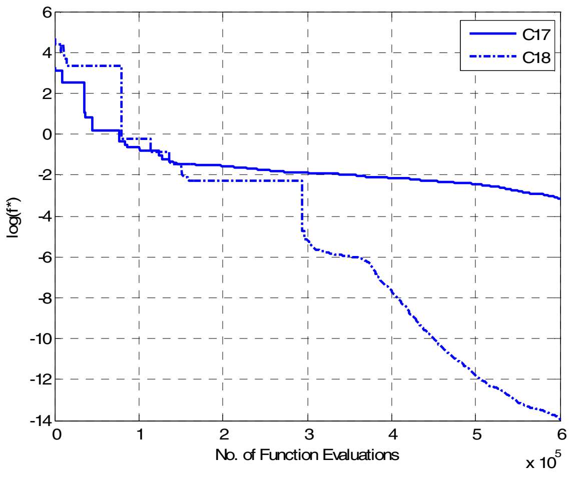

Fig 2: Convergence plot of 10D problems C17 and C18

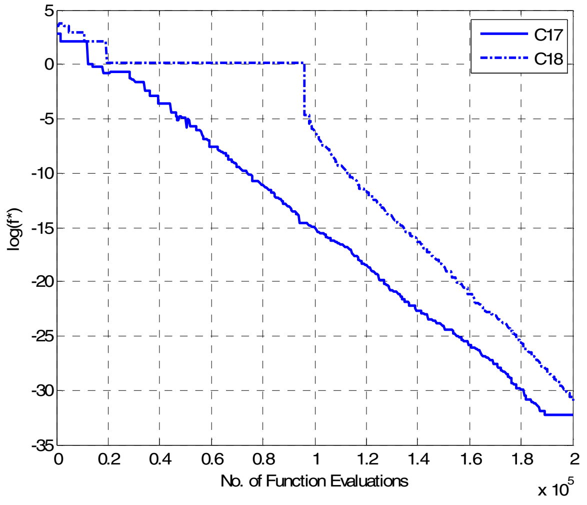

Fig 3: Convergence plot of 30D problems C17 and C18

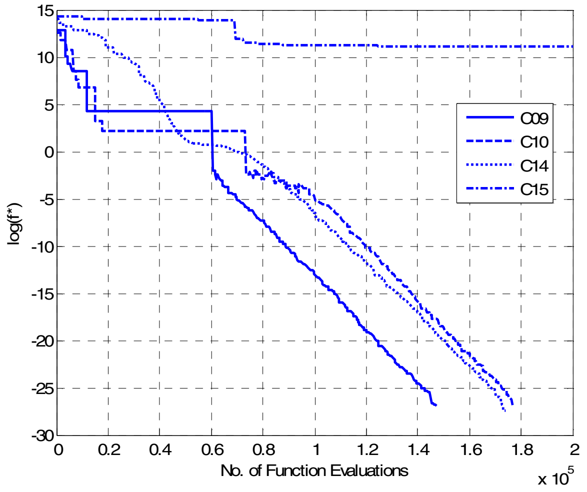

Fig 4: Convergence plot of 10D problems C09, C10, C14 and C15

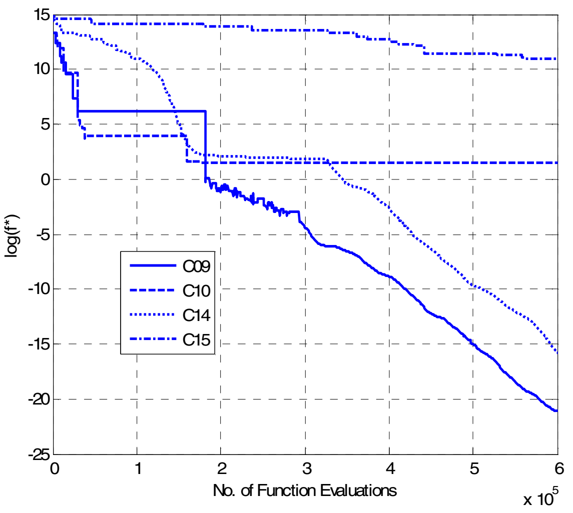

Fig 5: Convergence plot of 30D problems C09, C10, C14 and C15

PC Configuration

System: Windows XP; CPU: 3.00GHz; RAM: 1GB; Language: Matlab 7.1; Algorithm: ECHT-DE

Parameter Setting

The maximum number of function evaluations (Max_FEs) in each run set to 200000 and 600000 for 10D and 30D respectively. The population size corresponding to each constraint handling technique is set to 50. The parameters corresponding to the constraint handling are presented in Section II.

