Int. J. Mach. Learn. & Cyber.
DOI 10.1007/s13042-017-0695-3

ORIGINAL ARTICLE

Evolutionary dynamic multi-objective optimization algorithm based on Borda count method

Maysam Orouskhani[1] · **Mohammad Teshnehlab**[2] · **Mohammad Ali Nekoui**[2]

Received: 29 December 2016 / Accepted: 9 May 2017

Abstract In this paper, a novel dynamic multi-objective optimization algorithm is introduced. The proposed method is composed of three parts: change detection, response to change, and optimization process. The first step is to use Sentry solutions to detect the environmental change and advises the algorithm when a change occurs. Then, to increase the diversity of solutions, the worst solutions should be elected and removed from population and re-initialized with new solutions. The main idea is to use Borda count method which is an optimal rank aggregation technique that ranks the solutions in order of preference and nominates the worst solutions that should be removed. The last step is optimization process which is done by multi-objective Cat swarm optimization (CSO) in this paper. CSO utilizes the population that has been improved from the previous step to estimate the best solutions and converges to optimal Pareto front. The performance of the proposed algorithm is tested on dynamic multi-objective benchmarks, and the results are compared with the ones achieved by previous algorithms. The simulation results indicate that the proposed algorithm can effectively track the time-varying optimal Pareto front and achieves competitive results in comparison with traditional approaches.

Keywords Dynamic multi-objective optimization · Borda count ranking method · Cat swarm optimization

✉ Mohammad Teshnehlab
Teshnehlab@eetd.kntu.ac.ir

[1] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

[2] Industrial Control Centre of Excellence, Electrical Engineering Department, K. N. Toosi University, Tehran, Iran

1 Introduction

Many optimization problems require the simultaneous optimization of some functions which may be in conflict with one another. These optimization problems are called: multi-objective optimization (MOO). The best solutions in this case are referred to as non-dominated solutions or Pareto optimal. The set of Pareto optimal solutions is called the Pareto optimal front (POF), while the corresponding set of decision variables is referred to as the Pareto optimal set (POS). Some algorithms have been proposed to solve this type of engineering problems [42–44]. Furthermore, Some real-world multi-objective optimization problems exhibit dynamic behavior in objective functions, constraints, the number of variables or objectives [8, 31, 32, 35, 40]. In this case, the optimal solutions are likely to change over time and the main goal is to approximate and track the time-varying Pareto optimal front. Problems with such characteristics are known as dynamic multi-objective optimization problems (DMOOP) [18, 19]. Solving DMOOPs requires managing two most important issues: loss of diversity and slow convergence. To manage these, some studies have been done by researchers. DNSGA-II [8] re-initializes randomly some percentage of population and creates new solutions after a change occurred to increase the diversity of solutions. DVEPSO with archive management [16] applies various ways to manage the archive solutions. Also, QICCA was proposed by [37] and adopts entire cloning and evolves the theory of quantum to design a quantum updating operation, which improves the searching ability of the algorithm. As a prediction-based technique, a forward-looking approach was proposed by [14] that hybrid a forecasting technique with an evolutionary algorithm to estimate the location of the next optima. Finally, dMOEAD-DI [27] decomposes a dynamic multi-objective

problem into several dynamic single sub-problems and an adaptive diversity introduction is adopted to respond to the environment change.

How to response to change is the main issue addressed in this study. When a change occurs, optimization algorithm should be able to give a true and fast response to change to explore new positions to improve the diversity of solutions; otherwise, it is likely that current non-dominated solutions may be changed into dominated in a new environment. Two most used strategies for handling this issue are predictive models and re-initialization methods. In predictive models such as linear regression, it is assumed that the size of the change is so small that the information of consecutive steps is depended on together. By this method, computational cost and time complexity will be increased. The latter, re-initializes some percentage of the population where solutions are randomly selected to be re-initialized or mutated to increase the chance of exploration in search space. Here the main drawback is the random selection of solutions that may mislead the trajectory of the algorithm. Also, this random selection may pick and remove the elite solutions from the population while using a wise approach can determine the best solutions to be re-initialized and enhance the rate of diversity. That's why the paper applies a novel method to do a wise selection of the solutions in response to change process.

This paper introduces a novel dynamic multi-objective optimization algorithm composed of three parts: (1) change detection by sentry solutions, (2) response to change to enhance the diversity of solutions and improve the convergence rate. Here contrary to traditional approaches in which the selection process of individuals for re-initialization is random, for the first time in dynamic multi-objective algorithms, this selection is done by Borda count. (3) The last part is using a multi-objective version of Cat swarm optimization to produce the diverse solutions and estimate the POF.

The rest of paper is written as follows: Sect. 2 presents background information of DMOO and pure cat swarm optimization. The proposed algorithm included of three parts is presented in Sect. 3 and information about the experiments is provided in Sect. 4. Finally, conclusions about this research are presented in the last section.

2 Background

Section 2.1 investigates the theory and background information of the multi-objective optimization. The definition of the dynamic multi-objective problem (DMOOP), types of dynamic environment, challenges and issues, and a review of DMOO algorithms are considered in Sect. 2.2. In Sect. 2.3 single cat swarm optimization is introduced.

2.1 Multi-objective optimization

Multi-objective optimization is concerned with optimization problems involving more than one function to be optimized simultaneously. Let $S \subseteq R^n$ denote the n-dimensional search space and $F \subseteq S$ the feasible space. Let $X = (x_1, x_2, \ldots, x_n) \in S$, referred to as a decision vector and single objective function $f_k(x)$ is defined as: [15] $f_k{:}R^n \rightarrow R$. Let $F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \in O \subseteq R^m$ be an objective vector containing m objective functions; 'O' is referred to as the objective space. The search space, 'S' is also referred to as the decision space. Definition of the multi-objective optimization for minimization problems is as follows where 'g' and 'h' are inequality and equality constraints respectively [5].

$$\begin{aligned} &\text{Minimize } F(x), \text{St:} g_i(x) \leq 0 \quad i = 1, 2, \ldots, n_g \\ &h_j(x) = 0 \quad j = 1, 2, \ldots, n_h, \; x \in [x_{min}, x_{max}]^n. \end{aligned} \tag{1}$$

2.2 Dynamic multi-objective optimization

Some multi-objective optimization problems change over time and need to be optimized in a time-varying environment and known as dynamic multi-objective problem. In these problems, dynamic behaviours are revealed in objective functions, constraints, or in the number of variables and objectives. This section investigates the definitions, types, issues, and approaches to solving dynamic multi-objective optimization problems.

2.2.1 Dynamic multi-objective problem

Dynamic single-objective optimization problem is included of just one fitness function that changes over time while dynamic multi-objective optimization problem (DMOOP) is involving more than one objective function to be optimized simultaneously. In other words, DMOOP is as an extension of the static multi-objective problem with an additional time parameter is defined as [15]:

$$\text{Minimize } F(X, W(t)), W(t) = (w_1(t), \ldots, w_N(t)), \tag{2}$$

where F(X) is a vector of fitness functions and W(t) is a vector of time-dependent control parameters of an objective function at time t. Pareto optimal set (POS) and Pareto optimal front (POF) are best compromised solutions of the problem in decision and objective space respectively. In this case, the definition of the dynamic Pareto-optimal front (POF) can be written as follows:

$$POF^* = \{F(t) = ((f_1(x^*, w_1(t)), \ldots, (f_N(x^*, w_N(t)))) | x^* \in P^*\}.$$

2.2.2 Dynamic environment types

This section discusses the categorization of DMOOPs, as well as the possible influences of a change in the environment on the POF [11].

Type I: the optimal set of decision variables (POS) changes, but the corresponding objective function values (POF) remains unchanged.

Type II: both the POS and the POF change.

Type III: just the POF changes.

Type IV: POS and the POF remain unchanged. Also when a change occurs in the environment, the POF can change as one the following way over time [15]:

- Existing solutions in the POF become dominated.
- The shape of the POF remains the same, but its location in the objective space change over time.
- The shape of the POF changes over time.

2.2.3 Dynamic multi-objective algorithms

The proposed methods for solving DMOOPs are classified by Helbig as follows: [18, 19, 22].

2.2.3.1 Adapted approaches Multi-objective algorithms adapted for solving dynamic multi-objective optimization problems [1–3, 10]. For example, NSGA-II [8], as one of the most successful algorithms was adapted for DMOO. In this algorithm, some solutions are re-initialized by randomly created individuals (DNSGA-II-A) or are replaced with mutated solutions (DNSGA-II-B). Also Camara et al. [2, 3; 2] proposed an algorithm that uses an island model and allows the communication and coordination among the subpopulations in the different islands.

2.2.3.2 Convert to static' approaches These algorithms try to convert a dynamic multi-objective problem into various static multi-objective optimization problems. In [6] the period of the DMOOP is divided into several smaller time intervals. For each time interval, a statics optimization problem is defined by limiting the DMOOP to the specific time interval. Each static problem is then transformed into a new two-objective optimization problem where one objective is to increase the diversity of the solutions and the other objective is to increase the quality of the found non-dominated solutions.

2.2.3.3 Prediction-based approaches These approaches incorporate a forecasting technique with an evolutionary algorithm to predict the population of next generations. Also some authors tried to estimate the time-varying POF, and environmental changes [24, 28–30, 34, 39, 45, 46]. Here, authors try to predict the most important section of a problem. Some try to assess the population of the next generation while some prefer to determine the changing size of the environment. For example, Koo et al. [24] applied the history of previously discovered solutions using a weighted average approach to assessing the size of the next change while Zhou and Zhang [45] tried to predict the next population in a dynamic multi-objective problem. Muruganantham et al. [34] proposed a hybrid approach of an evolutionary algorithm and Kalman filter that uses historical information to predict for future generations. Liu et al. [29, 30] introduced two predictive models: the first is an orthogonal model that estimates the new individuals after the change while the second, is a modified prediction model using the historical information of last two times to initialize the new population.

2.2.3.4 Memory-based approaches Some dynamic multi-objective algorithms employ an extra memory that implicitly or explicitly stores the useful information from past generations to guide the future search. In [13], authors extended the competitive-cooperative evolutionary algorithm (CCEA) to detect and respond to changes in the environment. This version of CCEA is called dynamic COEA where the main idea is to allow the decomposition process of the optimization problem to adapt and emerge rather than being hand designed and fixed at the start of the evolutionary optimization process. The external population is used in order to store the potentially useful information about past POF.

2.2.3.5 Generic approaches In addition to above approaches, some extensions can be added to dynamic multi-objective evolutionary algorithms to improve the rate of performance and handle some issues.

Re-initialization strategy is one the traditional approaches to increase the rate of diversity when a change occurs. Four re-initialization strategies have been proposed by Zhou as follows: randomly re-initialize all new solutions (RND), add Gaussian noise (VAR), sample solutions around predicted locations of the POS (PRE), and create half of the solutions using PRE and the other half using VAR (V&P) [46].

Helbig, as one of the pioneers in the field of DMOO, recently has investigated some aspects of solving DMOOPs. Although these have been just applied on PSO-based algorithms, but can be implemented on any other evolutionary algorithm. The author investigated the effect of various approaches to manage boundary constraint violations [15], different behavior of each individuals in heterogeneous PSO [20], various ways to manage the archive solutions when solving DMOOPs [16], the influence of different guide update approaches [17], and local guide

selection by headless chicken macro-mutation operator [21].

2.2.4 Challenges of solving a DMOOP

Due to structure of the dynamic environment, there are many challenges that an optimization algorithm should be able to manage. The most important issue is change detection and gives a true response to change. In a static environment there is no need to use change detection mechanism and handle the challenges of after changing. Contrary to static environment, when a change occurs, how to response and react to change is a very important issue. At this moment the optimization algorithm should be able to give a true response to the environmental change in order to track the time-varying Pareto optimal front. Re-evaluating all solutions and re-initializing some percentage of population are two main approaches for handling this issue. It is significant that by handling the response to change, the issue of diversity will be controlled. This paper tries to investigate this issue and proposes a logical method to enhance the performance. Some challenges may be problematic if no strategy is used to handle the changes. These can be summarized as follows:

1. The volume of the change: if the size of change is huge, re-initialization can explore the environment to adapt the algorithm with new conditions. In contrast, for an environment with small changes, using previously found solutions can be useful. However, what information should be kept in each environment and how to reuse it is a major problem.
2. Diversity deterioration: in this case, the optimization algorithm is not able to track the changing optimum over time or it may take a long time to converge to new solutions.
3. Memory and archive management: to have a fresh archive removing all the non-dominated solutions of previous steps from archive or re-evaluating them is accepted.
4. The rate of convergence: if time of two consecutive changes is short, obtaining and tracking the changing optima will be difficult that causes slow convergence.
5. Number of objectives: by increasing the number of objective functions, most of the solutions are non-dominated to each other and multi-objective traditional approaches are inefficient.

