Hacettepe Journal of Mathematics and Statistics
Volume 48 (3) (2019), 931 – 950
Research Article

Hybrid Constrained Evolutionary Algorithm for Numerical Optimization Problems

Wali Khan Mashwani*†, Alam Zaib‡, Özgür Yeniay§, Habib Shah¶, Naseer Mansoor Tairan‖ and Muhammad Sulaiman**

Abstract

Constrained optimization are naturally arises in many real-life applications, and is therefore gaining a constantly growing attention of the researchers.Evolutionary algorithms are not directly applied on constrained optimization problems. However, different constraint-handling techniques are incorporated in their framework to adopt it for dealing with constrained environments. This paper suggests an hybrid constrained evolutionary algorithm (HCEA) that employs two penalty functions simultaneously. The suggested HCEA has two versions namely HCEA-static and HCEA-adaptive. The performance of the HCEA-static and HCEA-adaptive algorithms are examined upon the constrained benchmark functions that are recently designed for the special session of the 2006 IEEE Conference of Evolutionary Computation (IEEE-CEC'06). The experimental results of the suggested algorithms are much promising as compared to one of the recent constrained version of the JADE. The converging behaviour of the both suggested algorithms on each benchmark function is encouraging and promising in most cases.

Keywords: Constrained Functions, Evolutionary Computation(EC), Evolutionary Algorithm(EA) and Hybrid EAs.

Received : 14.04.2017 *Accepted* : 04.09.2018 *Doi* : 10.15672/HJMS.2018.625

*Kohat University of Science & Technology, KPK, Pakistan, Email: mashwanigr8@gmail.com
†Corresponding Author.
‡Kohat University of Science & Technology, KPK, Pakistan, Email: alamzaibmaths@gmail.com
§Hacettepe University, Ankara, Turkey, Email: yeniay@hacettepe.edu.tr
¶King Khalid University Abha, Saudi Arabia, Email: habibshah.uthm@gmail.com
‖King Khalid University Abha, Saudi Arabia, Email: nmtairan@kku.edu.sa
**Abdul Wali Khan University, Mardan, Pakistan, Email: msulaiman@awkum.edu.pk

1. Introduction

In the last two decades and so, numerical optimization has become an emerging area of research because of their wide application in different discipline of sciences and engineering [12, 11]. The optimization problems have many types including multi-quadratic programming, bilinear and biconvex, generalized geometric programming, general constrained nonlinear optimization, bilevel optimization, complementarity, semidefinite programming, mixed-integer nonlinear optimization, combinatorial optimization and optimal control problems [19]. Generally these problems can be categorized into constrained and unconstrained optimization problems. In this paper, we are interested in solving optimization problems with continuous variables. In this paper, we are interested in solving the constrained optimization problems that can generally formulated as follow [14]:

$$
\begin{array}{llll}
 & \text{Minimize} & f(\mathbf{x}), \quad x \in S & \\
(1.1) & \text{subject to} & g_i(\mathbf{x}) \leq 0, & i = 1, 2, 3, ..., p; \\
 & & h_j(\mathbf{x}) = 0, & j = 1, 2, 3, ..., q,
\end{array}
$$

where S denotes the whole search space, p is the number of inequality constraints and q is the number of equality constraints. If a problem at hand has some equality constraints, they can transform into inequalities as follow:

$$h_j - \epsilon \leq 0 \tag{1.2}$$

where ϵ is the tolerance rate. The inequality constraints that satisfy $g_i(\mathbf{x}) = 0$ are called active constraints, it is also to be noted that equality constraints are always active.

In the last two decades and so, very many optimization methods developed in the form of deterministic and stochastic natures. Deterministic approaches involve no randomness to perform their search process. Interval optimization [13], branch-and-bound [27, 55] and algebraic techniques [54] are commonly used deterministic methods.On the other hand, stochastic nature based algorithms evolve their set of solutions with randomness. Simulated annealing (SA) [23, 22], Monte Carlo sampling [15], stochastic tunneling [32],parallel tempering [34], Genetic Algorithm (GA) [17], Evolutionary Strategies (ES) [43], Evolutionary Programming (EP) [20, 21, 7], Particle Swarm Optimization (PSO) [25, 67], Ant Colony Optimization (ACO) [60], differential evolution (DE) [52], Krill herd algorithm based on cuckoo search [2, 62],Elephant Herding Optimization (EHO)[8, 58], Moth search algorithm [56], Monarch Butterfly Optimization (MBO)[57], Earthworm Optimization Algorithm (EWA)[59], Plant Prorogation Algorithm (PPA)[49, 50, 51, 44] and hybrid EAs [28, 47, 18] are well-known stochastic methods. Evolutionary computation is the collective name used for population base evolutionary algorithms. These algorithm are mainly inspired by biological process of evolution, such as natural selection and genetic inheritance [16].

In general, evolutionary algorithms employ penalty functions and other constraint handling techniques to maintain a reasonable ratio among feasible and infeasible solutions for dealing with constrained optimization problems [33, 64, 63, 65, 9, 36]. Penalty functions are very common and popular approaches while adopting unconstrained EAs to constrained one. Most of the Researchers prefer adaptive penalty methods in order to develop constrained EAs to handle complex COPs. Recently, hybrid evolutionary algorithms have got much attention due to their high potentialities and capabilities to solve problems with high complexity, noisy environment, imprecision, uncertainty and vagueness [35, 1]. In this paper, We have combined two popular EAs including the PSO and differential evolution (DE) and developed hybrid constrained evolutionary algorithm to handle test problems designed for the special session of the 2006 IEEE-congress on evolutionary computation (IEEE-CEC'06) [31]. The suggested hybrid constrained EA utilizes two penalty functions simultaneously.The suggested algorithms have tackled most of used test problems an affective manner.

The rest of the paper is organized as follows. Section 2 presents the framework of the proposed hybrid constrained evolutionary algorithm. Section 3 demonstrates experimental results. Section 4 concludes this research paper with future plan and directions.

2. Hybrid Constrained Evolutionary Algorithm

Evolutionary algorithms (EAs) have gained popularity and much attention of the researchers in academia and industrial applications. They have tackled various optimization and search problems comprising various complexities like noisy environments, imprecision, uncertainty and vagueness in their mathematical structures. EAs operate on set of solutions called population and ultimately they provide a set of optimal solutions in single simulation run. They do not require any derivative information regarding the objective function as well as constraint functions of the problems at hand. They use various intrinsic evolutionary operators like reproduction, mutation, recombination and selection to perform their search process.

In the last two decades and so, hybrid evolutionary algorithms(EAs) have got much attention for dealing with optimization problems with high complexity, noisy environment, imprecision, uncertainty and vagueness [1, 6, 42, 37, 38, 39, 40, 4, 5, 61, 3, 41, 26, 45, 46].In this paper, We have developed an efficient constrained hybrid constrained EAs by incorporating some existing penalty functions with static and self-adaptive procedures. The suggested HCEAs employs Differential evolution (DE) [52] and Particle Swarm Optimization(PSO)[25] as constituent search operators to perform their search process. The suggested algorithm also employs the penalty functions to improve the qaulity of feasible solutions. The penalty functions as given in equation 2.1 adopted with static and penalty function as explained in 2.2-2.8 are employed adaptive procedure in the framework of the Algorithm (1).