V. Conclusion

ECHT is a general method. It can be implemented with any search algorithm and with any of the existing constraint handling techniques present in the literature. The proposed ECHT-DE algorithm is evaluated on the CEC 2010 benchmark problem set.

References

[1] C. A. Coello Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering,* vol. 191, pp. 1245-1287, 2002.

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

[3] S. Koziel and Z. Michalewicz, "Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization," *Evolutionary Computation,* vol. 7, pp. 19-44, 1999.

[4] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation,* vol. 1, pp. 67-82, 1997.

[5] R. Mallipeddi and P. N. Suganthan, "Ensemble of Constraint Handling Techniques," *IEEE Transactions on Evolutionary Computation, available online*.

[6] V. L. Huang, A.K. Qin, and P. N. Suganthan., "Self-adaptive differential evolution algorithm for constrained real-parameter optimization," in *IEEE Congress on Evolutionary Computation* Vancouver, Canada: IEEE, 2006.

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

[8] B. Tessema and G. G. Yen, "A self adaptive penalty function based algorithm for constrained optimization," in *IEEE Congress on Evolutionary Computation*, 2006, pp. 246-253.

[9] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization,* vol. 11, pp. 341-359, 1997.

[10] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces," *Technical Report TR-95-012, ICSI, http://http.icsi.berkeley.edu/~storn/litera.html* 1995.

[11] J. Brest, V. Zumer, and M. S. Maucec, "Self-Adaptive Differential Evolution Algorithm in Constrained Real-Parameter Optimization," in *IEEE Congress on Evolutionary Computation* Vancouver, Canada: IEEE, 2006, pp. 215-222.

[12] M. Fatih Tasgetiren and P. N. Suganthan, "A Multi-Populated Differential Evolution Algorithm for Solving Constrained Optimization Problems," in *IEEE Congress on Evolutionary Computation*. Vancouver, Canada: IEEE, 2006, pp. 33-40.

[13] S. Kukkonen and J. Lampinen, "Constrained Real-Parameter Optimization with Generalized Differential Evolution," in *IEEE Congress on Evolutionary Computation* Vancouver, Canada, 2006, pp. 207 - 214.

[14] E. Mezura-Montes, J. Vel'azquez-Reyes, and C. A. C. Coello, "Modified Differential Evolution for Constrained Optimization," in *IEEE Congress on Evolutionary Computation* Vancouver, Canada: IEEE, 2006, pp. 25-32.

[15] T. Takahama and S. Sakai, "Constrained Optimization by the Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites," in *IEEE Congress on Evolutionary Computation* Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, 2006, pp. 1-8.

[16] K. Zielinski and R. Laur, "Constrained Single-Objective Optimization Using Differential Evolution," in *IEEE Congress on Evolutionary Computation* Vancouver, Canada: IEEE, 2006, pp. 223-230.

[17] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation,* vol. 13, pp. 398-417, April 2009.

[18] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering,* vol. 186, pp. 311-338, 2000.

[19] R. Farmani and J. A. Wright, "Self-Adaptive Fitness Formulation for Constrained Optimization," *IEEE Transactions on Evolutionary Computation,* vol. 7, pp. 445-455, 2003.

[20] T. P. Runarsson and X. Yao, "Search biases in constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 35, pp. 233-243, May 2005.

[21] S. Das, A. Abraham, and U. K. Chakraborthy, "Differential Evolution Using a Neighborhood-Based Mutation Operator," *IEEE Transactions on Evolutionary Computation,* vol. 13, pp. 526-553, JUN 2009.

