The Find-Fix-Finish-Exploit-Analyze (F3EA) Meta-Heuristic Algorithm with an Extended Constraint Handling Technique for Constrained Optimization

Ali Husseinzadeh Kashan, Alireza Balavand, and Somayyeh Karimiyan

Abstract As a novel population-based evolutionary algorithm, the Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm has been introduced for numerical optimization which develops new selection operators, a new parameter-free mutation operator, and a local search operator. The algorithm takes the surface of the objective function as the battleground and mimics the F3EA targeting process of object or installation selection for destruction in the warfare. It performs the main steps of Find-Fix-Finish-Exploit-Analyze (F3EA) in an iterative manner, wherein the Find step introduces a new individual selection mechanism via imitating the military radar detection rationale; in the Fix step, it is shown that how the reality of the target monitoring process can be transformed to a single variable constrained optimization problem to obtain a local search operator; In the finish step, new solutions are generated via a new adaptive mutation operator which is developed through the simulation of projectile motion using physics equations; in the exploit step it tries to take over opportunities presented by the generated potential solution and other members of the population; Finally, in analyze step, the population is updated. In this chapter, an extended epsilon constrained handling technique is used to handle constraints within the body of F3EA meta-heuristic algorithms that is called the εF3EA. In order to evaluate the efficiency of the F3EA algorithm, nine constraint benchmark functions are used. In addition, the εF3EA algorithm is compared with eight other algorithms, which included SAFFa, COPSO, ISR, ATMES, SMES, ECHT-EP2, HCOEA, and αSimplex. Results indicate that the algorithm is the best of all on benchmark test problem instances.

A. H. Kashan (✉)
Faculty of Industrial and Systems Engineering, Tarbiat Modares University, Tehran, Iran
e-mail: a.kashan@modares.ac.ir

A. Balavand
Department of Industrial Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
e-mail: a.balavand@srbiau.ac.ir

S. Karimiyan
Department of Civil Engineering, Islamshahr Branch, Islamic Azad University, Islamshahr, Iran
e-mail: s_karimiyan@iiau.ac.ir

A. J. Kulkarni et al. (eds.), *Constraint Handling in Metaheuristics and Applications*,
https://doi.org/10.1007/978-981-33-6710-4_1

Keywords Constrained optimization · Constraint-handling techniques · F3EA · Constraint violation

1 Introduction

Calculation of the optimum solutions for most of the optimization problems is a hard and complex process. In practice, these solutions are achieved by heuristics and meta-heuristic algorithms. Metaheuristic algorithms cover a set of approximate optimization techniques that have been developed for the past three decades. Metaheuristic methods find acceptable solutions in a reasonable time for engineering and science NP-hard problems. Unlike the exact optimization algorithm, metaheuristic algorithms do not guarantee that obtained solutions are optimum solutions, but most of the time, near the optimum solutions, are found by them. On the other hand, an optimization algorithm provides satisfactory results, and the same algorithm may have poor performance in other problems [1] especially in the constrained optimization problems. Most real-world optimization problems have constraints and usually, their decision variables are more than classic functions that increase the dimension of the problem. When constraints are added to the problem, the solution space has been more limited, and feasible solution space will be changed. So, they are more complex rather than classic optimization problems which are a challenge for meta-heuristic algorithms in recent years [2–6]. Constrained optimization problems usually simulate real-world optimization problems. Different categories of algorithms have been performed to solve the constrained optimization problem with different methods. A constrained optimization problem with inequality constraints, equality constraints, lower bound constraint, and upper bound constraint is shown as (1).

$$\begin{aligned} &\min f(x) \\ &s.t g_i(x) \leq 0 i = 1, \ldots, q \\ &h_i(x) = 0 i = q+1, \ldots, m \\ &l_i \leq x_i \leq u_i i = 1, \ldots, n, \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)$ is a vector of the decision variables, $f(x)$ is an objective function, $g_i(x) \leq 0$, $h_i(x) = 0$ are q inequality constraints and $m - q$ equality constraints,. The constraints in optimization problems increase the complexity and no longer a combination of some solutions is not feasible. There are a lot of constrained optimization problems in the field of engineering design [7–11]. The handling constraints in the constrained optimization problems is very important. Because choosing the appropriate Bound Constraint Handling Method (BCHM) makes that the optimization algorithms achieve the most efficiency in most of the problems. So, different methods have been proposed in the field of BCHM [12]. There are three methods with different approaches to handling constraints in the constrained optimization problem [13]. In the following, these methods will be explained.

Penalty functions: One of these methods includes the penalty function method [14, 15]. The basic idea of the penalty function is to transfer constraints to the objective function with a penalty factor and transform the optimization problem into an unconstrained problem [15]. According to (2), in the penalty function method, the values of constraints violation $\left(v_{gi}, v_{hj}, v_{kl}\right)$ are added to the objective functions and as many as the constraints violation are higher, the value of the objective function will be increased. It is important to mention that determining the penalty factor is an optimization problem and depends on the problem [15].

$$\min f(x)$$

s.t.

$$\begin{aligned} g_i(x) \geq g_0 \quad \forall i, \quad v_{gi} &= \max\left(1 - \frac{g_i(x)}{g_0}, 0\right) \\ h_j(x) \leq h_0 \quad \forall i, \quad v_{hj} &= \max\left(\frac{h_j(x)}{h_0} - 1, 0\right) \\ k_l(x) = k_0 \quad \forall, \quad k v_{kl} &= \left|\frac{k_l(x)}{k_0} - 1\right| \end{aligned} \tag{2}$$

The basic idea of the penalty function is shown in (3). According to this equation, the value of v_{gi} is added to $f(x)$ when the $g_i(x) \geq g_0$, v_{hj} is added to $f(x)$ when $h_j(x) \leq h_0$, and when $k_l(x) \geq k_0$, v_{kl} will be added to $f(x)$. How adding the penalty function to the objective function is formulated based on (2). One of the critical challenges in the penalty function is how determining the appreciated value of r [15]. The value of r is changed for each constrained optimization problem and determined to appreciate the value of r can have a great impact to reach the optimum value of decision variables.

$$v(x) = f(x) + \sum_{\forall i} r_i \times v_{gi} + \sum_{\forall j} r_j \times v_{hj} + \sum_{\forall l} r_l \times v_{kl} \tag{3}$$

Repair methods: in this method, a generated feasible solution is replaced instead of an infeasible solution. After generating infeasible solutions, this solution has been repaired and replaced instead of an infeasible one.