2.3 Cat swarm optimization

Cat swarm optimization (CSO) is a swarm based algorithm [4] that models the behavior of cats into 'Seeking mode' and 'Tracing mode'. These two modes are combined by mixture ratio (MR) which decides how many cats will be moved into seeking or tracing. Also every cat has its position composed of M dimensions, velocities for each dimension, a fitness value, and a flag to identify whether the cat is in seeking mode or tracing mode. The algorithm can be summarized as follows: first, the population of cats is created, initialized, and evaluated. Also, the position of the best cat will be kept into memory. Then, according to the rate of mixture ratio, cats are moved to seeking or tracing mode. After processing the two modes, new cats are obtained and compose the fresh population. This scenario will be continued until the last iteration. The flowchart of CSO is shown in Fig. 1.

2.3.1 Seeking mode

Seeking mode is a mode of thinking and deciding about moving and is included in the following parameters: the number of copies for each cat called seeking memory pool (SMP), mutation rate called seeking the range of the selected dimension (SRD), and counts of dimension to change (CDC). The process of seeking mode is

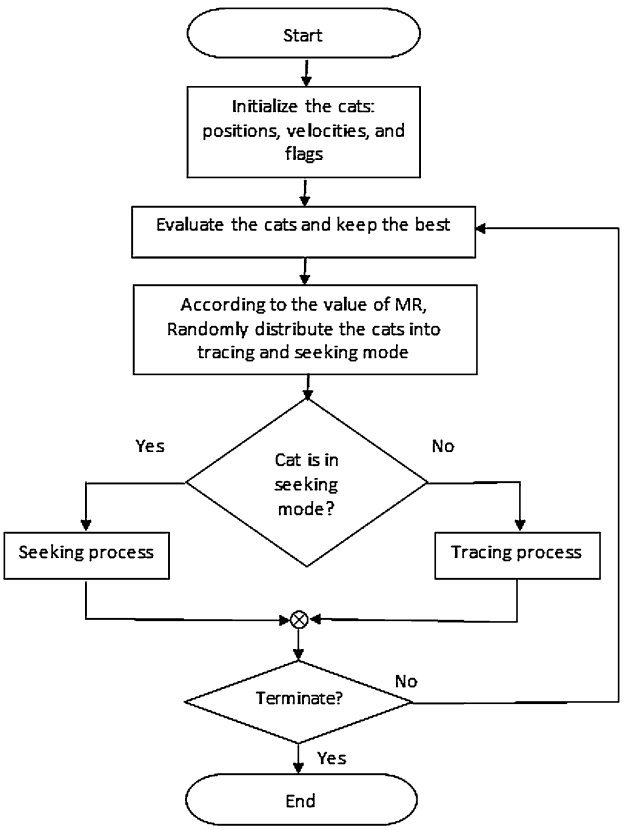

Fig. 1 Flowchart of the pure CSO

as follows: make 'SMP' copies of the present position of cat_k and retain the current cat as one of the candidates. Then, for each copy, according to 'CDC', randomly plus or minus 'SRD' percent the present values and replace the old ones. Calculate the fitness values (FS) of all candidate points and calculate the selecting probability of each candidate point by Eq. (3). Here, FS_{max} and FS_{min} are the maximum and minimum of obtained fitness values for each cat and its copies. Finally, pick a candidate randomly from 'SMP' copies and place it at the position of cat_k.

$$P_i = \frac{|FS_i - FS_{max}|}{FS_{max} - FS_{min}}. \tag{3}$$

2.3.2 Tracing mode

Tracing is the mode of targets and foods tracing. Here, the velocity and position of each cat for each dimension is determined by Eq. (4) to (5) where k is index of cat, d is related to dimension, $V_{k,d}$ is the velocity of cat_k in dimension of d, $X_{k,d}$ is the position of cat_k in dimension of d, $X_{best,d}$ is the position of the cat, who has the best fitness value, c_1 is an acceleration coefficient equal and r_1 is a random value in the range of [0,1].

$$V_{k,d} = V_{k,d} + r_1 c_1 (X_{best,d} - X_{k,d}), \tag{4}$$

$$X_{k,d} = X_{k,d} + V_{k,d}. \tag{5}$$

2.4 Borda count ranking method

In a voting system, voters rank the candidates in order of relative preference and the winner can be determined by different approaches. One of the most used is Borda Count ranking method, invented in 1770 by the French mathematician and physicist Jean-Charles, chevalier de Borda [36]. Borda is a voting method based on ranks, i.e.; it assigns a weight corresponding to the ranks in which a candidate appears within each voter's ranked list. Computationally they are very easy, as they can be implemented in linear time. The Borda count method has been considered in the context of the rank fusion problem and works as follows: each voter ranks a fixed set of c candidates in order of preference. For each voter, the top ranked candidate is given c points; the second ranked candidate is given c-1 points, and so on. If there are some candidates left unranked by the voter, the remaining points are divided evenly among the unranked candidates. The candidates are ranked in decreasing order of total points.

Definition: Borda Count

If we have an election with N candidates we will give 1 point for last place, 2 points for second to last, …, and N points for first place. The candidate with the highest total number of points is the winner. We will call such a candidate the Borda winner.

Given full ordered lists τ_1, …, τ_L, each a permutation of the underlying space T, we let $R\tau_l$ (u) be the rank of element u ∈ T in list τ_l. We let $B_l(u)$ denote the Borda score in general, with $B_l(u) = R_{\tau l}$ (u) being a special case. Let Bu = f ($B_1(u)$, $B_2(u)$,…, $B_L(u)$) be an aggregate function of the Borda scores. Then one sorts the Bu's to obtain an aggregate ranked list τ (T). Frequently suggested aggregation functions are included of median and P-norm functions.

2.5 Opposition-based learning method

Opposition-based learning (OBL) is a method based on machine learning that tries to improve the rate of convergence to optimal solution. In evolutionary algorithms, population initialization is done randomly in the first iteration where this may be a bad choice for starting an algorithm and deteriorate the speed of search and optimization process. So it is better to take a more logical method and look in all directions simultaneously, or more concretely, in the opposite direction [38]. Therefore, to increase the chance to achieve fast convergence, for each cat, we create an opposite cat. Suppose that the position of cat (x) is defined in the range of [a,b]. So the opposition-based learning method defines the location of the opposite cat as: ($\hat{x} = a + b - x$).

3 The proposed method

How to response to change after environmental changes is the main issue addressed in this paper. When a change occurs, an optimization algorithm should be able to give a true response to change to explore new points to move to and track the changing optima; otherwise, it is likely that current non-dominated solutions may be changed into dominated in a new environment. Predictive approaches and re-initialization methods are two most used approaches to handle this issue. The predictive model assumes that the volume of change is small and the behavior of two consecutive environments is similar together. So it tries to estimate the change and predict the solutions of next generations. The second strategy re-initializes some percentage of the population to enhance diversity. The main disadvantage is the random selection of solutions that should be re-initialized which may mislead the trajectory of the algorithm.

3.1 Main framework

This section presents the main framework of the proposed algorithm which its flowchart has been illustrated in Fig. 2.

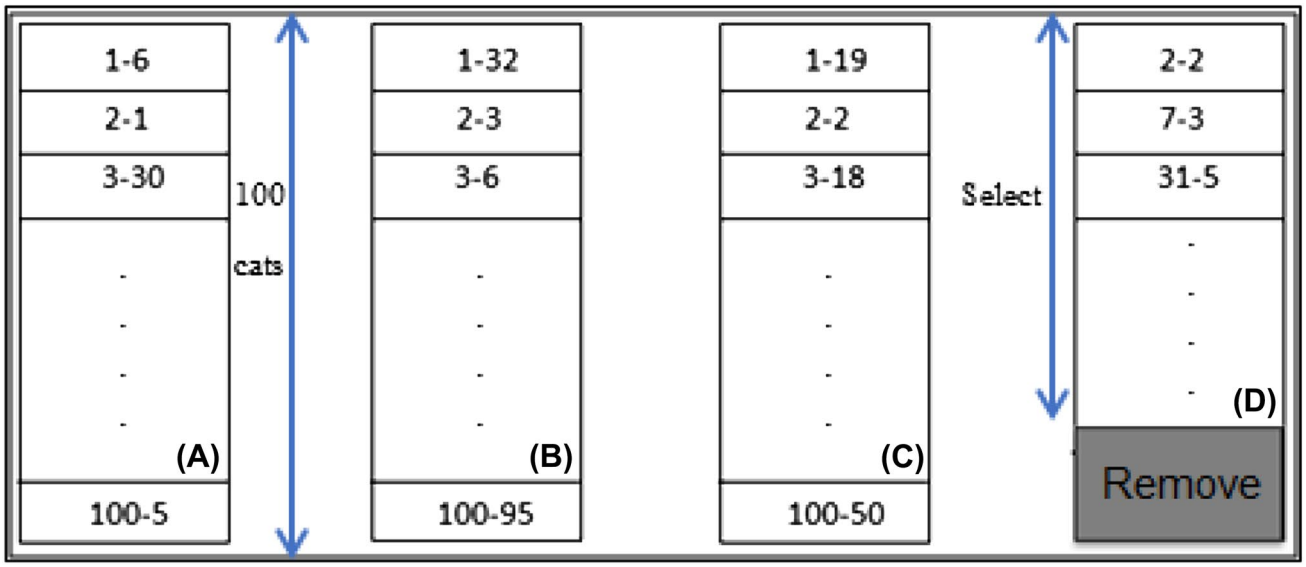

Fig. 2 Ranks just before change (**a**), ranks just after change (**b**), Borda scores (**c**) and Sorted Borda scores (**d**)

Algorithm 1 The proposed algorithm

```
1. INPUT: Population P, Maximum Iteration
2. OUTPUT: Approximated POF (Dynamic POF)
3. BEGIN
4.      Randomly initialize a population P
5.      Evaluate the fitness values of individuals
6.      for number of iterations do (until Maximum Iteration)
7.          Check whether a change has occurred
8.          If change has occurred
90.             Calculate the Borda score just before and after change
10.             Aggregate the Borda scores and calculate the Borda rank
11.             Sort the population according to Borda rank
12.             Remove cats with the worst Borda rank from population
13.             Create new cats and replace with removed ones
14.         Perform the static multi-objective CSO
15.         Perform the domination process among population
16.         Perform the crowding distance process among population
17.         Select the cats with highest domination and crowding distance as output POF.
18. END
```

3.2 The proposed dynamic multi-objective algorithm

In this paper, after detecting the environmental changes by sentry cats, in order to response to change and improve the diversity of solutions, it is necessary to remove the worst solutions and create new ones. To achieve this, all cats of population are assigned by Borda score. Here, Borda method assigns two ranks for each member of population based on its Borda score just before and after change. For each member, Borda score is determined by its number of wins in the competition of domination and crowding distance with other members. Then, rank of each member is determined by its Borda score. So each member has two ranks: the rank just before/after change. By combination of these ranks two lists of population are created. By joining these two lists and sorting in increasing order, the best candidates for removing from population are determined. Furthermore, new solutions should be created to enhance the diversity rate of algorithm and explore the new regions of search space. Finally, the new population is given to multi-objective cat swarm algorithm to estimate the optimal Pareto front. This scenario will be done by the three steps:

3.2.1 Change detection

This section investigates whether a change occurred or not. To do this, some cats (Sentry cats) are initialized, evaluated and their fitness values are stored in memory. Exactly just before the next iteration is performed, these cats are re-evaluated, and if their current fitness value differ more than a specified value from their fitness value just after the previous iteration, it means that a change has occurred.

3.2.2 Response to change

After detecting the change, how to response and react to change is an important issue. If the optimization algorithm fails to give a true response to change, it is assumed that there are not any environmental changes. As a result, the new environment will be considered as previous and the found solutions are ineligible. Using a predictive model to forecast the next generation and random re-initializing of whole or some parts of population are two main approaches. This paper uses re-initializing method where after detecting the change some solutions that should be

re-initialized are selected based on Borda count ranking method.

Borda count is the main technique used in this paper to response to change. Here rank of each cat just before and after a change is calculated based on non-dominated sorting and crowding distance. Then, by using a linear aggregation function, Borda score of each cat is determined. So when the change occurs and there is a need to change some percentage of the population in order to increase the rate of diversity, cats with worst Borda scores are selected to be removed from population. For example, suppose that the population size is '100'. For cat_i Borda rank is calculated based on the number of cats dominated by cat_i and higher crowding distance for the same number of dominated cats. Therefore, by summing the two Borda ranks of each cat, its Borda score is calculated. Population is sorted based on Borda score of each cat from the best to worst. Finally, cats with the worst score are selected to be removed and replaced by new ones. Here, the population has been refreshed by some new cats to generate diverse solutions. In this paper, 20 percentages of population (20 cats) are selected to be re-initialized.