2.1. Penalty Functions. In the last decades, various Penalty Functions developed and found in the existing literature of the evolutionary computing [35, 33, 64, 63, 65, 9, 36]. These penalty functions are used to penalize the candidate solutions that violate the constraint functions of the problem 1.1. The first penalty function which was proposed in [24] that defines different levels of violation keeping in view the magnitude of violation of the constraint functions. Penalty Function works as follow:

- Define l levels of violation for each constraint,
- Generate penalty coefficient R_{ij}, where $i = 1, \ldots, l$ and $j = 1, \ldots, m$ for each level of violation and each constraint. The bigger coefficients are given to the bigger violation levels.
- Generate a random population using both feasible and unfeasible individuals.
- Evaluate these individuals by using following formula

$$eval(\overline{x}) = f(\overline{x}) + \textstyle\sum_{j=1}^{m} R_{ij} max[0, g_i(\overline{x})]^2 \qquad (2.1)$$

where $R_{i,j}$ denoted the penalty coefficient with respect to j^{th} constraint and i^{th} violation level and m is the number of constraints. Homaifar et al. [24] transformed equality constraints to inequality constraints according to $|h_j(\overline{X})| - \epsilon \leq 0$, where ϵ is a small positive number. Adjustment of large number of parameters settings such as $m(2l + 1)$ is one of the main issue with static penalty functions proposed in [24]. For example, if $m = 5$ and $l = 4$ levels of violation then one has to adjust t 45 parameters at same time. Although, complexity of this strategy is very high but still quite useful strategy

Algorithm 1 Framework of the Hybrid Constrained Evolutionary Algorithm

N = Population Size.
n = Dimension of the Serach Space.
$X = \{x^1, \dots, x^N\}^T \leftarrow$ **Initialize-Population**(N, n)
$F = \{f(x^1), \dots, f(x^N)\} \leftarrow$ **Evaluate**$(\{x^1, \dots, x^N\}^T)$
$G = \{g(x^1), \dots, g(x^N)\} \leftarrow$ **Evaluate**$(\{x^1, \dots, x^N\}^T)$
Apply the penalty % (i.e, For Static Penalty Function referred to algorithm (2), or for Adaptive Penalty Function referred to algorithm (3)).
for $i \leftarrow 1 : N$ **do**
  if $rand < 0.15$ **then**
    Select x^i, x^{r_1}, x^{r_2} at random from X such that $x^i \neq x^{r_1} \neq x^{r_2}$
    $u^i = x^i + F(x^{r_1} - x^{r_2})$
    Apply the penalty % (i.e, For Static Penalty Function referred to algorithm (2), or for Adaptive Penalty Function referred to algorithm (3)).
    for $j \leftarrow 1 : n$ **do**
      $y^i_j = \begin{cases} u^i_j, & If rand \leq 0.5 \\ x^i_j, & \text{otherwise} \end{cases}$
    end for
    $F_C(i) = \{f(y^1), \dots, f(y^N)\} \leftarrow$ **Evaluate**$(\{y^1, \dots, y^N\}^T)$
    $G_c(i) = \{g(y^1), \dots, g(y^N)\} \leftarrow$ **Evaluate**$(\{y^1, \dots, y^N\}^T)$
  else
    $\nu^i = \omega\nu^i + a_1 r^1(pbest^i - x^i) + a_2 r^2(nbest^i - x^i)$
    $y^i = x^i + \nu^i$
    Apply the penalty % (i.e, For Static Penalty Function referred to algorithm (2), or for Adaptive Penalty Function referred to algorithm (3)).
    if $G(y^i) = 0$ **then**
      if $f(y^i) < f(x^i)$ **then**
        $x^i = y^i$
      else
        $x^i = x^i$
      end if
    end if
    if $G(y^i) \neq 0$ **then**
      $v(y^i) < v(x^i)$
      $x^i = y^i$
    else
      $v(x^i) < v(y^i)$
      $y^i = x^i$
    end if
  end if
end for

while developing the constrained EAs. The algorithm (2) explains the procedure of the suggested static penalty function.

2.2. Adaptive Penalty Functions. Static penalty functions are adjusted based on error-trial procedure. They are characterised by repeated, varied and continued attempts until success not achieved [61]. In general, these strategies are problem-dependent and the users have facing difficulties to settle down the parameters involved at different levels of constraints violation. This tedious and difficult task can overcome with the strategy

Algorithm 2 Procedure for Penalty Function in HCEA with Static Strategy

Input=N, v
Output=f(x)
N:Number of constraints;
v:Constrained function value;
l:Number of violation level;
$f(x)$:Penalized Constrained function value;
$v_m = mean(v)$
for $i \leftarrow 1 : N$ **do**
  $v_{max} = max(v)$;
  $v_{l_1} =$ if$((v > 0) \quad \& \quad (v \leq 0.1 * v_m))$
  $R_1 = 0.1 * v_m$
  $v(v_{l_1}) = R_1 * v(v_{l_1})$
  $v_{l_2} =$ if$(v > 0.1 * v_m) \quad \& \quad (v \leq 0.2 * v_m)$
  $R_2 = 0.2 * v_m$
  $v(v_{l_2}) = R_2 * v(v_{l_2})$
  $v_{l_3} =$ if$(v > 0.2 * v_m) \quad \& \quad (v \leq v_m)$
  $R_3 = v_m$
  $v(v_{l_3}) = R_3 * v(v_{l_3})$
  $v_{l_4} =$ if$(v > v_m) \quad \& \quad (v \leq v_{max})$
  $R_4 = v_{max}$
  $v(v_{l_4}) = R_4 * v(v_{l_4})$
end for
Return($f(x)$);

of adaptive penalty functions procedures [22, 9, 36]. The adaptive approaches utilize previous information in order to adjust the coefficient of the penalty functions. The suggested algorithm employ adaptively the following Penalty functions.

$$F(x) = d(x) + p(x) \tag{2.2}$$

Where $p(x)$ is the penalty value. In equation (2.2), the distance value $d(x)$ is computed as follow:

$$d(x) = \begin{cases} \nu(x), & \text{if } r_f = 0 \\ \sqrt{f(x)''^2 + \nu(x)^2}, & \text{otherwise} \end{cases} \tag{2.3}$$

$$r_f = \frac{\text{Number of feasible solution}}{\text{population size}} \tag{2.4}$$

Where $\nu(x)$ is the overall constrain violation.

$$f(x)'' = \frac{f(x) - f_{min}}{f_{max} - f_{min}} \tag{2.5}$$

Where f_{max} and f_{min} are maximum and minimum value of objective function. The penalty value is defined as follow

$$p(x) = (1 - r_f)M(x) + r_f N(x) \tag{2.6}$$

Where $M(x)$ and $N(x)$ are given by

$$M(x) = \begin{cases} 0, & \text{if } r_f = 0 \\ \nu(x), & \text{otherwise} \end{cases} \tag{2.7}$$

$$N(x) = \begin{cases} 0, & \text{if } x \text{ is feasible solution.} \\ f(x)'', & \text{if } x \text{ is an infeasible solution.} \end{cases} \tag{2.8}$$

Algorithmic procedure for the adaptive penalty function used in suggested algorithm is given in the algorithm 3.

Algorithm 3 Procedure for Adaptive Penalty Function in the Framework HCEA

Input=F,G,N
Output=f_2
F:Fitness function value;
G:Constrained function value;
N:Population size;
x_f:Number of feasible solution;
f_2:Penalized Constrained function value;
$G = (G > 0) * G$
$g_{max} = max(G)$
$w = \text{find}(g_{max} \neq 0)$
if $w = \phi$ **then**
    $v = 0$
else
    $\nu(x) = \frac{\sum_{i=1}^{m} \omega_i (G_i(x))}{\sum_{i=1}^{m} \omega_i}$
end if
if $f_{max} = f_{min}$ **then**
    $f(x)'' = 1$
else
    $f(x)'' = \frac{f(x) - f_{min}}{f_{max} - f_{min}}$
end if
$x_f = \text{find}(v = 0)$
$r_f = \frac{x_f}{N}$
if $r_f = 0$ **then**
    $X = 0$
    $d = v$
else
    $X = v$
    $d = \sqrt{(f(x)''^2 + v^2)}$
end if
$Y = f(x)''$
$Y(x_f) = x_f$
$p = (1 - r_f) * X + (r_f * Y)$
$f_2 = d + p$

3. Discussion on Experimental Results

All experiments were carried out in the following platform and parameter settings:

- Operating system: Windows XP Professional;
- Programming language of the algorithms: Matlab;
- CPU: Core $i3$ Quad 1.8 GHz;
- RAM: 4 GB DDR2 500 GB;
- Execution: 25 times each algorithm with different random seeds.

Table 1. Classification and Properties of the used Benchmark Functions [28].

