Available online www.ejaet.com

**European Journal of Advances in Engineering and Technology, 2020, 7(6):57-65**

**Research Article** **ISSN: 2394 - 658X**

# Metaheuristic-based Adaptive Hybrid Algorithm for solving Constrained Optimization Problems

**Julius Kilonzi Charles, Peter Musau Moses and Jackson Mwangi Mbuthia**

*Department of Electrical & Information Engineering, University of Nairobi, Nairobi, Kenya*
*charles.kilonzi@gmail.com*

---

**ABSTRACT**

*In this paper a novel Adaptive Hybrid Optimization technique based on Evolutionary and Swarm Intelligence Meta-heuristic methods is formulated and tested in solving complex optimization problems. The hybrid utilizes some of the mostly studied and applied metaheuristic methods in the hybridization and adaptation process with the aim of suppressing their individual weaknesses while taking advantages of the associated individual strengths. The proposed approach combines the strengths of Differential Evolution (DE) and Bacterial Foraging Optimization Algorithms (BFOA) in the hybridization while their weaknesses are mitigated by the introduction of important Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) characteristics in the algorithm formulation. The developed algorithm is tested on the high dimensional Standard Benchmark Functions (F1-F10) as well as two constrained engineering optimization problems (Pressure vessel design and tension/compression spring design). The obtained results are compared with those obtained by other researchers using other well-known metaheuristic optimization methods. When subjected to solving the standard benchmark functions the developed algorithm outperformed the rest of the optimization methods in eight out of the ten test functions. In addition, the developed algorithm produced superior results for the two constrained engineering optimization problems when compared to other meta-heuristic methods.*

**Key words:** Meta-heuristic methods, BFOA, GA, PSO, DE

---

## INTRODUCTION

Different methods and techniques have been formulated to solve various optimization problems. These methods can broadly be classified into for main groups namely: deterministic (mathematic/exact) methods, heuristic (approximate) methods, metaheuristic methods and hybrid methods.Deterministic methods include the unconstrained methods which convert constrained problems into unconstrained form. These methods include all mathematical models which are focused on optimization processes with objective function minimization/maximization subject to sets of constraints [1]. They mainly include the classical programming approaches (e.g. Linear Programming (LP), Dynamic Programming (DP), Quadratic Programming, Non-Linear Programming (NLP) etc.) and decomposition techniques (e.g. Benders Decomposition (BD), Hierarchical Decomposition (HD), Branch & Bound Algorithm (BBA) among others). When solving complex optimization problems which are often nonlinear and non-convex, the computational effort in these deterministic methods is usually huge. In such scenarios, many of these methods require the relaxation of the binary to continuous variables to lower computation burden, however this may lead to solutions far from the optimum [2]. In addition, due to intrinsic limitations of the searching process there is a possibility that the obtained optimal solution corresponds to a local optimum.

Heuristic methods are inventive techniques based on users' experience and hence their computational performance is usually better than that of the mathematical methods. They can be interactive or non-interactive. Interactive heuristic methods interact with the planner in their step-by-step generation, evaluation, and selection of expansion options, while non-interactive do not [2]. Most combinatorial problems cannot be solved to optimality in reasonable computation times, due to their dimensionality or other characteristics. Being exact in most practical optimization problems may be meaningless, since one is dealing with not very precise data in addition to simple simplifications of reality. However, the integrity of the input data, technique used and the solution should be within the acceptable limits. Though heuristic methods can give good feasible solutions with reasonable computation efforts, they are however problem-dependent and require parameter tuning. Thus, they do not guarantee global solution attainment. Such methods include: Sensitivity Analysis, Depth First Search (DeFS), Best First Search (BeFS), Scenario Analysis etc.

Metaheuristic methods combine the attributes of both deterministic and approximate methods. Unlike heuristic methods, they are not problem-dependent however some intrinsic parameter fine tuning is necessary in their adaptation to specific problems. In these approaches, the constraints and objective functions in the problem formulation are not differentiated since the approaches need no prior knowledge of the problem. The fact that these methods are not gradient-based (derivative-free) helps them avoid premature convergence as a result of being trapped in local optima. Their independence from the starting point (initial solution) eliminates the necessity for convexity in solving optimization problems. Available metaheuristic methods can be classified into three groups namely: Evolutionary Algorithm (EA) approaches (e.g. Evolutionary Programming (EP), Genetic Algorithm (GA), Evolution Strategies (ES), Differential Evolution (DE), Artificial Immune Systems (AIS) etc.); Swarm Intelligence Approaches (e.g. Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Bat Algorithm (BA), Bee Colony Optimization (BCO), Artificial Bee Colony (ABC), Bacterial Foraging Optimization Algorithm (BFOA) etc.) and Trajectory Metaheuristic Approaches (e.g. Hill Climbing (HC), Simulated Annealing (SA), Tabu Search (TS), Greedy Randomized Adaptive Search Procedures (GRASP), Teacher Learning Algorithm (TLA), Biogeography-Based Optimization (BBO) etc.