Feasibility preserving genetic operators: in this method, the operators of the genetic algorithm always generate feasible individuals. According to this method, the mutation and crossover are done in a feasible area of solution space and as a result, generated solutions are feasible.

The first idea of handle constraint was proposed in the form of the penalty function method in the DE algorithm. There are different types of penalty functions [16]. The death penalty, quadratic penalty, and substitution penalty. In the death penalty, a constant large value is added to the infeasible individual that makes the value of fitness increase. While the square of constraints violations and fitness value of the

repaired solution is the fitness of additive quadratic penalty method. The last method is the substitution quadratic penalty which is the combination of the death penalty and quadratic penalty. The feasibility is not computed for the feasible individual. Instead, the fitness values are the sum of the squared distance from the exceeded bound and a large value [12].

In this paper, an extended epsilon constrained handling technique is proposed to handle constraints within the body of F3EA meta-heuristic algorithm that is called εF3EA. In the epsilon constraint method, the violation is defined as the sum of all the constraint violations $\emptyset(x)$ [17] based on the Eq. (4):

$$\emptyset(x) = \sum_{\forall i} {v_{gi}}^p + \sum_{\forall j} {v_{hj}}^p + \sum_{\forall l} {v_{kl}}^p \tag{4}$$

The εF3EA includes several steps of generating random solutions, initialization of the ε level, ranking the solutions, the fix step, the Find step, the Finish step, the Exploit step, the Analyze step, and checking stopping criteria. In generating random solutions step, random solutions are generated with uniform distribution and create the initial population. In the initialization of the ε level step, the ε value is defined. In the ranking of the solutions, the population is sorted based on constraint violations and function values. The Fix step, the Find step, the finish step, and the exploit step are implemented based on the body of the F3EA algorithm. In the analyze step, lexicographic order is used which will be explained. So, the proposed algorithm will be described in Sect. 3. In Sect. 4, the results of using the proposed method are presented on some real-world problems and will be compared with state of art methods, as well as the εF3EA algorithm is examined, and the conclusion is presented in Sect. 5.

2 Background

In the ε constraint handling method, the constraint violation is defined based on the sum of all violations of constraint in which if $violation > \varepsilon$, the solution is not feasible and the worth of this point is low. In this method, $\emptyset(x)$ precedes $f(x)$ which means the importance of feasibility is more than the objective function value of $f(x)$. The ε levels have adjusted the value of precedence.

2.1 ε Constrained Method

Assume that f_1, f_2 are the values of the objective function and $\emptyset_1$, $\emptyset_2$ are constraint violations based on solution x_1 and x_2. ε levels comparison for $\varepsilon > 0$ in $(f_1, \emptyset_1)$, $(f_2, \emptyset_2)$ are defined as (5) and (6):

$$(f_1, \emptyset_1) <_\varepsilon (f_2, \emptyset_2) \leftrightarrow \begin{cases} f_1 < f_2, \text{ if } \emptyset_1, \emptyset_2 \leq \varepsilon \\ f_1 < f_2, \text{ if } \emptyset_1 = \emptyset_2 \\ \emptyset_1 < \emptyset_2, \text{ otherwise} \end{cases} \tag{5}$$

$$(f_1, \emptyset_1) \leq_\varepsilon (f_2, \emptyset_2) \leftrightarrow \begin{cases} f_1 \leq f_2, \text{ if } \emptyset_1, \emptyset_2 \leq \varepsilon \\ f_1 \leq f_2, \text{ if } \emptyset_1 = \emptyset_2 \\ \emptyset_1 < \emptyset_2, \text{ otherwise} \end{cases} \tag{6}$$

The lexicographic order is used when $\varepsilon = 0$, $\varepsilon < 0$ and $\varepsilon \leq 0$ in which constraint violation $\emptyset(x)$ precedes the objective value of $f(x)$. The ordinal comparisons between function values are used in the case of $\varepsilon = \infty$. In the constraint handling method, the constraints transfer into objective function and a constrained optimization problem converts into an unconstrained optimization problem using ε level comparisons. So, constrained optimization problems can be solved by incorporating ε level comparisons. Ordinary comparisons with ε level comparisons are used for comparing until the value of ε converges to 0. By reaching ε to 0, the feasible solution has been reached. On the other hand, if the violation of a point is greater than ε, the point is infeasible and its worth is low.

The ε constrained optimization method converts a constrained optimization problem into an unconstrained one with ε level comparisons that use order relation for replacing. An optimization problem based on ε a constrained optimization method is defined as (7):

$$\begin{aligned} &minimize\ f(x) \\ &subject to\ \emptyset(x) \leq \varepsilon \end{aligned} \tag{7}$$

It is obvious that by converging ε to 0, the optimal solution obtained as well as in the penalty function method the optimal solution obtained by increasing the penalty coefficient to infinity.

2.2 The F3EA Algorithm

In the field of population-based evolutionary algorithm, the Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm has been introduced for numerical optimization which develops new selection operators, new parameter-free mutation operator, and a local search operator [6]. The algorithm takes the surface of the objective function as the battleground and mimics the F3EA targeting process of object or installation selection for destruction in the warfare. This algorithm includes five main steps of the Find, Fix, Finish, Exploit, and Analyze step. The Find step mimics based on the object detection process follow by military radars. In the Fix step, the highest peak is found by a local heuristic method. In the Finish step, the intended target is destroyed. In the Exploit step contains information gathering from the target

area. Finally, the Analyze step relates to the results of the attack on the target to identify more targeting opportunities [18]. In this chapter, an extended epsilon constrained handling technique is used to handle constraints by using F3EA meta-heuristic algorithm that is called εF3EA. In the following, the structure of εF3EA will be explained.

