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

Differential Evolution with a Constraint Consensus Mutation for Solving Optimization Problems

Noha M. Hamza, Daryl L. Essam, and Ruhul A. Sarker

***Abstract*— in the literature, a considerable number of mutation operators have been proposed, which are the key search operators in differential evolution algorithm for solving optimization problems. Although those operators were developed in the context of unconstrained optimization, they were widely used in constrained optimization. However, those operators did not contain any mechanism that would reduce the constraint violation in the search process. Therefore, in this paper, a new mutation operator based on the constraint consensus method is proposed, which can help infeasible points reach the feasible region quickly. The algorithm is tested on the CEC2010 constrained benchmark problems. The experimental results show that the proposed algorithm is able to obtain better solutions in comparison with the state-of-the-art algorithms.**

***Index Terms*—Constrained optimization, constraint consensus, differential evolution**

I. Introduction

THERE are many real-world decision processes that require solving constrained optimization problems (COPs). In COPs, an acceptable solution is the one that is feasible and either optimal or near optimal. An optimal solution in a COP may exist either on the surface (/boundary) of the feasible region or well-inside the feasible space. The constraints in COPs are usually handled using one of two approaches: (i) converting the constrained problem into an equivalent unconstrained problem and then solving it using an appropriate search algorithm, and (ii) dealing the constraints separately where the search approach optimizes the objective function, while satisfying the constraint set. One serious drawback of the first approach is that it may produce a high quality objective value without satisfying one or more constraints. For this reason, the second approach is popular in practice however the approaches are much more complex. Note that, in this paper, all analysis and discussions are in the context of the second approach mentioned above.

Over the last few decades, population based stochastic algorithms, such as genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], differential evolution (DE) [3], evolutionary strategies (ES) [4], and evolutionary programming [5], have shown their ability in solving constrained optimization problems. In this research, we would like to deal with one of such algorithms that is DE. From the literature, DE has shown a tremendous success in solving continuous optimization problems. However, the algorithm converges prematurely when dealing with those problems that contain multiple local optima by losing diversity [6-9].

There have been numerous studies that were proposed to improve DE's performance. Such improvements lie in proposing new mutation strategies, ensemble of DE operators, and ensemble of constraint handling techniques. The search operators are designed mainly to improve the fitness value, while maintaining the diversity of the population. Such operators are well-suited with the unconstrained optimization problems. To solve the constrained optimization problems, researchers implemented the same search operators with an additional mechanism, known as constraint handling technique. However, in almost all cases, the constraint handling technique contributes to the evolution process, indirectly, via the ranking and selection methods. In other words, the search operators in DE do not play any direct role in reducing the constraint violations. It is expected that the introduction of a new mutation operator, that can improve the fitness value and reduce the constraint violation, will significantly improve the performance of the algorithm.

The constraint consensus (CC) method, which uses different types of projection algorithms, are used for solving feasibility problems with nonlinear and non-convex constraints. This method assists the infeasible solution vector to quickly move towards the feasible region by making a consensus among the currently violated constraints to determine the direction and distance that are required to achieve feasibility [28]. Although the CC methods can improve the feasibility of individuals in the population, in our earlier work, we applied them as an additional step with EAs (see for details in [10-12]). However, no research reported in the literature on a DE mutation that combines the usual movement operator with the concept of CC for improving the performance of the algorithm. In this paper, we propose a new mutation operator that combines the search technique with a method of reducing constraint violation. The performance of the proposed algorithm has been tested by solving a set of constrained benchmark problems (from CEC2010 competition) [13]. The results show that the proposed algorithm is able to obtain better solutions in comparison with a state-of-the-art algorithm.

This paper is organized as follows. After the introduction, section II describes the DE algorithm. Section III discusses an overview of constraint consensus methods. Section IV

The authors are with School of Engineering and Information Technology, University of New South Wales at Canberra, 2600, Australia, emails: noha.hamza@student.adfa.edu.au, {d.essam and r.sarker}@adfa.edu.au.

demonstrates the design of the proposed algorithm, while the experimental results and the analysis are presented in section V. Finally, the conclusions are given in section VI.

II. Differential Evolution

Differential Evolution is a popular evolutionary algorithm (EA) for solving optimization problems [3, 14]. DE has two main search operators: mutation and crossover operators.

In the traditional DE mutation operator, a mutant vector, $\vec{V}$, is generated by multiplying the difference of two random vectors by a scaling factor F and the result is then added to another third random vector. For an individual z, $\vec{V}_z$ is calculated as follows:

$$\vec{V}_z = \vec{x}_{r_1} + F.(\vec{x}_{r_2} - \vec{x}_{r_3}), \quad (1)$$

where, r_1, r_2, r_3 are random integer numbers $\in$ {1,2, ..., NP}\{z}, $r_1 \neq r_2 \neq r_3$ x is a decision vector, NP is the population size, F is a positive control parameter.