Problem	n	Type of Function	ρ	LI	NI	LE	NE	a	Known Optimal
g01	13	Quadratic	0.0111%	9	0	0	0	6	−15.000000000000
g02	20	Nonlinear	99.9971%	0	2	0	0	1	−0.803619000000
g03	10	Polynomial	0.0000%	0	0	0	1	1	−1.000000000000
g04	5	Quadratic	52.1230%	0	6	0	0	2	−30665.539000000001
g05	4	Cubic	0.0000%	2	0	0	3	3	5126.498100000000
g06	2	Cubic	0.0066%	0	2	0	0	2	−6961.813880000000
g07	10	Quadratic	0.0003%	3	5	0	0	6	24.306000000000
g08	2	Nonlinear	0.8560%	0	2	0	0	0	−0.095825000000
g09	7	Polynomial	0.5121%	0	4	0	0	2	680.630057300000
g10	8	Linear	0.0010%	3	3	0	0	6	7049.248000000000
g11	2	Quadratic	0.0000%	0	0	0	1	1	0.750000000000
g12	3	Quadratic	4.7713%	0	1	0	0	0	−1.000000000000
g13	5	Nonlinear	0.0000%	0	0	0	3	3	0.053949800000
g14	10	Nonlinear	0.0000%	0	0	3	0	3	−47.764888459500
g15	3	Quadratic	0.0000%	0	0	1	1	2	961.715022289900
g16	5	Nonlinear	0.0204%	4	34	0	0	4	−1.905155258600
g17	6	Nonlinear	0.0000%	0	0	0	4	4	8853.533874806501
g18	9	Quadratic	0.0000%	0	13	0	0	6	−0.866025403800
g19	15	Nonlinear	33.4761%	0	5	0	0	0	32.655592950200
g20	24	Linear	0.0000%	0	6	2	12	16	0.2049794002
g21	7	Linear	0.0000%	0	1	0	5	6	193.724510070000
g22	22	Linear	0.0000%	0	1	8	11	19	236.4309755040
g23	9	Linear	0.0000%	0	2	3	1	6	−400.055100000000
g24	2	Linear	79 : 6556%	0	2	0	0	2	−5.508013271600

Experiments were conducted with following parameter settings.

- The sized of population, $N = 60$;
- The tolerance value ϵ for the equality constraints is set to 0.0001
- The Parameters of PSO were settled as $\omega = 1$;
- Differential Evolution (DE) has used with $F = 0.7$ and $CR = 1.0$;
- The maximum number of function evaluations $300,000$;
- The tolerance value $\Delta = 0.0001$ for the problems consisting equality constraints.
- The maximum of generations is set to 2500.

Due largely to the nature of evolutionary algorithms (EAs), their behaviors and performances are mainly experimentally analyzed over different kinds of test suites of optimization and search problems. Several continuous test functions are already proposed for EC community over the last few years. These test functions played crucial role in developing and in studying the algorithmic behavior of particular evolutionary algorithm.

In this paper, we have used 24 benchmark functions that were designed for the special session of the IEEE Congress of Evolutionary Computation (CEC'2006). This CEC'2006 test suit consist of 24 benchmark functions comprising different characteristics like linear, nonlinear, polynomial, quadratic and cubic of objective functions with high dimensionality and wide range of linear inequalities (LI), nonlinear inequalities (NI), linear equalities (LE), and nonlinear equalities (NE) and number of other constraints [28]. The characteristics of the used Benchmark Functions are summarized in the Table 1.

Table 1 provides the features of the used 24 benchmark $CEC'06$ problems, where n is the number of decision variables, $\rho = \frac{|F|}{|S|}$ is the estimated ratio between the feasible region and the search space, LI denotes is the number of linear inequality constraints, NI stand fir the number of nonlinear inequality constraints, LE is the number of linear

Table 2. Comparative Analysis of the a) HEA-static and b) HEA-adaptive versus c) CJADE-D [53].

Problem	Method	Minimum	Mean	Median
g01	a	**-15.000000000000**	**-15.000000000000**	**-15.000000000000**
	b	−14.999843000000	−14.999843000000	−14.999843000000
	c	−14.999339	−12.969465	−14.664120
g02	a	**-0.803619000000**	**-0.803619000000**	**-0.803619000000**
	b	−0.783840000000	−0.783840000000	−0.782222000000
	c	−0.802628	−0.801371	−0.801388
g03	a	−1.000500000000	−1.000500000000	−1.000500000000
	b	−0.562186000000	−0.562186000000	−0.562186000000
	c	**-99675.835134**	**-99689.240788**	**-99690.148408**
g04	a	−30665.538671999999	−30665.538672000162	−30665.538671999999
	b	−30664.917622000001	−30664.917621999986	−30664.917622000001
	c	**-32196.152588**	**-32192.275678**	**-32192.360668**
g05	a	5126.496714000000	5126.496713999783	5126.496714000000
	b	5126.659844000000	5126.659843999903	5126.659844000000
	c	**1362.819559**	**1081.791600**	**1074.561136**
g06	a	−6961.813876000000	−6961.813875999722	−6961.813876000000
	b	−6961.781377000000	−6961.781377000246	−6961.781377000000
	c	−7962.000000	−7944.403151	−7943.279110
g07	a	24.306209000000	**24.306209000001**	24.306209000000
	b	25.857273000000	25.857273000001	25.857273000000
	c	24.306209	24.321855	24.306209

equality constraints and NE is the number of nonlinear equality constraints and a is the number of active constraints [28].

Tables 2-3-4 provide the experimental results of the suggested algorithms, namely, HCEA-static and HCEA-adaptive in comparison with recently developed constrained version of JADE aestivated as CJADE [53]. Tables 2-3-4 clearly show that the minimum functions values are much closer to known global optimal values of the test problems. Furthermore, Table 2-3-4 clearly indicated that the suggested HEA-static has found promising results in terms of better convergence toward the know optimal values while solving the g01, g02, g13, g18 and g21 problems. Similarly HCEA-static has tackled the problems g01, g02, g07, g13, g18, g19 and g21 with better mean functions values. The same is the case with median values for g01, g02, g13, g18 and g21 problems. It is also important to noted here that HCEA-static has performed better than our HCEA-adaptive and existing sate-of-the-art CJADE. The better performance could be attributed to better choice of efficient penalty functions and the combined use of DE [52] and PSO [25] in the framework of the proposed algorithm.

3.1. Graphical Result of HCEA-static. Figures 1-2-3 demonstrate the graphical results displayed by HCEA-static penalty functions for CEC'2006 benchmark functions. The figure 1 represents the evolution in minimum function values of the benchmark functions,g1,g2,g3,g4, g6 and g7 provided by HCEA-static algorithm in 25 independent runs of simulations with different random seeds.Simliary, figure 2 shows the convergence behaviour of the HCEA-static over g8,g9,g10,g11,g12 and g14 benchmark functions. Figure 3 display the convergence speed of the suggested HEA-static over g15,g16,g18,g19,g23 and g24 of the IEEE-CEC06 benchmark functions.

