Applied Soft Computing 113 (2021) 107981

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A soft computing based multi-objective optimization approach for automatic prediction of software cost models

Shailendra Pratap Singh [a], Gaurav Dhiman [b], Prayag Tiwari [c,*], Rutvij H. Jhaveri [d,*]

[a] CSED, Bundelkhand Institute of Engineering and Technology Jhansi, Jhansi, UP, India
[b] Department of Computer Science, Government Bikram College of Commerce, Patiala, India
[c] Department of Computer Science, Aalto University, Espoo, Finland
[d] Department of Computer Science & Engineering, Pandit Deendayal Energy University, Gandhinagar, India

ARTICLE INFO

Article history:
Received 12 April 2021
Received in revised form 1 September 2021
Accepted 1 October 2021
Available online xxxx

Keywords:
Adaptation
Optimization
Multiobjective differential evolution
Multiobjective evolutionary algorithms
Software cost estimation

ABSTRACT

This paper tries to extend the idea of single-objective differential evolution (DE) algorithm to a multi-objective algorithm. Most of the existing algorithms face the problem of diversity loss and convergence rate. In this paper, we propose a novel multi-objective DE algorithm to deal with this problem. In the validation process, the proposed method is validated in two steps. Firstly, the new homeostasis factor-based mutation operator incorporates multi-objective differential evolution algorithms (MODE). In this method, we use the Pareto optimality principle. We incorporate a new adaptive-based mutation operator (MODE) to create more diversity and enhance convergence rate among candidate solutions which provide better solutions to help the evolution. The effectiveness of the proposed method is evaluated on eight benchmarks of bi-objective and tri-objective test functions. Our proposed method performed well compared to the latest variants of multi-objective evolutionary algorithms (MOEAs). Secondly, the proposed method is used for an application-based test by applying it for software cost estimation. This method also incorporates multi-objective parameters, i.e., two objectives-based software cost estimation and three objectives-based software cost estimation. The proposed approach achieves better results in most software projects in terms of reducing effort and minimum error.

1. Introduction

DE is a stochastic population-based optimization algorithm developed by Storn and Price [1]. It is considered to be the most promising algorithm to solve optimization problems in the real world [2,3]. However, the current variant of DE is mainly focused on getting one optimum solution for the problem, and the concept of the Pareto front was completely ignored by the researchers [1]. In the Pareto approach, the author addresses the numerous objectives rather than single-objective optimization. In addition, the objective function is a vector rather than a scalar value. From the literature, it has been confirmed that Pareto-based approaches give better optimum solutions for multi-objective problems. The authors suggest many different variants of DE. Still, the Pareto-based DE algorithm (MODE) [4–8] is more prominent, easy, and has a minimal rate of convergence to generate new offspring.

In this work, we propose a homeostasis factor-based mutation operator that overcomes the problems of being stuck in local optima, maintains the diversity, and enhances the convergence speed and non-dominated fronts algorithm in MODE algorithms. This paper also uses a homeostasis factor-based mutation operator to incorporate the Pareto-based variant of the multi-objective differential evolution algorithm.

One of the complex activities in the software engineering domain is to provide an efficient software effort. Many factors can affect the effort and cost of the software, but the most important factors are the size of the project, number of persons, and schedule for the estimation of cost. Ezghari et al. [9] made a detailed study of how misleading and irrelevant information can affect software estimation. To overcome the issue, the Sheta model [10] proposed a differential evolution (DE) for tuning the parameters of single-objective optimization to estimate effort cost. In addition, [9,11] suggested Evidence-based guidelines for resolving the uncertainties in cost assessments. The drawback of the effort estimation approach is diluted with a multi-objective differential evolution algorithm with a homeostasis factor-based mutation operator in software cost estimation.

* Corresponding authors.
E-mail addresses: shail2007singh@gmail.com (S.P. Singh), gdhiman0001@gmail.com (G. Dhiman), prayag.tiwari@aalto.fi (P. Tiwari), rutvij.jhaveri@sot.pdpu.ac.in (R.H. Jhaveri).

https://doi.org/10.1016/j.asoc.2021.107981
1568-4946/

1.1. Problem description

In this section, the detailed problem description is presented. The first is related to evolutionary computation, and the next software cost estimation.

I. In DE, the mutation operator is mainly used for maintaining diversity and improving the convergence rate among solutions. However, from the literature, it is observed that the diversity and convergence rate provided by the existing mutation operator is not satisfactory [12–17]. Therefore, it is essential to embed a better-modified mutation operator that generates more diverse solutions and also solve the problem of stagnation in multi-objective optimization [12–14]. Consequently, coping with the stagnation problem enhances the diversity and convergence speed of the MODE method.

II. During the software development life cycle (SDLC) phase, the primary concern is optimally managing the total budgetary and human resources involved in software development. Therefore, the major problem of software cost estimation is to minimize the project cost and minimize the time span.

1.2. Contribution

The Author's description of all these contributions as follows.

1. A single-objective-based DE algorithm has been extended to solve multi-objective optimization problems using a homeostasis factor mutation approach. The proposed algorithm uses the concept of Pareto optimality termed as multi-objective differential evolution (MODE). In addition, this work incorporates a new mutation method with MODE for creating more diversity among candidate solutions.
2. The proposed method is applied for software cost estimation to show its applicability.
3. The proposed algorithm is evaluated with the other standard multi-objective evolutionary algorithms (MOEA) for the bi-objective and tri-objective multi-objective benchmark functions.
4. The proposed software cost estimation-based algorithm is evaluated with the other standard MOEA algorithms for the bi-objective (like Prediction and MMRE) and tri-objective(like Prediction, MMRE, and RMSE) multi-objective benchmark functions.
5. The proposed approach obtains a better solution in both benchmark function and application-based tests.

1.3. Organization

The rest of the paper is organized as follows; Section 2 explain the background of multi-objective optimization techniques and software cost estimation approaches. In Section 3, the proposed MODE using a new mutation operator has been discussed and applied to the software cost estimation. In Section 4, results are discussed for multi-objective-benchmark functions. After finding the favorable outcomes on multi-objective functions, we have applied the same algorithm to estimate software cost. These results are also discussed in the same subsection. Finally, Section 5 concludes the paper.

2. Background method

In this section, a detailed description of the literature survey is presented. Firstly, we survey multi-objective optimization problem-based techniques and then multi-objective differential evolution. Next, software cost estimation approaches are discussed.

2.1. MOP (Multiobjective optimization problem)

MOP's involves the concurrent optimization of numerous objective functions [18–21]. However, a Multi-objective optimization algorithm has to obtain a set of candidate solutions. These solutions are better for the pool of candidate solutions. The multi-objective problems are explained here:

$$Minimize\, f(X)(f(x_1), f(x_2), \ldots .f_m(x_n))^T \tag{1}$$

$$G_i(x) \le I = 1, 2, \ldots, k \tag{2}$$

$$X \epsilon \omega \tag{3}$$

where $X = (x_1, x_2, \ldots, x_n)^T$ represent the decision variable of vector, ω denotes the decision space, and n denote the number of variable. Therefore, f(x) is a function of MOP's, which contains m objective functions.

2.1.1. Pareto front (dominance)

Vilfredo F. D. Pareto proposed the Pareto concept. This Pareto applies the multiobjective algorithm and checks the performance of Pareto dominance. Pareto dominance uses the multiobjective-based NSGAII by Deb et al. [22] and NSGA III by Ahmed and Adjabi [23]. This dominance is a set of solutions of a nondominated sorting algorithm. This algorithm performs better than the other solutions, and hence Pareto dominated over those outcomes. Pareto uses two types of solution known as dominating and non-dominating. The solution set for this is defined as:

$x = (x_1, x_1, \ldots, x_n)^T$ will be said to be dominating over $y = (y_2, y_2, \ldots, y_2)^T$ if the function below: for all iϵ ϵ $(1, 2, \ldots, k)$: $f_i(x) \le f_i(y)$ there exist i ϵ $(1, 2, \ldots, k)$: $f_i(x) < f_i(y)$.

2.1.2. Non-dominating sorting

A solution is dominated with respect to other solutions depending on the following conditions.

- Let, S represent the solution, then S^1 is not worst than S^2 in all M Objectives.
- S^1 outperforms over S^2 at least one of the objectives in all M objectives.

when solution i dominated to solution j then rank is $r_i < r_j$.

2.2. Multi-objective differential evolution

Plenty of approaches have been proposed to resolve the diversity issue in MODE Algorithm [4–8,24–27]. In [4], the MODE/D-DE algorithm was proposed. In this algorithm, the concept of three-component is suggested by introducing a novel concept of reasonable time to find the Pareto front. In addition, it adopts the idea of the self-adaptive method for proposed mutation and crossover operators. But, still, the convergence issue was not resolved when tested on various benchmark functions. In [5], MOEA-D is suggested by introducing a novel concept of reasonable time to find the Pareto front. But, still, the diversity issue was not resolved as observed on most of the tested benchmark functions. In [6], the author's give the concept of a new mutation method-based multi-objective MOEA-D. They also adopted the idea of DE/rand/1/bin and a non-dominating sorting algorithm. However, the diversity problem was still not resolved as observed in experimental analysis when tested benchmark functions. In [7], the authors proposed a Handle multi-modal-based multi-objective optimization. They tried to capture the search space issue, i.e., exploration and exploitation strategy. However, the proposed strategy resolved the diversity issue to some extent, but the convergence rate was reduced when tested on multi-objective benchmark functions. In [8], researchers have implemented the Pareto-based evolutionary algorithm called RDMOE (Reference Direction for Multi/many-objective algorithm). This algorithm has adopted the Pareto front concept, but the algorithm is still unable to maintain the diversity. In [24], researchers have proposed a many-objective optimization algorithm. They opted for the concept of a self-balance adaptive decomposition-based algorithm. This approach used the Pareto dominance. The experimental analysis observed that the proposed algorithm maintains diversity when tested on many-objective benchmark functions. In [25], the author's presented the many-objective optimization algorithm called DBMOA. This algorithm solved the many-objective problems, namely WFG and DTLZ. This algorithm, similarly, generates the Pareto front for the DTLZ series. In [26], the author has given the concept of a new MODE algorithm. This algorithm solved the problem of local convergence rate. In the experimental analysis, the proposed algorithm enhanced the convergence speed and maintained the diversity. But, still, the diversity issue was not resolved as observed on most of the tested benchmark functions. In [27], the PBDE-HBM algorithm has been proposed by the authors. In PBDE-HBM, a new mutation strategy using Homeostasis based vectors is devised. These proposed vectors are applied in the existing mutation vector to enhance the convergence speed and maintain the bandwidth of diversity as well. The outcomes show that PBDE-HBM gets better performance than other standard MOEA algorithms in the experimental analysis. It provides accurate prediction and minimization of the error like MMER, MMRE, MSE, and RMSE. The model is tested on the COCOMO dataset. There is a strong relationship between the management of software, effort, and cost estimation, which affects the execution of the software project. The major concern is optimally managing the total budgetary and human resources involved in software development [9,28].