Similar to other EAs, there are many strategies for mutation in DE, such as:

- DE/best/1 ([15]):

  $$\vec{V}_{z,t} = \vec{x}_{best,t} + F \times (\vec{x}_{r_2,t} - \vec{x}_{r_3,t}) \quad (2)$$

  where $\vec{x}_{best,t}$ is the best individual vector at generation t.

- DE/rand-to-best/1 ([16]):

  $$\vec{V}_{z,t} = \vec{x}_{r_1,t} + F \times (\vec{x}_{best,t} - \vec{x}_{r_2,t}) + F \times (\vec{x}_{r_3,t} - \vec{x}_{r_4,t}) \quad (3)$$

- DE/rand/2/dir ([17]):

  $$\vec{V}_{z,t} = \vec{v}_{1,t} + \frac{F}{2} \times \left((\vec{v}_{1,t} - \vec{v}_{2,t}) + (\vec{v}_{3,t} - \vec{v}_{4,t})\right) \quad (4)$$

  where $f(\vec{v}_{1,t}) \leq f(\vec{v}_{2,t})$ and $f(\vec{v}_{3,t}) \leq f(\vec{v}_{4,t})$.

- DE/current-to-rand/1 ([18]):

  $$\vec{V}_{z,t} = \vec{x}_{r_1,t} + F \times (\vec{x}_{r_2,t} - \vec{x}_{z,t}) + F \times (\vec{x}_{r_3,t} - \vec{x}_{r_4,t}) \quad (5)$$

- DE/current-to-best/1([19]):

  $$\vec{V}_{z,t} = \vec{x}_{r_1,t} + F \times (\vec{x}_{best,t} - \vec{x}_{z,t}) + F \times (\vec{x}_{r_3,t} - \vec{x}_{r_4,t}) \quad (6)$$

- DE/current-to-best ([20]):

  $$\vec{V}_{z,t} = \vec{x}_{z,t} + F \times \left((\vec{x}^{\psi}_{best,t} - \vec{x}_{z,t}) + (\vec{x}_{r_1,t} - \vec{x}_{r_2,t})\right) \quad (7)$$

  where $\vec{x}^{\psi}_{best,t}$ is randomly chosen as one of the top $100 \times \psi\%$ individuals in the current generation.

In DE, the well known crossover operator is the *Binomial crossover*, which used in this paper. In it, for each of the j^{th}variables, whenever a randomly picked number ($rand \in [0,1]$) is less than or equal to Cr value, or it is a random $jrand \in [1 - D]$ variable, the j^{th} variable is inherited from the mutant vector, such as:

$$u_{ij,t} = \begin{cases} v_{i,j,t}, & if\ (rand \leq Cr\ or\ j = jrand) \\ x_{i,j,t}, & otherwise \end{cases} \quad (8)$$

Over the last two decades, DE was successfully applied to solve different COPs. A few such examples are discussed here. Takahama and Sakai [21], introduced an ε constrained DE with an archive and a gradient-based mutation (εDEag) for solving the CEC2010 competition on constrained real-parameter optimization [13]. The εDEag utilized an archive to maintain the diversity of individuals and adopted a new way of selecting the ε-level control parameter in εDEg. Elsayed *et al.* [8], proposed a multi-operator based DE (SAMO-DE) algorithm for solving two different sets of COPs. The feasibility rule was used to handle the constraints. SAMO-DE was found to be better than other state-of-the-art algorithms. Elsayed et al. [22] proposed a DE algorithm that used multiple search operators, and two constraint handling techniques (the feasibility rule and the epsilon constraint methods) were adopted to handle constraints. The results showed that the proposed algorithm was superior to the-state-of-the-art algorithms. Brest et al. [23], employed the epsilon constraint method to handle the constraints in their proposed differential evolution algorithm. The algorithm has shown competitive performance over a set of test problems.

III. Constraint Consensus Methods

Constraint Consensus (CC) methods [24-26] use different types of projection algorithms, as a heuristic, for solving the feasibility problems with nonlinear and non-convex constraints**.** The key idea is to help a currently infeasible solution to quickly move towards the feasible region by making a consensus among the currently violated constraints [10-12, 25]. More precisely, CC starts from an initial infeasible solution and then constructs a *feasibility vector* for each violated constraint at the existing solution. The feasibility vector approximates the move from the current infeasible point to the closest feasible solution for each violated constraint.