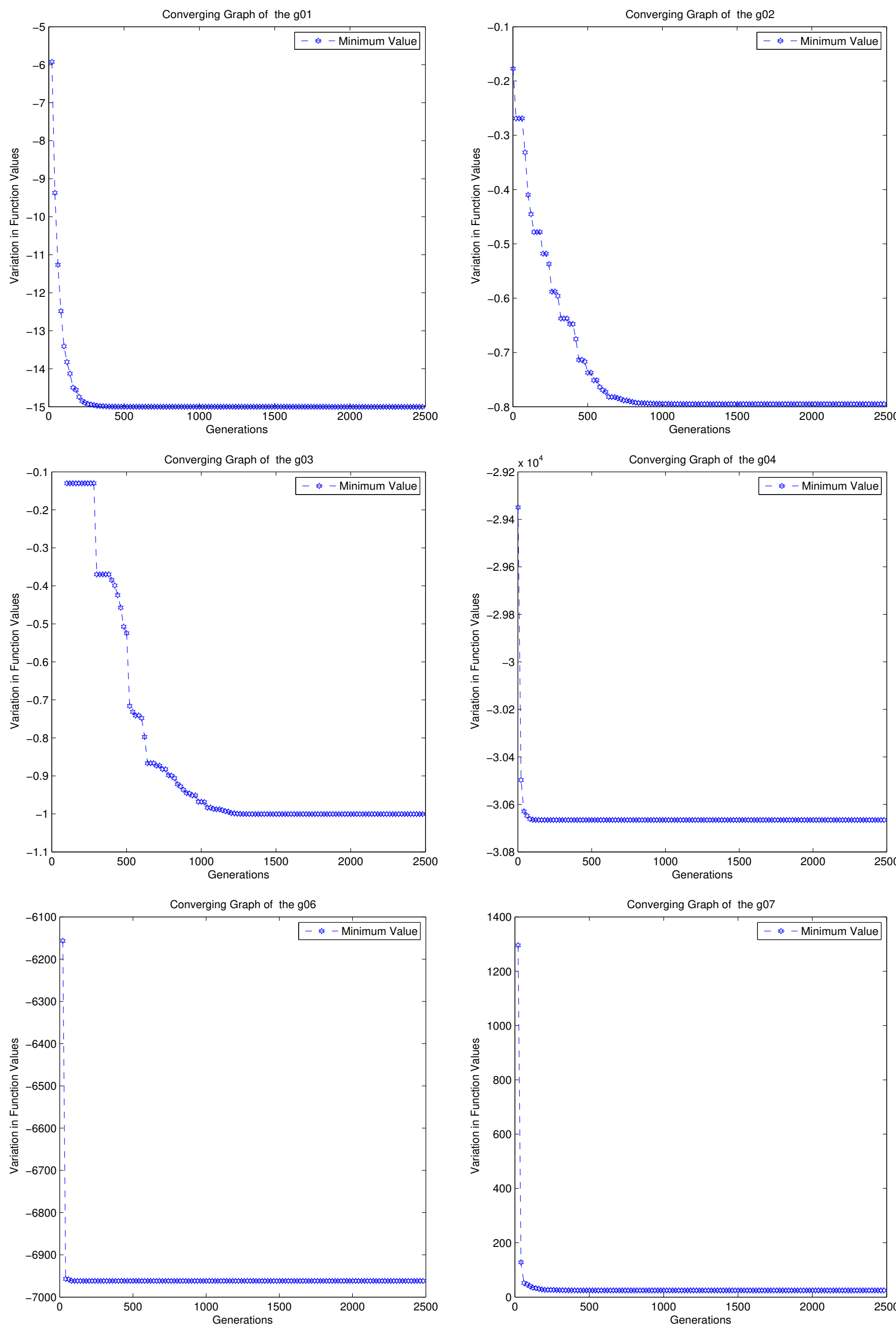

Figure 1. Convergence Graph of HCEA-static for CEC'06 Benchmark Functions.

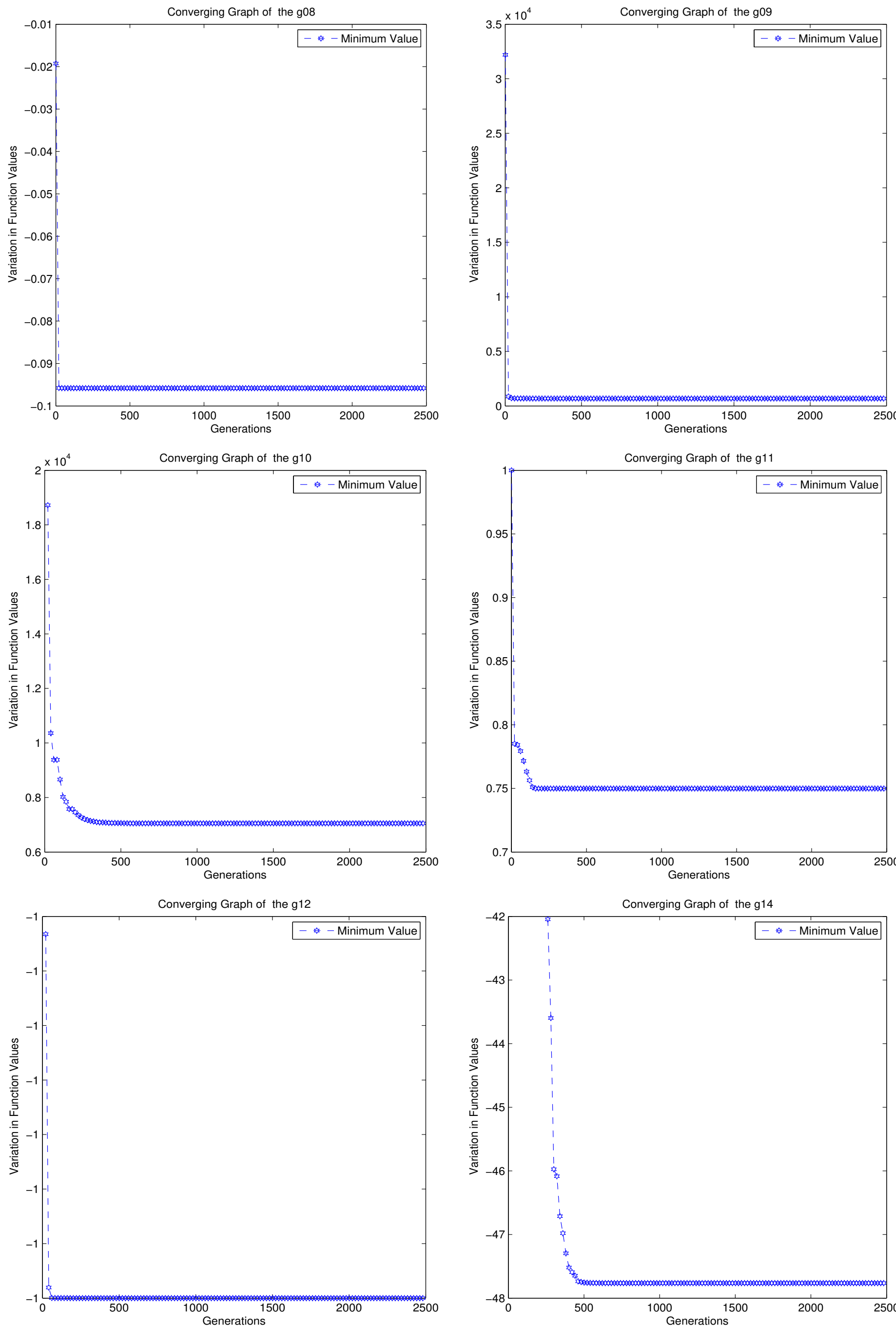

Figure 2. Convergence Graph of HCEA-static for the CEC'06 Benchmark Functions.

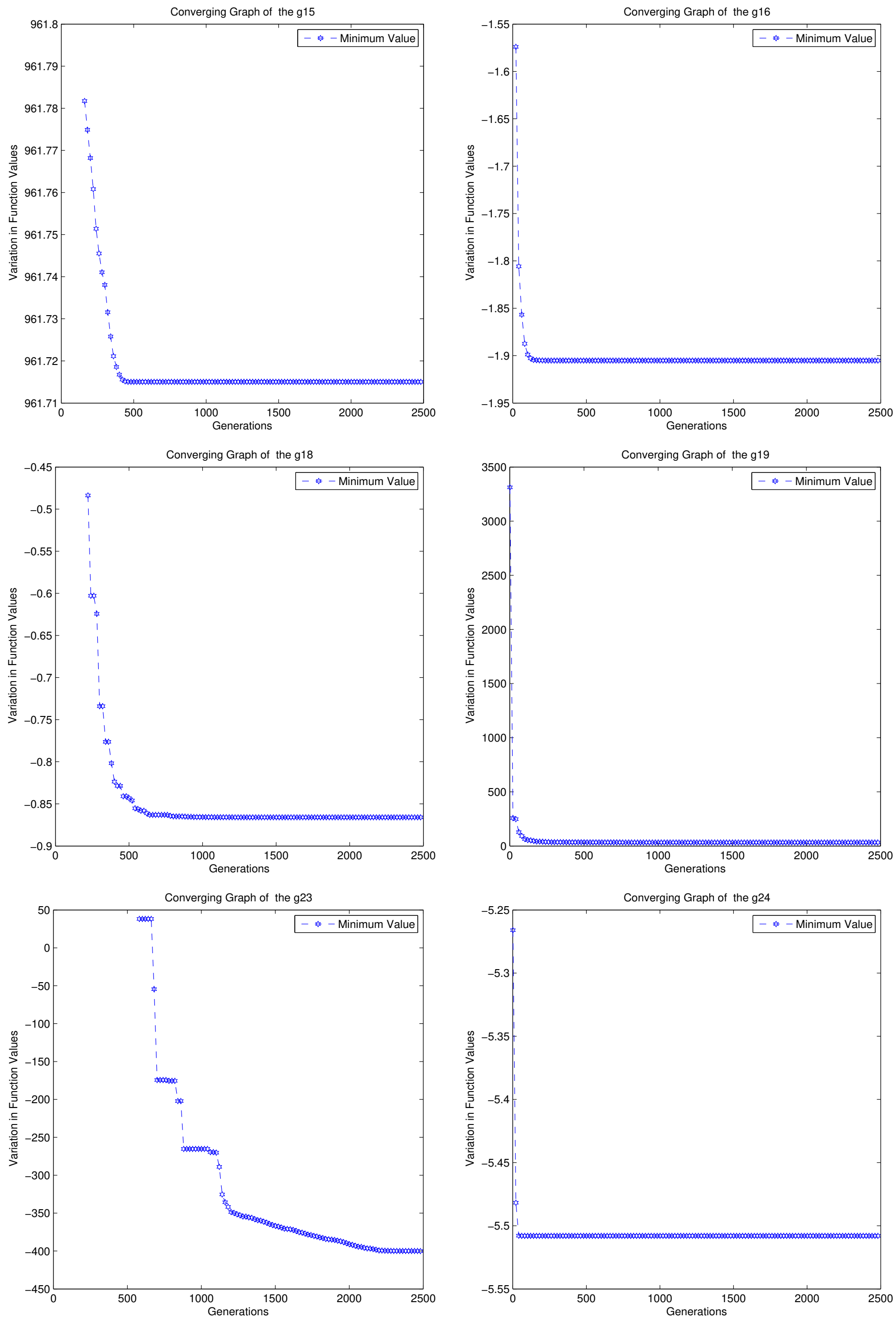

Figure 3. Convergence Graph of HCEA-static for the CEC'06 Benchmark Functions.