3 The F3EA Adapted to Constrained Optimization

The F3EA is a stochastic direct search method based on the population. F3EA has been successfully performed in some nonlinear, multimodal, and constrained engineering design problems. The results have been shown that F3EA is fast and has good convergence in these test functions [6]. F3EA has used the penalty function method to solve the constrained optimization problem so far. In this paper to develop the performance of F3EA for solving the constrained optimization problem, epsilon constrained handling with an automatic update of ε is proposed. The algorithm of the εF3EA is as follows:

Step 1: **generating random solutions** by which the initial population is randomly generated.

Step 2: **initialization of the ε level** by which an initial ε level is generated as $\varepsilon(0)$.

Step 3: **ranking the solutions**. by which the best members of the population are selected based on the best value of the constraint violations and objective function value, and the rank of individuals is defined by the ε level comparison.

Step 4: **the Fix step**. In this step, the highest peak is found by $fminbnd$ function which is a function for optimization in the Matlab toolbox. In order to destroy the military tank, the projectile launch angle must be located in the highest peak position. So, the highest peak position is found by $fminbnd$ as a local optimum.

Step 5: **the Find step**. In this step, a new solution is generated as p_i^t (solution i at iteration t) which temporarily considers an artificial radar and other solutions (p_j^t) consider military facilities that may or may not be detected by the artificial radar. Then the distance between p_i^t parent solution i at iteration t and child solution j at iteration t (p_j^t) is calculated as Eq. (8):

$$R_{ij}^t = \left| f\left(p_i^t\right) - f\left(p_j^t\right) \right| \tag{8}$$

This equation $f\left(p_i^t\right)$ shows the value of the objective function of p_i^t and $f\left(p_i^t\right)$ shows the value of the objective function of p_j^t that show in Eq. (9).

$$R_{maxij}^t = \left(f_{max}^t - f_{min}^t\right) \times u\left(p_i^t\right) \times \sqrt[4]{\frac{\left(1 - u\left(p_j^t\right)\right) \exp\left(-au\left(p_j^t\right)\right)}{u\left(p_i^t\right)}}, i \neq j. \tag{9}$$

where R^t_{maxij} is the maximum range and based on the above equation, if $R^t_{ij} \leq R^t_{maxij}$, p^t_j is detectable by p^t_i. Among detectable solutions p^t_i, one of the solutions will be selected in the Finish step.

Step 6: **the Finish step**. In this step, new solutions are generated via a new adaptive mutation operator which is developed through the simulation of projectile motion using physics equations. Assume that p^t_i impersonates the position of a projectile launcher and p^t_j shows the position of a target facility. E^t_i is generated by p^t_i and p^t_j which is considered as the explosion position of the artificial projectile launched from p^t_i toward p^t_j. How generating E^t_i is shown in Eq. (10).

$$E^t_i = p^t_i + x\left(T^t_{ij}\right)\frac{\left(p^t_j - p^t_i\right)}{p^t_j - p^t_i} + z\left(T^t_{ij}\right)\frac{z^t_{ij}}{z^t_{ij}} \tag{10}$$

$x\left(T^t_{ij}\right)$ and $z\left(T^t_{ij}\right)$ are the projected explosion position on the x-axis and z-axis, respectively, which is defined as Eqs. (11) and (12):

$$x\left(T^t_{ij}\right) = w^t_{x_{ij}} T^t_{ij} + m^t_{ij}\left(v^t_{0_{ij}} \cos\alpha^t_{ij} - w^t_{x_{ij}}\right)\left(1 - e^{-T^t_{ij}/m^t_{ij}}\right) \tag{11}$$

$$z\left(T^t_{ij}\right) = w^t_{z_{ij}} T^t_{ij} - m^t_{ij} w^t_{z_{ij}}\left(1 - e^{-T^t_{ij}/m^t_{ij}}\right) \tag{12}$$

Based on these equations, α^t_{ij} is a lunch angel, $v^t_{0_{ij}}$ is the initial velocity of the projectile, the mass of projectile is m^t_{ij}, wind vector is w^t_{ij}, and z^t_{ij} is perpendicular to the hyper-line connecting p^t_i and p^t_j. In order to the further investigate the Finish step please refer to [6].

Step 7: **the Exploit step**. In the Exploit step, the information gathering from the target area is done in this step. This step utilizes the result of the Finish step because the result of the Finish step may have premature convergence. So, c^t_i shows the number of changes made in p^t_i and truncated geometric distribution [19] is used for simulating c^t_i as Eq. (13):

$$c^t_i = \left\lceil \frac{\ln\left(1 - \left(1 - \left(1 - q^t_i\right)^n\right) rand(0,1)\right)}{\ln\left(1 - q^t_i\right)} \right\rceil, c^t_i \in \{1, 2, \ldots, n\} \tag{13}$$

where $q^t_i < 1, q^t_i \neq 0$. By the above equation, c^t_i the number of dimensions of E^t_i is randomly selected which are assigned by U^t_i. U^t_i is the output of the Exploit step.

Step 8: **Analyze step**. In the Analyze step, the new solution is analyzed that way a comparison between $f\left(U^t_i\right)$ and $f\left(p^t_i\right)$ is performed and if $f\left(U^t_i\right)$ is better than $f\left(p^t_i\right)$, $f\left(U^t_i\right)$ will be replaced. In this step, the comparison between solutions is based on two values of $\emptyset(x)$ and $f(x)$ value. Given that $\emptyset(x)$ precedes $f(x)$, So, at first, the value of $\emptyset(x)$ is checked and if the value of $\emptyset(x)$ is zero, then the current

solution is compared based on $f(x)$ value. $f\left(U_i^t\right)$ is replaced instead of G_{best}^t, if in terms of violation and objective function value, $f\left(U_i^t\right)$ is better than $f\left(G_{best}^t\right)$. The process of replacing in this step is defined as (14) and (15):

$$(f_i, \emptyset_i) <_\varepsilon (G_{best}, \emptyset_{best}) \leftrightarrow \begin{cases} f_i < G_{best}; if\ \emptyset_i, \emptyset_{best} \leq \varepsilon \\ f_i < G_{best}; if\ \emptyset_i = \emptyset_{best} \\ \emptyset_i < \emptyset_{best}; otherwise \end{cases} \tag{14}$$

$$(f_i, \emptyset_i) \leq_\varepsilon (G_{best}, \emptyset_{best}) \leftrightarrow \begin{cases} f_i \leq G_{best}; if\ \emptyset_1, \emptyset_2 \leq \varepsilon \\ f_i \leq G_{best}; if\ \emptyset_1 = \emptyset_2 \\ \emptyset_i < \emptyset_{best}, ; otherwise \end{cases} \tag{15}$$

In the following, the pseudocode of εF3EA will be explained based on Fig. 1.