[22] S. Das, A. Konar, and U. K. Chakraborty, "Two Improved Differential Evolution Schemes for Faster Global Search," in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, Washington DC, USA, 2005, pp. 991 - 998.

[23] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential Evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing (accepted),* 2010.

[24] R. Mallipeddi and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization," Nanyang Technological University, Singapore 2010.

Table 1: Function Values when FES =2 X 10^4, FES =1 X 10^5, FES = 2 X 10^5 for 10D Problems H01-H06.

FEs		C01	C02	C03	C04	C05	C06
2 X 10^4	Best	-0.6462 (0)	-2.2317 (0)	1.5356E+01 (1)	5.0409E+00 (4)	-473.8343 (2)	-579.0555 (2)
	Median	-0.5392 (0)	-2.2045 (0)	5.5553E+01 (1)	1.5681E+01 (4)	-313.4170 (2)	-444.5916 (2)
	worst	-0.4660 (0)	2.2740 (1)	4.2168E+05 (1)	3.1378E+01 (4)	-301.6915 (2)	-235.0772 (2)
	c	0, 0, 0	0, 0, 0	0, 1, 1	3, 3, 4	0, 2, 2	0, 2, 2
	$\bar{v}$	0	0	2.3270E-01	1.1745E+00	2.4613E-01	1.6791E-01
	Mean	-0.5478	-0.2241	4.3579E+04	1.2324E+01	-155.1486	-468.8162
	std	0.0432	1.9882	1.2722E+05	6.5408E+00	218.3213	128.3444
1 X 10^5	Best	-0.7473 (0)	-2.2777 (0)	2.1393E-17 (0)	-7.2351E-06 (0)	-483.6106 (0)	-578.6623 (0)
	Median	-0.7473 (0)	-2.2777 (0)	7.0651E-16 (0)	-5.4652E-06 (0)	-431.8835 (0)	-578.6623 (0)
	worst	-0.7406 (0)	-2.2612 (0)	1.1201E-03 (0)	-3.5136E-06 (0)	-451.2842 (2)	-367.4681 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-0.7470	-2.2743	1.0130E-04	-5.5656E-06	-407.5695	-561.5201
	std	0.0014	0.0067	2.1573E-04	8.6400E-02	78.6318	46.1943
2 X 10^5	Best	-0.7473 (0)	-2.2777 (0)	0 (0)	-1.0000E-05 (0)	-483.6106 (0)	-578.6624 (0)
	Median	-0.7473 (0)	-2.2777 (0)	0 (0)	-1.0000E-05 (0)	-434.9495 (0)	-578.6624 (0)
	worst	-0.7406 (0)	-2.2612 (0)	0 (0)	-1.0000E-05 (0)	-270.5665 (0)	-368.5512 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-0.7470	-2.2744	0	-1.0000E-05 (0)	-411.4532	-562.4688
	std	0.0014	0.0067	0	0	76.3137	45.1479

Table 2: Function Values when FES = 2 X 10^4 , FES =1 X 10^5, FES = 2 X 10^5 for 10D Problems H07-H12.