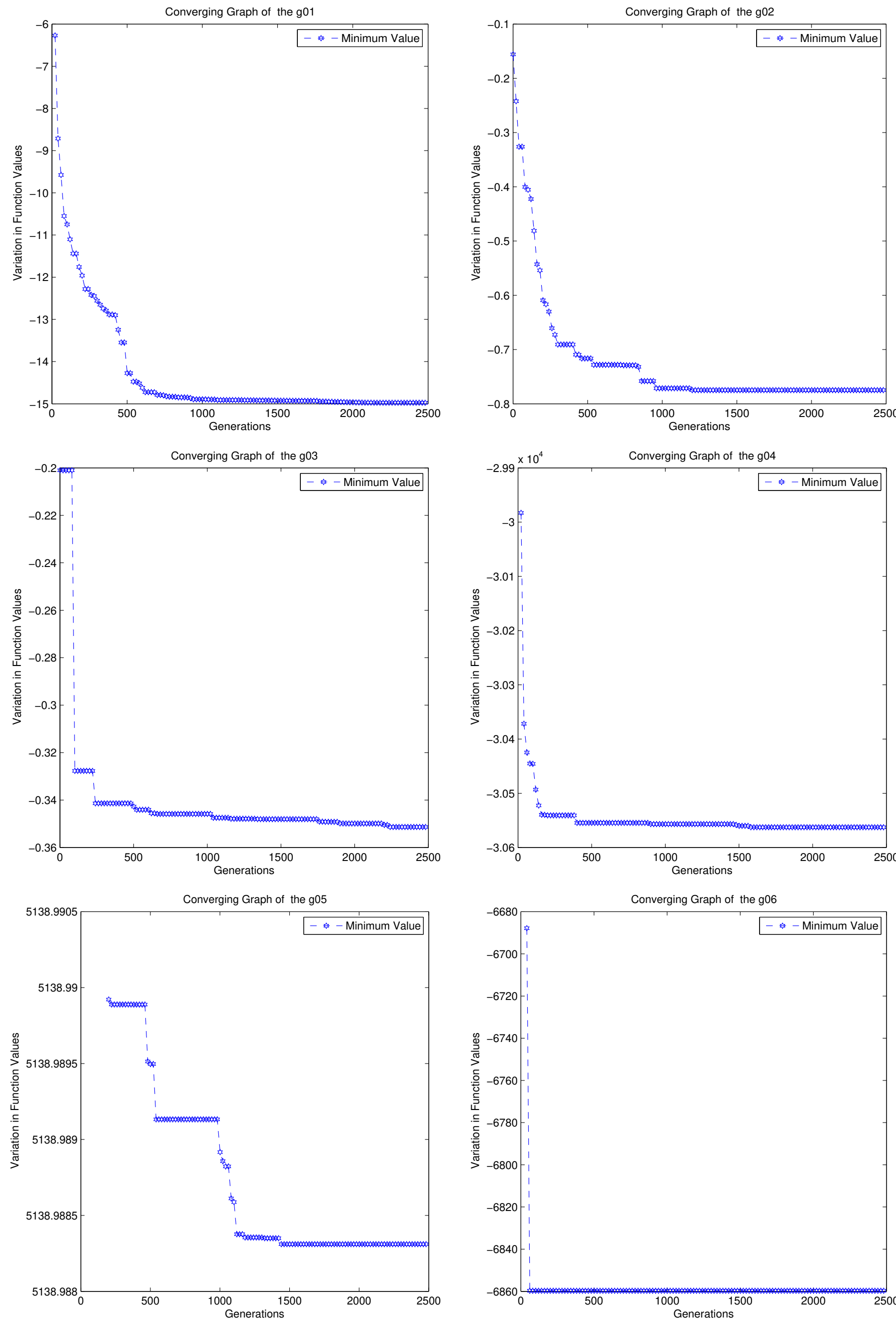

Figure 4. Convergence Graph of HCEA-adaptive for theCEC'06 Benchmark Functions.

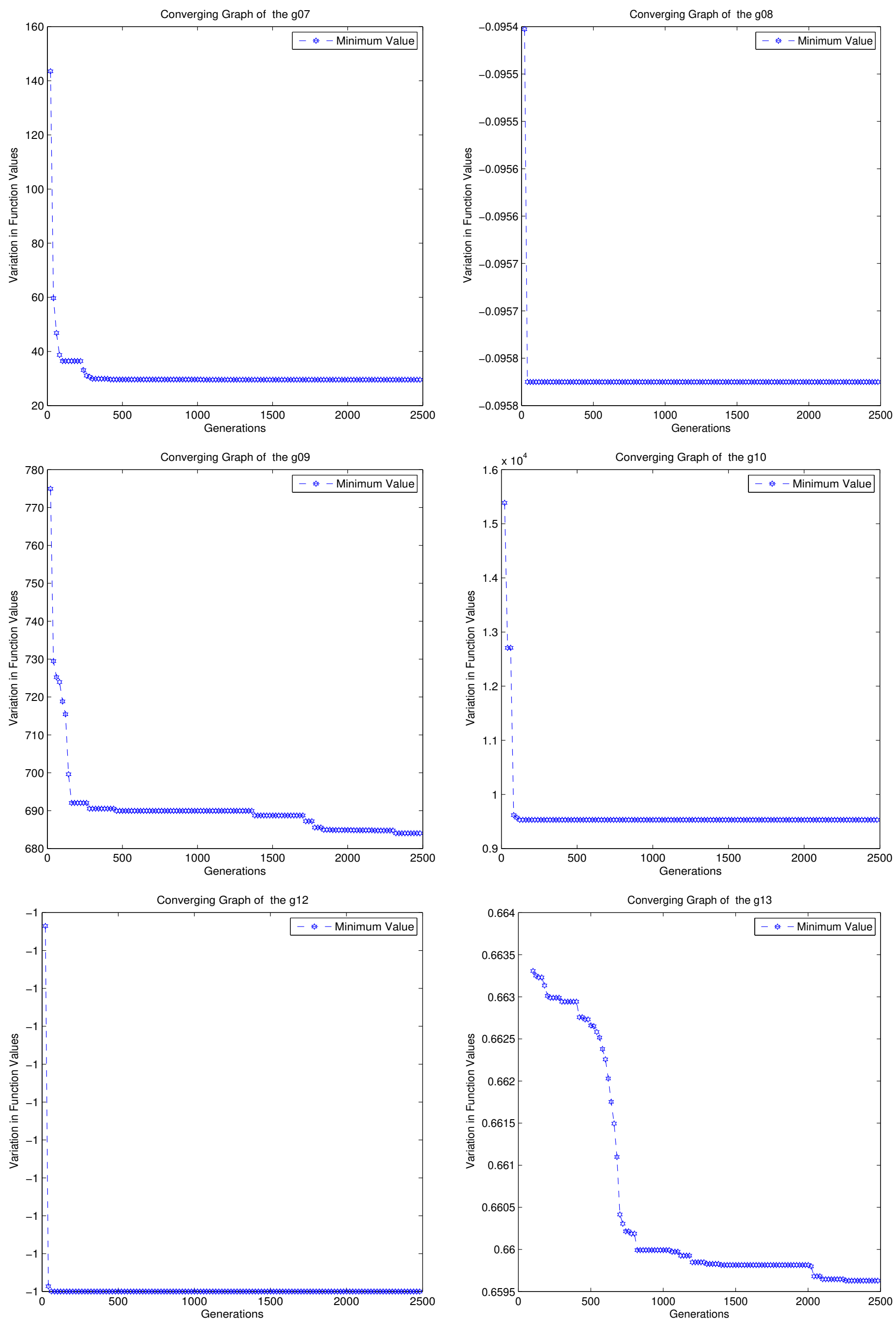

Figure 5. Convergence Graph of HCEA-adaptive for the CEC'06 Benchmark Functions.