2.3. Advantages of proposed over existing approaches

The list of advantages over the existing approaches are shown as follows [4–8,23,29–31]:

- The proposed algorithm improves the convergence rate of global search space, in contrast to the existing approaches [23,30,31], where convergence rate is low on tested benchmark functions.
- Existing methods [7,8,24–27,32,33] face the problem of stagnation after a certain number of iterations. For resolving this issue, a new homeostasis factor-based multi-objective differential evolution algorithm (HFMODE) is proposed.
- The proposed method uses the Pareto optimal principle with a non-dominant front algorithm and reduces time and space complexity. Therefore, it outperforms the other tested optimization algorithms on standard benchmark functions [18–21,34–37].
- Introduces the homeostasis factor-based novel approach proposed in this article. Compared with other multi-objective problem algorithms, the proposed method provides better exploitation and exploration among candidate solutions.
- The proposed approach provides accurate predictions and minimizes errors such as MMRE and RMSE.

Most of the above-discussed methods take more time to find the optimum Pareto front and face the problem of stagnation after a certain number of iterations. To resolve all these issues, we have introduced a new mutation operator for the MODE algorithm and applied it to the software cost estimation problem, discussed in the next section.

3. Proposed approach

In this section, a detailed description of the proposed approaches is discussed. First, we proposed a multiobjective-based mutation technique. Next, the software cost estimation model is framed using the proposed method.

3.1. Homeostasis factor-based multi-objective differential evolution algorithm (HFMODE)

In this section, the proposed approach is composed of various phases, namely, (1) Initialization population methodology, (2) Homeostasis factor-based mutation operator, (3) Non dominating fronts, (4) Crossover, and (5) Selection. The detailed description of all these phases is as follows.

3.1.1. Initialization population methodology

In this process, initially, the population is sorted according to the fitness value. Thereafter, the population is divided into two parts. The first one consists of those population members having high fitness values. The next one consists of the remaining population members. Finally, ranking is given to the better candidate's solutions. Further, $Best_{(pop)}$ is assigned to best population members as shown in Eq. (4). Similarly, $Rem_{(pop)}$ is given to remaining population members as shown in Eq. (5).

$$Best_{(pop)} = fit_b * \sum_{i=1}^{N} (Solution_i) \tag{4}$$

Where, $Best_{(pop)}$ denotes the best population members of the search space, and fit_b denotes the candidate solutions according to feasible search space, $i = 1, 2, \ldots .n$, which is stored the increasing order ranking of candidate solutions ($solution_i$).

$$Rem_{(pop)} = fit_w * \sum_{j=1}^{N} (Solution_j) \tag{5}$$

Where, $Rem_{(pop)}$ denotes the remaining population members of the search space, and fit_w, $\forall_i 1 \le i \le n$ denotes the index of remaining solutions to whole search space which is stored the increasing order of candidate solutions ($solution_j$).

The best populations based vectors help to provide better search space of the global environment. But, this method takes a little more time to find optimum value and also faces the problem of stagnation after a certain number of iterations. Therefore, we have incorporated the homeostasis factor into our proposed work.

3.1.2. Homeostasis factor-based mutation operator

The proposed approach is inspired by the "homeostasis factor" based biology evolution process. This evolution process maintains the balance in the environment and also enhances the convergence rate. Therefore, this strategy incorporates an evolutionary algorithm like DE. The proposed DE algorithm is to improve the performance of computation by using the homeostasis factor. This factor is the ability to maintain living environment systems like local and global search space. Also, this factor has to enhance the internal environment and improve the functioning of the search space. Therefore, a novel operator is devised that maintains the diversity in the search space and improves the convergence rate. The detailed description of the Homeostasis factor methodology is as follows.

(I) Homeostasis factor (Hf): This process applies the homeostasis factor(Hf) in the best and remaining candidate solutions. The (Hf) is helping to improve the environmental condition of search space locally and globally. The factor is incorporated according to the candidate vectors obtained from the multiplication by Hf values, which lies rand(0, 1). This factor value provides sufficient diversity to the current vector and improves the performance of convergence speed. We have designed two homeostasis factor-based vectors fv_1 and fv_2, maintaining a better search space environment. The adaption of DE can be preserved based on the new vector formation of the mutation operator designed in the next section.

(II) Homeostasis factor-based new vector formation process: In this process, two new current vectors are estimated. The first vector is framed from the $Best_{(pop)}$ candidate solutions by considering the homeostasis factor Hf as shown in Eq. (6). Afterward, the next vector is estimated using the $Rem_{(pop)}$ candidates solutions and considering the Hf as shown in Eq. (7). Hence, the adaption of DE can be preserved based on the homeostasis factor-based mutation operator as given by Eqs. (6) and (7).

$$fv_1 = Best_{(pop)} * Hf \tag{6}$$

$$fv_2 = Rem_{(pop)} * Hf \tag{7}$$

where, fv_1 and fv_2 represents the homeostasis factor-based vectors in global search space, and Hf denotes the homeostasis factor. This factor provides sufficient diversity to the current vector and improves the performance of convergence speed. The detailed description of the generation of new donor vector methodology is as follows.

3.1.3. Generation of new donor vector

In this section, the optimization-based mutation scheme of $DE/BEST/1$ is presented. In this process, the donor vector is obtained using the fv_1 and fv_2 candidate's solutions, which are taken as shown in Eq. (9). Thereafter, to consider the effect of the donor vector obtained from Eq. (8). Finally, donor vectors obtained from Eq. (8), and Eq. (9) are compared, among the best is considered for the crossover process.

Illustration of Eqs. (8), & (9): Conventional way to obtain the donor vector is shown in Eq. (8). The devised equation for obtaining the donor vector is shown in Eq. (9).

$$\vec{\gamma}_{i,G+1} = \vec{\alpha}_{\text{best},G} + \delta_1 \cdot (\vec{\alpha}_{r_1^i,G} - \vec{\alpha}_{r_2^i,G}) \tag{8}$$

where, $\vec{\gamma}_{i,G+1}$ denotes the donor vector, $\vec{\alpha}_{\text{best}}$ denotes the best vector. $(\vec{\alpha}_{r_1^i,G} - \vec{\alpha}_{r_2^i,G})$ denote the difference vector of search space, and δ_1 denote the mutation factor.

$$\vec{H}_{i,G+1} = \vec{\alpha}_{\text{best},G} + \delta_1 \cdot (f\vec{v}_{1_{r_1^i},G} - f\vec{v}_{2_{r_2^i},G}) \tag{9}$$

Where, $\vec{H}_{i,G+1}$ represents the obtained donor vector, G denote the generation, and δ_1 represents the mutant factor. The $\vec{\alpha}_{\text{best}}$ denotes the best vector of current population. The $\vec{H}_{i,G+1}$ newly obtained donor vector is incorporate to frame the novel mutation strategy known as homeostasis factor-based mutation strategy.

The proposed algorithm is used as homeostasis factor-based vectors, providing sufficient diversity for MOP's and reducing the time complexity. These solutions are applied to non-dominating solutions. A detailed description of the non-dominating front is designed next section.

3.1.4. Non-dominating fronts incorporates the novel mutation operator

In this phase, all the solution vectors are arranged using the non-dominating Pareto fronts, but instead of making many Pareto fronts, we have created only two fronts. The first rank solution is a non-domination Pareto front, and these solutions have been used to the first Pareto front. This solution uses the best hierarchy of non-dominating fronts. Hierarchy of the best solution (All non-dominated) which is used for the first Pareto front. Therefore, the non-dominated solution is a better search for a feasible global space. This process incorporates the novel mutation operator as given in Eq. (9). Another dominated solution finds a better solution according to the second Pareto front. This type of front will be guided by Eq. (8). Therefore, for the determination of the solution is close to the suppressed solution, this uses random individuals for non-dominating Pareto fronts.

3.1.5. Crossover

After the completion of the mutation operation, the crossover operator is applied to the offspring. This operator mainly focuses on the tuned of the parameter according to the feasible environment. In this paper, we have used the crossover rate between 0.01 to 0.5. These values varied according to the need for vectors based on the tuning of the self-adaptive for mutant Vectors. Therefore, the tuning values are adjusted according to the crossover rate (CR), which is kept low for the non-dominated solutions and high for other solutions.

3.1.6. Selection

After completion of the crossover operation, a selection operator is applied for finding the solution of survival of fittest in the trail vectors and target vectors. This strategy chooses the best vector of first higher fitness according to the non-dominating front algorithm. The pseudo-code for the HFMODE is given in Algorithm 1.

3.1.7. HFMODE algorithm

The pseudo-code of the proposed HFMODE algorithm is shown in Algorithm 1.

3.2. Proposed approach used for the software cost estimation (SCE)

The proposed method provides diversity among solutions for choosing better mutant outcomes. Being motivated, we have also applied this algorithm to software projects to estimate their cost during the development phase. In this approach, the proposed DE algorithm is applied to software projects to estimate their cost. It optimizes various constraints, i.e., software cost estimation parameters. The detailed description of software performance parameters are as follows:

3.2.1. Software performance parameters

Five types of performance parameters are generally used for the software cost estimation, i.e., Effort (Eff), Mean relative error (MRE), Prediction (Prid), Mean magnitude of relative error (MMRE), and Root mean square error (RMSE) metrics. These matrices are extensively used for testing the effectiveness of the proposed algorithm (HFMODE). The values of these metrics are calculated using the simulation work, and it uses 100, 200, and 500 generations for optimal solutions from software projects. The details of these parameters are given below [9,10,28,38–43]:

(I) Effort (Eff): In the first objective, the amount of effort and the schedule for software projects are considered. It is calculated using the COCOMO model as shown in Eq. (10). This effort reflects the required staffing to complete a project in time [11,44–48]. The first objective(Obj1) in terms of effort is calculated by using the following formula:

$$Effort(E) = a(KLOC)^b EAF \tag{10}$$

where, EAF is the effort adjustment factor, a and b are parameter constants (values are based on the type of model), and E is the effort in person-months.