FEs		C07	C08	C09	C10	C11	C12
2 X 10^4	Best	5.1954 (0)	7.4917E+01 (0)	7.2521E+02 (1)	2.4310E+02 (0)	0.4277 (1)	-937.6291 (1)
	Median	7.5906 (0)	2.5182E+02 (0)	4.7072E+03 (1)	2.9557E+02 (1)	1.5060 (1)	-688.5143 (1)
	worst	69.3562 (0)	1.2677E+03 (0)	4.6283E+02 (1)	9.7417E+05 (1)	-17.3312 (1)	-791.5072 (1)
	c	0, 0, 0	0, 0, 0	0, 1, 1	0, 0, 1	1, 1, 1	1, 1, 1
	$\bar{v}$	0	0	1.0732E-02	2.9357E-03	2.4321E+06	1.7125E-03
	Mean	9.4031	3.6189E+02	1.2612E+07	9.3046E+10	-7.5013	-614.7990
	std	11.4072	2.7918E+02	5.1895E+07	5.0953E+11	8.3109	350.1145
1 X 10^5	Best	0 (0)	0 (0)	0 (0)	0 (0)	0.0015 (1)	-105.0916 (1)
	Median	0 (0)	7.0979 (0)	0 (0)	0 (0)	0.0065 (1)	-320.9763 (1)
	worst	3.9866 (0)	2.6115E+01 (0)	4.4082 (0)	4.1727E+01 (0)	1.1679 (1)	-731.7487 (1)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 1	1, 1, 1
	$\bar{v}$	0	0	0	0	7.7321E-03	1.0644E-08
	Mean	0.1329	6.1566E+00	1.4691E-01	1.7117E+00	0.0906	-395.5151
	std	0.7278	6.4527E+00	8.0482E-01	7.6554E+00	0.2757	283.6947
2 X 10^5	Best	0 (0)	0 (0)	0 (0)	0 (0)	-0.0015 (0)	-0.1992 (0)
	Median	0 (0)	7.0979 (0)	0 (0)	0 (0)	-0.0015 (0)	-0.1992 (0)
	worst	3.9866 (0)	2.6115E+01 (0)	4.4082 (0)	4.1727E+01 (0)	-0.0873 (1)	-554.3466 (1)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	0.1329	6.1566E+00	1.4691E-01	1.7117E+00	-0.0044	-171.8714
	std	0.7278	6.4527E+00	8.0482E-01	7.6554E+00	0.0157	221.0436

Table 3: Function Values when FES = 2 X 10^4 , FES =1 X 10^5, FES = 2 X 10^5 for Problems H13-H18 of 10D.

FEs		C13	C14	C15	C16	C17	C18
2 X 10^4	Best	-68.3739 (0)	3.2072E+07 (0)	1.9834E+12 (0)	1.6815E-01 (0)	2.1361E+01 (0)	2.9105E+02 (0)
	Median	-62.0621 (0)	6.2137E+10 (0)	4.8184E+13 (0)	8.3815E-02 (0)	1.8312E-02 (1)	2.7912E+02 (2)
	worst	-54.1040 (0)	1.1945E+12 (0)	2.3289E+14 (0)	7.9195E-01 (3)	1.2452E+00 (1)	1.8151E+03 (2)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 1, 1	0, 0, 2
	$\bar{v}$	0	0	0	0	7.9093E-13	1.8601E-05
	Mean	-61.7134	1.1355E+11	6.8517E+13	1.8991E-01	1.0609E+00	2.2596E+02
	std	4.1980	2.1633E+11	7.0929E+13	2.7261E-01	3.8728E+00	5.0437E+02
1 X 10^5	Best	-68.4294 (0)	0 (0)	1.4559E+07 (0)	0 (0)	0 (0)	0 (0)
	Median	-63.5175 (0)	0 (0)	6.7401E+10 (0)	3.0437E-02 (0)	0 (0)	0 (0)
	worst	-61.6487 (0)	1.7191E+07 (0)	1.8693E+14 (0)	1.6351E-01 (0)	1.0884E+00 (0)	0 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-65.1208	7.0242E+05	2.5153E+13	3.9327E-02	1.1152E-01	0
	std	2.3750	3.1937E+06	5.7338E+13	4.2815E-02	3.3152E-01	0
2 X 10^5	Best	-68.4294 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
	Median	-63.5175 (0)	0 (0)	1.2216E+10 (0)	3.0437E-02 (0)	0 (0)	0 (0)
	worst	-61.6487 (0)	1.7191E+07 (0)	1.8693E+14 (0)	1.6351E-01 (0)	1.0884E+00 (0)	0 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-65.1208	7.0242E+05	2.3392E+13	3.9327E-02	1.1152E-01	0
	std	2.3750	3.1937E+06	5.2988E+13	4.2815E-02	3.3152E-01	0

Table 4: Function Values when FES = 6 X 10^4 , FES =3 X 10^5, FES = 6 X 10^5 for Problems H01-H06 of 30D.