In the recent past, researchers in this area are hybridizing various techniques to come up with powerful but less complex methods which can be used to solve the different optimization problems. These hybrids techniques are formed by combining two or more of the above reviewed techniques. Hybrids can be between/among methods in the same category of in different categories [2]. In most cases these researchers combine a deterministic approach with a heuristic or meta-heuristic approach. Recently, there is increasing hybridization of heuristic and meta-heuristic methods. The reason for this increased use of hybrids is because they exalt the strengths and improve the weaknesses of the methods concerned.This paper presents a novel metaheuristic-based adaptive hybrid approach developed by combining the attributes of evolutionary techniques with those of swarm intelligence techniques. Table-1 gives the attribute(s) of interest for each optimization technique used in the hybridization process and the reason behind its selection.

**Table -1 Details on selection of Optimization Techniques**

| **Optimization Technique** | **Attribute of Interest** | **Reason for Selection** |
|---|---|---|
| DE | • Real-valued continuous space application<br>• Differential recombination | • Ease of application to a wide variety of real valued problems with multi-modal, multi-dimensional spaces [3].<br>• Gives better results in comparison to other EA in most cases [4]. |
| BFOA | • Easily adaptable<br>• Relatively new with increasing application<br>• Powerful among swarm intelligence techniques | • Its formulation accommodates best attributes from other techniques easily (ease of improvement) [5].<br>• Often outperforms other swarm intelligence techniques [6]. |
| GA | • Cross over<br>• Mutation | • These properties of the GA bring diversity to the candidate solutions thus discouraging premature convergence [7].<br>• Can provide a good guidance for PSO particles thus improving its efficiency [8]. |
| PSO | • Global best<br>• Individual best | • This attribute can be used to bring the useful social/historical information of particle positions leading to faster convergence [9]. |

Evolutionary techniques are based on the powerful principle of evolution, that is, survival of the fittest. They explore biological evolution mechanisms such as reproduction, mutation, recombination (crossover) and selection [10-12]. On the other hand, Swarm Intelligence (SI) approaches exhibit the swarm intelligence phenomenon where the collective behavior of agents interacting locally within their environment in a system causes coherent functional global patterns to emerge [2]. The evolutionary techniques utilized in this hybridization are Genetic Algorithm and Differential Evolution while the swarm intelligence techniques are Particle Swarm Optimization and Bacterial Foraging Optimization Algorithm. The hybridization and adaptation process aims at maximizing on individual benefits while avoiding the associated weaknesses. Detailed steps in the formulation of each variant/type of the four metaheuristic techniques used in this hybridization are given in [5-8].

## PROPOSED ALGORITHM FORMULATION

As mentioned in the introduction, the metaheuristic-based adaptive hybrid algorithm proposed in this paper combines both genetic and swarm intelligence characteristics. The developed algorithm is majorly based on a hybrid of Differential Evolution (DE) and Bacterial Foraging Optimization Algorithms (BFOA), however, various steps in the developed hybrid are adapted through selected Genetically Improved Particle Swarm Optimization (GIPSO) attributes so as to be able to mitigate some of the foreseeable weakness in the optimization process. The algorithm is thus abbreviated ADEBFOA (Adaptive Differential Evolution - Bacteria Foraging hybrid optimization algorithm).

**Proposed Hybridization and Adaptation Steps**

1. The algorithm starts by initializing all the parameters of the techniques involved. These include:
   - Population size (number of bacteria/particles), $N$
   - Chemotactic steps, $N_c$
   - Swim length, $N_s$
   - Reproduction steps, $K$
   - Elimination/Dispersal steps, $L$
   - Step-size limits, $C_{min}$ &$C_{max}$
   - Mutation probability, $P_{mut}$
2. The N population is randomly initialized taking into account all relevant constraints for which the already formulated objective function is being optimized (minimized/maximized) subject to.
3. The fitness of each bacterium/particle is evaluated based on the optimization problem objective function. The value of each bacterium $P$, becomes its personal best denoted, $P_{best}$. The bacterium with the best fitness in this step is denoted as global best denoted, $G_{best}$.

   $$P^i_{best} = P^i, \quad \forall i \tag{1}$$

   $$G_{best} = P^i_{best} \; if \; f\left(P^i_{best}\right) = \min\{f(P^i)\}, \; i \in N \tag{2}$$
4. The iterations are initialized in this stage starting with Elimination/dispersal loop;

   $$l = 1, \; l \in L \tag{3}$$
5. Start reproduction loop;

   $$k = 1, \; k \in K \tag{4}$$
6. Start chemotaxis loop;

   $$j = 1, \; k \in N_c \tag{5}$$

   (i) Unlike in normal BFOA, the chemotactic step here is performed employing an adapted step-size based on GIPSO attributes:

   (ii) The chemotactic movement for a classical BFOA is represented in (6) and (7).

   $$P^i_{(j+1,k,l)} = P^i_{j,k,l} + C(i)\emptyset(i) \tag{6}$$

   $$P^i_{(j+1,k,l)} = P^i_{j,k,l} + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{7}$$

   Where: $P^i_{j,k,l}$ is the position of the $i^{th}$ bacterium in population *N* at the $j^{th}$ chemotactic step in *Nc* steps, $k^{th}$ reproduction step in *K* steps and $l^{th}$ elimination in *L* elimination steps, $C(i)$ is the step size in the random direction and $\emptyset$(i) is a unit vector in the random direction.