(II) Mean Relative Error (MRE): In the second objective, MRE [41–43,49] is used to calculate the error of projects. We have the MRE metric calculated by Eq. (11) and also find the difference of error between actual effort and estimated effort. If

Algorithm 1: Proposed HFMODE Algorithm

```
Input: Initializations
(1)fun_obj, obj = 1, 2, ...n. Multi-objective problem with obj
is a objectives;
(2) Search_space of D ;
(3) Number of generation according to fitness functions;
(4) Set itr=0, t=0.;
Output: Generate the better candidate solutions
while (t! = Max_T) do
    while (itr! = n) do
        1.1 Set each solution according to fv_i better vector
        within feasible search space.;
        1.2 Apply the non-dominated sorting algorithm and
        generate their Pareto front number with assigned
        ranks to the solutions ;
        1.3 Apply proposed mutation operator used in Eq.
        (9);
        1.4 or Existing mutation operator used in Eq. (8);
        1.5 Generate Nondominating fronts according to the
        compared mutant vectors step1.3 & step1.4, if
        mutant vector (step1.3) is not able to find a better
        solution, i.e., first Pareto front, then mutant vector
        (step1.4) will be adopted;
        1.6 Apply crossover and use the tuning parameter;
        1.7 Apply selection strategy and first ranking of best
        solutions according to fitness for
        fun_obj, obj = 1, 2, ...n
        1.8 itr=itr+1;
    end
    t=t+1;
end
```

the difference is minimum, then a better quality of projects will be achieved. Therefore, the second objective (Obj2) in terms of error is calculated as follows:

$$MRE = |Est\_Eff - Act\_eff|/|Act\_Eff| \tag{11}$$

where *Est_Eff* represents the estimated effort of the projects and *Act_Eff* denotes the actual effort.

(III) Prediction (P): In the third objective, an accurate prediction performance measure for the optimization algorithm in the software cost estimation model is taken into consideration. It is defined as the percentage within 25% of the actual value [41–43,49] as shown in Eq. (12). If the high value of prediction is achieved, then better diversity and convergence rate will be achieved. Therefore, the third objective (Obj3) in terms of prediction is estimated as follows:

$$P(0.25) = 1/N \sum_{i=1}^{N} |Est\_Eff - Act\_Eff|/|Act\_Eff| \tag{12}$$

where N number of projects for prediction of actual effort and schedule. *Est_Eff* denotes the estimated effort of the projects, and *Act_Eff* denotes the actual effort. This measure estimates the convergence speed and diversity of the population according to the DE algorithms. The high value of the prediction reflects better diversity and convergence speed of the algorithm.

(IV) Mean Magnitude of Relative Error (MMRE): In the fourth objective, MMRE [42,43,49–51] metric is taken into consideration, which is calculated using Eq. (13). If MMRE error is minimum, then a better quality of software projects will be achieved. Hence, the fourth objective(Obj4) in terms of MMRE is estimated as follows:

$$MMRE = 1/N \sum_{i=1}^{N} |Est\_Eff - Act\_eff|/|Act\_Eff| \tag{13}$$

where *Est_Eff* represents the estimated effort of the projects and *Act_Eff* denotes the actual effort.

(V) Root Mean Square Error (RMSE): In the fifth objective, RMSE [28,41–43,49] metric is taken into consideration which is calculated using Eq. (14). If the RMSE error value is less, then a better quality of software projects will be achieved. Hence, the fifth objective(Obj5) in terms of RMSE is estimated as follows:

$$RMSE = \sqrt{1/N \sum (Act\_Eff - Est\_Eff)^2} \tag{14}$$

Where *Est_Eff* represents the estimated effort of the projects and *Act_Eff* denotes the actual effort. The software performance metric value is minimum from the software projects. It means achieving the minimum optimum value like an error.

3.2.2. Fitness function formulation

In this section, fitness functions calculated by Eqs. (10), (11), (12), (13), and (14) are non-conflicting to each other. Therefore, all the objectives ($Obj1$, $Obj2$, $Obj3$, $Obj4$, and $Obj5$) are converted into the single objective function using the sum of weighted approach as shown in Eq. (15).

$$\begin{aligned} Fitness &= fv_1 \times Obj1 + fv_2 \times Obj2 \\ &+ fv_3 \times Obj3 + fv_4 \times Obj4 + fv_5 \times Obj5 \end{aligned} \tag{15}$$

Here, values of $fv_1, fv_1, fv_3, fv_4 \& fv_5$ are the weights, which are assigned to each of the objective function. Similarly, the fitness function is applied to the MOPSO, MOEA-D, NSGA-III, and whale optimization algorithm (MOWOA) to compare software models.

3.2.3. Two objective-based formulation

In this process, we have designed the two objective functions based on software cost estimation that are conflicting with each other. All objective is converted into the multi-objective functions using software cost estimation parameters as shown in Eq. (16).

$$\begin{aligned} f1(max) &= Obj3 \\ f2(min) &= Obj4 \end{aligned} \tag{16}$$

Here, $f1(max)$ denotes the maximization problem of $obj3$, which calculates the Prediction(25%) for software projects. $f2(min)$ denotes the minimization problem of $obj4$, which is the measurement to calculate the error rate for software models.

3.2.4. Three objective-based formulation

In this process, we have designed the three objective-based software cost estimations, conflicting with each other. Finally, all objective is converted into the multi-objective functions using software cost estimation parameters as shown in Eq. (17).

$$\begin{aligned} f1(max) &= Obj3 \\ f2(min) &= Obj4 \\ f3(min) &= Obj5 \end{aligned} \tag{17}$$

Here, $f1(max)$ denotes the maximization problem of $obj3$, which is the measurement to calculate the Prediction(25%) for software projects. $f2(min)$ denotes the minimization problem of $obj4$, which is the measurement to calculate the error rate for software models, and $f3(min)$ denotes the minimization problem of $obj5$, which is the RMSE in error values for software projects.

3.2.5. The proposed algorithm with SCE

The proposed method is applied to the software projects to estimate its cost during the development phase. We need to optimize various constraints like software cost estimation parameters. The pseudo-code of the proposed software cost estimation-based algorithm is shown in Algo 2.

Algorithm 2: Proposed multi-objective software cost estimation model

Result: Obtain the software cost estimation model with reduced effort and error

Initialization: Random initialization of solution vectors ;
Select the homeostasis factor based vectors fv_i, where i is the rank of first nondominating fronts ;
while *(t!=Max)* **do**
    (1) Evaluate the fitness value of each fv_i Vector using Eq. (15);
    (2) Evaluate the two objective based software cost functions using Eq. (16);
    (3) Evaluate the three objective based software cost functions using Eq. (17);
    (4) Evaluate new vector using proposed mutation operator as mentioned in above section;
    (5) Apply the selection process ;
    (6) go to step 1 until it reaches convergence;
end

4. Result analysis and discussions

This section mentions a description of the test-bed, performance indicators, and results obtained with respect to the performance indicators.

4.1. Analysis of multi-objective benchmark functions

In this paper, we have taken eight benchmark functions [18–22,34–37,52] to test our proposed algorithm. These functions are based on multi-objective problems (MOP's), also known as bi-objective, and tri-objective functions. The multi-objective problems are based on ZDT and DTLZ series. The proposed approach (HFMODE) is applied to three standard functions of the ZDT series, known as bi-objective functions. Also proposed approach is tested on three standard functions of the DTLZ series, known as tri-objective benchmark problems. Finally, experimental outcomes produced by the proposed algorithm are compared with standard Algorithms like MOEA-D [5], NSGA-III [23], MOPSO [30], and MOWOA [31]. The features of the objective functions are described in Table 2.

4.1.1. Multiobjective benchmark functions

Biobjectives function (ZDT series): This ZDT family proposed by Zitzler contains eight multi-objective problems or standard functions [18–21,34–37,52]. These functions have two objectives, just like two different functions $f1$ and $f2$. These functions are based on the minimization problem of the ZDT series. In our study, we have taken 4 benchmark functions of this family. The Pareto optimal fronts of these problems are shown in Table 2.

Triobjectives Function (DTLZ Series): This DTLZ family proposed by Deb etc., contains 7 multi-objective problems or standard functions [18–22,34–37,52]. These functions have three objectives just like three different functions $f1$, $f2$, and $f3$. These functions are based on the minimization problem of the DTLZ series. In our study, we have taken 4 benchmark functions of this family. This is an essential implementation feature of three objectives on the Pareto front. The Pareto optimal fronts of these problems are shown in Table 2.

4.1.2. Performance metric

The performance metrics used by the proposed algorithm for evaluating the performance of the objective functions are given in Table 2. There are three types of performance metrics such as generational distance (GD), inverted generational distance (IGD), and Spacing (Sp) metric. The tested values of these metrics are calculated using the simulation work, and it uses 150 to 350 uniformly spaced Pareto optimal solutions for bi-objective and tri-objective problems. These matrices are discussed given below:

I. Generational Distance: Generation Distance (GD) was developed by Veldhuizen et al. [36,37]. GD is a metric that is calculated by Eq. (18), reflecting the closeness of the Pareto optimal front from different objectives. The GD metric is applied to a non-dominated set. Thus, the GD metric measures the closeness for the Pareto front set.

$$GD(P, P*) = \sqrt{\sum x_{\epsilon_P} Dis(x, P*)^2/|P|)} \quad (18)$$

Where $Dis(x, P*)$ denotes the Euclidean distance between population (P) and Pareto optimal front $|P*|$. The GD value is the minimum distance between the nondominated solution set from the population (P) and Pareto optimal front $|P*|$. Therefore, that is a value called the better adjacency set.

II. Inverted generational distance: Inverted generational distance (IGD) metric [34,35] developed by Zilzter et al. is a distance performance measure for the optimization algorithm. IGD measures the close distance for benchmark functions. IGD also measures diversity and convergence rates for multi-object optimization problems. This measure is calculated using Eq. (19):

$$IGD(P, P*) = \sqrt{\sum x_{\epsilon_{P*}} Dis(x, P)^2/|P*|)} \quad (19)$$

Where $Dis(x, P)$ denotes the minimum Euclidean distance between candidate solution x and the candidate solution P (population), and $|P*|$ measure the convergence speed and diversity of the population. Thus, the proposed algorithm is achieved by the minimum value of the IGD, which is called better diversity and convergence speed of the algorithm.

III. Spacing: Spacing (Sp) was developed by J. Schott [21]. This metric is used to calculate the closeness of the Pareto optimal front. The Sp metric is calculated by Eq. (20), which calculates the distance (D) between one objective solution to another. Therefore, minimum values of Sp indicate a better generation of the Pareto front set.

$$Sp = \sqrt{1/(m-1)\sum_{i=1}^{m}(\bar{ED} - ED_i)^2} \quad (20)$$

$ED_i = min_i, j \neq i \left(\sum_{k=1}^{k} \left|f_i^k - f_j^k\right|\right)$

Where m is the number of objectives according to internal environment vectors. This vector's distance calculates between the objective solution to another nearest objective solution in the Pareto front. k is the number of objectives according to the non-dominating solution set, and all ED_i is the distance between the nondominated solution set from the population mean of $\bar{ED}$. Thus, the metric spacing value is minimum for the Pareto front set.