The calculation of the feasibility vectors is exact for linear constraints, however a linear approximation is generated (known as *linear feasibility vectors*) for nonlinear constraints. The linear feasibility vector moves the infeasible solution in a parallel direction to the gradient of the violated constraint and the step size of the movement is calculated by using a first order Taylor series expansion of the constraint function around the current solution [12, 25, 27], such as:

$$(x_{k+1} - x_k) = \frac{-g_i(x_k)}{\|\nabla g_i(x_k)^T\|^2} \nabla g_i(x_k)^T \quad (9)$$

where

- $g_i(x_k) : R^n \rightarrow R$ is the i^{th} constraint, $\nabla g_i(x_k)^T$is the transposition of its gradient, and $\| \nabla g_i(x_k)^T \|$ is its length.
- k: Number of steps.
- x_k: Current infeasible point at k^{th} step.
- x_{k+1}: Estimated feasible point for each violated constraint at $k+1^{th}$ step.
- To accommodate the non-differential and discontinuous functions, an approximate gradient $(\nabla g_i(x_k))$ is calculated numerically for our algorithm as follows

($\frac{g_i(x+\Delta)-g_i(x)}{\Delta}$), where Δ represents a small change in x (here it is equal to 0.0001).

The feasibility vectors for all violated constraints are joined into a *consensus vector* which is then used to update the current point. The CC steps are repeated until satisfying predefined stopping conditions [25, 28], such as: (1) the length of every feasibility vector is less than a predefined feasibility distance threshold (α) (e.g.10^{-6}), (2) the length of the consensus vector is less than a movement tolerance (β), in which the consensus vector becomes too short, or (3) more μ generations have been completed.

Over the last few decades, many variants of CC method have been proposed such as traditional CC, Direction Based maximum (DBmax) algorithm and the Feasibility Distance far (FDfar) algorithm. They differ by the method of constructing the consensus vector. In this paper, we use the traditional CC method, in which all eligible feasibility vectors are treated equally and the movement is created by averaging the non-zero components of the feasibility vectors.

Recently, CC was employed with differential evolution as an additional mechanism to deal with the infeasible solutions. Hamza *et al.* [10] proposed different variants of DE with multiple CC methods. The proposed algorithms were tested and analyzed by solving the CEC2006 test problems. The performance of the algorithms was further improved by incorporating a local search approach. The algorithm showed competitive performance while using significantly lower computational time. The algorithm was also applied to solve a complex real-world problem [11].

IV. DE WITH CONSTRAINT CONSENSUS BASED MUTATION

In this section, the proposed algorithm is described, followed by the description of constraint handling method used in this research. We introduced the algorithm as *DEwCCM* (Differential Algorithm with Constraint Consensus Mutation).

A. DEwCCM

As indicated earlier, the CC methods were used as an additional step within EAs to deal with infeasibility [10] [11]. In this section, our goal is to propose a new mutation operator that combines the concept of CC method with a traditional mutation operator. It is expected that a such mutation operator will not only improve the quality of fitness value but also help the infeasible solutions to reach the feasible region quickly. That means, this mutation will be beneficial for infeasible solutions only, which is introduced below. The main steps of the proposed algorithm are shown in Algorithm 1.

Firstly, the initial population is generated within the boundary based on the following equation:

$$x_{zj} = L_j + rand \times (U_j - L_j) \quad (10)$$

where L_j, U_j are the lower and upper bound for decision variable x_{ij} and $rand$ is a random number, $rand \in [0,1]$. Then, for some of the infeasible solutions, say P individuals, new offspring are generated based on:

ALGORITHM 1: DEwCCM

STEP 1: In generation $t = 0$, generate an initial random population of size PS. The variables in each individual (z) must be within the range using (11).

STEP 2 : for each (z) in ***NP***, randomly generate its F_z, Cr_z and φ_z

STEP 3: At each generation (t)**: If** there are infeasible solutions **then**: Randomly **Select** P individuals of the infeasible solutions, , and generated new offspring using (12)-(14)

STEP 4: For each individual in the remaining ($PS - P$) individuals, a new offspring is generated using (15).

STEP 5: if $\vec{U}_i$ is better than $\vec{x}_i$, based on the fitness function and/or constraint violation, accept it for the next generation. Sort the entire population based on the objective value and/or constraint violation.

STEP 6: Apply an expansion mechanism on the successful offspring, using (16), and update the number of fitness evaluations. then, Sort the entire population based on the objective value and/or constraint violation.