In order to create an initial population of the F3EA, a set of random variables is generated between the lower and upper bounds. Given that feasibility always precedes objective function value, at first the constraint violation of the current population is checked and then the solution is compared based on objective function value. If the constraint violation is smaller than the epsilon value and the objective function value is smaller than the global best, the current solution is replaced as shown in (15), the global best solution is updated and the algorithm enters the main loop.

```
1- generate a set of initial random population and set initial parameters
2- if constraint violation<epsilon threshold
3-    if objective function value<global best value
4-       update global best
5- calculate R_ij^f, R_ij^cv
6- while FE<FEmax
7-    perform the fix step
8-       if constraint violation<epsilon threshold
9-          if objective function value<global best value
10-            update global best
11-   perform the find step
12-      calculate R_max^f, R_max^cv and p_i, p_j
13-   for offspring=1:N0
14-      perform the finish step
15-         calculate E_i^f, E_i^cv
16-      perform the exploit step
17-         perform crossover methods and generate offspring
18-            compare offspring with their parent in analyze step
19-               if constraint violation<epsilon threshold
20-                  if objective function value<global best value
21-                     update global best
22-   end for
24-   update epsilon threshold
25- check stopping criteria
26-    if stopping criteria has not been met
27-       return to line 7
28-       save results
28-    else
29-       break
```

Fig. 1 The pseudocode of εF3EA

In the Fix step, the new solution is generated based on $fminbnd$ [20] function that this function is one of the classic optimization methods available in the Matlab optimization toolbox. After generating a new solution in the Fix step, the feasibility of the solution is checked and this solution is replaced global best if the better solution is found based on (15). The values of R_{maxij}^{cv} and R_{maxij}^{ofv} are calculated in the Find step. In original paper of F3EA R_{maxij}^{cv} is calculated for all population based on Eq. (9), but in εF3EA approach the constraint violation (cv) and objective function value (ofv) are calculated for all solutions in population, therefore, the values of R_{maxij}^{cv} and R_{maxij}^{ofv} are calculated as follow:

$$\begin{cases} R_{maxij}^{ofv} = \left(f_{max}^{t} - f_{min}^{t}\right) \times u\left(p_i^{tobv}\right) \times \sqrt[4]{\dfrac{\left(1-u\left(p_j^{tobv}\right)\right)\exp\left(-au\left(p_j^{tobv}\right)\right)}{u\left(p_i^{tobv}\right)}}, i \neq j.ifcv < ep \\ R_{maxij}^{cv} = \left(cv_{max}^{t} - cv_{min}^{t}\right) \times u\left(p_i^{tcv}\right) \times \sqrt[4]{\dfrac{\left(1-u\left(p_j^{tcv}\right)\right)\exp\left(-au\left(p_j^{tcv}\right)\right)}{u\left(p_i^{tcv}\right)}}, i \neq j.ifcv > ep \end{cases} \tag{16}$$

According to Eq. (16), cv_{max}^{t} and cv_{min}^{t} are the maximum value of constraint violation of population and the minimum value of constraint violation of population, respectively. p_i^{tobv} is the value of the objective function of population ith and p_i^{tcv} is the value of the constraint violation of population ith. In this approach, if constraint violation (cv) of population ith is smaller than epsilon threshold (ep), the above section of Eq. (16) is performed, otherwise, the bottom section is performed. Finding detectable solutions is done in the Find step, that way if $R_{ij}^{tofv} \leq R_{maxij}^{ofv}$, p_j^t is detectable by p_i^t. This process is also performed based on R_{maxij}^{cv}. Mutation is done in the finish step. In this step to increase the exploitation power of εF3EA, the offspring of each selected solution from the find step are generated. This idea is taken from LCA [3]. In order to identify the number of offspring, the following Eq. (17) is used.

$$N0 = 2 \times 0.1 \times N \tag{17}$$

As mentioned before in the εF3EA algorithm each solution is investigated by two criteria of objective function value and constraint violation. This method has been also used in the Finish step. That means all of the solutions are once compared based on constraint violation and then are compared based on objective function value. Thus, this process is one of the differences between εF3EA and F3EA. Generating a new solution in the Finish step depends on p_i^t, p_j^t. Assume that p_i^t impersonates the position of the projectile launcher and p_j^t simulates the position of a target facility. E_i^t is the position of explosion that has been thrown from p_i^t toward p_j^t. Let $\left[p_i^t, u\left(p_i^{tobv}\right)\right]$, $\left[p_i^t, u\left(p_i^{tcv}\right)\right]$, $\left[p_j^t, u\left(p_j^{tobv}\right)\right]$ and $\left[p_j^t, u\left(p_j^{tcv}\right)\right]$. In order to simulate the launch, it is needed to the inclination (β_{ij}), launch angle (β_{ij}), the projectile initial velocity (v_{0ij}), and distance between p_i^t and p_j^t. All of the Finish stage has been explained in [6].