4.1.3. Experimental results and discussion

The proposed algorithm is evaluated using MATLAB 2019a in the platform of Windows 10 with the core-i7 @3.6 GHz processor and 8GB RAM. The proposed algorithm is a simulation of the bi-objective like the ZDT family and tri-objective like the DTLZ family given in Table 2. The proposed approach is considered as

Table 1
Parameter of proposed algorithm.

Sr. no.	Parameter name	Description	Value
1	Population	Size of population	100
2	obj	No. of objectives	Objective functions
3	n	No. of variable s	Objective functions
4	g	No. of maximum generations	150, 200, 250, 350
5.	$\delta 1$	Mutation factor	0.12 to 1.6
6	Biobj	Biobjectives	ZDT Series
7	Biobj	Triobjectives	DTLZ series
8	Cr	Crossover rate	0.12 to 0.9
9	Hf	Homeostasis factor	rand(0,1)
10	fv	Best vectors	Search space

Table 2
HFMODE Algorithm based objective functions.

Sr. no.	Objective problems (Biobj and Triobj)	Feature of objective problems	Domain variable of objective problems
1	Biobj1: Biobjectives (ZDT1)	Convex	[0, 1] in between of two objectives
2	Biobj2: Biobjectives (ZDT2)	Nonconvex	[0, 1] in between of two objectives
3	Biobj3: Biobjectives (ZDT3)	Convex disconnected	[0, 1] in between of two objectives
4	Biobj4: Biobjectives (ZDT4)	Non convex	[0, 1] in between of two objectives
5	Triobj1: Triobjective (DTLZ1)	Linear	[0, 1] in between of three objectives
6	Triobj2: Triobjective (DTLZ2)	Concave	[0, 1] in between of three objectives
7	Triobj3: Triobjective (DTLZ3)	Concave	[0, 1] in between of three objectives
8	Triobj4: Triobjective (DTLZ4)		[0, 1] in between of three objectives

Table 3
GD: Proposed algorithm comparing with standard algorithms.

Objectives/Algorithm	MOPSO		MOWOA		MOEA-D		NSGA-III		Proposed Algo	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Biobj1	0.000358	0.000035	0.000868	0.000043	0.000397	0.000636	0.000925	0.000545	0.000166	0.000057
Biobj2	0.000421	0.000065	0.000868	0.000284	0.000838	0.000198	0.000575	0.000075	0.000106	0.0000099
Biobj3	0.000684	0.000082	0.000472	0.000136	0.000352	0.000092	0.000263	0.000094	0.000185	0.000026
Biobj4	0.000478	0.000139	0.000492	0.000092	0.000484	0.000237	0.000355	0.000173	0.000123	0.000079
Triobj1	0.000697	0.000125	0.000576	0.000298	0.000637	0.000319	0.000522	0.000244	0.000157	0.000044
Triobj2	0.000515	0.000085	0.000753	0.000235	0.000575	0.000075	0.000638	0.000207	0.000169	0.000088
Triobj3	0.000458	0.000229	0.000664	0.000083	0.000625	0.000092	0.000435	0.000293	0.000097	0.000017
Triobj4	0.000287	0.000065	0.004296	0.000335	0.000439	0.000189	0.000462	0.000055	0.000108	0.000025

Table 4
IGD: Proposed algorithm comparing with standard algorithms.

Objectives/Algorithm	MOPSO		MOWOA		MOEA-D		NSGA-III		Proposed Algo	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Biobj1	0.000106	0.000045	0.000449	0.000024	0.000248	0.000011	0.000525	0.000095	0.000079	0.000014
Biobj2	0.000770	0.0000505	0.000418	0.000084	0.000768	0.000026	0.000758	0.000061	0.000043	0.000016
Biobj3	0.001094	0.000528	0.004512	0.000225	0.000399	0.000143	0.000923	0.000974	0.000126	0.000024
Biobj4	0.008172	0.004336	0.001376	0.000688	0.006416	0.000153	0.006504	0.000635	0.000362	0.000071
Triobj1	0.006713	0.000256	0.000814	0.000105	0.000340	0.000174	0.001348	0.000729	0.000796	0.000096
Triobj2	0.000874	0.000143	0.004992	0.000749	0.000817	0.000121	0.002655	0.000649	0.000184	0.000044
Triobj3	0.008458	0.000229	0.000664	0.000083	0.000194	0.000094	0.000335	0.000060	0.000147	0.000054
Triobj4	0.001483	0.000415	0.001864	0.000532	0.000567	0.00635	0.000612	0.000191	0.000925	0.000204

Table 5
Spacing: Proposed algorithm comparing with standard algorithms.

Objectives/Algorithm	MOPSO		MOWOA		MOEA-D		NSGA-III		Proposed Algo	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Biobj1	0.001561	0.000782	0.001251	0.006252	0.000164	0.005944	0.001173	0.00054	0.000975	0.000480
Biobj2	0.004218	0.002109	0.003374	0.001687	0.000329	0.000163	0.000316	0.000147	0.000296	0.000095
Biobj3	0.001539	0.000769	0.001231	0.000636	0.001202	0.000849	0.001154	0.000769	0.000821	0.000472
Biobj4	0.008104	0.000522	0.004836	0.000418	0.006324	0.000797	0.006078	0.000283	0.000764	0.000248
Triobj1	0.003547	0.000917	0.005376	0.000988	0.006067	0.000373	0.007376	0.003442	0.0014335	0.000473
Triobj2	0.000536	0.000153	0.002425	0.000122	0.006139	0.000816	0.000538	0.002918	0.000893	0.000288
Triobj3	0.003145	0.000725	0.000516	0.000145	0.002531	0.000951	0.003588	0.000908	0.000579	0.000241
Triobj4	0.004502	0.001912	0.005901	0.000978	0.000116	0.000090	0.000765	0.000577	0.000249	0.000046

100 to 300 population size. The proposed algorithm has achieved reaching up to 60 generations or some termination criteria. Other parameters of the HFMODE algorithm (Proposed Algo) are given in Table 1.

The performance matrices are calculated for all types of GD, IGD, and Sp. From Tables 3, 4, and 5, it is clear that proposed algorithm performance is better than other algorithms. It was able to obtain good solutions that were very close to the optimum Pareto and deviations sufficiently. Also, the ZDT and DTLZ series

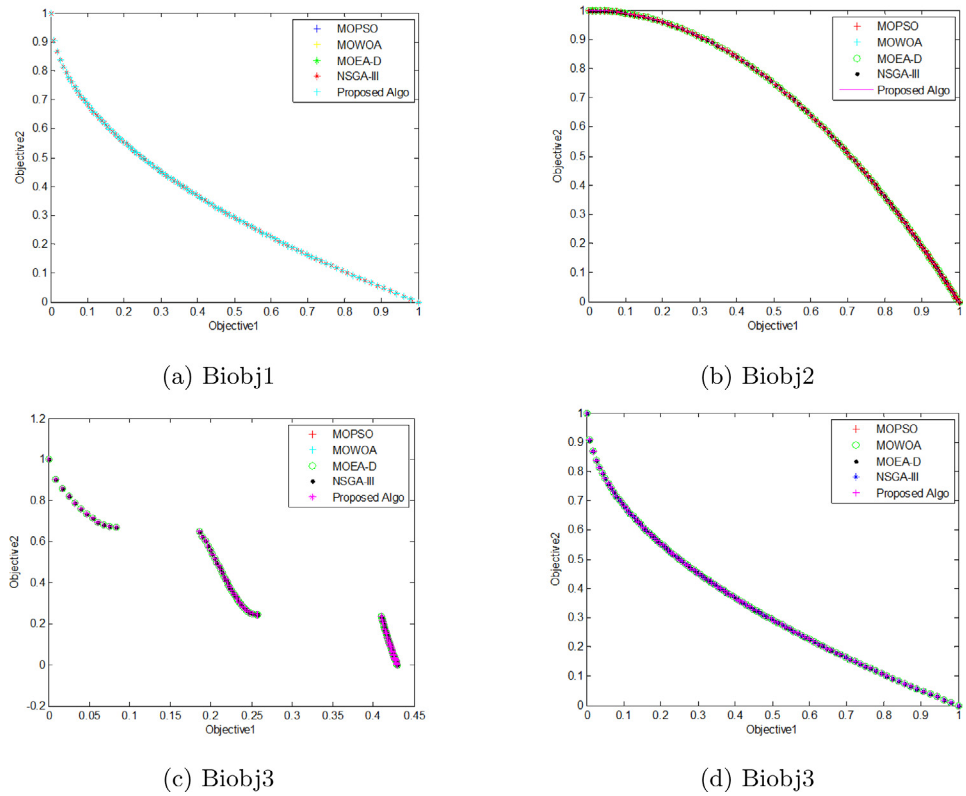


Fig. 1. Pareto optimal solution: Proposed Algorithm comparing with other standard algorithm on Biobjectives series.

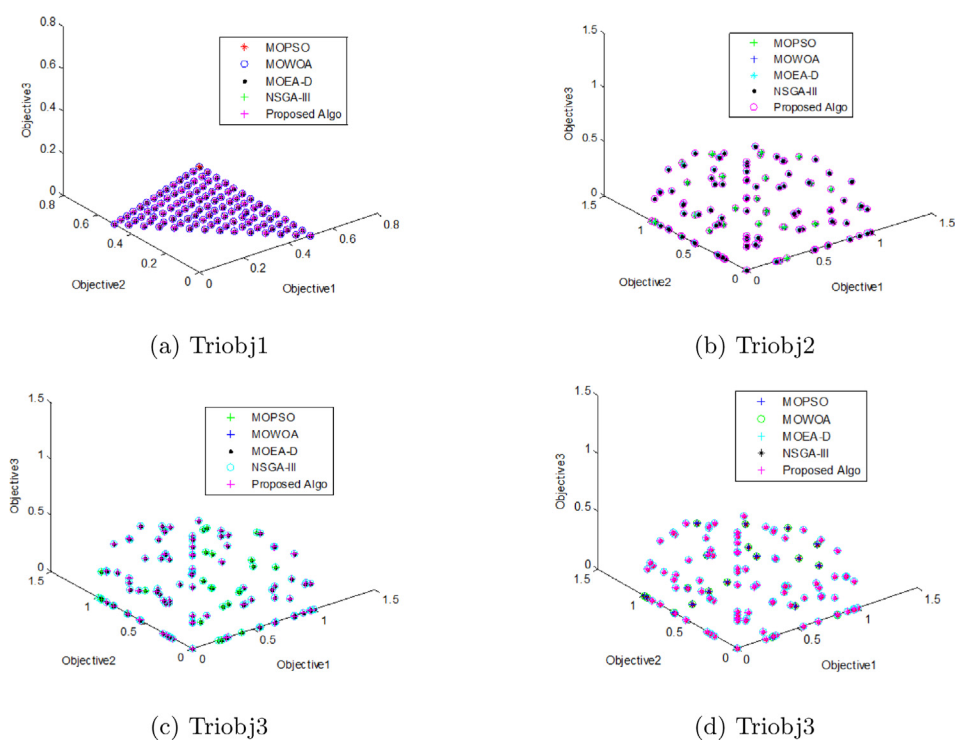


Fig. 2. Pareto optimal solution: Proposed Algorithm comparing with other standard algorithm on Triobjectives series.

results show that this algorithm is strong and scalable for any purpose. Figs. 1, and 2 shows the better optimal Pareto front of capturing the objectives functions using IGD, GD, and Spacing. Further, each objective function runs 20 times independently and also calculates the statistics values. These are including "mean" and "standard deviation(SD)" value of 8 test functions are shown in Tables 3, 4, and 5.