Figure 2 shows the selection process by Borda count. First of all, ranks of all cats just before the change are calculated (A). For example, rank of cat_1 is 6, cat_2 is 1 and cat_3 is 30. Just after a change occurred, rank of each cat is calculated in new environment (B). In this case, rank of cat_1 is 32, cat_2 is 3, and cat_3 is 6. Then, by summing the before and after change rank of each cat, the average rank is calculated (C). Here, rank of cat_1 is 19, cat_2 is 2 and cat_3 is 18. Finally ranks are sorted form the best to the worst rank (D). Figure 2 shows that cat_2, cat_7, and cat_{31} has the best rank while the grey region has the worst rank and should be selected to remove.

3.2.3 Multi-objective CSO

One of the main issues in dynamic environment is existence of optimal solutions in the search space. So it is necessary to manage this issue by using algorithm with capacity of covering all peaks of search space. Cat swarm optimization with ability of global and local search simultaneously can be an appropriate choice. The comparison of CSO with other evolutionary algorithms such as PSO shows the better convergence and performance of CSO [4]. By using mixture ratio, CSO can switch between local search and global search. Furthermore, multi-population is one of the techniques of solving dynamic problems to investigate the many optimal solutions (peaks). In this way, the population is divided into sub-populations and works parallel to solve the problem where this technique is inserted to CSO automatically by using seeking and tracing phase. Optimization phase is the responsible of creating solutions to converge to optimal solutions and finds the solution as diverse as possible.

CSO is composed of seeking and tracing mode. Seeking mode applies the mutation operator on parents to produce offspring solutions while tracing uses the information of best cat in velocity equation to move to the next position. In this research in order to produce diverse solutions in the search space multi-objective version of CSO is used. To get this, non-dominated sorting method is applied to obtain the solutions as close as to Pareto optimal front the crowding distance approach is used to keep and find solutions in less crowded parts of Pareto front. So the pure CSO needs some changes on seeking and tracing mode.

3.2.3.1 Changes on seeking mode First, according to values of SRD, CDC and SMP, positions of all cats in this mode are updated. In this mode, cats have two additional parameters: rank and crowding distance. Rank is the number of loss in competition process and crowding distance is used to demonstrate the density. So, a competition based on domination concept is performed among the cats and for each cat, the number of cats dominating is assigned as rank. Then cats are sorted: the better the rank, the higher the priority is. Calculating the crowding distance for each cat is the second important task in this mode. Higher crowding distance value indicates that cat may locate in less density. For example, for an optimization problem with two objectives, the value of crowding distance for cat_i is calculated as Eq. (6). For cat_i the average distance from the two solutions immediately before and immediately after, along with each objective first and second dimension are indicated by d_i^1 and d_i^2 respectively. The last step in this mode is to select the best cats. Here, algorithm sorts the solutions (consist of parents and offspring: cats and their copies) based on non-dominated solutions (solutions with less rank) and higher crowding distance. It is necessary to indicate that the main selection operator is the rank of each cat and crowding distance is applied for cats with the same rank [7]

$$d_i^1 = \frac{\left|f_1^{i+1} - f_1^{i-1}\right|}{f_1^{max} - f_1^{min}}, d_i^2 = \frac{\left|f_2^{i+1} - f_2^{i-1}\right|}{f_2^{max} - f_2^{min}} \quad (6)$$

Crowding distance for $cat_i = d_i^1 + d_i^2$.

3.2.3.2 Changes on tracing mode Here cats try to move toward to food to find it. So this mode is called moving mode and because of using the position of the best cat in velocity equation, the most important issue is how to select the leader of cats (best cat). Here some percent of the best cats are stored in an external archive and the leader is randomly selected from this archive.

3.3 Various used strategies to response to change

In this paper, various strategies (from A to I) have been proposed to analyze the selection process, creating new solutions, and effect of using OBL on the performance of the proposed algorithm. These nine strategies are mentioned in Table 1. For example, this table indicates that in approach "E", selection of the worst cats for re-initializing is performed by Borda method. Also new solutions are generated by mutating the position of the best cats and the OBL technique is used to improve the convergence rate.

4 Experimental study

This section describes experiments that were conducted, using benchmark functions, performance metrics and different approaches of the proposed algorithm including random and Borda count strategy. All experiments consist of 20 independent cycles and each cycle consisted of 1000 iterations. For all benchmark functions the severity of change (n_t) is set to 10 and the frequency of change (τ_t) is set to either 10, 50. Population size is set to 100 and the CSO parameters are set to SMP = 4, CDC = 0.8, SRD = 0.2, MR = 0.95, and $C_1 = 2$.

Ten cats play the role of sentry cats for detecting the change and 20 cats are selected to be removed after a change occurred. Also codes are implemented by Java Eclipse and on a PC that has an Intel Processor of 2.7 GHz, Quad core CPU, and 12 GB of Memory.

Table 1 Summary of various strategies

Approach	Selection of cats to be re-initialized	Generate new solutions	OBL
A	No re-initialization	No generation	No
B	Random	Random	No
C	Random	Mutation	No
D	Borda	Random	No
E	Borda	Mutation	No
F	Random	Random	Yes
G	Random	Mutation	Yes
H	Borda	Random	Yes
I	Borda	Mutation	Yes

A: No response to change strategy is used

B: This model uses random selection to select cats for re-initialization and new solutions are made randomly

C: This model uses random selection for re-initialization and new solutions are made by mutating the best cats

D: Selection of cats for re-initialization is done by Borda method and new solutions are made randomly

E: Selection of cats for re-initialization is done by Borda method and new solutions are made by mutating the position of best cats

F: This model uses random selection to select cats for re-initialization and new solutions are made randomly. Also OBL technique is applied into algorithm

G: Selection of cats for re-initialization is done by Borda method and new solutions are made randomly. Also OBL technique is applied into algorithm

H: Selection of cats for re-initialization is done by Borda method and new solutions are made randomly. Also OBL technique is applied into algorithm

I: Selection of cats for re-initialization is done by Borda method and new solutions are made by mutating the position of best cats. Also OBL technique is applied into algorithm. Figure 3 shows the flowchart of the proposed algorithm

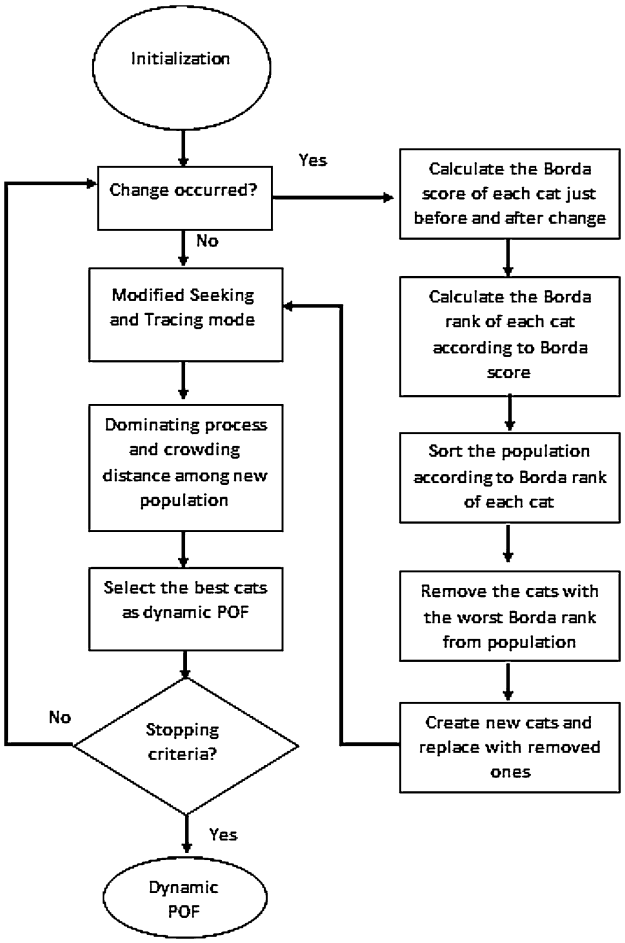

Fig. 3 Overall flow chart of the proposed algorithm

4.1 Benchmark functions

To determine whether an optimization algorithm can solve DMOOPs efficiently, DMOO benchmarks should be used to test the ability of the algorithm to overcome certain difficulties such as changing POF. Table 2 investigates the benchmark functions used in this paper [11].

Table 2 Dynamic multi-objective Benchmarks

Function	Definition	POF	POS
$dMOP_1$	$\begin{cases} \text{Minimize: } f(x,t) = (f_1(X_1), g(X_2) \times h(f_1(X_1), g(X_2,t))) \\ f_1(X_1,t) = x_1 \\ g(X_2,t) = 1 + 9 \sum_{x_i \in X_2} (x_i)^2 \\ h(f_1,g,t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ \text{where:} \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ x_1 \in [0,1],\ X_2 = (x_2, \ldots, x_n) \in [0,1]^{n-1} \end{cases}$	$1 - f_1^{H(t)}$	$x_i = 0$
$dMOP_2$	$\begin{cases} \text{Minimize: } f(x,t) = (f_1(X_1), g(X_2,t) \times h(f_1(X_1), g(X_2,t), t)) \\ f_1(X_1) = x_1 \\ g(X_2,t) = 1 + 9 \sum_{x_i \in X_2} (x_i - G(t))^2 \\ h(f_1,g,t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ \text{where:} \\ H(t) = 0.75\sin(0.5\pi t) + 1.25,\ G(t) = \sin(0.5\pi t),\ t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ x_1 \in [0,1], X_2 = (x_2, \ldots, x_n) \in [0,1]^{n-1} \end{cases}$	$1 - f_1^{H(t)}$	$x_i = G(t)$
$dMOP_2iso$	$\begin{cases} \text{Minimize:} f(x,t) = (f_1(X_1), g(X_2,t) \times h(f_1(X_1), g(X_2,t), t)) \\ f_1(X_1) = x_1 \\ g(X_2,t) = 1 + 9 \sum_{x_i \in X_2} (y_i - G(t))^2 \\ h(f_1,g,t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ \text{where:} \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, G(t) = \sin(0.5\pi t), t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ y_i = y_i(x_i, A, B, C) \forall x_i \in X_2 \\ x_1 \in [0,1], X_2 = (x_2, \ldots, x_n) \in [0,1]^{n-1} \end{cases}$	$1 - f_1^{H(t)}$	$x_i = G(t)$
HE_2	$\begin{cases} \text{Minimize:} f(x,t) = (f_1(X_1), g(X_2) \times h(f_1(X_1), g(X_2)), t))) \\ f_1(X_1) = x_1 \\ g(X_2) = 1 + \frac{9}{n-1} \sum_{x_i \in X_2} x_i \\ h(f_1,g,t) = 1 - \left(\sqrt{\frac{f_1}{g}}\right)^{H(t)} - \left(\frac{f_1}{g}\right)^{H(t)} \sin(10\pi f_1) \\ \text{where:} \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ x_1 \in [0,1], X_2 = (x_2, \ldots, x_n) \in [0,1]^{n-1} \end{cases}$	$1 - \sqrt{f_1}^{H(t)} - f_1^{H(t)}\sin(0.5\pi\ f_1)$	$x_i = 0$
FDA_1	$\begin{cases} \text{Minimize:} f(x,t) = (f_1(X_1), g(X_2,t) \times h(f_1(X_1), g(X_2,t))) \\ f_1(X_1) = x_1 \\ g(X_2,t) = 1 + \sum_{x_i \in X_2} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ \text{where:} \\ G(t) = \sin(0.5\pi t), t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ x_1 \in [0,1], X_2 = (x_2, \ldots, x_n) \in [-1,1]^{n-1} \end{cases}$	$1 - \sqrt{f_1}$	$x_i = G(t)$
FDA_3	$\begin{cases} \text{Minimize:} f(x,t) = (f_1(X_1), g(X_2,t) \times h(f_1(X_1), g(X_2,t))) \\ f_1(X_1,t) = \sum_{x_i \in X_2} (x_i)^{F(t)} \\ g(X_2,t) = 1 + G(t) + \sum_{x_i \in X_2} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ \text{where:} \\ G(t) = \lvert\sin(0.5\pi t)\rvert, t = \frac{1}{n_t}\frac{\tau}{\tau_t} \\ x_{i \in X_1} \in [0,1],\ x_{i \in X_2} \in [-1,1]^{n-1} \end{cases}$	$(1 + G(t))\left(1 - \sqrt{\frac{f_1}{1+G(t)}}\right)$	$x_i = G(t)$

4.2 Performance measures

In order to determine the performance of the proposed algorithm, and compare with previous approaches, this section investigates performance measures introduced in dynamic multi-objective environments. In this research, each metric is calculated every time just before a change occurs and the average of all these values is then calculated for each of the runs. To determine the algorithm with the best performance for a specific function, the algorithm's overall rank is calculated. For each measure the algorithm is ranked according to its performance on the specific metric. The algorithm's average rank value is calculated and then the algorithm is ranked accordingly.