STEP 7: Stop if the termination criterion is met**; else** set $t = t + 1$, and go to **STEP 3.**

$$u_{z,j} = \begin{cases} x_{\varphi_z,j} + F_z.\left(x_{best,j} - x_{CC,j}\right), if\ (rand \le Cr_z\ or\ j = j_{rand}) \\ x_{z,j} \qquad otherwise \end{cases} \quad (11)$$

where $\varphi \in [0 - \frac{NP}{2}]$ which aim at maintaining a balance between two DE mutation strategies (DE/rand and DE/best) [29].

$$x_{CC} = \frac{x_{k+1,C}}{nc} \quad (12)$$

nc is the total number of the violated constraints for each infeasible solution, ($x_{k+1,C}$) the new point considering each violated constraint (C) and

$$x_{k+1,C} = \frac{-g_i(x_k)}{\|\nabla g_i(x_k)\|^2}\ \nabla g_i(x_k)^T + x_k \quad (13)$$

This mutation is not applied to all infeasible individuals, but to some of them. This is to minimize the additional computational time required for computing them as well as maintaining the diversity in the population. So, for the remaining *NP-P* individuals (either feasible or infeasible), a new offspring is generated according to:

$$u_{z,j,t} = \begin{cases} x_{\varphi_z,j,t} + (\emptyset.F_z).\left(x_{r_{z1},j,t} - x_{r_{z2},j,t}\right) + \\ ((1-\emptyset).F_z).\left(x_{best,j,t} - x_{r_{z3},j,t}\right)\ if\ (rand \le Cr_z\ or\ j = j_{rand}) \\ x_{z,j} \qquad otherwise \end{cases} \quad (14)$$

where φ is a random integer number within a range [a, b], $a = 1$, $b = NP/2$ [30], and $\emptyset = 0.75$. In (10), a bit more emphasis on the diversity appears from using $0.75F_z$ for the first difference vector. For each new generated individual, if it is better than its parent, it survives for the next generation, and the entire population is sorted based on the fitness function and/or constraint violation.

C. Expansion Mechanism

For a better convergence pattern, at each generation, we employ a stretching mechanism on the successful offspring generated by (15). This is done by using following mutation:

$$u_{z,j,t} = \begin{cases} x_{\varphi_z,j,t} + ((1-\emptyset).F_z).\left(x_{r_{z1},j,t} - x_{r_{z2},j,t}\right) + \\ (\emptyset.F_z).\left(x_{best,j,t} - x_{z,j,t}\right) \quad if\ (rand \le Cr_z\ or\ j = j_{rand}) \\ x_{z,j} \qquad otherwise \end{cases} \tag{15}$$

The key idea of this expansion mechanism is that using the same $\vec{x}_{r_{z1},t}$ and $\vec{x}_{r_{z2},t}$ along with F_z and Cr_z used in (15), may be a good seed to generate a new offspring using information gained from the offspring generated by (15) , $\vec{x}_{z,t}$, and the new best individual in the population. Note that (16) employs a more exploitation mechanism in comparison with (15) as shown by using $\emptyset$, instead of (1-$\emptyset$) in (15), in the second difference vector.

C. Constraint Handling Method

In this paper, the superiority of feasible solutions technique is used to select individuals [31], in which: **1**) between two feasible solutions, the fittest one (according to fitness function) is better, i.e. $f(\vec{x}_1) < f(\vec{x}_2)$, **2**) a feasible solution is always better than an infeasible one, **3**) between two infeasible solutions, the one having the smaller sum of its constraint violation is preferred, i.e. $vio(\vec{x}_1) < vio(\vec{x}_2)$. Equality constraints are transformed into inequalities of the following form, where ε is a small tolerance, i.e. 0.0001, and E is the number of equality constraints:

$$|h_e(\vec{x})| - \varepsilon \le 0, for\ e = 1, \dots, E \tag{16}$$

V. Experimental Results

In this section, we present and analyze the experimental results of the proposed algorithm by solving a set of benchmark problems introduced in the CEC2010 constrained problems [13]. These problems contain different properties, such as the objective function and/or the constraints are either linear or non-linear. The constraint signs may be equality or inequality. The objective function may be multi-modal or unimodal. Moreover, the optimal solution may lie on the boundary and the feasible space may be very tiny.