The convergence metric GD for the five algorithms is also given in Table 3. From this table, it is evident that the proposed algorithm achieves better "mean" and "SD" values as compared

Table 6
Proposed algorithm comparison with standard algorithms of Effort and MRE for organic, Semidetached, and embedded models.

Sr. no.	Software models	KLOC	Actual effort	Proposed Algo_effort	MOEA-D_effort	MOPSO_effort	MOWOA_effort	NSGA-III_effort	Proposed Algo_MRE	MOEA-D_MRE	MOPSO_MRE	MOWOA_MRE	NSGA-III_MRE
1	Semidetached	25.9	117.6	73.80351067	86.06785553	83.58364076	77.82804673	100.8251586	0.20892713	0.227910908	0.223304671	0.229635683	0.112688135
2	Semidetached	24.6	117.6	70.02692733	81.3279564	79.01716494	73.69736483	95.12547786	0.226942974	0.262170383	0.253281876	0.253485453	0.150976807
3	Semidetached	7	31.2	19.43172082	20.40887594	20.05456302	19.4719975	22.98798821	0.211602712	0.293988957	0.27577812	0.255234413	0.20793235
4	Semidetached	8.2	36	22.83501971	24.28882352	23.83318887	23.02398801	27.48842353	0.205154276	0.276513889	0.260910505	0.244742004	0.186781817
5	Semidetached	9.7	25.2	27.10306648	29.21865022	28.62722186	27.50698226	33.23473652	0.042365885	0.13554971	0.10499267	0.062160355	0.251882613
6	Semidetached	2.2	8.4	5.967362157	5.713157418	5.672759692	5.715803629	6.215524888	0.162465456	0.27188288	0.250646371	0.21697254	0.205444683
7	Semidetached	3.5	10.8	9.582099287	9.521077286	9.414331313	9.345870087	10.5035733	0.063263176	0.100655954	0.099049651	0.091421686	0.021683064
8	Semidetached	66.6	352.8	193.3993344	243.2388233	234.2187321	211.5976741	293.133909	0.253468745	0.26396542	0.259480552	0.271758445	0.133606043
9	Semidetached	7.5	72	20.84844892	22.01803874	21.62235932	20.94795145	24.85188916	0.398555835	0.59006482	0.540160258	0.48144918	0.517319549
10	Semidetached	20	72	56.69723201	64.76557669	63.04272353	59.18919675	75.28424087	0.119234067	0.085406386	0.096041909	0.120812992	0.036035421
11	Semidetached	6	24	16.60448984	17.22572926	16.95017771	16.53917548	19.31306016	0.17287005	0.239922089	0.226769284	0.21107916	0.154278436
12	Semidetached	100	360	292.7596784	380.3743662	364.9314071	325.425575	464.0237189	0.104782835	0.048106142	0.010575129	0.065211207	0.228274272
13	Semidetached	1.3	36	3.489261196	3.20294093	3.195386817	3.274294141	3.430016179	0.50662568	0.774375006	0.703476705	0.617243174	0.714730201
14	Semidetached	100	215	292.7596784	380.3743662	364.9314071	325.425575	464.0237189	0.20289851	0.653805634	0.538358355	0.348739374	0.915017386
15	Semidetached	20	48	56.69723201	64.76557669	63.04272353	59.18919675	75.28424087	0.101648899	0.296890421	0.241937137	0.158280512	0.449053131
16	Semidetached	100	360	292.7596784	380.3743662	364.9314071	325.425575	464.0237189	0.104782835	0.048106142	0.010575129	0.065211207	0.228274272
17	Semidetached	150	324	442.7151108	594.1712401	567.9718418	499.956621	733.7080823	0.205553016	0.708782574	0.581315623	0.368748598	0.998979583
18	Semidetached	31.5	60	90.11311992	106.746305	103.4827725	95.7552418	125.7856523	0.281557671	0.662239321	0.55947834	0.404630153	0.866177755
19	Semidetached	15	48	42.27896485	47.19669923	46.06030457	43.64478156	54.39053156	0.066864598	0.014225118	0.031196768	0.061608194	0.105177499
20	Semidetached	32.6	60	93.32397173	110.8538043	107.4315009	99.29997046	130.7603482	0.311579136	0.720428894	0.610285312	0.444744666	0.931677918
21	Semidetached	9.7	60	27.10306648	29.21865022	28.62722186	27.50698226	33.23473652	0.307586328	0.436069122	0.403663079	0.367712651	0.352409302
22	Semidetached	66.6	300	193.3993344	243.2388233	234.2187321	211.5976741	293.133909	0.199343245	0.160823334	0.169277129	0.200083931	0.018080706
23	Semidetached	29.5	120	84.28100733	99.31514003	96.33565646	89.32914881	116.7989823	0.166986291	0.146517758	0.15224061	0.1735459	0.021073367
24	Semidetached	15	90	42.27896485	47.19669923	46.06030457	43.64478156	54.39053156	0.297461119	0.404253396	0.376904943	0.34972437	0.312572001
25	Semidetached	38	210	109.1165258	131.2118921	126.985796	116.7999104	155.4875586	0.269502995	0.318904246	0.305176026	0.301346956	0.205070613
26	Semidetached	10	48	27.95833223	30.21420988	29.594516	28.40872095	34.39855351	0.234236992	0.3149567	0.296021534	0.277134968	0.22385714
27	Semidetached	15.4	70	43.42925672	48.58296707	47.40196531	44.87827177	56.03231875	0.212945528	0.260063971	0.24922404	0.243680764	0.15763526
28	Semidetached	48.5	239	139.9483668	171.6039002	165.7125798	151.2349552	204.8464043	0.232501951	0.239693242	0.236727566	0.249340859	0.112892639
29	Semidetached	16.3	82	46.01957652	51.71512831	50.43220051	47.66047363	59.74645933	0.246158751	0.313928548	0.297199283	0.284348029	0.214393867
30	Semidetached	12.8	62	35.96378812	39.64078155	38.74157911	36.89665868	45.46607313	0.235585724	0.306537672	0.289604854	0.274922077	0.210674229
31	Semidetached	32.6	170	93.32397173	110.8538043	107.4315009	99.29997046	130.7603482	0.253030893	0.295730979	0.284134596	0.282384236	0.18234897
32	Semidetached	35.5	192	101.799157	121.7481666	117.8990688	108.6784678	143.9786905	0.263555588	0.311010721	0.297947494	0.294663127	0.19758768
33	Semidetached	5.5	18	15.19431776	15.65345468	15.41512093	15.08327916	17.50451247	0.087443763	0.110809085	0.110862591	0.110025192	0.021746397
34	Semidetached	10.4	50	29.09948262	31.54626265	30.88834326	29.61351689	35.95736461	0.234503805	0.313713535	0.29508398	0.276848441	0.221873639
35	Semidetached	14	60	39.40595506	43.74738327	42.72055918	40.5696509	50.3112214	0.19255432	0.230245404	0.222328805	0.219886784	0.127568918
36	Semidetached	6.5	42	18.01701687	18.81117545	18.49693259	18.00225568	21.14132845	0.320344132	0.46929764	0.432008763	0.387963533	0.392341679
37	Semidetached	13	60	36.5370501	40.32263729	39.40246928	37.50746312	46.26964512	0.219378582	0.278762638	0.265021562	0.254540542	0.180783006
38	Semidetached	90	444	262.9290795	338.7489846	325.3043116	291.0680245	411.9402385	0.228785555	0.201494061	0.206380792	0.233875701	0.057043269
39	Semidetached	8	42	22.26706865	23.63797274	23.19970259	22.42972645	26.73202544	0.263575583	0.371612457	0.345567371	0.316386089	0.287183331
40	Semidetached	16	114	45.15581069	50.66910377	49.4203871	46.73204029	58.50537412	0.338785879	0.472204051	0.437328607	0.400657409	0.384568021
41	Semidetached	177.9	1248	526.854542	716.8112036	684.1529455	598.9464703	889.6905994	0.324168752	0.361787241	0.348790005	0.353130887	0.226814444
42	Semidetached	302	2400	903.8957475	1282.977024	1218.705627	1049.008563	1617.886406	0.349714369	0.395612304	0.379983023	0.382217994	0.257445725
43	Semidetached	282.1	1368	843.1841016	1190.295075	1131.360492	975.9522038	1497.944311	0.215220555	0.110416072	0.133542179	0.194590975	0.07504094
44	Semidetached	284.7	973	851.1115377	1202.368128	1142.741418	985.4804395	1513.554342	0.070276904	0.200372979	0.134676644	0.008709371	0.438887904
45	Semidetached	79	400	230.1923542	293.4952265	282.177511	253.5354958	355.5157075	0.238155223	0.226322644	0.227397404	0.248623496	0.087856478
46	Semidetached	423	2400	1274.613271	1858.598772	1760.145552	1498.80844	2367.580725	0.263059148	0.191746268	0.205819847	0.254962112	0.010671345
47	Semidetached	190	420	563.4299571	770.6198698	735.0746069	642.17256	958.3667761	0.191581443	0.709587832	0.579137135	0.359178972	1.012642269
48	Semidetached	47.5	252	137.0057338	167.7158904	161.9884201	147.9347538	200.0801293	0.25599914	0.284291639	0.275749761	0.280398024	0.162764674
49	Semidetached	21	107	59.59021358	68.33645818	66.489412	62.32781667	79.55142865	0.248569067	0.307140286	0.292281999	0.28348049	0.202657676
50	Semidetached	78	571.4	227.2206281	289.4111805	278.2828569	250.1381106	350.4346823	0.337914994	0.419479343	0.396021061	0.381758528	0.305499827
51	Semidetached	11.4	98.8	31.95613224	34.89848534	34.14243054	32.63727707	39.88803686	0.379548682	0.549759995	0.505219065	0.454701304	0.471057195
52	Semidetached	19.3	155	54.67385751	62.27651216	60.63931208	56.99763962	72.31359284	0.363115909	0.508483643	0.469977104	0.429313566	0.421433946
53	Semidetached	101	750	295.7461247	384.56057	368.9146149	328.8728453	469.2705854	0.339781899	0.414164687	0.39226389	0.381260451	0.29570165
54	Semidetached	219	2120	651.2747765	900.9480882	858.2933961	746.4178187	1125.231886	0.388657948	0.488770814	0.45945165	0.439935048	0.370691892
55	Semidetached	50	370	144.3645847	177.4509164	171.3118939	156.1927658	212.019734	0.342112076	0.442342489	0.414560048	0.392365168	0.337309217
56	Semidetached	227	1181	675.5502137	937.2159793	892.5559551	775.3237071	1171.788952	0.240099348	0.17545844	0.188551061	0.233238106	0.006161497
57	Semidetached	70	278	203.4750961	256.9325255	247.2937859	223.0542514	310.0994284	0.150390184	0.06441494	0.085270494	0.134202026	0.0912178
58	Semidetached	0.9	8.4	2.397941705	2.137361598	2.139389301	2.21816832	2.263779986	0.40085175	0.633719362	0.575379936	0.499698061	0.577096882
59	Semidetached	980	4560	3003.045128	4683.385694	4401.887006	3648.884318	6118.227841	0.191546422	0.022999526	0.026768252	0.135668322	0.269956139
60	Embedded	350	720	1050.656294	1508.989058	1431.493822	1226.364978	2293.603362	0.257636363	0.931445415	0.762879487	0.477530306	1.726592578
61	Embedded	70	458	203.4750961	256.9325255	247.2937859	223.0542514	372.119314	0.311765221	0.37316016	0.355164186	0.348314767	0.148134808
62	Embedded	271	2460	809.3566475	1138.878677	1082.881011	935.3327904	1717.816323	0.376427204	0.45648501	0.432169049	0.420832941	0.238343539
63	Organic	90	162	262.9290795	338.7489846	325.3043116	291.0680245	494.3282862	0.349513664	0.927386648	0.778215608	0.5409703	1.620613248
64	Organic	40	150	114.9773918	138.8280528	134.294644	123.3199136	197.7193694	0.130984555	0.063307701	0.080830232	0.120771858	0.251322012
65	Embedded	137	636	403.6140166	537.7790326	514.4857857	454.1912481	794.7227901	0.204981976	0.131270161	0.147498386	0.194100853	0.197155667
66	Embedded	150	882	442.7151108	594.1712401	567.9718418	499.956621	880.4496988	0.279409096	0.277385993	0.274863649	0.29411276	0.001388592
67	Embedded	339	444	1016.985951	1456.903913	1382.480813	1185.58627	2212.315619	0.723975492	1.939117851	1.631772944	1.134092517	3.14632734
68	Organic	240	192	715.0340574	996.4226879	948.4658673	822.4232622	1497.476893	1.528240136	3.561246275	3.041623175	2.229465599	5.371493465
69	Semidetached	144	576	424.6596557	643.8247166	543.2312191	478.8037685	840.7580661	0.14739919	0.100088557	0.043919269	0.114576808	0.363123042
70	Semidetached	151	432	445.7257738	678.3339381	572.1041409	503.4870075	887.0852885	0.017824442	0.484684832	0.250371289	0.112360366	0.832216153
71	Semidetached	34	72	97.41363665	131.5817666	112.4746733	103.8216326	164.547998	0.198014586	0.703395855	0.433978441	0.300095674	1.0154572
72	Semidetached	98	300	286.7885833	421.6164889	356.9758943	318.5371536	544.2605949	0.024705349	0.344580052	0.146617968	0.041955758	0.643219567
73	Semidetached	85	300	248.0381969	360.5202988	305.6379581	273.9720903	463.4093718	0.097168572	0.17147418	0.014508346	0.058909836	0.430311346
74	Semidetached	20	240	56.69723201	73.40098691	63.04272353	59.18919675	90.34108904	0.42847022	0.590038171	0.569212573	0.511543898	0.492627249
75	Semidetached	111	600	325.6422147	483.5308419	408.9390762	363.4533828	626.5220965	0.256524529	0.164997974	0.245831722	0.267691922	0.03492076
76	Semidetached	162	756	478.8688376	732.884303	617.720654	542.4104915	960.4469747	0.205648918	0.025989871	0.141205893	0.191835022	0.21364168
77	Semidetached	352	1,200	1056.780469	1720.940415	1440.42048	1233.787487	2308.418979	0.066955131	0.368999461	0.154670509	0.019118087	0.729709161
78	Semidetached	165	97	487.9158028	747.8271748	630.2113563	553.0535416	980.5692805	2.260863561	5.703124728	4.243702754	3.192374791	7.196079707
79	Embedded	60	409	191.2573887	245.7736275	209.0134606	189.4583955	312.6312157	0.324750594	0.339223513	0.377480705	0.36447127	0.186140195
80	Embedded	100	703	322.0356462	431.0909483	413.5889281	356.9183725	556.8284627	0.330566509	0.328766279	0.317816995	0.334266607	0.164261045
81	Embedded	32	1350	100.7295571	123.0931485	119.3129353	106.7880684	153.6529497	0.564485163	0.772496907	0.703770677	0.625289557	0.70008457
82	Embedded	53	480	168.5253872	214.4234901	206.8968496	182.2120314	271.7397422	0.39583232	0.470291736	0.4392409	0.421245897	0.342761674
83	Embedded	41	599	129.7010465	161.6706359	156.3565545	138.8375714	203.3139531	0.477917131	0.62058424	0.570485375	0.521619848	0.521856389
84	Embedded	24	430	75.11374459	89.70182314	87.17247341	78.74311825	111.0094957	0.503443293	0.672682443	0.615495001	0.554659123	0.586052322
85	Embedded	165	4178.2	536.7073831	747.8271748	714.2395372	606.574852	980.5692805	0.531642931	0.697864368	0.640030989	0.580425417	0.604597259
86	Embedded	65	1772.5	207.5274598	268.3944904	258.4981124	226.1747655	342.2264084	0.538580113	0.721291782	0.659412952	0.592358158	0.637470317
87	Embedded	70	1645.9	223.8226057	291.1901955	280.2662907	244.6401467	372.119314	0.527047336	0.699619256	0.640542696	0.578076092	0.611389964
88	Embedded	50	1924.5	158.8010432	201.1110386	194.1534798	171.3081948	254.4236808	0.559665557	0.761174652	0.694116661	0.618559229	0.685560037
89	Embedded	7.25	648	22.15382462	24.04034585	23.6155285	22.16495495	28.70141934	0.589145319	0.818465596	0.743865451	0.655774685	0.755009072
90	Embedded	233	8211	763.1448644	1093.101316	1040.768679	874.1758901	1448.21692	0.553305521	0.736842514	0.674146703	0.606710945	0.650663577
91	Embedded	16.3	480	50.62153417	58.61047875	57.15649391	52.27277753	71.6957512	0.545668467	0.746210611	0.680073306	0.6050558	0.672000743
92	Embedded	6.2	12	18.88615153	20.23949921	19.90986142	18.78069688	24.0504959	0.350046036	0.583631194	0.508867752	0.383674432	0.793324314
93	Embedded	3	48	9.006740006	9.1075651	9.017957681	8.706434013	10.58932907	0.495539346	0.688720201	0.626961181	0.555840236	0.615717292