The goal of a dynamic multi-objective optimization algorithm is two important targets: estimating the optimal Pareto front and adaptation to environmental changes and track the Pareto front over time. Some performance measures described below investigate both targets while some can do only one. This section introduces VGD, IGD, MS, Spacing, HVR, ACC, and Stab.

1. Variable generational distance (VGD) measures the distance between POF^* (approximated POF) and true POF as follows: [13]

$$VD(t) = \frac{1}{\tau}\sum_{t=0}^{\tau} VD(t)I(t),$$
$$VD(t) = \frac{1}{n}\sqrt{n\sum_{i=1}^{n} d_i(t)^2}, I(t) = \begin{cases} 1, & \text{if } t\%\tau_t = 0 \\ 0, & \text{otherwise} \end{cases}, \quad (14)$$

where n is the number of found non-dominated solutions at time t and d_i is the Euclidean distance between i-th solution of POF^* and the nearest solution of POF. It is notable that in the original definition, VGD is used to POS while in this literature this measure is applied to compare the found POF.

2. Inverted generational distance (IGD) measures the average distance of the known Pareto front to the generated Pareto front making it possible to measure convergence diversity of the solution found with respect to POF^* (t) [28]

$$IGD = \frac{\sum_{i=1}^{n} d_i}{n}, \quad (15)$$

where d is the Euclidean distance between the each point in true POF and nearest non-dominated individual of approximated POF.

3. Maximum Spread (MS) measures how well POF^* covers the POF and is defined as: [13]

$$MS = \sqrt{\frac{1}{n_k}\sum_{k=1}^{n_k}\left[\frac{\min\left[\overline{POF^*}_k, \overline{POF'}_k\right] - \max\left[\underline{POF^*_k}, \underline{POF'_k}\right]}{\overline{POF'}_k - \underline{POF'_k}}\right]^2}, \quad (16)$$

where 'M' is the number of objectives, 'n' is the number of non-dominated solutions in POF^* at time t, $\overline{POF^*}_i$ and $\underline{POF^*_i}$ refer to the maximum and minimum of i-th objective of non-dominated solutions in POF^* and $\overline{POF'}_i$ and $\underline{POF'_i}$ refer to the maximum and minimum of i-th objective of non-dominated solutions in POF respectively.

4. Spacing measures how evenly the non-dominated solutions are distributed along the POF^* can be done using the metric of spacing: [12]

$$\overline{S}^i = \frac{1}{n_c}\sum_{j=1}^{n_c} S^i_j, S = \frac{1}{n}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - \overline{d})^2}, \quad (17)$$

where n_c is the number of changes, n is the number of non-dominated solutions found at time t and di is the Euclidean distance, in the objective space, between non-dominated solution i and its nearest solution in POF^*.

5. Hypervolume ratio (HVR) is the HV ratio of obtained POF to true POF. HV computes the size of the region that is dominated by a set of non-dominated solutions, based on a reference vector [25, 47]

$$HVR = \frac{1}{n_c}\sum_{i=1}^{n_c} HVR(t), HVR(t) = \frac{HV(POF^*(t))}{HV(POF(t))}, \quad (18)$$

6. Alternative accuracy: accuracy metric measures the quality of the solutions as a relation between the HV of POF* and the maximum HV that has been found so far as follows: [3]

$$acc(t) = \frac{HV(POF^*(t))}{HV_{max}(POF^*(t))}. \quad (19)$$

The alternative accuracy as follows: [2, 3]

$$acc_{alt}(t) = |HV(POF(t)) - HV(POF^*(t))|. \quad (20)$$

7. Stability measures the effect of the changes in the environment on the accuracy of the algorithm and is defined as: [3]

$$Stab = \frac{1}{n_c}\sum_{i=1}^{n_c} stab(t),$$
$$stab(t) = \max\{0, acc(t-1) - acc(t)\}. \quad (21)$$

Table 3 Simulation results of running different approaches for test functions at re-initialization rate of 0.3 of population

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
$dMOP_1$	10–10	Spacing	Mean	3.2195E−4	5.625E−4	3.949E−4	3.752E−4	3.428E−4	5.315E−4	3.332E−4	3.994E−4	2.135E−4
			Std	4.410E−8	1.202E−7	3.700E−8	2.251E−8	2.602E−8	1.041E−7	1.66E−8	4.128E−8	1.152E−9
		IGD	Mean	0.01778	0.00172	0.00172	0.00219	0.00226	0.00204	0.00201	0.00167	0.00216
			Std	9.614E−4	1.135E−7	2.089E−7	6.88E−7	3.573E−7	2.923E−7	2.982E−7	2.478E−7	2.55E−7
		VGD	Mean	0.03426	0.0483	0.0362	0.0514	0.0452	0.0615	0.0314	0.0351	0.0309
			Std	6.36E−4	6.919E−4	2.50E−4	7.91E−4	2.30E−4	0.00138	2.265E−5	1.404E−4	1.98E−4
		MS	Mean	0.7977	0.99491	0.99595	0.99772	0.99639	0.99426	0.99477	0.99572	0.99198
			Std	0.15795	1.68E−6	4.27E−6	2.39E−6	5.060E−6	1.808E−5	2.108E−5	3.829E−6	3.747E−4
		HVR	Mean	0.78969	0.98673	0.9877	0.98618	0.98709	0.98347	0.98604	0.98811	0.98420
			Std	0.15400	6.83E−6	1.33E−6	9.02E−6	2.919E−6	3.492E−5	1.210E−5	2.974E−6	2.696E−5
		Acc alt	Mean	0.10767	0.0169	0.0177	0.0234	0.0245	0.0232	0.0179	0.0149	0.0182
			Std	0.0271	3.61E−5	2.26E−5	4.4E−5	5.93E−5	3.99E−5	1.86E−5	2.60E−5	5.41E−5
		Stab	Mean	1.225E−4	0.00175	1.794E−4	8.50E−4	3.702E−4	0.0144	7.672E−4	8.235E−4	3.819E−4
			Std	8.589E−4	9.85E−7	1.169E−8	1.69E−7	6.44E−8	6.39E−5	5.39E−7	3.461E−7	8.48E−8
		R_1		**7**	**5**	**1**	**6**	**4**	**8**	**2**	**1**	**3**
		R_2		**8**	**3**	**2**	**6**	**5**	**7**	**4**	**1**	**5**
	10–50	Spacing	Mean	1.09E−4	1.10E−4	1.091E−4	1.119E−4	5.36E−5	1.107E−4	1.145E−4	1.10E−4	1.113E−4
			Std	8.6E−12	6.3E−12	9.13E−13	1.34E−12	2.88E−9	3.07E−12	1.32E−12	1.0E−11	7.05E−12
		IGD	Mean	9.37E−4	9.50E−4	9.439E−9	9.336E−4	0.0379	9.710E−4	9.629E−4	8.87E−4	9.25E−4
			Std	3.9E−10	1.5E−13	1.10E−11	1.85E−9	0.0013	5.17E−11	4.31E−9	8.4E−10	4.57E−10
		VGD	Mean	0.00529	0.00559	0.00538	0.005578	0.00269	0.005534	0.005900	0.00529	0.005517
			Std	1.18E−9	8.85E−8	5.90E−9	4.06E−8	7.28E−6	4.28E−8	2.06E−7	1.7E−10	3.46E−8
		MS	Mean	0.99966	0.99988	0.999548	0.999233	0.49966	0.99993	1	1	1
			Std	1.14E−7	1.24E−8	2.03E−7	5.87E−7	0.2496	3.64E−9	0	0	0
		HVR	Mean	0.98950	0.98926	0.98913	0.98957	0.4949	0.98899	0.98922	0.9900	0.9896
			Std	5.58E−7	5.0E−11	1.29E−8	1.69E−6	0.2449	2.75E−11	1.34E−6	5.53E−7	4.99E−8
		Acc alt	Mean	0.00375	0.00388	0.00390	0.00376	0.17783	0.00398	0.00396	0.00358	0.00375
			Std	7.57E−8	2.93E−9	9.92E−10	2.25E−7	0.0303	6.57E−10	2.00E−7	8.23E−8	5.15E−9
		Stab	Mean	0.25E−5	0.0112	8.83E−6	6.13E−5	5.68E−5	0.00260	0.1.1E−5	1.00E−4	2.84E−5
			Std	5.0E−12	1.20E−4	1.35E−11	1.66E−10	3.23E−9	6.37E−6	3.68E−11	2.5E−10	7.74E−11
		R_1		**2**	**7**	**4**	**5**	**8**	**9**	**6**	**3**	**1**
		R_2		**3**	**6**	**5**	**4**	**9**	**8**	**7**	**2**	**1**

Table 3 (continued)

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
$dMOP_2$	10–10	Spacing	Mean	5.72E−4	3.612E−4	3.24E−4	3.276E−4	3.258E−4	3.04E−4	3.19E−4	3.22E−4	3.12E−4
			Std	7.1E−11	1.18E−9	7.1E−11	8.09E−11	1.26E−9	2.2E−12	2.0E−13	5.1E−11	1.5E−11
		IGD	Mean	0.00475	0.00365	0.00377	0.003594	0.003592	0.00349	0.00364	0.00362	000368
			Std	1.33E−8	5.109E−9	1.80E−9	1.05E−9	1.36E−10	1.10E−9	6.9E−10	3.6E−11	8.4E−9
		VGD	Mean	0.0960	0.0812	0.05766	0.0626	0.0520	0.05312	0.04607	0.06779	0.0459
			Std	3.38E−8	2.47E−4	4.70E−6	4.3E−5	6.15E−7	8.30E−5	1.67E−7	8.7E−9	7.9E−11
		MS	Mean	0.98336	0.098697	0.9881	0.98862	0.98848	0.98884	0.98865	0.98768	0.98750
			Std	4.21E−6	7.52E−7	8.89E−8	5.85E−7	6.28E−9	8.58E−9	1.49E−9	4.19E−5	6.9E−7
		HVR	Mean	0.89034	0.91512	0.90384	0.91047	0.90643	0.90797	0.90331	0.91348	0.90228
			Std	4.29E−5	2.02E−6	1.53E−7	1.38E−7	6.78E−7	2.65E−5	5.28E−7	1.83E−6	2.4E−6
		Acc alt	Mean	0.0727	0.0538	0.0556	0.05297	0.0529	0.05093	0.05364	0.0524	0.05368
			Std	3.90E−6	1.821E−6	4.95E−8	2.79E−7	2.99E−12	6.49E−7	2.22E−7	7.7E−6	2.2E−6
		Stab	Mean	0.00720	0.00594	0.00928	0.00830	0.00458	0.00508	0.00759	0.00496	0.00551
			Std	4.05E−7	1.367E−6	4.18E−6	4.01E−6	5.91E−7	1.28E−6	2.34E−6	4.24E−7	3.2E−6
		R_1		**9**	**8**	**7**	**5**	**2**	**1**	**3**	**4**	**6**
		R_2		**9**	**6**	**7**	**3**	**3**	**1**	**2**	**4**	**5**
	10–50	Spacing	Mean	2.38E−4	1.50E−4	1.039E−4	1.27E−4	1.24E−4	1.96E−4	1.091E−4	1.552E−4	1.04E−4
			Std	1.08E−10	6.00E−10	5.00E−10	6.16E−11	1.87E−10	5.08E−9	6.73E−10	7.9E−11	1.05E−9
		IGD	Mean	0.00340	0.00188	0.1659	0.00208	0.08187	0.00208	0.2475	0.00214	0.16590
			Std	4.76E−10	1.17E−8	0.02691	6.41E−8	7.58E−4	1.92E−8	0.06037	2.36E−8	0.02690
		VGD	Mean	0.0775	0.02493	1.5528	0.02056	0.7472	0.04178	2.339	0.02570	1.5521
			Std	2.94E−4	1.15E−5	2.355	1.05E−5	0.06131	3.24E−4	5.386	8.1E−7	2.358
		MS	Mean	0.98814	0.99893	1.6680	0.99754	0.9000	0.99790	2.2192	0.99808	1.6680
			Std	8.35E−8	1.01E−6	0.4464	3.31E−6	0.0089	1.04E−6	1.486	1.31E−6	0.44692
		HVR	Mean	0.92549	0.94478	0.5494	0.9371	0.3463	0.9401	0.5355	0.9387	0.5535
			Std	5.78E−5	1.80E−5	0.1555	4.37E−5	0.0048	2.61E−5	0.1670	2.20E−5	0.1544
		Acc alt	Mean	0.04514	0.02318	1.6875	0.2591	0.91626	0.02611	2.484	0.0263	1.686
			Std	1.16E−6	2.94E−6	2.77E−2	1.25E−5	0.0779	3.38E−6	6.062	4.64E−6	2.771
		Stab	Mean	0.00124	0.01291	0.01042	0.00614	0.00387	0.02960	0.00461	0.00310	0.00162
			Std	1.00E−7	6.96E−5	4.88E−5	2.66E−5	8.87E−7	4.78E−4	2.08E−5	1.59E−6	7.57E−7
		R_1		**6**	**1**	**7**	**2**	**9**	**4**	**8**	**3**	**5**
		R_2		**5**	**1**	**6**	**2**	**8**	**3**	**9**	**4**	**7**