FEs		C01	C02	C03	C04	C05	C06
6 X 10^4	Best	-0.2795 (0)	-1.8578 (0)	3.7265E+01 (1)	1.2279E+01 (4)	-37.7609 (2)	55.4721 (2)
	Median	-0.2369 (0)	-0.8244 (0)	1.9392E+02 (1)	2.1296E+01 (4)	27.3541 (2)	480.4163 (2)
	worst	-0.2138 (0)	3.9798 (0)	5.8861E+03 (1)	7.8676E+00 (4)	-110.2899 (2)	448.8929 (2)
	c	0, 0, 0	0, 0, 0	1, 1, 1	3, 3, 4	0, 2, 2	0, 2, 2
	$\bar{v}$	0	0	8.8619E+00	1.5547E+00	9.9435E-02	1.5523E-01
	Mean	-0.2409	-0.1371	1.6843E+05	1.8173E+01	-41.3592	-21.4720
	std	0.0173	1.6248	5.7506E+05	5.6033E+00	136.4506	171.3966
3 X 10^5	Best	-0.7704 (0)	-2.1720 (0)	1.3123E-03 (0)	-1.3257E-07 (0)	-169.7941 (0)	-127.8888 (0)
	Median	-0.5671 (0)	-1.9307 (0)	4.0030E+01 (1)	5.5331E-04 (0)	-131.4861 (0)	-197.2119 (1)
	worst	-0.3802 (0)	-1.1807 (0)	1.7168E+02 (1)	2.5983E-01 (0)	-208.5683 (2)	-143.2744 (2)
	c	0, 0, 0	0, 0, 0	0, 0, 1	0, 0,0	0, 0, 1	0, 0, 1
	$\bar{v}$	0	0	6.7251E-04	0	1.0373E-05	2.8861E-05
	Mean	-0.5651	-1.8651	1.2322E+02	9.9671E-03	-102.6552	-134.2772
	std	0.1248	0.2479	5.9815E+01	4.7315E-02	167.7271	99.5973
6 X 10^5	Best	-0.8217 (0)	-2.2251 (0)	3.2433E-21 (0)	-3.3015E-06 (0)	-213.6844 (0)	-295.7192 (0)
	Median	-0.8012 (0)	-2.0662 (0)	1.0983E+02 (0)	-2.9456E-06 (0)	-163.0036 (0)	-147.3193 (0)
	worst	-0.7557 (0)	-1.3511 (0)	1.8496E+02 (0)	4.6205E-01 (0)	477.1882 (0)	263.5327 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-0.7994	-1.9943	9.8920E+01	-1.0257E-06	-106.4228	-137.6152
	std	0.0179	0.2099	6.2594E+01	9.0135E-02	167.1481	98.8995

Table 5: Function Values when FES = 6 X 10^4 , FES =3 X 10^5, FES = 6 X 10^5 for Problems H07-H12 of 30D.