to the MOPSO, MOWOA, MOEA-D, and NSGA-III for the following ten objective problems, i.e., Biobj1, Biobj2, Biobj3, Biobj4, Triobj1, Triobj2, Triobj3, and Triobj4. Thus, the proposed approach gives significantly encouraging GD values for all four algorithms. Furthermore, this algorithm also achieves a better convergence rate and diversity for ZDT(Biobj) and DTLZ(Triobj) series problems.

The convergence metric IGD for the five algorithms is given in Table 4. This table shows that the proposed algorithm achieves better IGD "mean" and "SD" values compared to the MOPSO, MOWOA, MOED, and MODE for all the objective problems. Furthermore, the mean and SD values of the IGD for designing the Pareto front provide encouraging performances for the Biobj1, Biobj2, Biobj3, and Biobj4. This algorithm is also tested on DTLZ (Triobj) series like Triobj1, Triobj2, Triobj3, and Triobj4, and found to achieve a better convergence rate and diversity for the proposed algorithm.

Table 7
Two objectives problem: Comparison with Proposed algorithm and other optimization algorithm.

Software model	Proposed algorithm		MOEA-D		MOPSO		MOWOA		NSGA-III	
	Pridiction	MMRE	Pridiction	MMRE	Pridiction	MMRE	Pridiction	MMRE	Pridiction	MMRE
Organic	0.0045725	0.0016003	0.003497	0.002085	0.009616	0.008635	0.015985	0.0176041	0.0437245	0.0271561
Semidetached	0.0025843	0.0009045	0.0197704	0.0078472	0.039098	0.011549	0.0203382	0.0099498	0.0247138	0.0153485
Embedded	0.0057841	0.00202443	0.044248	0.007549	0.005901	0.005847	0.0455208	0.0222687	0.0553101	0.0343517

The space metric Sp for the five algorithms is given in Table 5. This table reveals that the proposed algorithm achieves better Sp "mean" and "SD" values than the MOPSO, MOWOA, MOEA-D, and NSGA-III for the following eight objective problems, i.e., Biobj1, Biobj2, Biobj3, Biobj4, Triobj1, Triobj2, Triobj3, and Triobj5. From this table, the proposed approach achieves significantly encouraging Sp values for all four algorithms. This algorithm also achieves a better convergence rate and diversity of the ZDT(Biobj) and DTLZ(Triobj) series problems.

4.2. Experimental results for software cost estimation

We have applied the proposed HFMODE algorithm on the COCOMO model for estimating the software cost by tuning the parameters of this model. This framework is being used by organic, semidetached, and embedded types of projects, where we have to optimize the various constraints for effective cost estimation. This paper incorporates and all the tuned parameters as given in Table 1, and Table 2 to maintain the evolutionary process of the proposed HFMODE algorithm. The NASA software project dataset is used for the testing of the proposed algorithm [42].