   (iii) In a classical PSO algorithm, the velocity and position is updated based on equations (8) and (9) respectively.

   $$V^i_{j+1} = wV^i_j + c_1 r\left(P_{best_i} - S^i_j\right) + c_2 r\left(G_{best} - S^i_j\right) \tag{8}$$

   $$S^i_{j+1} = S^i_j + V^i_{j+1} \tag{9}$$

   Where: $V^i_j$ and $S^i_j$ are the velocity and position of the $i^{th}$ particle in the population, $c_1$ and $c_2$ are weight coefficients for each term respectively and r is a random integer between 0 and 1.

   (iv) These classical PSO equations can be used to improve the chemotactic movement for a classical BFOA by incorporating social behavior between the bacteria. In this case, the new BFOA movement is represented as in (10);

   $$P^i_{(j+1,k,l)} = P^i_{j,k,l} + C(i)\{\left(P_{best_i} - P^i_{j,k,l}\right) + \left(G_{best} - P^i_{j,k,l}\right)\} \tag{10}$$

   (v) This social cooperation ensures both exploration and exploitation in the search process. As a result, it enhances the probability of searching/moving towards better areas as good information is fully utilized.

   (vi) However, premature convergence may arise when $P_{best}$ and $G_{best}$ are located in the same local optimum. In addition, if $P_{best}$ and $G_{best}$ are located on opposite sides of $P^i_{j,k,l}$ oscillations will result. To avoid these limitations the social cooperation analysis is modified using the attributes of Evolutionary Algorithms, i.e. cross-over and mutation. Arithmetic crossover commonly used in Differential Evolution is performed between the $P_{best}$ of each bacterium and the $G_{best}$ to generate a off-spring $P_{ideal_i}$ which is mutated using a mutation probability, $P_{mut}$ as shown in (11) and (12) respectively:

   $$P^i_{ideal} = \begin{cases} \alpha P_{best_i} + (1-\alpha)G_{best}, \text{if } f(P_{best_i}) < f(P_{r,d}) \\ P_{r,d} \quad\quad Otherwise \end{cases} \tag{11}$$

   $$P^i_{ideal} = \begin{cases} P^i_{ideal} + r\Delta P^i_{ideal}, \quad \text{if } r < P_m \\ P^i_{ideal} \quad\quad Otherwise \end{cases} \tag{12}$$

   (vii) Using the $P^i_{ideal}$ obtained in (12) the chemotactic movement given in (10) becomes:

   $$P^i_{(j+1,k,l)} = P^i_{j,k,l} + C(i)\{\left(P^i_{ideal} - P^i_{j,k,l}\right) \tag{13}$$

(viii) To balance between the exploration (diversification) and exploitation (intensification) ability of the DE-ABFOA algorithm, the step size is varied to enhance exploration at earlier stages of chemo-taxis and exploitation at later stages.

$$C^i_{j,k,l} = C_{max} - \frac{(C_{max} - C_{min})}{Nc} . j \quad (14)$$

Equation (14) ensures larger step size at initial stages to guarantee the exploration ability while as the iteration move towards the stopping criterion smaller step sizes are adopted to intensify search around the promising areas and thus enhance algorithm's convergence.

(ix) The $P_{best}$ and $G_{best}$ for each bacterium and the population respectively are then updated using (15) and (16).

$$P^i_{best\,(j+1,k,l)} = \begin{cases} P^i_{(j+1,k,l)} iff(P^i_{(j+1,k,l)}) < f(P^i_{best\,(j,k,l)}) \\ P^i_{best\,(j,k,l)} otherwise \end{cases} \quad (15)$$

$$G^i_{best\,(j+1)} = \begin{cases} P^i_{best\,(j+1)} \;\; if \;\; f(P^i_{best\,(j+1)}) < f(G^i_{best\,(j)}) \\ G^i_{best\,(j)} \quad otherwise \end{cases} \quad (16)$$

(x) Start Swim loop inside the chemotactic step for $Ns$ swims,

$$s = 1, \quad s \in N_s \quad (17)$$

a) Update the position of the bacteria using (13).
b) Evaluate the fitness of the new bacteria population.
c) Update bacterium's$P_{best}$ and $G_{best}$ using (18) & (19).