In the Finish step, the mutation is done based on Eq. (10) with the difference that each parent has the number of $N0$ offspring. After generating E_i^t in each iteration, the

crossover method is done on E_i^t in the exploit step. There are two kinds of crossover in εF3EA that one of them is related to Eq. (13) and another crossover will be explained in the following:

- To the number of population, a permutation vector was generated as (p).
- The first three members of p were chosen as x^1, x^2, x^3 in each iteration of the current population.
- x_{ap}^t was created based on Eq. (18):

$$x_{ap}^{t+1} = \frac{\left(x^{1,t}.x^{2,t}\right)}{x^{3,t}} \tag{18}$$

$x^{1,t}$ ، $x^{2,t}$, and $x^{3,t}$ represent the first, second, and third members of the vector p in the iteration t, respectively. Equation (18) improved the quality and diversity of the solutions produced. In the exploit step for each parent, the $N0$ offspring are generated by crossover methods and all of the offspring are compared with their parents. All of the generated solutions are compared based on Eq. (15) in analyze step which means at first, the constraint violation, and then the objective function value is checked. If a solution is better than the global best, the global best is updated. Finally, the value of epsilon becomes smaller in each iteration that the value of epsilon is calculated as Eq. (19):

$$\varepsilon = \varepsilon \times \left(1 - \frac{FE}{Fmax}\right)^{\delta} \tag{19}$$

According to Eq. (19), FE is the number of function evaluations, $Fmax$ is the maximum number of function evaluations and δ is a constant number and has been considered 50. This process makes the value of epsilon is converged to zero that means that in the initial iterations the solution with positive constraint violation is accepted and as many as the algorithm approach the final iterations, the constraint violation of all solution should be zero and the solution are compared based on objective function value.

If the number of functions evaluation (FE) exceeds the maximum number of functions evaluation (FE_{max}), the algorithm is stopped.

4 Experiments

In this section, the εF3EA performs to evaluate the benchmark constrained optimization problems. These benchmark functions include 13 well-studied problems (g01–g09) of the CEC 2006 test suite [21], where we compare εF3EA with the result of a study that has been written by Husseinzadeh Kashan [3]. In that study,

the league champion algorithm (LCA) [22] has been compared with eight state of the art algorithms. G01 to g09 has some features of the number of decision variables (n), estimates the ratio between the feasible region and the entire search space (ρ), the number of linear inequalities, the number of nonlinear inequalities, the number of linear equalities, and the number of nonlinear equalities. In this study, the εF3EA will be compared with eight other algorithms of SAFFa [23], COPSO [24], ISR [25], ATMES [26], SMES [27], ECHT-EP2 [28], HCOEA [29], and αSimplex [30].

In order to solve the benchmark functions, it is necessary to explain that with 225,000 number of function evaluations, εF3EA almost finishes its search, since all individuals converge to the global optimum with acceptable precision.

4.1 G01 Benchmark Function

According to (20), the objective function is a Quadratic function with 13 decision variables. Estimating the ratio between the feasible region and the entire search space (ρ) is 0.0111. The number of linear inequalities constraints is nine, this function does not have the linear inequalities constraints, in nonlinear equalities constraints, and nonlinear equalities constraints. Constraints of g1, g2, g3, g7, g8, and g9 are active.

$$Minimize\ f(X) = 5\sum_{d=1}^{4} x_d - 5\sum_{d=1}^{4} x_d^2 - \sum_{d=5}^{13} x_d$$

subject to:

$$\begin{aligned} g_1(X) &= 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0 \\ g_2(X) &= 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0 \\ g_3(X) &= 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0 \\ g_4(X) &= -8x_1 + x_{10} \leq 0 \\ g_5(X) &= -8x_2 + x_{11} \leq 0 \\ g_6(X) &= -8x_3 + x_{12} \leq 0 \\ g_7(X) &= -2x_4 - x_5 + x_{10} \leq 0 \\ g_8(X) &= -2x_6 - x_7 + x_{11} \leq 0 \\ g_9(X) &= -2x_8 - x_9 + x_{12} \leq 0 \end{aligned} \tag{20}$$

In Table 1, BV is the best value, AV is the average value, WV is the worst value and STD shows the standard deviation value. The results of Table 1 show that there is no significant difference between BV, AV, WV, and STD. The best result is related to the εF3EA, SMES, COPSO, and SAFFa algorithms.

Table 1 Comparative results of εF3EA with other algorithms in g01 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−15.0000	−15.000000	−15.000	−15.000	−15.0000	−15.0000	−15.000	−14.9999998	−15
AV	−15.0000	−15.000000	−15.000	−15.000	−15.0000	−15.0000	−15.000	−14.9999998	−15
WV	−15.0000	−15.000000	−15.000	−15.000	−15.0000	−15.0000	−15.000	−14.9999998	−15
STD	0	0	5.8E−14	1.6E−14	0	0.00E+00	4.3E−7	6.4E−06	0

4.2 G02 Benchmark Function

This function is nonlinear which has 20 variables based on (21). The value of ρ is 99.8474. The number of linear inequalities constraints, linear inequalities constraints, nonlinear equalities constraints, and nonlinear equalities constraints are 1, 1, 0, 0, respectively. The constraint g1 is close to being active.

$$Minimize\ f(X) = -\left|\left(\sum_{d=1}^{n}\cos^4(x_d) - 2\prod_{d=1}^{n}\cos^2(x_d)\right) \Big/ \sqrt{\sum_{d=1}^{n} dx_d^2}\right|$$

subject to:

$$g_1(X) = 0.75 - \prod_{d=1}^{n} x_d \le 0$$

$$g_2(X) = \sum_{d=1}^{n} x_d - 0.75n \le 0 \tag{21}$$

where $n = 20$ and $0 \le x_d \le 1$. The best known objective value is $f(X^*) = -0.80361910412559$.

The results in Table 2 show that the COPSO, ISR, αSimplex, and ECHT-EP2 have had the best value among the algorithms. Whatever the value of AV, WV, and STD is less, it means that the performance of the algorithm is better. Accordingly, in terms of the AV, WV, and STD, the εF3EA has had the best values among eight other algorithms.