4.2.1. Discussion

The proposed approach is evaluated on NASA Dataset [42] to compare the Pareto Front on the biobj and triobj series. Further, this approach is also applied to NASA Dataset for calculating the Effort, Prediction, MMRE, and RMSE. Further, the detailed description of the analysis of each software cost estimation objective result is as follows.

I. Comparison of Effort for software cost estimation Models: The developed COCOMO based proposed algorithm is capable of an effective estimation of software development costs. In Eq. (10), the calculated amount of estimated Effort for the actual dataset (NASA Project 93) is shown in Table 6. From this Table, it is clear that the proposed method achieves better effort values (person/months) than MOPSO, MOWOA, MOEA-D, and NSGA-III standard evolutionary algorithms. The results of the Effort obtained by four algorithms are given in Table 6. The proposed algorithm provides diversity, convergence rate, and effective effort values (person/months) to the software model in most of the projects, i.e., organic, semidetached, and embedded, shown in Table 6.

II. Two objectives based Pareto front on software cost parameters like Prediction and MMRE: The proposed algorithm performance is verified for the bi-objective problems in software cost estimation such as Prediction and MMRE. The proposed algorithm reduces the costs and Effort of software projects by using parameter tuning with the Pareto front. Further, the proposed algorithm incorporates the bi-objectives problem, representing the X-axis as a prediction, and the Y-axis as an MMRE. The investigator has created the Pareto front on convex, convex disconnected, and non-convex. So, the proposed algorithm creates the Pareto front in between zero and one. Therefore, the proposed algorithm is to provide a feasible solution for the global search space and also provides the diverse solutions and rate of convergence which depicts a faster estimation model that reaches the desired value as calculated from Table 2, and biobj series functions. The proposed algorithm achieves the Pareto front trade-off between Prediction and MMRE for two objective functions are shown in Fig. 3. Also, from Fig. 3, it is evident that the proposed approach gives encouraging results with a well-distributed Pareto front between Prediction and MMRE. From Table 7, it is clear that the proposed algorithm achieves a better Prediction rate and minimizes the error rate, i.e., MMRE, in comparison to MOPSO, MOWOA, MOEA-D, and NSGA-III standard algorithms. When estimating software cost, the proposed algorithm optimizes the tuning parameters, minimizes the error rate (MMRE), and accurately gives the prediction.

III. Three objectives based Pareto front on software cost parameters like Prediction, MMRE, and RMSE: The proposed algorithm performance is verified for the tri-objective problems in software cost estimation such as Prediction, MMRE, and RMSE. Software cost estimation is to reduce effort and time by using the proposed algorithm. The proposed algorithm incorporates the tri-objectives problem, representing the X-axis called the Prediction, Y-axis called the MMRE, and Z-axis called RMSE. We have created the Pareto front on linear, concave, and triobj4. So, the proposed algorithm creates the Pareto front in between zero and one. Therefore, the proposed algorithm provides a feasible solution for the global search space. It also provides the diverse solutions and rate of convergence which depicts a faster estimation model that reaches the desired value as calculated from Table 2, Triobj series functions. The trade-off between Prediction, MMRE, and RMSE for triobj functions is shown in Fig. 4. From Fig. 4, it is evident that the proposed algorithm gives encouraging results with a well-distributed Pareto front among Prediction, MMRE, and RMSE. From Table 8, it is clear that MOSADE achieves an accurate Prediction rate and minimizes the error rate, i.e., MMRE and RMSE, in comparison to MOPSO, MOWOA, MOEA-D, and NSGA-III standard evolutionary algorithms. During the software cost estimation, the proposed algorithm optimizes the tuning parameters with accurate prediction and minimizes errors like MMRE and RMSE.

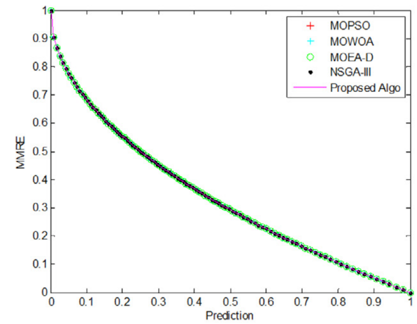
IV. Comparison of Prediction, MMRE, and RMSE for software cost estimation Models:

The proposed algorithm is capable of an accurate prediction of estimation of software development costs. From the calculation of prediction using Eq. (12), it is clear that the proposed algorithm achieves better accurate prediction values in comparison to MOPSO, MOWOA, MOEA-D, and NSGA-III standards algorithms. The results of the accurate prediction obtained by five algorithms are given in Tables 7 and 8.

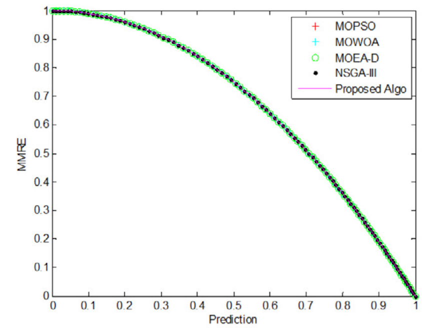
An error calculation of MMRE and RMSE of the proposed approach using Eqs. (13) and (14), It is clear that the proposed algorithm achieves lower error values in comparison to MOPSO, MOWOA, MOEA-D, and NSGA-III standard evolutionary algorithms as given in Tables 7 and 8. Moreover, the proposed algorithm provides diversity, convergence rate, and minimum error values of the software model in most projects, i.e., organic, semidetached, and embedded.

5. Conclusion

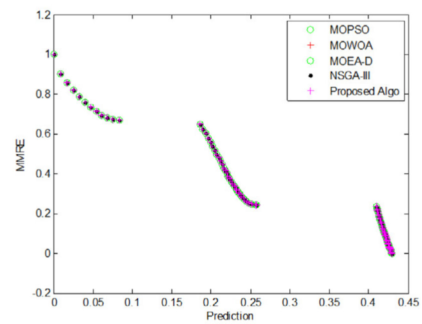
In this work, a novel multi-objective differential evolution is proposed. The investigation ability of MODE is enhanced by a new mutation operator capable of doing better diversification and intensification. As a result, sufficient diversity is achieved as well, along the convergence speed of the optimal Pareto front was also



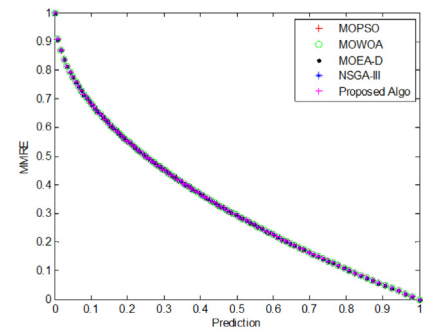
(a) Biobj1



(b) Biobj2

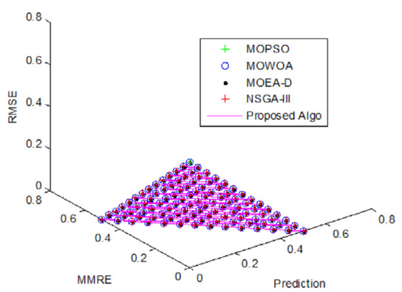


(c) Biobj3

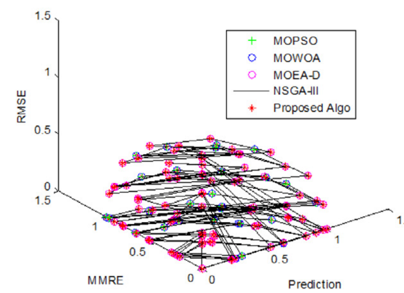


(d) Biobj3

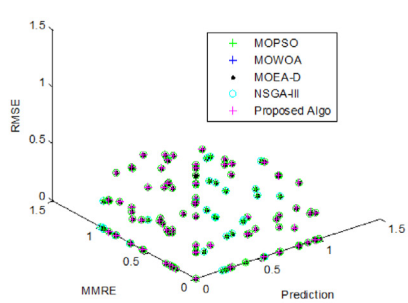
Fig. 3. Proposed algorithm comparison with standard algorithms for Pareto front on Prediction, and MMRE.



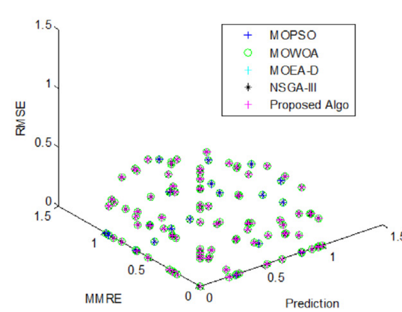
(a) Triobj1



(b) Triobj2



(c) Triobj3



(d) Triobj3

Fig. 4. Proposed algorithm comparison with standard algorithms for Pareto front on Prediction, MMRE, v/s RMSE.

Table 8
Three objectives problem: Comparison with proposed algorithm and other optimization algorithm.

Software model	Proposed algorithm			MOEA-D			MOPSO			MOWOA			NSGA-III		
	Pridiction	MMRE	RMSE	Pridiction	MMRE	RMSE	Pridiction	MMRE	RMSE	Pridiction	MMRE	RMSE	Pridiction	MMRE	RMSE
Organic	0.03654	0.023751	0.025578	0.036174	0.029373	0.028939	0.035443	0.026126	0.030693	0.034311	0.029451	0.02890314	0.030350	0.036423	0.032701
Semidetached	0.03587	0.0233155	0.025109	0.035511	0.028835	0.028409	0.034793	0.025647	0.030130	0.033681	0.028911	0.02837317	0.028192	0.035755	0.032102
Embedded	0.02871	0.0186615	0.020097	0.028422	0.023079	0.022738	0.027848	0.020527	0.024116	0.026958	0.023140	0.02270961	0.026689	0.028618	0.025694

improved. It is worth mentioning that the result of the proposed algorithm performed well compared to the latest variants of multi-objective DE optimization methods (tested bi-objective and tri-objective functions). For the sake of applicability, we have also applied the proposed algorithm on software cost estimation models to estimate its cost during the development phase. The proposed algorithm has been applied to explore the multi-objective software cost estimation problems: two objectives problems like prediction, and MMRE, and three objectives problems like prediction, MMRE, and RMSE. The proposed method optimizes the accurate prediction and reduces the effort and error rate of the software model in most projects, i.e., organic, semidetached, and embedded models. Moreover, the performance of the proposed method is commendable in maximum variants of functions. In the future research direction, we will work on a dynamic-based mutation and crossover approach that can be adopted in the initialize process, i.e., framing the population.