The algorithm has been coded using Matlab, and has been run on a PC with a 2.8 GHz Core (TM) i7 processor, 8G RAM and Windows 7. For the parameters settings, *NP* is set to 100 individuals; *P* = 10% of the infeasible solutions, this value was based on those discussed in [10]. F_z was a random number $\in [0.4 - 0.9]$. In regard to Cr, for each individual in the population at generation t, if $rand_{z,t} \le 0.75$ then $Cr_{z,t} = 0.95$; else $Cr_{z,t} = 0.4$ [29], $\varphi \in [1 - \frac{NP}{2}]$ the total Fitness Function evaluations (FEs) were 20000D, 25 runs were used, In regard to CC parameters, $\alpha = 10^{-6}$; β=0.0001 and μ=1.

A. The Effect of the CC based Mutation

Here, all the test problems are solved by DE with and without the CC based mutation, known as DEwCCM and DE, respectively. In this paper, DE used is that shown in (15). Appendix A shows the detailed results of both algorithms. Note that the "*" shown in Appendix A means that the value is of an infeasible point.

Table I. Comparison Among DEwCCM, DE and εDEag

D	Comparison	Results	Better	Equal	Worse	Dec.
10D	DEwCCM – to – DE	Best	4	14	0	≈
		Average	14	4	0	+
	DEwCCM – to – εDEag	Best	4	13	1	≈
		Average	7	8	3	≈
30D	DEwCCM – to – DE	Best	14	3	1	+
		Average	16	1	1	+
	DEwCCM – to – εDEag	Best	16	1	1	+
		Average	16	0	2	+

To start with, in regard to the best values obtained in 10D, DEwCCM obtained better results for four problems, and both were similar for 14 problems. Based on the average results, DEwCCM was superior to DE for 13 test problems instances, and was able to obtain better results in five problems.

For the 30D test problems, DEwCCM was found superior to DE for 14 and 16 test problems, based on the best and average results obtained, respectively, while it was inferior to DE for only one test problem, and similar to DE for three and one test problem(s), considering the best and average results, respectively.

In addition to the quality of solutions obtained, we have compared the computational time required to obtain the optimal solution with an error of 0.0001, i.e. the stopping criteria is $[f(\vec{x}) - f(\vec{x^*}) \le 0.0001]$, where $f(\vec{x^*})$ was the best known solutions. We found that DEwCCM requires 2.0% lower execution time in comparison with DE to reach the optimal solutions for the 30D test problems. This saving in time, although it is small, is a great advantage, especially when using gradient calculations, which normally consume time.

Furthermore, a statistical significance testing is performed. A non-parametric test, the Wilcoxon Signed Rank Test [8, 32] is performed. The Wilcoxon decision results regarding the best and average fitness values are presented in Table I. As a null hypothesis, it is assumed that there is no significant difference between the best and/or mean values of two samples, while the alternative hypothesis is that there is a significant difference in the best and/or average fitness values of the two samples, using the 5% significance level. Based on the test results/rankings, we have assigned one of three signs (+, -, and ≈) for the comparison of any two algorithms (shown in the last column), where the "+" sign means that the first algorithm is significantly better than the second, the "-" sign means that the first algorithm is significantly worse, and the "≈" sign means that there is no significant difference between the two algorithms. From Table I, it is clear that DEwCCM was significantly better than DE.

To this end, as an example, a convergence plot is depicted in figure 1, which shows that using the proposed algorithm converges faster than DE.

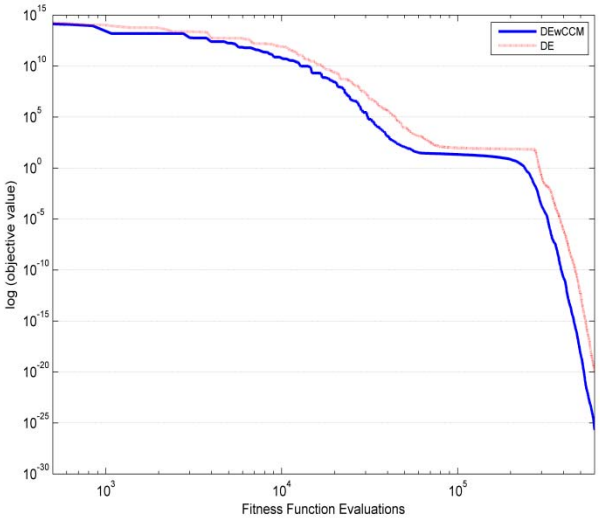

Fig.1. Convergence plots for both DEwCCM and DE for C14 (30D)

B. Comparison to State-of-the-art Algorithms

Here the computational results of DEwCCM are compared with a state-of-the-art algorithm, εDEag [21], which won the CEC2010 constrained optimization competition. The detailed results are shown in Appendix A.

Considering the quality of the solutions obtained, a summary is reported in Table I. From this table, for the 10D test problems, DEwCCM was found superior to εDEag for 4 and 7 test problems, based on the best and average results, respectively. However, εDEag was better for one problem based on the best results obtained, and 3 test problems considering the mean results reached.