$$P^i_{best\,(s+1)} = \begin{cases} P^i_{(s+1)} iff(\theta^i_{(s+1)}) < f(P^i_{best\,(s)}) \\ P^i_{best\,(s)} otherwise \end{cases} \quad (18)$$

$$G_{best\,(s+1)} = \begin{cases} P^i_{best\,(s+1)} iff(P^i_{best\,(s+1)}) < f(G_{best\,(s)}) \\ G_{best\,(s)} otherwise \end{cases} \quad (19)$$

d) Increment $s$, if $s > Ns$ go to step (x) else go to step (a)

(xi) Increment $j$, if $j > Nc$ go to step (7) else go to step (v)

7. Perform population reproduction. The BFOA reproduction stage is also modified using GA and DE variants. Roulette wheel selection is used to get the parents from the current population. The probability of a bacterium to be chosen/selected as a parent is given by (20).

$$p_{(\theta^i)} = \frac{f(\theta^i_{best\,(Nc,k,l)})}{\sum_{i=1}^{N} f(\theta^i_{best\,(Nc,k,l)})}, \quad i \in N \quad (20)$$

Where, $f(\theta^i_{best\,(Nc,k,l)})$ is the fitness of $i^{th}$ individual in the population.

Based on arithmetic crossover, the new population is obtained from the parents as given in (21) and (22).

$$\theta^{i(new_1)}_{Nc,k+1,l} = \lambda\theta^{i(old_1)}_{best\,(Nc,k,l)} + (1-\lambda)\theta^{i(old_2)}_{best\,(Nc,k,l)} \quad (21)$$

$$\theta^{i(new_2)}_{Nc,k+1,l} = \lambda\theta^{i(old_2)}_{best\,(Nc,k,l)} + (1-\lambda)\theta^{i(old_1)}_{best\,(Nc,k,l)} \quad (22)$$

Where, λ is a random integer between 0 & 1.

8. Increment $k$, if $k > K$ go to step (11) else go to step (6)
9. Perform Elimination/Dispersal stage: Half of the population (those with the worst fitness) are replaced with randomly assigned new positions in the solution space (similar to the population initialization in step 2) and the other bacteria with the better fitness values are maintained.
10. Increment$l$, if $l > L$ go to step (11) else go to step (5).
11. Output the positions and the fitness of all bacteria in the population. The bacteria with the latest $G_{best}$ becomes the optimal solution for the optimization problem.

## TEST RESULTS AND DISCUSSION

The proposed meta-heuristic based adaptive hybrid algorithm was tested on various standard benchmark functions and constrained engineering test problems. The developed algorithm and the test problems were programmed on MATLAB 2014 software. The obtained results were compared with those obtained by other researchers using available optimization techniques. The following parameters for the developed algorithm were used in the test analysis:

**Table -2 ADEBFOA Parameter Setting**

| Parameter | Symbol | Value |
|---|---|---|
| Population size | $N$ | 100 |
| Chemotactic steps | $N_c$ | 25 |
| Swim length | $N_s$ | 4 |
| Reproduction steps | $K$ | 4 |
| Elimination/Dispersal steps | $L$ | 2 |
| Step-size limits | $C_{min}$ , $C_{max}$ | 0.03, 0.07 |
| Mutation probability | $P_{mut}$ | 0.025 |

**Standard Benchmark Functions**

These functions are often used by researchers to examine the performance of developed optimization algorithms/ techniques. In this analysis both high dimensional (F1-F10) and low dimensional (F11-F20) test functions were employed. Emphasizes was on the high dimensional continuous functions whose dimensionality makes them difficult to solve [13]. Detailed information on the standard benchmark functions applied here can be obtained in [13 & 14] where the dimensionality and modality (unimodal or multimodal) of individual functions are given. Metaheuristic based optimization techniques are usually stochastic in nature and thus their performance cannot be judged in a single run [13], as a result an average of 50 runs was used for the comparisons in this paper. The normalization procedure given in [15] is used to facilitate authentic comparison between results obtained from different algorithms. The procedure employs equation (23):

$$f_{i,norm} = \{1 - \frac{(f_i - f_{min})}{(f_{max} - f_{min})}\} \tag{23}$$

Where $f_{i,norm}$ is the normalized value, $f_i$ is the fitness value, $f_{min}$ and $f_{max}$ are the minimum and maximum fitness values of solution *i*. Table-3 gives the comparison between results of the developed metaheuristic-based adaptive hybrid technique and those of other metaheuristic algorithms.