Table 3 (continued)

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
$dMOP_{2iso}$	10–10	Spacing	Mean	1.379E−4	1.618E−4	1.810E−4	1.527E−4	1.528E−4	1.933E−4	1.401E−4	1.510E−4	1.44E−4
			Std	5.21E−11	4.9E−11	5.84E−9	3.26E−10	3.92E−10	2.43E−9	4.30E−10	2.68E−10	5.40E−10
		IGD	Mean	9.398E−4	9.703E−4	9.911E−4	9.405E−4	9.772E−4	9.207e−4	9.538E−4	9.524E−4	9.324E−4
			Std	5.55E−10	5.01E−10	4.14E−9	1.04E−9	1.009E−9	7.74E−10	2.05E−10	2.124E−9	3.03E−10
		VGD	Mean	0.0138	0.01472	0.0335	0.01318	0.02111	0.02433	0.01475	0.01397	0.01309
			Std	3.46E−6	4.10E−6	0.00136	6.47E−6	1.71E−4	1.42E−4	3.95E−6	2.04E−6	1.68E−5
		MS	Mean	0.99792	0.99752	0.99768	0.99773	0.99709	0.99806	0.99770	0.99775	0.99778
			Std	1.102E−7	9.11E−8	5.01E−7	3.124E−7	8.60E−7	4.158E−7	3.33E−7	4.41E−7	7.79E−7
		HVR	Mean	0.98974	0.98899	0.98857	0.98962	0.98894	0.98986	0.98937	0.98939	0.98957
			Std	5.34E−7	4.9E−8	2.84E−6	4.414E−7	1.39E−6	1.06E−6	2.68E−7	7.32E−7	4.83E−7
		Acc alt	Mean	0.00618	0.0064	0.00900	0.00639	0.00720	0.00742	0.00653	0.00643	0.00584
			Std	2.69E−7	3.11E−7	2.54E−5	8.32E−7	2.01E−6	6.01E−6	3.75E−7	4.04E−7	3.79E−7
		Stab	Mean	0.00415	0.01007	0.00571	0.00897	0.00726	0.00980	0.00776	0.00576	0.00359
			Std	6.96E−7	1.1E−4	9.78E−6	3.54e−5	2.39E−5	9.75E−6	1.19E−5	1.61E−5	9.12E−6
		R_1		**2**	**7**	**8**	**3**	**6**	**4**	**5**	**3**	**1**
		R_2		**2**	**7**	**9**	**3**	**8**	**4**	**6**	**5**	**1**
	10–50	Spacing	Mean	1.16E−4	1.58E−4	1.032E−4	1.338E−4	1.576E−4	1.488E−4	1.624E−4	1.861E−4	1.15E−4
			Std	1.27E−11	6.11E−10	5.74E−10	7.81E−10	3.35E−10	1.66E−16	5.43E−10	8.68E−11	4.4E−10
		IGD	Mean	0.02724	0.00223	0.19267	8.373E−4	0.02721	9.022E−4	0.00154	8.44E−4	0.02722
			Std	6.95E−4	1.82E−6	0.0367	2.71E−10	6.96E−4	1.02E−9	4.97E−7	3.08E−14	6.95E−4
		VGD	Mean	0.25231	0.03008	1.8120	0.01804	0.25191	0.00934	0.02295	0.02784	0.2549
			Std	0.0568	3.6E−5	3.257	3.98E−5	0.0593	4.68e−6	7.82E−5	7.16E−5	0.0583
		MS	Mean	0.90279	0.99024	1.8492	0.99770	0.90208	0.99824	0.99321	0.99854	0.90103
			Std	0.00916	5.20E−5	0.72490	2.9E−8	0.00937	4.76E−8	3.04E−5	7.47E−7	0.0094
		HVR	Mean	0.72822	0.9677	0.5835	0.99036	0.74603	0.9899	0.9746	0.98969	0.72808
			Std	0.0682	3.85E−4	0.1662	5.03E−7	0.0600	3.5E−7	2.29E−4	5.38E−7	0.06830
		Acc alt	Mean	0.3144	0.0249	1.945	0.00613	0.3138	0.00423	0.01604	0.0066	0.3148
			Std	0.09532	2.59E−4	3.770	3.69E−6	0.09616	2.59E−7	1.15E−4	2.05E−6	0.09602
		Stab	Mean	0.01171	8.23E−4	0.00244	0.00735	0.00372	0.02185	0.00214	0.00573	0.00587
			Std	2.59E−5	3.45E−7	5.41E−6	3.34E−5	1.44E−6	4.71E−4	3.36E−6	2.96E−5	3.16E−5
		R_1		**8**	**5**	**7**	**1**	**6**	**2**	**4**	**3**	**9**
		R_2		**7**	**5**	**8**	**2**	**6**	**1**	**4**	**3**	**9**

Table 3 (continued)

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
HE_2	10–10	Spacing	Mean	6.71E−5	8.56E−5	5.89E−5	9.13E−5	7.596E−5	8.707E−5	5.76E−5	8.93E−5	7.26E−5
			Std	1.1E−14	6.6E−12	7.4E−13	1.9E−12	1.76E−11	1.84E−12	1.5E−13	3.6E−15	9.8E−12
		IGD	Mean	0.00658	0.00611	0.00795	0.00608	0.00631	0.00613	0.01129	0.00611	0.00631
			Std	9.0E−10	7.2E−10	1.38E−6	6.3E−11	1.144E−9	2.61E−10	2.29E−6	4.8E−10	2.72E−9
		VGD	Mean	0.03606	0.04145	0.03774	0.03963	0.03814	0.04294	0.03781	0.04126	0.03808
			Std	1.11E−7	2.21E−7	2.22E−7	5.52E−8	2.78E−8	3.77E−7	1.35E−6	4.00E−9	2.81E−7
		MS	Mean	0.86186	0.86061	0.8331	0.86142	0.860545	0.860531	0.76865	0.86135	0.95909
			Std	2.5E−10	4.55E−7	3.64E−4	2.02E−8	3.27E−8	1.41E−7	9.54E−4	2.76E−7	2.14E−7
		HVR	Mean	0.95328	0.95795	0.93699	0.95877	0.95625	0.95809	0.91616	0.95840	0.95547
			Std	6.34E−7	2.77E−7	2.61E−5	2.66E−8	3.05E−7	1.20E−11	3.91E−5	6.30E−8	1.70E−7
		Acc alt	Mean	0.0560	0.05192	0.06529	0.0519	0.05370	0.05202	0.0788	0.05188	0.05349
			Std	1.22E−7	1.20E−7	3.25E−5	6.59E−9	4.13E−8	6.08E−9	2.10E−5	2.41E−9	9.01E−8
		Stab	Mean	5.41E−5	1.58E−4	4.65E−5	1.63E−4	7.906E−5	1.36E−4	4.46E−5	1.40E−4	7.63E−5
			Std	5.6E−11	2.9E−15	7.6E−12	2.9E−10	5.80E−10	4.00E−10	1.2E−10	3.8E−11	2.9E−12
		R_1		**1**	**4**	**6**	**2**	**5**	**7**	**8**	**3**	**5**
		R_2		**4**	**3**	**7**	**1**	**5**	**5**	**8**	**2**	**6**
	10–50	Spacing	Mean	8.042E−5	8.77E−5	7.323E−5	8.357E−5	6.109E−5	8.090E−5	8.895E−5	8.149E−5	8.0216E−5
			Std	2.27E−10	1.27E−11	8.12E−12	4.43E−13	8.10E−13	7.24E−12	6.16E−10	1.30E−11	9.09E−11
		IGD	Mean	0.00638	0.00624	0.00627	0.00614	0.00659	0.00622	0.00624	0.00616	0.00641
			Std	2.07E−9	9.12E−9	5.66E−9	1.16E−10	1.05E−7	3.62E−9	3.22E−9	2.01E−10	6.14E−8
		VGD	Mean	0.02749	0.03497	0.03003	0.03307	0.02707	0.03361	0.03182	0.03363	0.03100
			Std	9.07E−6	4.41E−6	1.67E−6	9.52E−9	3.3E−7	2.77E−6	6.21E−6	2.53E−7	1.43E−5
		MS	Mean	0.84082	0.840311	0.84277	0.84251	0.84295	0.84103	0.84266	0.84303	0.8396
			Std	6.84E−6	7.48E−6	5.29e−7	5.79E−8	3.53E−8	1.62E−6	1.60E−9	1.56E−7	6.02E−6
		HVR	Mean	0.91825	0.920483	0.91906	0.92201	0.91483	0.92102	0.91900	0.92159	0.9187
			Std	2.19E−6	5.14E−7	1.76E−7	1.37E−7	2.79E−5	7.44E−7	1.21E−7	4.39E−8	1.23E−5
		Acc alt	Mean	0.0665	0.06445	0.06590	0.06351	0.06535	0.06432	0.6596	0.0638	0.06613
			Std	1.43E−6	3.04E−7	1.31E−7	9.10E−8	1.84E−5	5.30E−7	8.49E−8	3.26E−8	8.06E−8
		Stab	Mean	1.137E−4	2.050E−4	1.000E−4	1.525E−4	5.767E−5	1.87E−4	4.181E−5	1.366E−4	1.909E−4
			Std	6.04E−11	2.49E−9	4.32E−11	2.93E−10	5.71E−10	4.82E−9	1.03E−10	1.59E−11	1.81E−8
		R_1		**7**	**9**	**3**	**2**	**4**	**5**	**6**	**1**	**8**
		R_2		**8**	**7**	**4**	**2**	**6**	**3**	**5**	**1**	**9**

Table 3 (continued)