In regard to the 30D test instances, DEwCCM was clearly better than to the majority of the test problems.

Finally, based on the Wilcoxon test, DEwCCM was significantly better than εDEag.

VI. CONCLUSION AND FUTURE WORK

Differential Evolution is known as a good performing algorithm in solving optimization problems. In the literature, many DE mutation strategies were proposed. However, none of them contained any mechanism that can guide the infeasible individuals to move towards the feasible region. Although Constraint Consensus methods can do this task, it was not combined with the search approach of any mutation strategy.

In this research, a differential evolution algorithm with a new mutation strategy, based on a concept of constraint consensus method, was introduced. To add to this, a mechanism, to stretch successful points based on their parents, those points used to get them generated, was employed.

The algorithm was analysed by solving a set of well-known test problems where it showed good performance, in terms of the computational results and time, in comparison to the same algorithm without the new mutation. To add to this, the algorithm was superior to the CECE2010's competition winner.

For future work, we wish to analyze all components of the algorithm and apply it on real-world applications.

REFERENCES

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*: Addison-Wesley Publishing Corporation, 1989.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimisation," in *the IEEE International Conference on Neural Network*, 1995, pp. 1942–1948.

[3] R. Storn and K. Price, "Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces," International Computer Science Institute Technichal Report TR-95-012, 1995.

[4] I. Rechenberg, *Evolutions strategie: Optimierung Technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart: Fromman-Holzboog, 1973.

[5] L. Fogel, J. Owens, and M. Walsh, *Artificial Intelligence Through Simulated Evolution*. New York: John Wiley & Sons, 1966.

[6] J. Lampinen and I. Zelinka, "On stagnation of the differential evolution algorithm," in *6th Int. Mendel Conference on Soft Computing*, Brno, Czech Republic, 2000, pp. 76–83.

[7] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *IEEE Congress on Evolutionary Computation*, 2004, pp. 980-1987.

[8] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Multi-operator based evolutionary algorithms for solving constrained optimization Problems," *Computers and Operations Research,* vol. 38, pp. 1877-1896, 2011.

[9] N. Hamza, R. Sarker, and D. Essam, "Hybridizing constraint consensus methods with evolutionary algorithms for constrained optimization," Engineering & Information Technology, UNSW Canberra, Canberra, 2012.

[10] N. Hamza, R. Sarker, and D. Essam, "Differential evolution with multi-constraint consensus methods for constrained optimization," *Journal of Global Optimization,* pp. 1-29, 2012/10/01 2012.

[11] N. M. Hamza, R. A. Sarker, and D. L. Essam, "Differential evolution with a mix of Constraint Consensus methods for solving a real-world Optimization Problem," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*, 2012, pp. 1-7.

[12] N. M. Hamza, S. M. Elsayed, D. L. Essam, and R. A. Sarker, "Differential evolution combined with constraint consensus for constrained optimization," in *IEEE Congress on Evolutionary Computation* 2011, pp. 865-872.

[13] R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2010 competition and special session on single objective constrained real-parameter optimization," Technical Report, Nangyang Technological University, Singapore2010.

[14] 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.

[15] R. Storn, "On the usage of differential evolution for function optimization," in *Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS)*, 1996, pp. 519–523.

[16] 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, 2009.

[17] E. Mezura-Montes, J. V. Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *the 8th annual conference on Genetic and evolutionary computation*, Seattle, Washington, USA, 2006, pp. 485-492.

[18] A. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Australian Conference on Artificial Intelligence*, 2004, pp. 861–872.

[19] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: a practical approach to global optimization*. Berlin: Springer, 2005.

[20] Z. Jingqiao and A. C. Sanderson, "JADE: Adaptive Differential Evolution With Optional External Archive," *IEEE Transactions on Evolutionary Computation,* vol. 13, pp. 945-958, 2009.

[21] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1-9.

[22] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Integrated strategies differential evolution algorithm with a local search for constrained optimization," in *IEEE Congress on Evolutionary Computation*, 2011, pp. 2618-2625.

[23] J. Brest, B. Bos□kovic□, and V. Z□umer, "An improved self-adaptive differential evolution algorithm in single objective constrained real-parameter optimization," in *IEEE Congress on Evolutionary Computation* 2010, pp. 1-8.

[24] Y. Censor and S. Zenios, *Parallel Optimization: theory, algorithms, and applications*. New York: Oxford University Press, 1997.