**Table -3 Statistical Result Comparison for Benchmark Functions**

| Benchmark Function | Result Feature | PSO [16] | BBO [16] | DE [16] | FFA [16] | ADEBFOA [This Method] |
|---|---|---|---|---|---|---|
| F1 (Ackley) | Best | 0.8561 | 0.9125 | 0.1279 | 0.9878 | 0.99889 |
| | Mean | 0.7351 | 0.8924 | 0.0000 | 0.9733 | 0.98053 |
| | Std dev. | 0.7742 | 0.2514 | 0.9875 | 0.7126 | 0.54172 |
| F2 (Griewank) | Best | 0.8016 | 0.9235 | 0.0001 | 0.9616 | 0.96640 |
| | Mean | 0.6842 | 0.9014 | 0.0000 | 0.9324 | 0.91320 |
| | Std dev. | 0.5585 | 0.5197 | 0.1013 | 0.9102 | 0.62140 |
| F3 (Rosenbrock) | Best | 0.9954 | 0.9672 | 0.2541 | 0.9871 | 1.00000 |
| | Mean | 0.9512 | 0.9201 | 0.2435 | 0.9239 | 0.99255 |
| | Std dev. | 0.7649 | 0.5148 | 0.3512 | 0.6284 | 0.26356 |
| F4 (Schwefel 2.26) | Best | 0.9012 | 0.8921 | 0.6214 | 0.8743 | 0.93511 |
| | Mean | 0.8903 | 0.8315 | 0.4240 | 0.8272 | 0.87468 |
| | Std dev. | 0.5541 | 0.5148 | 0.8476 | 0.7513 | 0.59305 |
| F5 (Schwefel 2.22) | Best | 0.7549 | 0.7894 | 0.6259 | 0.9006 | 0.98624 |
| | Mean | 0.7158 | 0.7515 | 0.3682 | 0.8851 | 0.90264 |
| | Std dev. | 0.5541 | 0.8457 | 0.9845 | 0.6022 | 0.63540 |
| F6 (Schwefel 2.21) | Best | 0.8128 | 0.9459 | 0.7547 | 1.0000 | 0.98973 |
| | Mean | 0.7420 | 0.9025 | 0.6789 | 1.0000 | 0.96246 |
| | Std dev. | 0.3518 | 0.4875 | 0.8452 | 0.9638 | 0.47513 |
| F7 (Schwefel 1.2) | Best | 0.6742 | 0.9845 | 0.0000 | 0.9920 | 0.99072 |
| | Mean | 0.6315 | 0.9125 | 0.0000 | 0.9770 | 0.95284 |
| | Std dev. | 0.6842 | 0.5148 | 0.0000 | 0.7516 | 0.70122 |
| F8 (Sphere) | Best | 0.7155 | 0.8965 | 0.6025 | 1.0000 | 1.00000 |
| | Mean | 0.6879 | 0.8823 | 0.5942 | 0.9703 | 0.98564 |
| | Std dev. | 0.6658 | 0.5129 | 0.9551 | 0.7125 | 0.81546 |
| F9 (Rastrigin) | Best | 0.9727 | 0.9621 | 0.6745 | 0.9615 | 0.97762 |
| | Mean | 0.9523 | 0.9222 | 0.6424 | 0.9324 | 0.94285 |
| | Std dev. | 0.5135 | 0.6541 | 0.8845 | 0.9103 | 0.87583 |
| F10 (Quatric) | Best | 0.9021 | 0.9925 | 0.8992 | 0.9872 | 0.99925 |
| | Mean | 0.8999 | 0.9401 | 0.8422 | 0.9238 | 0.92856 |
| | Std dev. | 0.3513 | 0.6846 | 0.6584 | 0.6284 | 0.78961 |

From the results tabulated in Table II above, the developed metaheuristic-based adaptive hybrid algorithm (ADEBFOA) produced better results in eight (out of the ten standard benchmark functions) tests when compared to the other metaheuristic methods. Only in F6 (Schwefel 2.21) and F7 (Schwefel 1.2) functions where the developed algorithm was outperformed by the FireFly Algorithm (FFA). The results obtained in these two functions are however very close to those of FFA. The testing of the algorithm performance having shown promising results from the high dimensional standard benchmark functions was extended to selected representative constrained engineering optimization problems.

**Constrained Engineering Test Problems**

Two constrained engineering problems have been used frequently in open literature to test effectiveness of developed optimization algorithms. These two problems are the pressure vessel design and spring design.

**Pressure Vessel Design Optimization**

Equation (24) gives the cost function of the pressure vessel design optimization problem as given in [17].

$$Cost(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^3 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (24)$$

The equation is minimized subject to constraints (25-28):

$$f_1(x) = (-x_1 + 0.0193x_3) \leq 0 \quad (25)$$

$$f_2(x) = (-x_2 + 0.00954x_3) \leq 0 \quad (26)$$

$$f_3(x) = \{-(\pi x_3^2 x_4^2) - (\tfrac{4}{3}\pi x_3) + 1296000\} \leq 0 \quad (27)$$

$$f_4(x) = (x_4 - 240) \leq 0 \quad (28)$$

The thickness of the cylinder and head, $x_1$ and $x_2$ respectively are discrete variables and can only take integer multiples of 0.0625 inches while the diameter and length of the vessel, $x_3$ and $x_4$ are continuous variables.

The bounds for $x_1$ and $x_2$ are given by $x_1 \geq 1 \times 0.0625$, $x_2 \leq 99 \times 0.0625$ respectively.

The problem is solved in two regions where:

Region I:$x_4 \leq 200$

Region II:$10 \leq x_3 \leq 200$ and $10 \leq x_4 \leq 240$

Tables-4&5 give the statistical result comparison between the developed algorithm and other metaheuristic-based algorithms in regions I & II respectively.