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
FDA_1	10–10	Spacing	Mean	0.001340	6.15E−4	4.87E−4	6.02E−4	4.94E−4	6.83E−4	5.561E−4	4.963E−4	5.39E−4
			Std	3.39E−8	6.14E−9	1.13E−9	8.46E−5	3.98E−9	1.75E−8	5.65E−9	8.16E−9	4.32E−9
		IGD	Mean	0.02580	0.01338	0.0325	0.01877	0.0339	0.01220	0.0286	0.0133	0.0265
			Std	4.46E−6	1.28E−5	5.28E−5	8.36E−6	1.48E−5	7.53E−6	2.69E−5	9.15E−6	2.97E−5
		VGD	Mean	1.1603	0.4721	0.6957	0.5783	0.7880	0.4649	0.6765	0.5052	0.7200
			Std	0.0035	0.0066	0.0113	0.0104	0.0079	0.0062	0.0118	0.0136	0.0094
		MS	Mean	0.91044	0.94774	0.9067	0.9354	0.9027	0.9529	0.9084	0.9483	0.9152
			Std	7.6E−6	1.65E−4	1.6E−4	1.84E−4	4.85E−5	1.17E−4	1.40E−4	1.14E−4	1.03E−4
		HVR	Mean	0.9030	0.9088	0.8374	0.8900	0.8394	0.9180	0.8489	0.9097	0.8707
			Std	1.48E−5	2.87E−4	9.03E−4	1.02E−4	1.290E−4	8.71E−5	4.33E−4	9.60E−5	2.69E−4
		Acc alt	Mean	0.3913	0.2048	0.4808	0.2815	0.5035	0.1869	0.4308	0.2067	0.3972
			Std	8.35E−4	0.00298	0.0109	0.0016	2.56E−4	0.00169	0.0054	0.0019	0.0058
		Stab	Mean	0.0021	0.1024	0.0113	0.0096	0.0084	0.01384	0.0077	0.0103	0.0091
			Std	2.13E−6	5.79E−6	1.33E−25	1.49E−25	7.46E−26	4.43E−26	4.84E−26	3.68E−26	1.56E−26
		R_1		**6**	**3**	**8**	**4**	**8**	**1**	**7**	**2**	**5**
		R_2		**5**	**2**	**8**	**4**	**9**	**1**	**7**	**3**	**6**
	10–50	Spacing	Mean	1.15E−24	2.69E−24	1.085E−24	1.076E−24	9.781E−25	1.92E−24	1.03E−24	1.346E−24	8.927E−25
			Std	1.6E−210	7.83E−29	1.92E−210	2.18E−210	1.18E−211	1.20E−29	1.0E−210	1.76E−9	7.39E−213
		IGD	Mean	0.00174	0.00312	0.00135	0.00133	0.001511	0.00404	0.00278	0.00135	0.00130
			Std	1.40E−27	2.64E−26	7.9E−29	7.57E−210	2.47E−29	7.74E−26	2.10E−26	1.87E−29	6.10E−210
		VGD	Mean	0.04705	0.05718	0.02024	0.02871	0.02287	0.07091	0.03587	0.07578	0.01675
			Std	7.10E−24	7.17E−24	1.95E−26	3.96E−25	1.61E−27	8.58E−24	2.85E−24	0.00358	2.96E−27
		MS	Mean	0.99737	0.99125	0.999439	0.000559	0.99891	0.98566	0.99215	0.999213	0.999660
			Std	3.03E−26	6.69E−25	2.01E−28	2.49E−28	4.02E−28	2.03E−24	5.74E−25	3.67E−28	3.36E−210
		HVR	Mean	0.97517	0.95962	0.97852	0.97981	0.97625	0.95472	0.96160	0.98101	0.97845
			Std	9.04E−26	2.63E−24	2.36E−26	1.10E−210	1.80E−26	8E−24	3.00E−24	7.94E−26	6.93E−27
		Acc alt	Mean	0.02186	0.04171	0.01607	0.01566	0.01863	0.0559	0.03634	0.01612	0.01540
			Std	3.11E−25	5.36E−24	2.37E−26	2.24E−27	3.72E−27	0.00171	4.38E−24	3.48E−27	1.58E−27
		Stab	Mean	0.00423	0.02533	0.005100	0.00803	0.00202	0.01415	0.00711	0.005223	0.00365
			Std	1.23E−26	6.27E−28	3.54E−26	3.31E−27	1.54E−28	1.06E−24	3.29E−27	9.05E−27	3.36E−27
		R_1		**6**	**8**	**2**	**3**	**4**	**9**	**7**	**5**	**1**
		R_2		**5**	**7**	**2**	**2**	**4**	**8**	**6**	**3**	**1**

Table 3 (continued)

Function	n_t–τ_t	Measure		A	B	C	D	E	F	G	H	I
FDA_3	10–10	Spacing	Mean	6.356E−24	7.238E−24	6.903E−24	7.510E−24	6.740E−24	7.723E−24	6.542E−24	6.967E−24	5.56E−24
			Std	2.91E−29	7.18E−29	2.79E−210	2.73E−29	7.86E−210	1.08E−29	5.12E−212	4.7E−210	4.16E−212
		IGD	Mean	0.00155	0.00130	0.00174	0.00129	0.00115	0.00134	0.00146	0.00117	0.00152`
			Std	3.49E−27	1.00E−28	9.41E−210	6.54E−29	1.62E−28	6.11E−29	5.21E−28	2.72E−29	1.35E−27
		VGD	Mean	0.7365	0.7899	0.7519	0.7838	0.7564	0.7759	0.7555	0.7769	0.7611
			Std	4.37E−25	3.16E−27	3.70E−25	7.49E−27	3.06E−26	3.01E−24	7.39E−25	1.98E−25	1.12E−24
		MS	Mean	0.9977	0.9979	0.9965	0.9990	0.9972	0.9981	0.9985	0.9990	0.9982
			Std	8.61E−27	1.39E−26	7.57E−26	8.43E−25	1.31E−27	9.59E−27	1.32E−27	6.49E−28	2.91E−27
		HVR	Mean	2.640	2.5013	2.4839	2.399	2.761	2.3159	2.401	2.443	2.7600
			Std	0.0053	0.0462	0.0060	0.1073	4.86E−24	0.0868	0.1325	0.1058	5.14E−26
		Acc alt	Mean	2.573	2.6453	2.6332	2.633	2.707	2.613	2.668	2.660	2.664
			Std	8.15E−26	0.00347	2.78E−28	0.00305	6.35E−25	0.00296	0.00223	0.00280	0.00399
		Stab	Mean	0.00587	0.00462	0.00488	0.01004	0.00541	0.00583	0.00665	0.00908	0.00650
			Std	1.86E−27	1.49E−26	3.21E−27	1.94E−27	7.84E−27	4.91E−27	1.03E−26	5.63E−27	5.02E−26
		R_1		**1**	**7**	**4**	**8**	**2**	**6**	**6**	**5**	**3**
		R_2		**1**	**7**	**6**	**4**	**2**	**5**	**6**	**2**	**3**
	10–50	Spacing	Mean	3.455E−24	4.998E−24	6.276E−24	4.541E−24	4.574E−24	4.692E−24	4.595E−24	4.757E−24	4.093E−24
			Std	1.40E−210	4.48E−29	1.928E−28	3.94E−210	2.89E−29	2.44E−29	5.24E−29	9.11E−210	2.36E−210
		IGD	Mean	0.00322	0.00163	0.00167	0.00112	0.00262	0.00178	0.00206	0.00125	0.00121
			Std	3.80E−26	5.20E−28	2.18E−28	3.73E−210	1.90E−26	5.61E−29	4.98E−27	8.01E−29	4.00E−213
		VGD	Mean	0.58382	0.59304	0.61417	0.5868	0.60218	0.60330	0.61990	0.61237	0.60005
			Std	2.79E−26	5.47E−25	4.59E−25	1.67E−25	1.218E−24	7.3E−25	5.43E−24	3.16E−25	2.90E−26
		MS	Mean	0.999919	0.999998	0.999417	0.999728	0.999993	0.999502	0.999834	0.999682	0.999869
			Std	5.97E−29	2.62E−212	3.313E−27	2.94E−28	3.84E−211	2.88E−28	1.199E−28	9.78E−28	1.03E−28
		HVR	Mean	1.0499	1.0759	1.0659	1.0765	1.0460	1.0648	1.0601	1.0794	1.07409
			Std	0.00149	3.69E−27	3.65E−25	4.02E−25	4.02E−24	1.62E−24	9.87E−24	3.07E−26	5.25E−26
		Acc alt	Mean	0.83345	0.94069	0.95399	0.94093	0.77987	0.96000	0.91422	0.93237	0.93836
			Std	0.00844	1.46E−25	9.05E−24	3.47E−25	0.00221	7.04E−24	0.00221	4.54E−26	1.38E−24
		Stab	Mean	0.00419	0.00874	0.00872	0.00358	0.00937	0.10684	0.01628	0.01475	0.00660
			Std	3.82E−26	3.10E−25	3.47E−25	5.30E−26	2.75E−25	2.40E−25	8.53E−25	1.09E−24	6.78E−26
		R_1		**3**	**4**	**8**	**1**	**5**	**9**	**7**	**6**	**2**
		R_2		**4**	**1**	**7**	**2**	**5**	**7**	**6**	**3**	**2**

Best found answers in each table are indicated in Bold

4.3 Simulation results

In this section, the results obtained from simulations of dynamic multi-objective problems are expressed and analyzed. Table 3 shows the obtained results of simulation for benchmarks, performance metrics and these approaches. These results are shown by mean value of the found solutions. Table 4 shows the ranks of approaches in each performance metric and the overall rank of algorithms is indicated in Table 5. To determine the approach with the best performance for a specific function, the overall rank of each approach is calculated. In this study two ranks are determined, namely R_1: the sum of all ranks calculated for each approach with regards to specific measure, and R_2: the sum of ranks calculated for each approach with regards to specific measure that rely on the true POF. A comparison between DNSGAII (A,B) [8], dCOEA [13] as a co-evolutionary paradigm that hybridizes competitive and cooperative mechanisms, and best approach of DVEPSO that hybrids the vector evaluating strategies with particle swarm optimization [18, 19] with approach 'H' as the best obtained approach is done and shown in Table 6. In addition, Table 7 compares the proposed method with new achievements of state of the art. Tables 8, 9 and 10 analyze the effect of number of solutions that should be re-initialized on the performance of the dynamic multi-objective optimization algorithm. Tables 11, 12 and 13 analyzes the effect of mixture ratio (MR) on the performance of the approach 'H' for 500 iterations at re-initialization rate of 30 of 100. Finally, Table 14 investigates the effect of CSO on the performance of the approach 'H' and compares it with pure PSO and some traditional variants.

4.3.1 Comparison of nine strategies

Table 3 shows the obtained performance measure values of nine approaches for $dMOP_1$, $dMOP_2$, $dMOP_{2iso}$, HE_2, FDA_1, FDA_3, with $n_t = 10$ and $\tau_t = 10$, and 50 by mean (M) and standard deviation (Std). In $dMOP_1$, 'H' and 'I' that use Borda selection performed the best with regards to {IGD-HVR-Acc alt} and {Spacing-VGD} respectively. The approach 'H' achieved the best rank for both R_1 and R_2 for $n_t = 10$ and $\tau_t = 10$ while approach 'I' is the best approach and performed the best with regards to {IGD-MS} for $n_t = 10$ and $\tau_t = 50$. Also statistical results indicate that the lowest standard deviation for all measures is obtained by I, B, G, B, C, G, and C respectively. Furthermore, it shows that approach 'I' and 'C' gets the best results for both mean and standard deviation for Spacing and Stability respectively. The obtained results of second dMOOP show that random selection performed better than other ways. Random selection and Re-initialization

Table 5 Overall Ranking of Algorithms

Approach	n_t–τ_t	Average R_1	Average R_2	Overall R_1	Overall R_2
A	10–10	3	6	4	3
	10–50	5	6		
B	10–10	7	5	7	2
	10–50	6	3		
C	10–10	7	9	5	4
	10–50	4	6		
D	10–10	5	2	3	**1**
	10–50	1	1		
E	10–10	4	7	5	4
	10–50	7	8		
F	10–10	4	3	6	2
	10–50	8	5		
G	10–10	6	8	8	4
	10–50	8	7		
H	10–10	1	1	**1**	**1**
	10–50	2	2		
I	10–10	2	4	2	2
	10–50	3	4		

Best found answers in each table are indicated in Bold

Table 4 Ranks of approaches in each performance measure

Approach	R_S	R_{IGD}	R_{VGD}	R_{MS}	R_{HVR}	$R_{acc\ alt}$	R_{Stab}
A	4	8	**1**	9	6	6	**1**
B	9	4	8	8	3	3	7
C	3	9	5	6	7	9	3
D	6	**1**	4	2	2	2	7
E	2	7	3	7	8	7	2
F	8	3	6	3	4	4	8
G	5	6	7	4	9	8	5
H	7	2	6	**1**	**1**	**1**	6
I	**1**	5	2	5	5	5	4