4.3 G03 Benchmark Function

The g03 function is Polynomial with 10 decision variables in which the value of ρ is zero. It has only one equality nonlinear constraints. The equation of g03 is as (22):

$$Minimize\ f(X) = -(\sqrt{n})^n \prod_{d=1}^{n} x_d$$

subject to:

$$h(X) = \sum_{d=1}^{n} x_d^2 - 1 = 0 \tag{22}$$

where $n = 10$ and $0 \le x_d \le 1$ $(d = 1, \ldots, n)$. The global objective value is $f(X^*) = -1$.

Table 2 Comparative results of εF3EA with other algorithms in g02 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−0.80297	−0.803619	−0.803619	−0.803388	−0.803601	−0.8036191	−0.803241	−0.8036191	−0.8036069
AV	−0.79010	−0.801320	−0.782715	−0.790148	−0.785238	−0.7998220	−0.801258	−0.7841868	−0.8035734
WV	−0.76043	−0.786566	−0.723591	−0.756986	−0.751322	−0.7851820	−0.792363	−0.7542585	−0.803489
STD	1.2E−02	4.59E−03	2.2E−02	1.3E−2	1.67E−2	6.29E−3	3.8E−03	1.3E−02	2.6324E-05

Given that the g03 function has less complexity compared to other benchmarks, the difference values between functions are not significant based on the results of Table 3. But in terms of the STD, the εF3EA has had the best performance compared to other results in this table and it means that in all iterations, the εF3EA has reached the optimal solution.

4.4 G04 Benchmark Function

The g04 function is quadratic with five decision variables. Estimating the ratio between the feasible region and the entire search space (ρ) is 52.1230. There are only six nonlinear equalities constraints. Constraints g1 and g6 are active which its equation is based on the (23):

$Minimize\ f(X) = 5.3578547x_3^2 + 08356891x_1x_5 + 37.293239x_1 - 40792.141$
subject to:

$$\begin{aligned}
g_1(X) &= 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \le 0\\
g_2(X) &= -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \le 0\\
g_3(X) &= 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \le 0\\
g_4(X) &= -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \le 0\\
g_5(X) &= 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \le 0\\
g_6(X) &= -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \le 0
\end{aligned} \tag{23}$$

With boundary conditions $78 \le x_1 \le 102$, $33 \le x_2 \le 45$, and $27 \le x_d \le 45 (d = 3, 4, 5)$. The optimum objective value is $f(X^*) = -30665.539$.

Table 4 shows the superior performance of all algorithms in the g04 function. The ATMES and εF3EA have had the best performance compared to other algorithms.

4.5 G05 Benchmark Function

The G05 benchmark is a cubic function in which $\rho = 0$. According to (24), this function has two linear inequalities constraints and three nonlinear equalities constraints. The equation of g05 has defined as follows:

$$Minimize\ f(X) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$$

subject to:

$$\begin{aligned}
g_1(X) &= -x_4 + x_3 - 0.55 \le 0\\
g_2(X) &= -x_3 + x_4 - 0.55 \le 0
\end{aligned}$$

Table 3 Comparative results of εF3EA with other algorithms in g03 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−1.00000	−1.000005	−1.001	−1.000	−1.000	−1.0005	−1.000	−1.0005001	−1.0000
AV	−0.99990	−1.000005	−1.001	−1.000	−1.000	−1.0005	−1.000	−1.0005001	−1.0000
WV	−0.99970	−1.000003	−1.001	−1.000	−1.000	−1.0005	−1.000	−1.0005001	−1.0000
STD	7.5E−05	3.16E−07	8.2E−09	5.9E−5	2.09E−4	0.00E+00	1.3E−12	8.5E−14	0

Table 4 Comparative results of εF3EA with other algorithms in g04 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−30665.50	−30665.538	−30665.539	−30665.539	−30665.539	−30665.538	−30665.539	−30665.538	−30665.539
AV	−30665.20	−30665.538	−30665.539	−30665.539	−30665.539	−30665.5387	−30665.539	−30665.538	−30665.539
WV	−30663.30	−30665.538	−30665.539	−30665.539	−30665.539	−30665.538	−30665.539	−30665.538	−30665.539
STD	4.85E−01	0	1.1E−11	7.4E−12	0	0.00E+00	5.4E−7	4.2E−11	1.0042E−11

$$h_3(X) = 1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0$$
$$h_4(X) = 1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$$
$$h_5(X) = 1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0 \tag{24}$$

With boundary conditions $0 \leq x_1 \leq 1200$, $0 \leq x_2 \leq 1200$, $-0.55 \leq x_3 \leq 0.55$, and $-0.55 \leq x_4 \leq 0.55$. The best known objective value is $f(X^*) = 5126.49671$.

In Table 5, the comparative results of all the algorithms have been shown for solving the g05 problem. These results show that all algorithms have had good results that except the SAFFa, the COPSO, and the SMES, other algorithms have reached the optimal solution.

4.6 G06 Benchmark Function

The g06 is a kind of cubic function with two decision variables and two nonlinear inequalities constraints. Estimating the ratio between the feasible region and the entire search space is 0.0066. in this function, all constraints are active. Its equation is based on (25):

$$Minimize\ f(X) = (x_1 - 10)^3 + (x_2 - 20)^3$$

subject to:

$$g_2(X) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0 \tag{25}$$

With boundary conditions $13 \leq x_1 \leq 100$, and $0 \leq x_2 \leq 100$. The optimum objective value is $f(X^*) = -6961.8139$.

Given that the g05 is two dimensions function and it has only two constraints, the comparative results are not a significant difference according to the results of Table 7. COPSO, αSimplex, ECHT-EP2, and εF3EA have had the best values. The values of AV and WV are similar to each other in four functions. But in terms of STD, the COPSO is outperformed.