**Table -4 Result Comparison for Pressure Vessel Design Optimization Problem - Region I**

| **Parameter** | **Optimization Algorithm** | | | | |
|---|---|---|---|---|---|
| | **PSO [18]** | **GA [19]** | **ACO [20]** | **ES [21]** | **ADEBFOA [This Method]** |
| **Cost(x)** | 6059.721 | 6059.946 | 6059.726 | 6059.746 | 6059.719 |
| **f1(x)** | -8.8E-07 | -2.02E-05 | -1.79E-06 | -6.9E-06 | 1.05E-06 |
| **f2(x)** | -0.03588 | -0.03589 | -0.03588 | -0.03588 | -0.03588 |
| **f3(x)** | -521.857 | -546.549 | -521.682 | -518.735 | -524.303 |
| **f4(x)** | -63.363 | -63.346 | -63.362 | -63.359 | -63.364 |
| **x1** | 0.8125 | 0.8125 | 0.8125 | 0.8125 | 0.8125 |
| **x2** | 0.4375 | 0.4375 | 0.4375 | 0.4375 | 0.4375 |
| **x3** | 42.0984 | 42.0974 | 42.0984 | 42.0981 | 42.0985 |
| **x4** | 176.6372 | 176.6541 | 176.6378 | 176.641 | 176.6364 |
| **Best** | **6059.721** | **6059.946** | **6059.726** | **6059.746** | **6059.719** |
| **Mean** | 6440.379 | 6177.253 | 6081.781 | 6850.005 | 6082.570 |
| **Std Dev.** | 448.471 | 130.930 | 67.242 | 426.000 | 45.702 |

**Table -5 Result Comparison for Pressure Vessel Design Optimization Problem - Region II**

| **Parameter** | **Optimization Algorithm** | | | | |
|---|---|---|---|---|---|
| | **PSO [22]** | **FFA [23]** | **HS [24]** | **EA [25]** | **ADEBFOA [This Method]** |
| **Cost(x)** | 5875.166 | 5850.383 | 5852.639 | 5850.383 | 5849.728 |
| **f1(x)** | -0.00340 | -7E-08 | -0.00031 | -7E-08 | -0.00019 |
| **f2(x)** | -0.00595 | -0.00427 | -0.00443 | -0.00427 | -0.00437 |
| **f3(x)** | -506.790 | -521.510 | -523.682 | -521.463 | -41.152 |
| **f4(x)** | -15.910 | -18.635 | -18.388 | -18.635 | -18.586 |
| **x1** | 0.7500 | 0.7500 | 0.7500 | 0.7500 | 0.7500 |
| **x2** | 0.3750 | 0.3750 | 0.3750 | 0.3750 | 0.3750 |
| **x3** | 38.6840 | 38.8601 | 38.8441 | 38.8601 | 38.8504 |
| **x4** | 224.09 | 221.3655 | 221.6125 | 221.3655 | 221.4136 |
| **Best** | **5875.166** | **5850.383** | **5852.639** | **5850.383** | **5849.728** |
| **Mean** | 6032.740 | 5937.338 | 6083.339 | 5925.650 | 5871.985 |
| **Std Dev.** | 315.149 | 164.547 | 140.450 | 150.534 | 44.514 |

**Tension/Compression Spring Design Optimization Problem**

The cost function for the spring design optimization problem is given by equation (29) while equations (30-33) give the associated constraints [17].

$$Cost(x) = (x_3 + 2)x_2x_1^2 \quad (29)$$

$$f_1(x) = \{1 - (\tfrac{x_2^3 x_3}{7178 x_1^4})\} \leq 0 \quad (30)$$

$$f_2(x) = \{(\frac{4x_2^2 - x_1 x_2}{12566\,(x_2 x_1^3) - x_1^4}) + \left(\frac{1}{5108\,x_1^2}\right) - 1\} \leq 0 \quad (31)$$

$$f_3(x) = \{1 - (\frac{140.45 x_1}{x_2^2 x_3})\} \leq 0 \quad (32)$$

$$f_4(x) = \{\frac{(x_1 + x_2)}{1.5} - 1\} \leq 0 \quad (33)$$

The simple bounds for the spring design problem are given by:

$0.05 \leq x_1 \leq 2.0$, $0.25 \leq x_2 \leq 1.3$ and $2.0 \leq x_3 \leq 15.0$

The comparison between the obtained best solution for the developed ADEBFOA algorithm and those obtained by various researchers using other metaheuristic-based algorithms is given in Table-6.