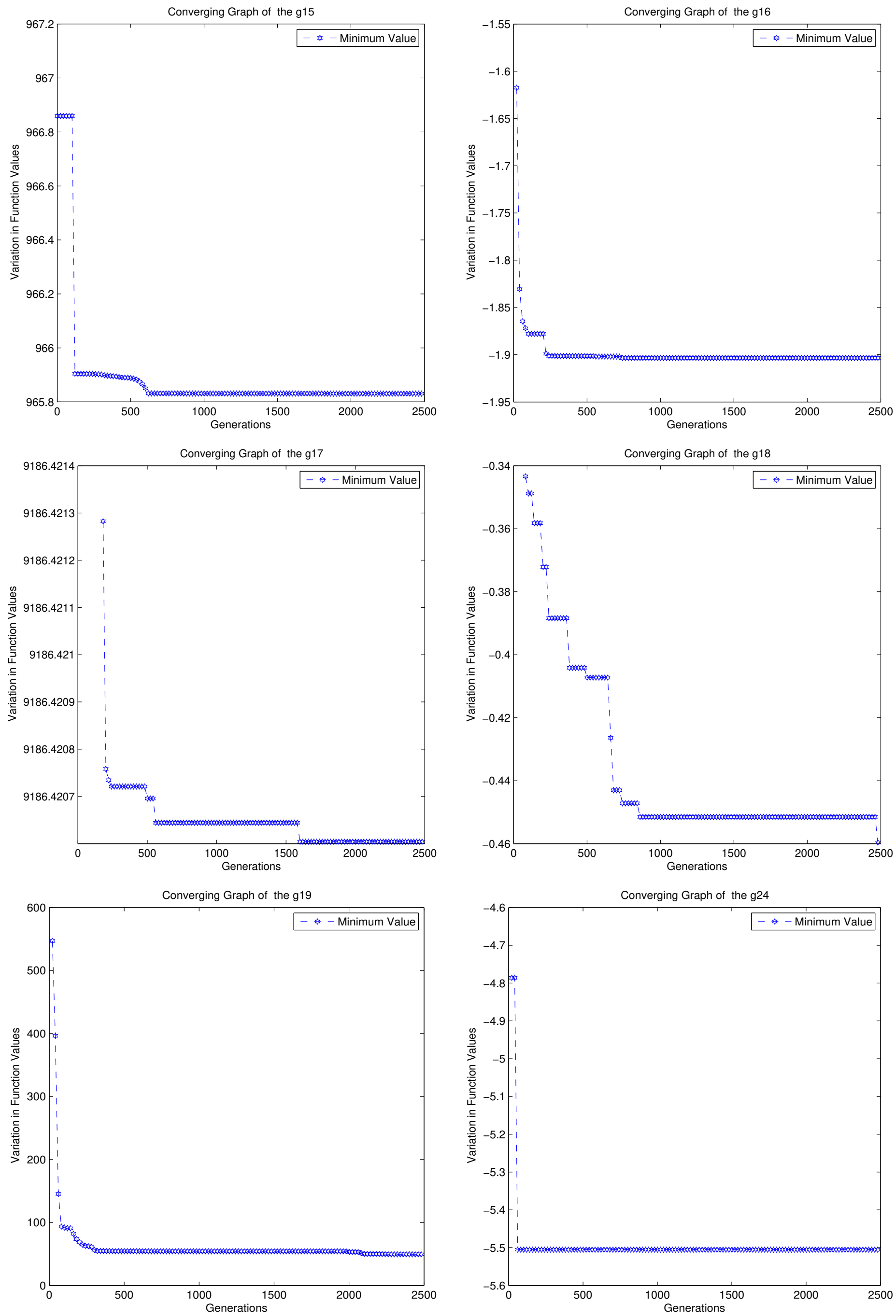

Figure 6. Convergence Graph of the HCEA-adaptive for the CEC'06 Benchmark Functions.

Table 3. Comparative Analysis of the a) HEA-static, b) HEA-adaptive and c) CJADE-D [53].

Problem	Method	Minimum	Mean	Median
g08	a	−0.095825000000	−0.095825000000	−0.095825000000
	b	−0.095825000000	−0.095825000000	−0.095825000000
	c	**-1296.806712**	**-1348.735625**	**-1346.209597**
g09	a	680.630057000000	680.630057000022	680.630057000000
	b	680.813616000000	680.813616000044	680.813616000000
	c	680.630057	680.630057	680.630057
g10	a	7049.248021000000	7049.248020999572	7049.248021000000
	b	7362.948704000000	7362.948704000238	7362.948704000000
	c	**2131.846129**	**2140.683593**	**2140.014032**
g11	a	0.749900000000	0.749900000000	0.749900000000
	b	0.749985000000	0.749985000000	0.749999000000
	c	**0.443553**	**1.834e-003**	**0.440157**
g12	a	−1.000000000000	−1.000000000000	−1.000000000000
	b	−1.000000000000	−1.000000000000	−1.000000000000
	c	−1.000000	−1.000000	−1.000000
g13	a	**0.062460000000**	**0.075469168400**	**0.484833000000**
	b	0.086195000000	0.086195000000	0.086195000000
	c	1.201125	1.837891	1.775458
g14	a	−47.764888000000	−47.764888000002	−47.764888000000
	b	−44.836489000000	−44.836488999998	−44.836489000000
	c	**-1420.022548**	**-1561.624708**	**-1570.929559**
g15	a	961.715022000000	961.715022000016	961.715022000000
	b	961.718600000000	961.718599999989	961.721756000000
	c	**771.641627**	**709.556725**	**709.110988**

3.2. Graphical Result of HCEA-adaptive for Benchmark Functions. Figures 1-2-3) demonstrate the convergence graph of the each CEC'06 test function displayed by HEA-static algorithm in single run of simulation. These figures clearly demonstrate that HEA-static have obtained approximate optimal solutions for g01, g04, g05, g07, g08, g09, g10, g11, g12, g14, g15, g16, g19 and g24 in almost 500 generations. For the test problems denoted by g02, g03 and g18, HEA-static have obtained optimal solution in almost 1000 generations. These sort of convergence behaviors of the HEA-static stamped their fast convergence speed.

The Figures 4-5-6) depicts the convergence graph of the CPSO-Adaptive. These figures show that CPSO-Adaptive have figured out the problem g06, g07, g08, g10, g11, g12, g16, g19 and g24 in 500 generations to reach near the know optimal values of these problems. Similarly, the approximated optimal of problems g01, g02, g04, g15 and g18 are hereby obtained by HEA-Adaptive in almost in 1000 generations while for the g03, g05, g09 g13 and g17 the optimal values are almost obtained 2000 generations.

From the above discussion, one can conclude that the convergence behavior of the HCEA-static is much better than HEA-adaptive for the most of the used test problems. This better performance of the HCEA-static can be attributed to the fact to wise adjustment of the intrinsic parameters of the HCEA-static keeping in view the mathematical formulations demand of the IEEE-CEC'06 test problems [31].

Table 4. Comparative Analysis of the a) HCEA-static, b) HCEA-adaptive and c) CJADE [53].

Problem	Method	Minimum	Mean	Median
g16	a	−1.905155000000	−1.905155000000	−1.905155000000
	b	−1.904886000000	−1.904886000000	−1.904867000000
	c	−1.905155	−1.905155	−1.905155
g17	a	8862.697056999999	8862.697057000150	8862.697056999999
	b	8875.006079999999	8876.330849559627	8904.658104000000
	c	**453.620550**	**1475.586258**	**1506.524787**
g18	a	**-0.866025000000**	**-0.866025000000**	**-0.866025000000**
	b	−0.756597000000	−0.753730160400	−0.747876000000
	c	32.185920	87.105215	90.087847
g19	a	32.655593000000	**32.655593000002**	32.655594000000
	b	41.652554000000	41.652554000000	41.652554000000
	c	32.655593	33.110947	**32.655593**
g21	a	**193.724510000000**	**193.724509999987**	**193.724510000000**
	b	194.219392000000	194.219391999999	194.219392000000
	c	475.884809	345.569134	345.311741
g23	a	−400.055100000000	−400.055099999998	−400.055100000000
	b	−373.588630000000	−373.588630000011	−373.588630000000
	c	**-3022.665686**	**-2458.257132**	**-2439.892090**
g24	a	−5.508013000000	−5.508013000000	−5.508013000000
	b	−5.507999000000	−5.507999000000	−5.507999000000
	c	−5.508013	**-5.628632**	**-5.631511**

4. Conclusion

In general, classical optimization methods are usually unable to solve the problems having complicated objective functions and concave feasible regions with very small part of the whole search space. In the recent few years evolutionary algorithms (EAs)have become a research interest to different domain of researchers for solving complex problems in science, engineering, management and financial real applications due to their population-based nature. They don't demand for any derivative information regarding the problems at hand. They provide a set of optimal solutions in single simulation unlike traditional optimization techniques.