4.7 G07 Benchmark Function

According to (26), it is obvious that g07 has 10 decision variables and is quadratic. The value of ρ is 0.0003. It has three linear inequalities constraints and five nonlinear inequalities constraints. The constraints g1, g2, g3, g4, g5, and g6 are active.

$$\text{Minimize } f(X) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 +$$

Table 5 Comparative results of εF3EA with other algorithms in g05 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP3	HCOEA	αSimplex	εF3EA
BV	5126.989	5126.498096	5126.497	5126.497	5126.599	5126.4967	5126.498	5126.4967	5126.4967
AV	5432.08	5126.498096	5126.497	5127.648	5174.492	5126.4967	5126.498	5126.4967	5126.4967
WV	6089.43	5126.498096	5126.497	5135.256	5304.167	5126.4967	5126.498	5126.4967	5126.4967
STD	3.89E+03	0	7.2E−13	1.80E+00	5.00E+01	0.00E+00	1.7E−7	3.5E−11	4.13E−13

Table 6 Comparative results of εF3EA with other algorithms in g06 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−6961.800	−6961.8139	−6961.814	−6961.814	−6961.814	−6961.8139	−6961.814	−6961.8139	−6961.8139
AV	−6961.800	−6961.8139	−6961.814	−6961.814	−6961.284	−6961.8139	−6961.814	−6961.8139	−6961.8139
WV	−6961.800	−6961.8139	−6961.814	−6961.814	−6952.482	−6961.8139	−6961.814	−6961.8139	−6961.8139
STD	0	0	1.9E−12	4.6E−12	1.85E+0	0.00E+00	8.5E−12	1.85E−10	1.85E−12

$$(x_5-3)^2+2(x_6-1)^2+5x_7^2$$
$$+7(x_8-11)^2+2(x_9-10)^2+(x_{10}-7)^2+45$$

subject to:

$$\begin{aligned}
g_1(X) &= -105+4x_1+5x_2-3x_7+9x_8 \le 0\\
g_2(X) &= 10x_1-8x_2-17x_7+2x_8 \le 0\\
g_3(X) &= -8x_1+2x_2+5x_9-2x_{10}-12 \le 0\\
g_4(X) &= 3(x1-2)^2+4(x_2-3)^2+2x_3^2-7x_4-120 \le 0\\
g_5(X) &= 5x_1^2+8x_2+(x_3-6)^2-2x^4-40 \le 0\\
g_6(X) &= x_1^2+2(x_2-2)^2-2x_1x_2+14x_5-6x_6 \le 0\\
g_7(X) &= 0.5(x_1-8)^2+2(x_2-4)^2+3x_5^2-x_6-30 \le 0\\
g_8(X) &= -3x_1+6x_2+12(x_9-8)^2-7x_{10} \le 0
\end{aligned} \tag{26}$$

where $-10 \le x_d \le 10\,(d = 1,\ldots,10)$. The global objective value is $f(X^*) = 24.3062$.

According to the results of Table 7, the best performance is related to εF3EA. Because the standard deviation of εF3EA is smaller than other algorithms that means that the best values have been generated close to the best solution.

4.8 G08 Benchmark Function

This benchmark function is nonlinear and it has only two nonlinear inequalities constraints. The number of variables is two and the value of $\boldsymbol{\rho}$ is equal to 0.8560. The g08 equation is as (27):

$$Minimize\ f(X) = -\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)}$$

subject to:

$$\begin{aligned}
g_1(X) &= x_1^2-x_2+1 \le 0\\
g_2(X) &= 1-x_1+(x_2-4)^2 \le 0
\end{aligned} \tag{27}$$

where $0 \le x_d \le 10(d = 1, 2)$. The global objective value is $f(X^*) = -0.095825$.

Given that the low number of constraints and decision variables, the g08 is a simple function. So, according to Table 8, there are no significant differences between the results of algorithms, and all algorithms have found the optimal solution.

Table 7 Comparative results of εF3EA with other algorithms in g07 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	24.48	24.3062	24.306	24.306	24.327	24.3062	24.306	24.3062	24.3062
AV	26.58	24.3062	24.306	24.316	24.475	24.3063	24.307	24.3062	24.3062
WV	28.40	24.3062	24.306	24.359	24.843	24.3063	24.309	24.3068	24.3062
STD	1.14E+00	3.34E−06	6.3E−05	1.1E−2	1.32E−1	3.19E−05	7.1E−04	1.3E−04	3.32716E−13

Table 8 Comparative results of εF3EA with other algorithms in g08 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
AV	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
WV	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
STD	0	0	2.7E−17	2.8E−17	0	0.00E+00	2.4E−14	3.8E−13	2.27598E−17

4.9 g09 Benchmark Function

The g09 function is Polynomial and has seven decision variables. The value of ρ is 0.5121 and it has only two nonlinear inequalities constraints. The constraint g1 and g4 are active which its equation is as (28):

$$Minimize\ f(X) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$$

subject to:

$$\begin{aligned} g_2(X) &= -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0 \\ g_3(X) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0 \\ g_4(X) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned} \tag{28}$$

where $-10 \leq x_d \leq 10$ $(d = 1, \ldots, 7)$. The global objective value is $f(X^*) = 680.630057$.

The results of Table 9 show that most algorithms have found the optimal solution and SAFFa has had the worst performance. In terms of STD, the εF3EA has had the best result among all algorithms.

5 Conclusion

In this paper, an extended epsilon constrained handling version of F3EA algorithm was proposed to handle constraints that was called εF3EA. The F3EA algorithm is classified into the population-based algorithm which simulates battleground and mimics the F3EA targeting process of object or installation selection for destruction in the warfare. The εF3EA algorithm was used for solving constraint optimization problems with ε constraint handling techniques. In this way, this algorithm was divided into nine steps of generating random solutions, initialization of the ε level, ranking the solutions, the Fix step, the Find step, the Finish step, the Exploit step, the Analyze step, and checking stopping criteria. For solving the constraint optimization problem, The lexicographic order was used when $\varepsilon > 0$ in which constraint violation precedes the objective value. That way if $\emptyset(x) \leq \varepsilon$ then the objective value was compared and if the better solution was found, it was replaced. In each iteration, the ε value was decreased at a constant rate. With this process, in the last iterations, the solutions without constraint violation were accepted and then in terms of objective function values, the solutions are accepted. In order to evaluate the efficiency of the εF3EA algorithm, nine constraint benchmark functions were used. Functions categories included three quadratic functions, two nonlinear functions, two polynomial functions, and two cubic functions. In addition, the εF3EA algorithm was compared