CRediT authorship contribution statement

Shailendra Pratap Singh: Methodology, Experiment, Writing manuscript. **Gaurav Dhiman:** Methodology, Experiment, Writing manuscript. **Prayag Tiwari:** Methodology, Experiment, Final proofreading. **Rutvij H. Jhaveri:** Experiment, Writing manuscript, Final proofreading.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Academy of Finland (grants 336033, 315896), Business Finland (grant 884/31/2018), and EU H2020 (grant 101016775).

References

[1] Storn Rainer, Kenneth Price, Differential Evolution-a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, International Computer Science Institute, Berkeley. Berkeley, CA, 1995, 1995.
[2] Victor H. Cantú, Catherine Azzaro-Pantel, Antonin Ponsich, Constraint-handling techniques within differential evolution for solving process engineering problems, Appl. Soft Comput. 108 (2021).
[3] H.A. Abbass, R. Sarker, C. Newton, PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems, in: Proceedings of the Congress on Evolutionary Computation 2001 (CEC2001), Vol. 2, 2001, pp. 971–978.
[4] Ryoji Tanabe, Hisao Ishibuchi, Review and analysis of three components of the differential evolution mutation operator in MOEA/D-DE, Soft Comput. (2019) 1–15.
[5] Z. Wang, Q. Zhang, A. Zhou, M. Gong, L. Jiao, Adaptive replacement strategies for MOEA/d, IEEE Trans. Cybern. 46 (2) (2016) 474–486.
[6] Y.Y. Tan, Y.C. Jiao, H. Li, X.K. Wang, A modification to MOEA/D-DE for multiobjective optimization problems with complicated Pareto sets, Inform. Sci. 213 (2012) 14–38.
[7] Ryoji Tanabe, Hisao Ishibuchi, A framework to handle multi-modal multi-objective optimization in decomposition-based evolutionary algorithms, IEEE Trans. Evol. Comput. (2020).
[8] S. Jiang, S. Yang, A strength Pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization, IEEE TEVC 21 (3) (2017) 329–346.
[9] Soufiane Ezghari, Azeddine Zahi, Uncertainty management in software effort estimation using a consistent fuzzy analogy-based method, Appl. Soft Comput. 67 (2018) (2018) 540–557.
[10] Alaa F. Sheta, Sultan Aljahdali, Software effort estimation inspired by COCOMO and FP models: A fuzzy logic approach, Int. J. Adv. Comput. Sci. Appl. 4 (11) (2013).
[11] Sun-Jen Huang, Nan-Hsing Chiu, Applying fuzzy neural network to estimate software development effort', Appl. Intell., Springer Science+Business Media 30 (2) (2007) 73–83, LLC.
[12] Ansif Arooj, Muhammad Shoaib Farooq, Aftab Akram, Razi Iqbal, Ashutosh Sharma, Garav Dhiman, Big data processing and analysis in internet of vehicles: Architecture, taxonomy, and open research challenges, Arch. Comput. Methods Eng. (2021) 1–37.
[13] Yongfu Shao, Jue Wu, Hongping Ou, Min Pei, Li Liu, Ali Akbar Movassagh, Ashutosh Sharma, Gaurav Dhiman, Mehdi Gheisari, Alia Asheralieva, Optimization of ultrasound information imaging algorithm in cardiovascular disease based on image enhancement, Math. Probl. Eng. 2021 (2021).
[14] A.K.M. Haque, Bharat Bhushan Bahalul, Gaurav Dhiman, Conceptualizing smart city applications: Requirements, architecture, security issues, and emerging trends. Expert Systems.
[15] Celestine Iwendi, Zunera Jalil, Abdul Rehman Javed, Thippa Reddy, Rajesh Kaluri, Gautam Srivastava, Ohyun Jo, Keysplitwatermark: Zero watermarking algorithm for software protection against cyber-attacks, IEEE Access 8 (2020) 72650–72660.
[16] Celestine Iwendi, Praveen Kumar Reddy Maddikunta, Thippa Reddy Gadekallu, Kuruva Lakshmanna, Ali Kashif Bashir, Md Jalil Piran, A meta-heuristic optimization approach for energy efficiency in the IoT networks, Softw. - Pract. Exp. (2020).
[17] Mehrdad Ahmadi Kamarposhti, Ilhami Colak, Celestine Iwendi, Shahab S. Band, Ebuka Ibeke, Optimal coordination of PSS and SSSC controllers in power system using ant colony optimization algorithm, J. Circuits Syst. Comput. (2021).
[18] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.
[19] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Trans. Evol. Comput. 7 (2) (2003) 117-13.
[20] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evol. Comput. 8 (2) (2000) 173–195, 173–195.
[21] J.R. Schott, Fault Tolerant Design using Single and Multicriteria Genetic Algorithm Optimization, Technical report, DTIC Document, 1995.
[22] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[23] Gai GeWang lZi MinGu, Improving NSGA-III algorithms with information feedback models for large-scale many-objective optimization, Future Gener. Comput. Syst. 107 (2020) 49–69.
[24] Y. Yuan, H. Xu, B. Wang, B. Zhang, X. Yao, Balancing convergence and diversity in decomposition-based many-objective optimizers, IEEE TEVC 20 (2) (2016) 180–198.
[25] H. Ishibuchi, Y. Setoguchi, H. Masuda, Y. Nojima, Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes, IEEE TEVC 21 (2) (2017) 169–190.
[26] Bili Chen, Yangbin Lin, Wenhua Zeng, Defu Zhang, Yain-Whar Si, Modified differential evolution algorithm using a new diversity maintenance strategy for multi-objective optimization problems, Appl. Intell. 43 (1) (2015) 49–73, 2015.
[27] Singh P. Shailendra, Kumar Anoj, Pareto based differential evolution with homeostasis based mutation, J. Intell. Fuzzy Systems 32 (5) (2017) 3245–3257.
[28] B. Boehm, et al., Software Cost Estimation with COCOMO II, Prentice Hall PTR, 2000.
[29] G. Dhiman, SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications, Knowl.-Based Syst. (2021).
[30] Adel Got, Abdelouahab Moussaoui, Djaafar Zouache, A guided population archive whale optimization algorithm for solving multiobjective optimization problems, Expert Syst. Appl. 141 (2020).
[31] Romit Beed, Arindam Roy, Sunita Sarkar, Durba Bhattacharya, A hybrid multiobjective tour route optimization algorithm based on particle swarm optimization and artificial bee colony optimization, Comput. Intell. 36 (2020).
[32] Xu Chen, Wenli Du, Feng Qian, Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization, Chemometr. Intell. Lab. Syst. 136 (2014) (2014) 85–96.
[33] X. Zhang, Y. Tian, R. Cheng, Y. Jin, An efficient approach to nondominated sorting for evolutionary multiobjective optimization, evolutionary computation, IEEE Trans. 19 (2) (2015) 201–213, (2015) 201–13.
[34] Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable multi-objective optimization test problems, in: Proceedings of the Congress on Evolutionary Computation (CEC-2002), (Honolulu, USA), 2002, (2002) pp. 825–830.
[35] K. Deb, L. Thiele, M. Laumanns, et al., Scalable Test Problems for Evolutionary Multi-Objective Optimization, Technical report, Computer Engineering and Networks Lab (TIK, ETH Zurich, Switzerland, 2001.

[36] D.A. Van Veldhuizen, G.B. Lamont, Multiobjective Evolutionary Algorithm Research: A History and Analysis, Technical report, Department of Electrical and Computer Engineering. Graduate School of Engineering, Air Force Inst. Technol, Wright Patterson, 1998.

[37] D.A. Veldhuizen, G.B. Van Lamont, On measuring multiobjective evolutionary algorithm performance. In: Proceedings of the Congress on Evolutionary Computation 2000 (CEC'2000), Vol. 1, 2000, pp. 204–211.

[38] Sultan Aljahdali, Alaa F. Sheta, Software effort estimation by tuning COOCMO model parameters using differential evolution, in: ACS/IEEE International Conference on Computer Systems and Applications-AICCSA 2010, 2010, pp. 1–6.

[39] C.H.V.M.K. Hari, P.V.G.D Prasad Reddy, A fine parameter tunning for COCOMO 81 software effort estimation using particle swarm optimization, J. Softw. Eng. 5 (1) (2011) 38–48.

[40] Alifia Puspaningrum, Riyanarto Sarno, A hybrid cuckoo optimization and harmony search algorithm for software cost estimation, Procedia Comput. Sci. 124 (2017) (2017) 461–469.

[41] S.P. Singh, A. Kumar, Multiobjective differential evolution using homeostasis based mutation for application in software cost estimation, Appl. Intell. 48 (3) (2018) 628–650.

[42] NASA Data SET: Project 93, http://promise.site.uottawa.ca/SERepository/datasets/cocomonasa_2.arff.

[43] Tirimula Rao Benala, Rajib Mall, DABE: Differential evolution in analogy-based software development effort estimation, Swarm Evol. Comput. (2018) 158–172.

[44] Shailendra Pratap Singh, Cost estimation model using enhance-based differential evolution algorithm, Iran J. Comput. Sci. (2019).

[45] John Chibuike Nwaiwu, Samuel Adebayo Oluwadare, Analytic study of fuzzy-based model for software cost estimation, in ocri, 2016, pp. 22–30.

[46] P.V.G.D Prasad Reddy, C.H.V.M.K. Hari, T. Srinivasa Rao, Multi objective particle swarm optimization for software cost estimation, Int. J. Comput. Appl. 32 (3) (2011) 13–17.

[47] Prerna Singala, Charan Kumari, Prabha Sharmaa, Estimation of software development effort: A differential evolution approach, Procedia Comput. Sci. 167 (2020) 2643–2652.

[48] Thamarai Ilango, S. Murugavalli, A study to improve the software estimation using differential evolution algorithm with analogy, J. Theor. Appl. Inf. Technol. (2017) 5587–5597.

[49] Leandro L. Minku, Xin Yao, 'software effort estimation as a multi-objective learning problem, ACM Trans. Softw. Eng. Methodol. (TOSEM) 22 (4) (2013).

[50] S.P. Singh, A. Kumar, Software cost estimation using homeostasis mutation based differential evolution, in: IEEE Conference 11^{th} International Conference on Intelligent Systems and Control (ISCO), 2017.

[51] S.P. Singh, V.P. Singh, A.K. Mehta, Differential evolution using homeostasis adaption based mutation operator and its application for software cost estimation, J. King Saud Univ.-Comput. Inf. Sci. (2018).

[52] J. Bader, E. Zitzler, HypE: An Algorithm for Fast Hyper Volume-Based Manyobjective Optimization, TIK Report 286, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2008.