Over the last two decades, several bio-inspired techniques have been developed based on nature collection, intelligence movement, thinking behaviors of social insects such as Ant, Honey Bees, Buffalos, Birds, Particles, Fishes etc. Particle swarm optimization (PSO) and differential evolution are two well known and the most effective EAs for engineering optimization problems. In this paper, we have combined both PSO and DE by employing two penalty functions with static manner and adaptive manner. The suggested algorithms have two versions called HCEA-adaptive and HCEA-static. The suggested algorithms have tackled most of the benchmark functions that were designed in 2006 IEEE-Congress on evolutionary computation (CEC'06). The simulation results offered by proposed algorithms are highly promising. Out of 24 benchmark function, 22 functions are solved by the suggested hybrid constrained EAs with good convergence speed as compared to the recent constrained version of JADE [53].

In near future, we intend to improve further the algorithmic structure of the suggested algorithms by employing some other novel and specialized constraint-handling techniques to cope with IEEE-CEC test instances [29, 10, 30].

Acknowledgements

The first author is thankful to Higher Education Commission Pakistan for the financial support under NRPU research project NO: 5892 to produce and publish this research article. The first author is also thankful to the Hybrid Metaheuristic Computing Research Group of King Khalid University (KKU), Kingdom of Saudi Arabia for the partial support to make it possible this research contribution.

References

[1] Abraham, Ajith, Grosan, Crin and Ishibuchi, Hisao,*Hybrid Evolutionary Algorithms*, Studies in Computational Intelligence, Springer, 2007.

[2] Abdel-Basset, Mohamed, Wang, Gai-Ge, Kumar Sangaiah, Arun and Rushdy, Ehab., *Krill herd algorithm based on cuckoo search for solving engineering optimization problems*, Multimedia Tools and Applications, 1-24,2017.

[3] Alam, Khug, Mashwani, Wali Khan and Asim, Muhammad,*Hybrid Biography Based Optimization Algorithm for Optimization Problems*, Gomal University Journal of Research,33(1), 134-142, 2017.

[4] Asim, Muhammad, Mashwani, Wali Khan and Jan, M.A., Hybrid Genetic Firefly Algorithm for Global Optimization Problems, Sindh University Research Journal, 49(4), 899-906, 2017.

[5] Asim, Muhammad, Mashwani, Wali Khan, Jan, Muhammad Asif and Iqbal, Javed, *Derivative Based Hybrid Genetic Algorithm: A Preliminary Experimental Results*, Punjab University Journal of Mathematics, Vol. 49(2), pp. 89-99, 2017.

[6] Asim, Muhammad, Mashwani, Wali Khan, Yeniay, Ozgur, Jan, Muhammad Asif, Hussian, Hazrat and Wang, Gai-Ge, *Hybrid Genetic Algorithms for Global Optimization Problems*, Hacettepe Journal of Mathematics and Statistics, 47 (3), 539 - 551, 2018.

[7] Blaha, Brian and Wunsch, Don, *Evolutionary programming to optimize an assembly program*, Proceedings of the 2002 Congress on *Evolutionary Computation, CEC02*, 2, 19011903, 2002.

[8] Bentouati,Bachir, Saliha, Chettih, El-Sehiemy, Ragab A. and Wang, Gai-Ge, *Elephant Herding Optimization for Solving Non-convex Optimal Power Flow Problem*,Journal of Electrical and Electronics Engineering, 10, 31-40, 2017.

[9] Coello Coello, Carlos, *Use of a self-adaptive penalty approach for engineering optimization problems*, Computers in Industry, 41(2), 113127, 2000.

[10] Chen, Q., Liu, B., Zhang, Q., Liang, J. J., Suganthan, P. N., Qu, B.Y. , *Problem Definition and Evaluation Criteria for CEC 2015 Special Session and Competition on Bound Constrained Single-Objective Computationally Expensive Numerical Optimization*, Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore, Nov, 2014.

[11] Chiong, Raymond, Weise, Thomas and Michalewicz, Zbigniew (Editors), *Variants of Evolutionary Algorithms for Real-World Applications*, ISBN 3642234232, Springer, 2012.

[12] Cagnoni, Stefano, Poli, Riccardo, Smith, George D., Corne, David, Oates, Martin, Hart, Emma, Lanzi, Pier L., Egbert J, Willem, Li, Yun, Paechter, Ben, Fogarty, Terence C. *Real-World Applications of Evolutionary Computing*, Springer-Verlag Lecture Notes in Computer Science, Berlin, 2000.

[13] Chen, Su Huan, Wu, Jie and Chen, Yu Dong, *Interval optimization for uncertain structures*, Finite Elements in Analysis and Design, 40(11), 13791398, 2004.

[14] Datta, Rituparna and Deb, Kalyanmoy,*Evolutionary Constrained Optimization*,Infosys Science Foundation Series in Applied Sciences and Engineering, ISSN: 2363-4995, Springer, 2015.

[15] Eidehall, Andreas and Petersson, Lars *Threat assessment for general road scenes using Monte Carlo sampling*, IEEE Intelligent Transportation Systems Conference, 2006.

[16] Eiben, A.E. and Smith, James E., *Introduction to Evolutionary Computing: Natural Computing Series*, Springer-Verlag Berlin Heidelberg, 2015.

[17] Engelbrecht, Andries P., *Computational Intelligence An Introduction*,Second Edition, John Wiley, 2007.

[18] El-Mihoub, Tarek A., Hopgood, Adrian A., Nolle, Lars and Battersby, Alan, *Hybrid Genetic Algorithms: A Review*, Engineering Letters, 13(2), 124-137, 2006.

[19] Floudas, Christodoulos A., Pardalos, Panos M., Adjiman, Claire, Esposito, William R., Gums, Zeynep H., Harding, Stephen T., Klepeis, John L., Meyer, Clifford A., and Schweiger, Carl A., *Handbook of test problems in local and global optimization*, Vol. 33. Springer Science & Business Media, 2013.

[20] Fogel, Lawrence J., Walsh, Alvin J. and Owens, Michael J., *Artificial Intelligence through Simulated Evolution*, John Wiley, 1966.

[21] Fogel, Lawrence J., *Intelligence through Simulated Evolution: Forty Years of Evolutionary Programming*, John Wiley, 1999.

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

[23] Geng, Xiutang, Xu, Jin, Xiao, Jianhua and Pan, Linqiang, *A simple simulated annealing algorithm for the maximum clique problem*, *Information Sciences*, 177, 22, 50645071, 2007.

[24] Homaifar, Abdollah, Qi, Charlene X. and Lai, Steven H., *Constrained optimization via genetic algorithms*, Simulation, 62, 242-254, 1994.

[25] Kennedy, James and Eberhart, Russell C., *Particle swarm optimization*, in Proceedings of the IEEE International Conference on Neural Networks, 4, 1942-1948, November, 1995.

[26] Khanum, Rashida Adeeb, Jan, Muhammad Asif, Mashwani, Wali Khan, Tairan, Naseer Mansoor, Khan, Hidayat Ullah and Shah, Habib, *On the hybridization of global and local search methods*, Journal of Intelligent & Fuzzy Systems, 35(3), 3451-3464, 2018.

[27] Lawler, Eugene L. and Wood, D. E., *Branch-and-Bound Methods: A Survey, Journal Operation Research*, 14, 4, 699-719, Institute for Operations Research and the Management Sciences, Linthicum, Maryland, USA, 1996.

[28] Liu, H., Cai, Z., and Wang, Y., *Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization*, Applied Soft Computing,10(2), 629-640, 2010.