Best found answers in each table are indicated in Bold

Table 6 Comparison between the proposed approach and traditional algorithms

Benchmarks	n_t–τ_t	Algorithm	Spacing	Stab	VGD	MS
FDAI	10–10	DVEPSOc	**0.00043**	**0.00154**	0.06593	0.9761
		DNSGAII(a)	0.00494	0.00339	0.83219	0.7869
		DNSGAII(b)	0.000612	0.00543	1.1339	1.1947
		dCOEA	0.00132	0.01328	1.1318	**2.485**
		Approach 'H'	0.00049	0.01030	**0.05052**	0.9483
	10–50	DVEPSOu	0.00033	0.00126	0.15148	0.9107
		DNSGAII(a)	0.00032	0.000030	0.1716	0.9885
		DNSGAII(b)	0.00033	**0.000022**	0.17261	0.9877
		dCOEA	0.00026	0.00017	0.1515	0.95904
		Approach 'H'	**0.000134**	0.00522	**0.07578**	**0.9992**
FDAIII	10–10	DVEPSOu	0.00081	0.00068	0.9748	0.8167
		DNSGAII(a)	0.00318	2E−253	1.326	1.099
		DNSGAII(b)	0.0049	2E−253	1.316	1.183
		dCOEA	0.00076	**1.5**E−**253**	1.085	**1.305**
		Approach 'H'	**0.00069**	0.00908	**0.77691**	0.9990
	10–50	DVEPSOu	0.00088	0.000036	0.6847	0.98049
		DNSGAII(a)	0.00137	3.24E−250	1.154	0.99744
		DNSGAII(b)	0.00141	**2.78**E−**250**	1.167	0.99743
		dCOEA	0.00065	2.08E−249	**0.07537**	0.9469
		Approach 'H'	**0.00047**	0.01475	0.6123	**0.9996**
dMOP1	10–10	DNSGAII(a)	0.00577	**0.000036**	0.1521	0.9834
		DNSGAII(b)	0.00497	0.000059	0.1535	0.9397
		dCOEA	0.00045	0.00253	0.0389	0.8623
		DVEPSOc	0.0040	0.00035	0.2634	0.8790
		Approach 'H'	**0.000399**	0.00082	**0.03517**	**0.9957**
	10–50	DNSGAII(a)	0.00034	0.000013	0.1178	0.9832
		DNSGAII(b)	0.00032	**0.000012**	0.121	0.9833
		dCOEA	0.00023	0.00021	0.0957	0.9783
		DVEPSOu	0.00126	0.0088	1.603	0.6624
		Approach 'H'	**0.000110**	0.00010	**0.00529**	**1**
dMOP2	10–10	DNSGAII(a)	0.00095	0.00064	0.9041	**1.456**
		DNSGAII(b)	0.00212	0.00068	1.037	1.439
		dCOEA	0.0011	0.00213	0.8129	1.409
		DVEPSOd	0.00062	**0.00042**	0.0740	0.9793
		Approach 'H'	**0.000332**	0.00496	**0.06779**	0.9876
	10–50	DNSGAII(a)	0.00032	0.00014	0.1564	0.9955
		DNSGAII(b)	0.00032	**0.00012**	0.1406	0.9963
		dCOEA	0.00027	0.0022	0.1524	0.9543
		DVEPSOu	0.00016	0.044	0.1593	0.8733
		Approach 'H'	**0.000155**	0.00310	**0.02570**	**0.99808**
HE2	10–10	DNSGAII(a)	0.00062	0.00213	0.20337	0.9193
		DNSGAII(b)	0.00061	0.00206	0.20331	0.9208
		dCOEA	0.0045	0.0159	0.2345	0.6925
		DVEPSOc	0.01986	0.0130	1.524	**0.9878**
		Approach 'H'	**0.000089**	**0.00014**	**0.04126**	0.8613
	10–50	DNSGAII(a)	0.00063	0.00048	0.1913	0.91808
		DNSGAII(b)	0.00062	0.00048	0.1753	0.9179
		dCOEA	0.0014	0.00346	0.2538	0.8177
		DVEPSOp	0.01708	0.00244	1.7375	**1.1905**
		Approach 'H'	**0.000081**	**0.000136**	**0.03363**	0.84303

Best found answers in each table are indicated in Bold

Results are indicated by Mean value

Table 7 Comparison of the proposed approach with new algorithms of state of the art

Function	nt–τt	Measure	Optimization Algorithm		
			OPMOED	PDNNIA	Approach 'H'
FDA_1	10–15	GD	2.63E−2	1.12E−22	8.5E−23
		Spacing	1.78E−22	1.02E−22	1.84E−24
	5–25	GD	1.42E−22	7.28E−23	1.1E−22
		Spacing	1.4E−22	7.22E−23	2.28E−24
$dMOP_1$	10–15	GD	4.16E−23	1.02E−22	6.7E−23
		Spacing	7.24E−23	7.73E−23	2.63E−24
	5–25	GD	1.93E−23	6.1E−24	1.16E−23
		Spacing	6.22E−23	3.81E−23	1.42E−24
$dMOP_2$	10–15	GD	9.55E−23	3.35E−23	8E−23
		Spacing	1.22E−22	5.75E−23	2.89E−24
	5–25	GD	5.53E−22	4.74E−23	4.77E−23
		Spacing	1.06E−22	6.87E−23	2.36E−24

Results are indicated by the mean value

without/with opposite search based algorithm approaches have the best performance. Simulation results of $dMOP2_{iso}$ indicate that for $n_t = 10$ and $\tau_t = 10$, approach 'I' again performed the best performance for VGD, accuracy, and stability. For a higher frequency of change Re-initialization based on mutation may mislead the algorithm and just using Borda selection is useful. Obtained results of simulating the HE_2 function indicate that approaches 'A', 'D' and 'D', 'H' obtained the best results for $\tau_t = 10$ and 50, respectively. While approach 'D' obtained the best rank for metrics of IGD and HVR. Although approach 'F' has the best rank for $\tau_t = 10$ in simulation of FDA_1, but it is notable that this approach is not remarkable for $\tau_t = 50$ and using a Borda selection with mutation strategy may lead the algorithm to true search trajectory. This approach achieved the best rank for the following metrics: spacing, IGD, VGD, MS, ACC alt, and Stab. Simulation results of FDA_3 show that for frequency of change 10 the approaches of 'A', 'B', 'E', 'H', and 'I' performed the best with regards to VGD, Stab, HVR, MS, and Spacing respectively. Also for $\tau_t = 50$, Borda selection with random re-initialization has the best rank with regards to all measures. Table 4 shows the ranks of approaches in each metric. For obtaining this, Ranks of approaches for each performance measure in each function with regards to frequency of change 10, and 50 has been calculated and then added. This Table indicates that approach 'H' beats the other approaches in three performance metrics includes of: MS, HVR, and Acc alt. In Table 5 ranks of each approach in each benchmark has been calculated and are then summed. As Table 4 shows based on first ranking method, R_1, approaches 'H', 'I' and 'D', 'H' are the best ranks for $\tau_t = 10$, and $\tau_t = 50$ respectively. Also approaches 'D' and 'H' obtained the best rank of R_2 for both $\tau_t = 10$, and $\tau_t = 50$. Finally, by summing ranks for two frequencies of change, it is obtainable that approach 'H' and approaches 'D', and 'H' can be the better candidate based on R_1 and R_2 respectively. It means that not only 'H' can be an approach for all metric types, but also is efficient for facing with measures just depending on true POF.

4.3.2 Comparison with traditional approaches

Table 6 shows the comparison between traditional algorithms with approach 'H' by four performance metrics (these comparisons are based on mean value). The results can be analyzed by per function, and per performance measure. The simulation results of function FDA1 indicate that the proposed method gets the best results for $n_t = 10$ and $\tau_t = 50$ in comparison with other algorithms. It shows that using Borda count as selection method gets the best performance of Spacing, VGD, and MS at $n_t = 10$ and $\tau_t = 50$. For function FDA3, dCOEA gets the best performance of Stability and MS while the proposed method has the better result for Spacing and VGD. For function dMOP1, Borda gets the best value of Spacing, MS, and VGD at both frequencies. For dMOP2, Borda obtained the best value for measures of Spacing and VGD for $n_t = 10$ and $\tau_t = 10$ and Spacing, VGD, and MS for $n_t = 10$ and $\tau_t = 50$. Finally, results show that for HE2 the best results of Spacing, Stability and VGD are obtained by Borda method for change frequencies of 10 and 50 while DVEPSO just gets the best for MS.

The result can be investigated by change frequency of environment too. It shows that the proposed method obtains quite satisfactory results in comparison with other approaches. It is notable that Borda method conquers all approaches for frequencies of 10 and 50 except for function of FDA_1 at change frequency of 10 where DVEPSO gets the best results. For example for function HE_2 with $\tau_t = 50$, approach 'H' in comparison with other algorithms gets the first rank for Spacing (0.000081), Stab (0.00013), and VGD (0.0336) while for MS performance metric is the fourth rank. Best found answers among five algorithms for each function and for each performance metric are indicated in bold.

4.3.3 Comparison with new achievements

Also the proposed method is compared with two new introduced algorithms, OPMOED [30] and PDNNIA [29], by measures of GD and SP. First, uses an orthogonal predictive model to predict new individuals after a change occurred while the second, hybrids a modified predictive model with a differential evolution operator. The comparison is done for three functions consisted of: FDA_1, $dMOP_1$, and $dMOP_2$ at $n_t = 10$ and $\tau_t = 15$, and $n_t = 5$ and

Table 8 Simulation results of running approach 'H' and 'W'(Selecting worst particles just before a change) for 500 iterations and population size of 100

Function	n_t–τ_t	R	Approach	Spacing	IGD	VGD	MS	HVR	Acc alt	Stab
$dMOP_1$	10–10	5	H	2.906E−4	0.00186	0.03030	0.99428	0.98203	0.01783	0.00200
			W	3.019E−4	0.00273	0.01385	0.98260	0.97179	0.01915	0.00343
		15	H	1.86E−4	0.00180	0.02533	0.99288	0.98306	0.01737	6.68E−4
			W	3.82E−4	0.00174	0.02698	0.99518	0.98196	0.0174	0.00319
		30	H	2.68E−4	0.01749	0.02134	0.79375	0.78689	0.101188	4.54E−4
			W	3.393E−4	0.00180	0.0370	0.99419	0.98255	0.0196	0.00869
	10–50	5	H	1.082E−4	0.00100	0.00511	0.999105	0.9883	0.00425	9.61E−5
			W	1.109E−4	0.00105	0.00505	0.99906	0.98732	0.00464	1.649E−4
		15	H	1.057E−4	0.00107	0.00510	0.99542	0.9877	0.00446	9.74E−5
			W	1.097E−4	0.00109	0.00503	0.999524	0.98665	0.00487	0.01504
		30	H	1.058E−4	0.00107	0.00515	0.99828	0.98721	0.00467	2.20E−4
			W	1.0875	0.00107	0.00516	0.9974	0.98784	0.00456	0.00713
$dMOP_2$	10–10	5	H	5.696E−4	0.14280	1.588	1.5146	0.6372	1.6801	0.00972
			W	5.295E−4	0.14306	1.6030	1.5117	0.63517	1.6820	0.00952
		15	H	5.726E−4	0.14252	1.6070	1.5169	0.6472	1.6767	0.00436
			W	5.238E−4	0.14268	1.5844	1.5142	0.6420	1.6790	0.00827
		30	H	8.105E−4	0.14234	1.6014	1.5180	0.64700	1.6730	0.00476
			W	5.321E−4	0.14241	1.5624	1.5169	0.64329	1.6744	0.01021
	10–50	5	H	3.080E−4	0.00425	0.10753	0.98548	0.90978	0.05935	0.00301
			W	2.453E−4	0.00345	0.07024	0.99128	0.9165	0.04745	0.00877
		15	H	3.55E−4	0.00323	0.0727	0.99329	0.92431	0.0439	0.00495
			W	2.744E−4	0.00339	0.07312	0.99138	0.91878	0.04626	0.01065
		30	H	3.937E−4	0.00333	0.08709	0.99236	0.91331	0.04532	0.00334
			W	2.788E−4	0.00286	0.07917	0.99652	0.9306	0.03902	0.01092
$dMOP_{2iso}$	10–10	5	H	1.935E−4	8.964E−4	0.2909	0.9964	0.9897	0.00734	0.01889
			W	1.340E−4	9.048E−4	0.0233	0.99722	0.9897	0.00688	0.01355
		15	H	1.638E−4	8.79E−4	0.02159	0.99727	0.9904	0.00623	0.02281
			W	1.913E−4	8.713E−4	0.02378	0.99783	0.99040	0.00631	0.0876
		30	H	1.932E−4	9.189E−4	0.0272	0.99670	0.98900	0.00742	0.0198
			W	2.1913E−4	8.835E−4	0.03239	0.99747	0.99012	0.00727	0.01480
	10–50	5	H	1.290E−4	0.00154	0.03249	0.99332	0.97784	0.01635	0.01561
			W	1.486E−4	0.00143	0.03200	0.99394	0.97580	0.01535	0.01182
		15	H	1.702E−4	8.44E−4	0.02682	0.99605	0.98989	0.00669	0.03191
			W	1.3718E−4	0.001515	0.02182	0.99338	0.9795	0.0148	0.01924
		30	H	1.652E−4	0.00184	0.02606	0.99313	0.97357	0.01486	0.01786
			W	1.1815E−4	0.002115	0.03412	0.9882	0.9588	0.02417	0.01955
HE_2	10–10	5	H	8.875E−4	0.00612	0.0425	0.86176	0.9594	0.05167	1.486E−4
			W	9.407E−5	0.00606	0.04377	0.86240	0.96014	0.05108	1.88E−4
		15	H	9.635E−5	0.00605	0.04452	0.86235	0.96033	0.0509	2.504E−4
			W	1.012E−4	0.00605	0.04452	0.86218	0.96025	0.05106	2.405E−4
		30	H	1.010E−4	0.00602	0.04498	0.86260	0.9605	0.05101	3.269E−4
			W	1.029E−4	0.00601	0.04563	0.8625	0.96031	0.05101	2.922E−4
	10–50	5	H	8.138E−5	0.00610	0.03400	0.8463	0.9266	0.06119	1.568E−4
			W	9.376E−5	0.00607	0.03651	0.8464	0.92703	0.06086	2.804E−4
		15	H	1.001E−4	0.00605	0.03817	0.8463	0.92811	0.0599	2.67E−4
			W	9.601E−5	0.00608	0.03646	0.84610	0.9277	0.06027	2.322E−4
		30	H	9.406E−5	0.00607	0.03895	0.8462	0.92780	0.0602	3.152E−4
			W	1.0133E−4	0.00604	0.04167	0.8466	0.92801	0.06002	2.165E−4