FEs		C07	C08	C09	C10	C11	C12
6 X 10^4	Best	25.2831 (0)	7.3454E+04 (0)	4.2807E+08 (1)	3.5962E+02 (0)	-0.5851 (1)	-571.4264 (1)
	Median	29.5979 (0)	2.0065E+05 (0)	7.7843E+07 (1)	6.9351E+03 (1)	-4.4613 (1)	-197.3924 (1)
	worst	303.6628 (0)	6.4820E+05 (0)	5.2726E+05 (1)	3.5513E+06 (1)	-4.1193 (1)	325.1862 (1)
	c	0, 0, 0	0, 0, 0	0, 1, 1	0, 0, 1	1, 1, 1,	1, 1, 1
	$\bar{v}$	0	0	1.3621E-02	2.2573E-03	1.6196E+07	1.5921E+10
	Mean	68.3499	2.2193E+05	6.8682E+10	3.7591E+07	-0.9614	-320.8379
	std	64.6705	1.2871E+05	3.6156E+11	1.1164E+08	1.7598	392.0726
3 X 10^5	Best	0 (0)	1.8156E-03 (0)	0 (0)	0 (0)	0.0453 (1)	-0.1947 (0)
	Median	2.2359 (0)	7.1975E+00 (0)	1.3521E-04 (0)	3.1311E+01 (0)	-0.0333 (1)	-588.1351 (1)
	worst	8.1438 (0)	5.8566E+02 (0)	6.6031E+02 (0)	4.7500E+02 (0)	0.1509 (1)	-665.4098 (1)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	1, 1, 1	1, 1, 1,
	$\bar{v}$	0	0	0	0	2.3761E+01	45.1278
	Mean	2.6609	4.1881E+01	4.4776E+01	6.3983E+01	0.0173	-390.0311
	std	2.2692	1.0968E+02	1.3836E+02	8.6686E+01	0.0472	365.6671
6 X 10^5	Best	0 (0)	0 (0)	0 (0)	0 (0)	-0.0004 (0)	-0.1993 (0)
	Median	0 (0)	0 (0)	0 (0)	3.1309E+01 (0)	-0.0002(0)	-0.1993 (0)
	worst	3.9866 (0)	5.8567E+02 (0)	6.5710E+02 (0)	4.7510E+02 (0)	0.0204 (1)	-748.163 (1)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	0.1329	3.3585E+01	4.2441E+01	5.3381E+01	0.0026	-25.1292
	std	0.7279	1.1072E+02	1.3762E+02	8.8308E+01	0.0060	136.5593

Table 6: Function Values when FES = 6 X 10^4 , FES =3 X 10^5, FES = 6 X 10^5 for Problems H13-H18 of 30D.

FEs		C13	C14	C15	C16	C17	C18
6 X 10^4	Best	-58.3015 (0)	3.1128E+11 (0)	8.0169E+13 (0)	7.6153E-03 (0)	1.2385E+00 (0)	1.1595E+01 (0)
	Median	-47.5518 (0)	2.1964E+12 (0)	1.7281E+14 (0)	2.8157E-03 (0)	1.9268E+00 (1)	4.6513E+03 (0)
	worst	-39.4831 (0)	5.4375E+12 (0)	3.8886E+14 (0)	3.4573E-03 (2)	3.4996E+00 (1)	5.4956E+03 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 1, 1	0, 0, 0
	$\bar{v}$	0	0	0	0	1.3909E-12	0
	Mean	-48.3975	2.7068E+12	1.7831E+14	6.0537E-02	2.2217E+00	1.4380E+03
	std	5.1796	1.4873E+12	6.7922E+13	2.2532E-01	3.2897E+00	2.7829E+03
3 X 10^5	Best	-68.4294 (0)	5.2972E+00 (0)	1.5797E+11 (0)	0 (0)	0 (0)	0 (0)
	Median	-64.6187 (0)	1.6880E+01 (0)	3.2873E+12 (0)	0 (0)	2.1051E-01 (0)	0 (0)
	worst	-60.9387 (0)	3.7101E+06 (0)	6.1436E+13 (0)	0 (0)	1.8992E+00 (0)	0 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-64.5831	1.2370E+05	1.3272E+13	0	2.8223E-01	0
	std	1.6690	6.7736E+05	1.7872E+13	0	3.781E-01	0
6 X 10^5	Best	-68.4294 (0)	0 (0)	1.9922E+09 (0)	0 (0)	0 (0)	0 (0)
	Median	-64.6187 (0)	0 (0)	8.5527E+10 (0)	0 (0)	1.9273E-01 (0)	0 (0)
	worst	-60.9387 (0)	3.7101E+06 (0)	2.3252E+12 (0)	0 (0)	1.8986E+00 (0)	0 (0)
	c	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
	$\bar{v}$	0	0	0	0	0	0
	Mean	-64.5831	1.2368E+05	1.9409E+11	0	2.7496E-01	0
	std	1.6690	6.7736E+05	4.3524E+11	0	3.7832E-01	0

Table 7: Time Complexity

n	*T*1 (sec)	*T*2 (sec)	(*T*2-*T*1)/*T*1
10D	32.1567	213.5635	5.6413
30D	63.3511	429.5212	5.7800