[25] W. Ibrahim and J. W. Chinneck, "Improving solver success in reaching feasibility for sets of nonlinear constraints," *Comput. Oper. Res.,* vol. 35, pp. 1394-1411, 2008.

[26] J. W. Chinneck, "Feasibility and Infeasibility in optimization," in *Algorithms and computational methods*. vol. 118, ed: Springer, International Series in Operation Research and Management Sciences, 2008.

[27] L. Smith, "Improved Placement of Local Solver Launch Points for Large-scale Global Optimization," Doctor of Philosophy, Electrical and Computer Engineering, Carleton University, Ottawa, Ontario, Canada, 2011.

[28] J. W. Chinneck, "The Constraint Consensus Method for Finding Approximately Feasible Points in Nonlinear Programs," *INFORMS J. on Computing,* vol. 16, pp. 255-265, 2004.

[29] S. Elsayed and R. Sarker, "Differential Evolution with Automatic Population Injection Scheme," in *IEEE Symposium Series on Computational Intelligence*, Singapore, accepted, 2013.

[30] R. Sarker, S. Elsayed, and T. Ray, "Differential Evolution with Dynamic Parameters Selection for Optimization Problems," *Evolutionary Computation, IEEE Transactions on,* vol. PP, pp. 1-1, 2013.

[31] K. Deb, "An Efficient Constraint Handling Method for Genetic Algorithms," *Computer Methods in Applied Mechanics and Engineering,* vol. 186, pp. 311-338, 2000.

[32] G. W. Corder and D. I. Foreman, *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. Hoboken, NJ: John Wiley, 2009.