Table 9 Comparative results of εF3EA with other algorithms in g09 problem

Parameters	SAFFa	COPSO	ISR	ATMES	SMES	ECHT-EP2	HCOEA	αSimplex	εF3EA
BV	680.64	680.630057	680.630057	680.630	680.632	680.630057	680.630	680.630057	680.630057
AV	680.72	680.630057	680.630057	680.639	680.643	680.630057	680.630	680.630057	680.630057
WV	680.87	680.630057	680.630057	680.673	680.719	680.630057	680.630	680.630057	680.630057
STD	5.92E−2	0	3.2E−13	1.0E−2	1.55E−2	2.61E−08	9.4E−8	2.9E−10	4.45339E−13

with eight other algorithms, which included SAFFa, COPSO, ISR, ATMES, SMES, ECHT-EP2, HCOEA, and αSimplex. According to the results, the εF3EA showed the fast convergence toward optimum solutions and it provided the appropriate results on all benchmark functions.

References

1. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)
2. Askarzadeh, A.: A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput. Struct. **169**, 1–12 (2016)
3. Husseinzadeh Kashan, A.: An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (LCA). Comput.-Aided Des. **43**(12), 1769–1792 (2011)
4. Sadollah, A., et al.: Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl. Soft Comput. **13**(5), 2592–2612 (2013)
5. Eskandar, H., et al.: Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput. Struct. **110**, 151–166 (2012)
6. Husseinzadeh Kashan, A., Tavakkoli-Moghaddam, R., Gen, M.: Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm: an effective algorithm with new evolutionary operators for global optimization. Comput. Ind. Eng. **128**, 192–218 (2019)
7. Yildiz, A.R.: A comparative study of population-based optimization algorithms for turning operations. Inf. Sci. **210**, 81–88 (2012)
8. Yildiz, A.R.: Comparison of evolutionary-based optimization algorithms for structural design optimization. Eng. Appl. Artif. Intell. **26**(1), 327–333 (2013)
9. Karagöz, S., Yıldız, A.R.: A comparison of recent metaheuristic algorithms for crashworthiness optimisation of vehicle thin-walled tubes considering sheet metal forming effects. Int. J. Veh. Des. **73**(1–3), 179–188 (2017)
10. Jalili, S., Husseinzadeh Kashan A., Hosseinzadeh, Y.: League championship algorithms for optimum design of pin-jointed structures. J. Comput. Civ. Eng. **31**(2), 04016048 (2017)
11. Husseinzadeh Kashan, A., Karimi, B., Noktehdan, A.: A novel discrete particle swarm optimization algorithm for the manufacturing cell formation problem. Int. J. Adv. Manuf. Technol. **73**(9–12), 1543–1556 (2014)
12. Biedrzycki, R., Arabas, J., Jagodziński, D.: Bound constraints handling in differential evolution: an experimental study. Swarm Evol. Comput. **50**, 100453 (2019)
13. Coello, C.A.C., Zacatenco, C.S.P.: List of references on constraint-handling techniques used with evolutionary algorithms. Power **80**(10), 1286–1292 (2010)
14. Michalewicz, Z.: A survey of constraint handling techniques in evolutionary computation methods. Evol. Prog. **4**, 135–155 (1995)
15. Coello, C.A.C.: Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput. Methods Appl. Mech. Eng. **191**(11–12), 1245–1287 (2002)
16. Jagodziński, D., Arabas, J.: A differential evolution strategy. In: 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE (2017)
17. Takahama, T., Sakai, S.: Solving constrained optimization problems by the ε constrained particle swarm optimizer with adaptive velocity limit control. In: 2006 IEEE Conference on Cybernetics and Intelligent Systems. IEEE (2006)
18. Husseinzadeh Kashan, A., Tavakkoli-Moghaddam, R., Gen, M.: A warfare inspired optimization algorithm: the Find-Fix-Finish-Exploit-Analyze (F3EA) metaheuristic algorithm. In: Proceedings of the Tenth International Conference on Management Science and Engineering Management. Springer (2017)

19. Husseinzadeh Kashan, A., Karimi, B., Jolai, F.: Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes. Int. J. Prod. Res. **44**(12), 2337–2360 (2006)
20. Forsythe, G.E.: Computer methods for mathematical computations. Prentice-Hall Ser. Autom. Comput. **259** (1977)
21. Liang, J., et al.: Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. J. Appl. Mech. **41**(8), 8–31 (2006)
22. Husseinzadeh Kashan, A.: League Championship Algorithm (LCA): An algorithm for global optimization inspired by sport championships. Appl. Soft Comput. **16**, 171–200 (2014)
23. Farmani, R., Wright, J.A.: Self-adaptive fitness formulation for constrained optimization. IEEE Trans. Evol. Comput. **7**(5), 445–455 (2003)
24. Aguirre, A.H., et al.: COPSO: Constrained optimization via PSO algorithm. In: Center for Research in Mathematics (CIMAT). Technical report no. I-07-04/22-02-2007 (2007), p. 77
25. Runarsson, T.P., Yao, X.: Search biases in constrained evolutionary optimization. IEEE Trans. Syst., Man, Cybern., Part C (Appl. Rev.) **35**(2), 233–243 (2005)
26. Wang, Y., et al.: An adaptive tradeoff model for constrained evolutionary optimization. IEEE Trans. Evol. Comput. **12**(1), 80–92 (2008)
27. Mezura-Montes, E., Coello, C.A.C.: A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans. Evol. Comput. **9**(1), 1–17 (2005)
28. Mallipeddi, R., Suganthan, P.N.: Ensemble of constraint handling techniques. IEEE Trans. Evol. Comput. **14**(4), 561–579 (2010)
29. Wang, Y., et al.: Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems. IEEE Trans. Syst., Man, Cybern. Part B (Cybern.) **37**(3), 560–575 (2007)
30. Takahama, T., Sakai, S.: Constrained optimization by applying the/spl alpha/constrained method to the nonlinear simplex method with mutations. IEEE Trans. Evol. Comput. **9**(5), 437–451 (2005)