[29] Liang, J. J., Qu, B-Y., Suganthan, P. N., *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*, Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore, December, 2013.

[30] Liang, J. J., Qu, B. Y., Suganthan, P. N. and Chen, Q., *Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization*, Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore, November, 2014.

[31] Liang, J. J., Runarsson, T. P., Mezura-Montes, E., Clerc, M., Suganthan, P. N., Coello Coello, C. A. and Deb, K., *Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization*, Technical Report, Nanyang Technological University, Singapore, 2006.

[32] Lin, Mingjie and Wawrzynek, John, *Improving FPGA placement with dynamically adaptive stochastic tunneling*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29, 12, 18581869, 2010.

[33] Mezura-Montes, Efrén and Coello Coello, Carlos, *A simple multimembered evolution strategy to solve constrained optimization problems*, IEEE Transactions on Evolutionary Computation, 9(1), 117, 2005

[34] Machta, Jon *Strengths and weaknesses of parallel tempering*, *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 80, 5, 2009.

[35] Mallipeddi, Rammohan, Das, S., Suganthan, P.N., *Ensemble of Constraint Handling Techniques for Single Objective Constrained Optimization*, In: Datta R., Deb K. (eds) Evolutionary Constrained Optimization. Infosys Science Foundation Series, Springer, New Delhi, 2015.

[36] Mallipeddi, Rammohan and Suganthan, P.N., *Ensemble of Constraint Handling Techniques*, in IEEE Transactions on Evolutionary Computation, 14(4), 561-579, 2010.

[37] Mashwani, Wali Khan and Salhi, Abdellah,*Multiobjective Memetic Algorithm Based on Decomposition*, Applied Soft Computing, No 21, 221-243, 2014.

[38] Mashwani, Wali Khan and Salhi, Abdellah, *Multiobjective Evolutionary Algorithm Based on Multimethod with Dynamic Resources Allocation Strategy*, Applied Soft Computing Journal, Vol 39, 292-309, 2016.

[39] Mashwani, Wali Khan, Salhi, Abdellah, Yeniay, Ozgur, Hussian, Hazrat, Jan, Muhmmad Asif, *Hybrid non-dominated sorting genetic algorithm with adaptive operators selection*, Applied Soft Computing, vol 56, 1-18, 2017.

[40] Mashwani, Wali Khan, Salhi, Abdellah, Yeniay, Ozgur and Khanum, R.A and Jan, Muhammad Asif, *Hybrid Adaptive Evolutionary Algorithm Based on Decomposition*, Applied Soft Computing, Volume 57, 363378, 2017.

[41] Mashwani, Wali Khan,Salhi, Abdel, Jan, Muhammad Asif, Khanum, R.A. and Sulaiman, M., *Impact Analysis of Crossovers in Multiobjective Evolutionary Algorithm*, Sci.Int.(Lahore), 27(6), 4943-4956, 2015.

[42] Mashwani, Wali Khan and Salhi, Abdellah, *A Decomposition Based Hybrid Multiobjective Evolutionary Algorithm with Dynamic Resources Allocation*, Applied Soft Computing, 12(9), 2765-2780, 2012.

[43] Pan, Changcheng, Xu, Chen and Li, Guo, *Differential evolutionary strategies for global optimization*, *Shenzhen Daxue Xuebao (Ligong Ban)*, Journal of Shenzhen University Science and Engineering, 25(2), 211-215, 2008.

[44] Salhi, Abdellah and Fraga, Eric S., *Nature-Inspired Optimisation Approaches and the New Plant Propagation Algorithm*, Proceedings of the ICeMATH2011, K2-1 to K2-8, 2011.

[45] Shah, Habib, Tairan, Nasser, Ghazali, Rozaida, Yeniay, Ozgur and Mashwani, Wali Khan, *Hybrid Honey Bees Meta-Heuristic For Benchmark Data Classification*, Exploring Critical Approaches of Evolutionary Computation, IGI Global Publisher, 2019

[46] Shah, Habib, Tairan, Nasser, Mashwani, Wali Khan, Ahmad Al-Sewari, Abdulrahman, Jan, Muhammad Asif and Badshah, Gran, Hybrid Global Crossover Bees Algorithm for Solving Boolean Function Classification Task. International Conference on Intelligent Computing ICIC (3), 467-478, 2017.

[47] Sharma, Jyoti and Singhal, Ravi Shankar,*Genetic Algorithm and Hybrid Genetic Algorithm for Space Allocation Problems-A Review*, International Journal of Computer Applications, 95(4), 33-37, 2014.

[48] Sulaiman, Muhammad, Salhi, Abdelah, Khan, Asfandyar,Muhammad, Shakoor and Mashwani, Wali Khan, *On the Theoretical Analysis of the Plant Propagation Algorithms*, Mathematical Problems in Engineering, Volume 2018.

[49] Sulaiman, Muhammad, Salhi, Abdellah, Mashwani, Wali Khan and Rashidi, Muhammad M., *A Novel Plant Propagation Algorithm: Modifications and Implementation*, Science International, 28(1), 201-209, 2016.

[50] Sulaiman, Muhammad and Salhi, Abdella, *A Seed-based Plant Propagation Algorithm: The Feeding Station Model*, The Scientific World Journal, 1-16, 2015

[51] Sulaiman, Muhammad, Salhi, Abdella, Selamoglu, Birsen Irem, Kirikchi, Omar Bahaaldin, *A Plant Propagation Algorithm for Constrained Engineering Optimisation Problems*, Mathematical problems in engineering, 1-10, 2014.

[52] Storn, Rainer M. and Price, Kenneth,*Differential Evolution: A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*, *Journal of Global Optimization*, 11(4), 341-359, 1997.

[53] Shah, Tayaba,Jan, Muhammad Asif, Mashwani, Wali Kahan and Wazir, Hamza, *Adaptive differential evolution for constrained optimization problems*, Science Int.(Lahore), 3(28) 41044108, 2016.

[54] Veltman, M., *Algebraic techniques*, Computer Physics Communications, 3, 7578, September, 1972.

[55] Vetschera, Rudolf, *A general branch-and-bound algorithm for fair division problems*, Journal Computers and Operations Research, 37(12), 2121-2130, December, 2010.

[56] Wang, Gai-Ge, *Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems*, Memetic Computing, 2016.

[57] Wang, Gai-Ge, Deb, Suash, Zhao, Xinchao and Cui, Zhihua, *A new monarch butterfly optimization with an improved crossover operator*, Operational Research, 2016.

[58] Wang, Gai-Ge, Deb, Suash and Coelho, Leandro *Elephant Herding Optimization*, 2015.

[59] Wang, Gai-Ge, Deb, Suash and Coelho, Leandro,*Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems*, International Journal of Bio-Inspired Computation, 2015.

[60] Wu, Chenhan, *Ant colony multilevel path optimize tactic based on information consistence optimize*, International Conference on "Computer Application and System Modeling, 1, 533536, November, 2010.

[61] Wazir, Hamza, Jan, Muhammad Asif, Mashwani, Wali Khan and Shah, Tayyaba, *A Penalty Function Based Differential Evolution Algorithm for Constrained Optimization*, The Nucleus Journal, 53(1), 155-161, 2016

[62] Wang, Gai-Ge, Gandomi, Amir, Alavi, Amir and Gong, Dunwei, *A comprehensive review of krill herd algorithm: variants, hybrids and applications* Artificial Intelligence Review, 2017.

[63] Wang,Yong, Liu, Hui, Cai, Zixing and Zhou, Yuren, *An orthogonal design based constrained evolutionary optimization algorithm* Engineering Optimization, 39 (6): 715736, 2007.

[64] Wang, Y, Cai, Z, Zhou, Y, Zeng, W., *An adaptive trade-off model for constrained evolutionary optimization*, IEEE Transactions on Evolutionary Computation,12(1): 8092, 2008.

[65] Wang, Yong and Cai, Zixing,*A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems*, Frontiers of Computer Science in China, 3(1),38-52, 2009.

[66] Wang, Yong, Cai, Zixing, Zhou, Yuren and Zeng, Wei, *An adaptive trade-off model for constrained evolutionary optimization*, IEEE Transactions on Evolutionary Computation, 12(1), 8092,2018.

[67] Yu, Jianbo, Xi, Lifeng and Wang, Shijin, *An improved particle swarm optimization for evolving feed forward artificial neural networks*, Neural Processing Letters, vol. 26, no. 3, pp. 217231, 2007.