**Table-6 Result Comparison for Tension/Compression Spring Design Optimization Problem**

| **Parameter** | **Optimization Algorithm** | | | | | |
|---|---|---|---|---|---|---|
| | **GA [26]** | **PSO [27]** | **ES [28]** | **DE [29]** | **FFA [30]** | **ADEBFOA [This Method]** |
| **Cost(x)** | 0.012705 | 0.012675 | 0.012698 | 0.012748 | 0.012667 | 0.012666 |
| **f1(x)** | -9.034065 | -9.008948 | -9.018026 | -9.000686 | -9.001002 | -8.990954 |
| **f2(x)** | -0.135661 | -0.134066 | -0.135133 | -0.122109 | -0.134734 | -0.134904 |
| **f3(x)** | -4.026318 | -4.051307 | -4.039301 | -4.149707 | -4.050127 | -4.054598 |
| **f4(x)** | -0.731239 | -0.727085 | -0.728665 | -0.689903 | -0.728850 | -0.728270 |
| **x1** | 0.051480 | 0.051728 | 0.051643 | 0.053862 | 0.051623 | 0.051665 |
| **x2** | 0.351661 | 0.357644 | 0.355360 | 0.411284 | 0.355102 | 0.355930 |
| **x3** | 11.632201 | 11.244543 | 11.397926 | 8.684380 | 11.385602 | 11.331890 |
| **Best Solution** | **0.012705** | **0.012675** | **0.012698** | **0.012748** | **0.012667** | **0.012666** |

The obtained results for the constrained engineering optimization problems studied were very comparable to those obtained from other metaheuristic-based algorithms. The developed Adaptive Differential Evolution/Bacterial Foraging Optimization hybrid algorithm (ADEBFOA) produced better results in both optimization regions (I & II) for the pressure vessel design problem. In region I, the obtained result of 6059.719 was quite close to the true global optimum of 6059.714335048436 obtained using both Mathematical analysis and Lagrange multiplier methods [31]. The developed adaptive hybrid algorithm outperformed the results of the other five metaheuristic techniques as given in Table 4.The results from the developed algorithm for the tension/compression spring design optimization problem were also superior to those from the other five optimization techniques in comparison. However, the obtained result of 0.012666 was very close to that obtained by Yuksel C. & Hakan K. using Firefly Algorithm, that is 0.012667 as given in [30].

## CONCLUSION

This paper presented a new methodology for solving constrained optimization problems. The paper gives a step by step analysis for developing an adaptive hybrid algorithm in which Differential Evolution (DE) & Bacterial Foraging Optimization Algorithm (BFOA) are hybridized and adapted using both Genetic and Swarm Intelligence operators. The developed ADEBFOA algorithm was tested using the Standard Benchmark Functions and produced promising results. The algorithm performed better than other metaheuristic methods in eight of the ten high dimensional functions (F1-F10) used. Having produced promising results on the Standard Benchmark Functions the developed algorithm was tested on constrained engineering optimization problems. The algorithm outperformed other metaheuristic optimization methods in the two constrained engineering problems solved (Pressure vessel design and tension/compression spring design problem). In both test cases, the developed algorithm was able to obtain better feasible solutions in majority of the test runs performed per problem. The results obtained show that the developed adaptive Differential Evolution/Bacterial Foraging Optimization hybrid algorithm (ADEBFOA) performs better in solving most complex constrained optimization problems. The future work of this research study is to use the developed algorithm in solving the highly dimensional, quite complex and non-linear power system expansion optimization problem.

## REFERENCES


[1]. Lee, C.W., Ng S.K.K. and Zhong, J., Value-based transmission expansion planning in deregulated market. *38th North American Power Symposium,* 2006, pp. 107-114.

[2]. Manuel José Costeira da Rocha, *Transmission Expansion Planning, a multiyear approach considering uncertainties,* PhD Dissertation Department of Electrical and Computer Engineering, University of Porto, 2011.

[3]. Sentinella, M. R., Comparison and integrated use of differential evolution and genetic algorithms for space trajectory optimization, *Evolutionary Computation, IEEE Congress,* 2007, pp. 973–978.

[4]. Brian Hegerty, Chih-Cheng Hung, and Kristen Kasprak, *A Comparative Study on Differential Evolution and Genetic Algorithms for Some Combinatorial Problems*, Southern Polytechnic State University, Marietta GA 30060, USA.

[5]. Xiaohui Yan, Yunlong Zhu, Hao Zhang, Hanning Chen, and Ben Niu, An Adaptive Bacterial Foraging Optimization Algorithm with Lifecycle and Social Learning *Hindawi Publishing Corporation, Discrete Dynamics in Nature and Society*, Volume 2012, Article ID 409478.

[6]. Chin-Ling Lee and Cheng-Jian Lin, A Novel Strategy Adaptation Based Bacterial Foraging Algorithm for Numerical Optimization, *International Journal of Swarm Intelligence and Evolutionary Computation,* 2016, ISSN: 2090-4908 SIEC, Volume 5, Issue 1.

[7]. Gong, Y.J., Li, J.J., Zhou Y., Li, Y., Chung H. S.H., Shi Y.H., and Zhang J., Genetic learning particle swarm optimization", *IEEE Transactions on Cybernetics*, 2015.

[8]. Tea Tusar and Bogdan Filipic, *Differential Evolution Versus Genetic Algorithms in Multiobjective Optimization*, Department of Intelligent Systems, Jozef Stefan Institute Jamova 39, Ljubljana, Slovenia.