Appendix A

FUNCTION VALUES OBTAINED BY DE, DEWCCM AND εDEAG FOR THE CEC2010 TEST PROBLEMS

Prob.	Alg.	10D			30D		
		Best	**Mean**	***St. d***	**Best**	**Mean**	***St. d***
C01	DE	**-7.473104E-01**	-7.462299E-01	2.526793E-03	**-8.218843E-01**	-8.179309E-01	4.061502E-03
	DEwCCM	**-7.473104E-01**	**-7.467701E-01**	**1.869859E-03**	-8.218840E-01	-8.148481E-01	5.533345E-03
	εDEag	**-7.473104E-01**	**-7.467701E-01**	**1.869859E-03**	-8.218255E-01	**-8.208687E-01**	**7.103893E-04**
C02	DE	**-2.277711E+00**	-2.068396E+00	2.670116E-01	-2.277661E+00	-2.260020E+00	3.283637E-02
	DEwCCM	**-2.277711E+00**	-2.261483E+00	**1.9696915E-02**	**-2.280970E+00**	**-2.272595E+00**	**7.532632E-03**
	εDEag	-2.277702E+00	**-2.269502E+00**	2.3897790E-02	-2.169248E+00	-2.151424E+00	1.197582E-02
C03	DE	**0.000000E+00**	2.770176E+00	1.031351E+01	2.0172674E-21	2.231244E+00	7.722545E+00
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**1.198614E-24**	**1.339530E-23**	**2.465790E-23**
	εDEag	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	2.867347E+01	2.883785E+01	8.047159E-01
C04	DE	**-1.000000E-05**	**-1.000000E-05**	**0.0000000E+00**	-3.316939E-06	4.024828E-02	2.012574E-01
	DEwCCM	**-1.000000E-05**	**-1.000000E-05**	**0.0000000E+00**	**-3.330669E-06**	**-3.313301E-06**	**1.355266E-08**
	εDEag	-9.992345E-06	-9.918452E-06	1.5467300E-07	4.698111E-03	8.162973E-03	3.067785E-03
C05	DE	1.337876E+02*	2.561201E+02*	1.069543E+02*	**-4.836106E+02**	-4.836076E+02	4.638421E-03
	DEwCCM	**-4.836106E+02**	**-4.836106E+02**	**0.0000000E+00**	**-4.836106E+02**	**-4.836106E+02**	**5.301109E-06**
	εDEag	**-4.836106E+02**	**-4.836106E+02**	3.89035E-13	-4.531307E+02	-4.495460E+02	2.899105E+00
C06	DE	**-5.786624E+02**	**-5.786624E+02**	3.462902E-06	-5.306368E+02	-5.304791E+02	6.862122E-01
	DEwCCM	**-5.786624E+02**	**-5.786624E+02**	**1.8945738E-07**	**-5.306378E+02**	**-5.306342E+02**	**2.492397E-03**
	εDEag	-5.786581E+02	-5.786528E+02	3.6271690E-03	-5.285750E+02	-5.279068E+02	4.748378E-01
C07	DE	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	1.296277E-25	1.707400E-13	5.737914E-13
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**3.893502E-26**	**5.987239E-24**	**1.359775E-23**
	εDEag	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	1.147112E-15	2.603632E-15	1.233430E-15
C08	DE	**0.000000E+00**	5.523709E+00	5.006128E+00	9.284612E-20	1.318268E+02	5.930584E+02
	DEwCCM	**0.000000E+00**	**4.562059E+00**	**5.052990E+00**	**8.206303E-26**	**1.617573E-20**	**4.713931E-20**
	εDEag	**0.000000E+00**	6.727528E+00	5.560648E+00	2.518693E-14	7.831464E-14	4.855177E-14
C09	DE	**0.000000E+00**	2.231778E+02	1.115889E+03	6.588174E-17	1.578058E+01	5.465172E+01
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**9.024904E-26**	**3.396387E-24**	**5.737653E-24**
	εDEag	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	2.770665E-16	1.072140E+01	2.821923E+01
C10	DE	**0.000000E+00**	1.009288E-26	3.904391E-26	1.780825E-21	5.954761E-03	2.977380E-02
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**6.721095E-26**	**6.008938E-24**	**2.303748E-23**
	εDEag	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	3.252002E+01	3.326175E+01	4.545577E-01
C11	DE	**-1.52271E-03**	**-1.52271E-03**	2.933590E-13	-3.923434E-04	-3.923318E-04	1.123230E-08
	DEwCCM	**-1.52271E-03**	**-1.52271E-03**	**2.6859647E-14**	**-3.923439E-04**	**-3.923428E-04**	**7.757013E-10**
	εDEag	**-1.52271E-03**	**-1.52271E-03**	6.3410350E-11	-3.268462E-04	-2.863882E-04	2.707605E-05
C12	DE	-1.992458E-01	-1.992458E-01	**1.415862E-08**	**-1.992635E-01**	-1.992634E-01	1.613874E-08
	DEwCCM	**-3.054888E+02**	-6.105263E+01	1.2421703E+02	**-1.992635E-01**	**-1.992634E-01**	**1.731444E-08**
	εDEag	**-5.700899E+02**	**-3.367349E+02**	1.7821660E+02	-1.991453E-01	3.562330E+02*	2.889253E+02
C13	DE	**-6.842937E+01**	-6.818071E+01	5.910213E-01	-6.842903E+01	-5.444431E+01	3.534682E+00
	DEwCCM	**-6.842937E+01**	-6.836890E+01	2.082451E-01	**-6.842917E+01**	-5.936727E+01	4.615222E+00
	εDEag	**-6.842937E+01**	**-6.842936E+01**	**1.0259600E-06**	-6.642473E+01	**-6.535310E+01**	**5.733005E-01**
C14	DE	**0.000000E+00**	5.659025E-12	2.829513E-11	1.090058E-20	3.189299E-01	1.103846E+00
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**1.991953E-26**	**5.953745E-20**	**2.926103E-19**
	εDEag	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	5.015863E-14	3.089407E-13	5.608409E-13
C15	DE	**0.000000E+00**	3.599233E-01	1.245728E+00	1.366993E-19	3.372009E-01	1.167084E+00
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**6.771582E-27**	**1.571328E-21**	**7.502248E-21**
	εDEag	**0.000000E+00**	1.798980E-01	8.8131560E-01	2.160345E+01	2.160376E+01	1.104834E-04
C16	DE	**0.000000E+00**	1.409414E-02	7.047069E-02	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**
	DEwCCM	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**	**0.000000E+00**	**0.000000E+00**	**0.0000000E+00**
	εDEag	**0.000000E+00**	3.702054E-01	3.7104790E-01	**0.000000E+00**	2.168404E-21	1.062297E-20
C17	DE	1.047706E-31	7.879282E-23	2.650742E-22	2.628795E-07	1.486175E-01	1.659793E-01
	DEwCCM	**0.00000E+00**	**1.573716E-27**	**3.0757252E-27**	**6.757731E-15**	**1.932266E-02**	**1.252069E-02**
	εDEag	1.463180E-17	1.249561E-01	1.9371970E-01	2.165719E-01	6.326487E+00	4.986691E+00
C18	DE	6.626432E-29	2.025350E-22	7.193852E-22	3.974207E-09	1.404993E+01	5.828927E+01
	DEwCCM	**0.000000E+00**	**2.435457E-26**	**4.8599586E-26**	**1.679787E-09**	**1.491371E-01**	**5.998427E-01**
	εDEag	3.731440E-20	9.678765E-19	1.8112340E-18	1.226054E+00	8.754569E+01	1.664753E+02