[9]. Tianlei Zang, Zhengyou He, and Deyi Ye, *Bacterial Foraging Optimization Algorithm with Particle Swarm Optimization Strategy for Distribution Network Reconfiguration*, School of Electrical Engineering, Southwest Jiaotong University, China.

[10]. Goldenberg D. E, *Genetic Algorithm in Search, Optimization and Machine Learning*, Addition-Wesley Publishing Co. Inc, 1989

[11]. Kalyanmoy Deb, *Optimization for engineering design: Algorithm and Examples*, Prentice Hall of India Ltd., New Delhi, 1995

[12]. Hsien-Yu Tseng, *A genetic algorithm for assessing flatness in automated manufacturing Systems*, © Springer Science + Business, 2006.

[13]. Bidar, M., & Kana, H. R., Modified firefly algorithm using fuzzy tuned parameters. *In Fuzzy Systems (IFSC), 13th Iranian Conference on (pp.1-4). IEEE*, 2013

[14]. Momin Jamil and Xin-She Yang, A literature survey of benchmark functions for global optimization problems, *Int. Journal of Mathematical Modelling and Numerical Optimization*, Vol. 4, pp. 150–194, 2013.

[15]. Ali, M. M., Khompartraporn, C., & Zabinsky, Z. B., A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *Journal of global optimization*, 2005. 31(4), 635-672.

[16]. Mahdi Bidar, Samira Sadaoui, Malek Mouhoub& Mohsen Bidar, Enhanced Firefly Algorithm Using Fuzzy Parameter Tuner, *Computer and Information Science;* Vol. 11, No. 1, 2018

[17]. Leticia C. Cagnina, Susana C. Esquivel & Carlos A. Coello Coello, Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer, *Informatica* 32 (2008) 319–326

[18]. L. S. Coelho, Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems, *Expert Systems with Applications*, vol.37, pp.1676–1683, 2010.

[19]. C. A. C. Coello, and E. M. Montes, Constraint- handling in genetic algorithms through the use of dominance-based tournament selection, *Advanced Engineering Informatics*, vol.16, pp.193–203, 2002.

[20]. A. Kaveh, and S. Talatahari, An improved ant colony optimization for constrained engineering design problems, *Engineering Computations*, vol.27, pp.155–182, 2010.

[21]. E. M. Montes, and C. A. C. Coello, An empirical study about the usefulness of evolution strategies to solve constrained optimization problems, *International Journal of General Systems*, vol.37, pp. 443–473, 2008.

[22]. Sulaiman Hassan, Kavi Kumar, Ch Deva Raj & Kota Sridhar, Design and Optimisation of Pressure Vessel Using Metaheuristic Approach, Applied Mechanics and Materials Vols. 465-466, pp 401-406, 2014,*Trans Tech Publications*, Switzerland.

[23]. A. H. Gandomi, X. S. Yang, and A. H. Alavi, Mixed variable structural optimization using firefly algorithm, *Computers & Structures*, vol. 89, pp. 2325–2336, 2011.

[24]. Liqun Gao, Dexuan Zou, Yanfeng Ge & Wenjing Jin, Solving Pressure Vessel Design Problems by an Effective Global Harmony Search Algorithm, *Chinese Control and Decision Conference*, 2010, IEEE.

[25]. G. G. Dimopoulos, Mixed-variable engineering optimization based on evolutionary and social metaphors, *Computer Methods in Applied Mechanics and Engineering*, vol.196, pp.803–817, 2007.

[26]. Carlos, A., Coello, C., Use of a self-adaptive penalty approach for engineering optimization problems, *Computers in Industry* 41(2), 113–127, 2000.

[27]. He, Q., Wang, L., An Effective Co-Evolutionary Particle Swarm Optimization for Constrained Engineering Design Problems,*Engineering Applications of Artificial Intelligence* 20, 89-99, 2007.

[28]. Montes, E. M., Coello C. A. C., An Empirical Study about the usefulness of Evolution Strategies to solve Constrained Optimization Problems,*International Journal of General Systems* 37, 443-473, 2008.

[29]. Raj, K. H., Sharma, R. S., Mishra, G. S., Dua, A. & Patvardhan, C., An Evolutionary Computational Technique for Constrained Optimization in Engineering Design, *Journal of the Institution of Engineers India Part Me Mechanical Engineering Division* 86,121-128, 2005.

[30]. Yuksel Celik & Hakan Kutucu, *Solving the Tension/ Compression Spring Design Problem by an Improved Firefly Algorithm*, Karabuk University, Department of Computer Engineering, Turkey.

[31]. Xin-She Yang, Christian Huyck, Mehmet Karamanoglu& Nawaz Khan, *True Global Optimality of the Pressure Vessel Design Problem: A Benchmark for Bio-Inspired optimization Algorithms*, School of Science and Technology, Middlesex University, The Burroughs, London, UK, 2014.