Table 8 (continued)

Function	n_t–τ_t	R	Approach	Spacing	IGD	VGD	MS	HVR	Acc alt	Stab
FDA_1	10–10	5	H	5.284E−4	0.01271	0.44617	0.95325	0.90732	0.19123	0.0103
			W	5.746E−4	0.01553	0.48213	0.9387	0.89139	0.22943	0.01365
		15	H	6.121E−4	0.01034	0.39405	0.96048	0.91541	0.15246	0.01093
			W	5.968E−4	0.00949	0.4176	0.96266	0.92314	0.14266	0.00935
		30	H	5.262E−4	0.00834	0.4066	0.9694	0.9264	0.12617	0.00900
			W	6.387E−4	0.00895	0.42831	0.9657	0.92319	0.1343	0.01542
	10–50	5	H	1.691E−4	0.00411	0.07855	0.98697	0.94645	0.05738	0.01225
			W	2.175E−4	0.00927	0.1663	0.96241	0.90953	0.1304	0.01156
		15	H	1.917E−4	0.00334	0.09459	0.99077	0.95703	0.04630	0.01135
			W	1.758E−4	0.00214	0.0639	0.99731	0.9687	0.02900	0.0017
		30	H	2.143E−4	0.00491	0.1333	0.98191	0.9405	0.06902	0.01096
			W	3.037E−4	0.00884	0.22365	0.96366	0.9177	0.12521	0.01133

'R' is referred to Re-initialization rate and results are indicated by mean value

Table 9 Calculated rank for running approach 'H' and 'W' by re-initialization rate of 5, 15, and 30

Function	n_t–τ_t	Approach	Rank(R_1)		
			Rate:5	Rate:15	Rate:30
$dMOP_1$	10–10	H	2	**1**	3
		W	2	**1**	3
	10–50	H	**1**	2	3
		W	**2**	3	1
$dMOP_2$	10–10	H	3	2	1
		W	3	2	1
	10–50	H	3	**1**	2
		W	3	**2**	1
$dMOP_{2iso}$	10–10	H	2	**1**	3
		W	2	**1**	3
	10–50	H	2	**1**	3
		W	2	**1**	3
HE_2	10–10	H	3	**1**	2
		W	3	**2**	1
	10–50	H	2	**1**	3
		W	3	**2**	1
FDA_1	10–10	H	3	2	1
		W	3	2	13
	10–50	H	2	**1**	3
		W	3	**1**	2
Overall rank	10–10	H	3	**1**	2
		W	3	**1**	2
	10–50	H	2	**1**	3
		W	3	**2**	1

Best found answers in each table are indicated in Bold

$\tau_t = 25$. The results indicated in Table 7 show that for function FDA1, the proposed method gets the quite satisfactory results for both GD and SP at $n_t = 10$ and $\tau_t = 15$. Also for function dMOP2 at frequencies 15 and 25, the proposed method and PDNNIA achieved the best performance for measures of SP and GD respectively. As a result, Table 7 shows that the proposed approach has the best performance for spacing in all conditions while the best overall performance for GD will be obtained by PDNNIA.

4.3.4 Re-initialization rate and selection process analysis

Table 8 analyses the effect of number of solutions that should be re-initialized on the performance of the dynamic multi-objective optimization algorithm. Here, the simulations are tested on the approach 'H' as the best method proposed in the previous results and approach 'W'. Although using Borda count ranking method is the main idea of this study to select solutions that should be re-initialized, but Table 8 investigates another method for selection process. Here, the worst cats just before a change occurred are selected to be re-initialized and indicated as 'W' in table. The re-initialization rate (indicated as 'R' in table) of this study is 5, 15, and 30. Simulation results of this table and calculated ranks showed in Table 9 indicate that the best performance of approach 'H' has been obtained by re-initializing the 15 percent of the population for both $\tau_t = 10$ and $\tau_t = 50$ while the best performance of the approach 'W' will be obtained by re-initialization rate of 15, and 30 for $\tau_t = 10$ and $\tau_t = 50$ respectively. This table shows that the best re-initialization rate of approach 'H' is 15 for function $dMOP_1$, $dMOP_{2iso}$, HE_2 and 30 for $dMOP_2$ and FDA_1 at $n_t = \tau_t = 10$. Also the best performance of the approach 'H' at $n_t = 10$ and $\tau_t = 50$ will be obtained by re-initialization rate of 5 for $dMOP_1$ and 15 for other functions. Table 10 compares the simulation results of running approach 'H' and 'W' obtained from Table 9 and shows the calculated score of approaches for each function and specific

Table 10 Comparison of overall wins of approach 'H', and 'W' for re-initialization rate of 5, 15, and 30

Function	n_t–τ_t	Rate	Score	
			H	W
$dMOP_1$	10–10	5	6	1
		15	5	2
		30	3	4
	10–50	5	6	1
		15	6	1
		30	4	2
$dMOP_2$	10–10	5	4	3
		15	5	2
		30	5	2
	10–50	5	1	6
		15	5	2
		30	1	6
$dMOP_{2iso}$	10–10	5	1	6
		15	3	3
		30	2	5
	10–50	5	2	5
		15	4	3
		30	6	1
HE_2	10–10	5	4	3
		15	5	1
		30	4	2
	10–50	5	3	4
		15	4	3
		30	2	5
FDA_1	10–10	5	7	0
		15	1	6
		30	7	0
	10–50	5	6	1
		15	1	6
		30	7	0
Overall wins	10–10		**10**	5
	10–50		**9**	6

Best found answers in each table are indicated in Bold

re-initialization rate (R). The word score is referred to the number of measures that the approach is won. For instance, for function $dMOP_1$, it is shown that approach 'H' has the better performance ('H' won six performance measures while 'W' just one) in comparison with approach 'W' at $n_t = 10$ and $\tau_t = 10$ with re-initialization rate of 5%. Overall wins of this table indicate the number of wins of each approach in each case. Final results show that approach 'H' wins 19 cases of 30.

4.3.5 Analysis of mixture ratio

Table 11 analyses the effect of mixture ratio (MR) on the performance of the approach 'H' for 500 iterations at re-initialization rate of 30 of 100. Mixture ratio divides the cats into seeking and tracing mode. So the obtained results of mixture ratio of 0, 0.5, and 1 are showed. Mixture ratio of zero and one indicates that the optimization algorithm is just in tracing mode (local search) and seeking mode (global search) while by using MR of 0.5 both local and global search ability are used simultaneously. Calculated ranks of using different mixture ratio are indicated in Table 13 and show that if the optimization algorithm uses mixture ratio of 0.5, it gets the best results for functions of $dMOP_1$, $dMOP_2$, and FDA_1 while by using mixture ratio of 1, it gets the best for the other two functions. Furthermore, Table 13 suggests that focusing on global search only seems to be a reasonable alternative while it is likely that using MR: 0.8 obtains the better performance against just global search. So in order to compare both algorithms with traditional approaches, two functions: HE_2 and $dMOP_1$ have been selected to test and the results are mentioned in Table 12. The table shows that the best results will be obtained by using MR: 0.8 (80% on seeking and 20 on tracing) for both frequencies of change. It is obtainable that this approach achieves the best performance for measures of {Stability- VGD} and {VGD-MS} for HE_2 and $dMOP_1$ respectively. Moreover, disregarding the results of this approach indicates that the second best method is obtained by using just global search algorithm (MR = 1). This approach achieves the best measures of Spacing, Stability, and VGD at both frequencies of HE_2 and best value for MS at both frequencies and best Spacing and VGD at frequency of 50 in $dMOP_1$.

4.3.6 CSO vs some variants of PSO

As previously mentioned, the final part of the proposed approach (H) is CSO to find the best solutions of search space. Here, performance effect of CSO in the proposed approach is compared with pure PSO [23] and four important variants of PSO consisted of fully informed PSO: FIPSO [33], comprehensive learning PSO: CLPSO [26], dynamic adaptive PSO: DAPSO [41], and heterogonous PSO: HPSO [9]. Table 14 investigates the effect of CSO on the performance of the approach 'H' and compares it with four PSO-based algorithms. The simulation is done for six test functions and then is evaluated based on different performance measures at re-initialization rate of 30% for running at 1000 iterations. Population size for each algorithm is 100 and other algorithms' parameters are as the same as original paper.

Table 11 Comparison of different mixture ratio: just local (MR:0), local + global (MR:0.5), just global (MR:1) ('MR' is referred to mixture rate and results are indicated by mean value)

Function	n_t–τ_t	MR	Spacing	IGD	VGD	MS	HVR	Acc alt	Stab
$dMOP_1$	10–10	0	5.536E−4	0.0189	0.0515	0.7851	0.7206	0.1335	0.0088
		0.5	1.412E−4	0.00102	0.0101	0.9984	0.9886	0.0065	3.94E−4
		1	0.00164	0.00253	0.2941	0.9868	0.9897	0.0293	0.0134
	10–50	0	2.83E−4	0.0262	0.0103	0.66239	0.6292	0.1465	7.14E−4
		0.5	9.85E−5	9.34E−4	0.0051	0.9992	0.9883	0.0043	4.46E−5
		1	1.049E−4	8.82E−4	0.0085	0.9955	0.9910	0.0032	0.0029
$dMOP_2$	10–10	0	0.00159	0.2089	2.096	1.564	0.3037	2.362	0.0731
		0.5	3.74E−4	0.14153	1.5321	1.5269	0.6591	1.6592	0.0140
		1	7.88E−4	0.14303	1.906	1.509	0.6926	1.682	0.0092
	10–50	0	0.00128	0.0552	0.3594	0.6106	0.3002	0.5428	0.0934
		0.5	2.697E−4	0.0027	–	0.99139	0.93100	0.0312	0.0089
		1	3.84E−4	0.0043	0.782	0.9756	0.9672	0.0662	0.0067
$dMOP_{2iso}$	10–10	0	0.00163	0.03012	0.2131	0.7867	0.6098	0.3339	0.0398
		0.5	1.348E−4	0.00150	0.0156	0.9837	0.9848	0.0114	0.00107
		1	1.956E−4	9.43E−4	0.0286	0.9930	0.9916	0.0051	0.0109
	10–50	0	3.77E−4	0.0389	0.1744	0.6778	0.5205	0.3188	0.0581
		0.5	1.057E−4	0.0052	0.0491	0.9624	0.9255	0.0631	0.0024
		1	1.088E−4	9.2E−4	0.0064	0.9892	0.9907	0.0036	0.0152
HE_2	10–10	0	1.14E−8	0.0814	1.11E−6	4.02E−4	1.16E−5	0.9598	0.3098
		0.5	7.04E−5	0.0060	0.0422	0.8623	0.9580	0.0513	4.67E−4
		1	1.04E−4	0.0060	0.0456	0.8623	0.9604	0.0509	3.12E−4
	10–50	0	3.106E−4	0.0144	0.0184	0.8459	0.6428	0.3032	0.0371
		0.5	7.68E−5	0.00608	0.0393	0.8463	0.92456	0.0620	6.65E−4
		1	9.35E−5	0.00610	0.0380	0.8459	0.9273	0.0605	4.23E−4
FDA_1	10–10	0	0.0027	0.1623	1.738	1.90	0.4241	2.047	0.0971
		0.5	6.01E−4	0.00290	0.1639	0.9958	0.9596	0.0433	0.0150
		1	5.717E−4	0.0516	1.302	0.8502	0.8164	0.7090	0.0073
	10–50	0	0.00282	0.0889	0.9305	0.8249	0.497	1.207	0.1073
		0.5	2.55E−4	0.00302	0.0361	0.9922	0.9520	0.0409	0.0102
		1	3.11E−4	0.0244	0.7119	0.900	0.8763	0.3383	0.00403

Simulation results of function $dMOP_1$ indicate that CLPSO and CSO achieved better performance in comparison with other algorithms. Here, results show that CLPSO has the better values for measures of MS, HVR, and ACC alt while CSO is the better candidate based on metrics of Spacing, VGD, and Stability. For function $dMOP_2$, it is obtainable that none of algorithms achieve the best results for all measures. In this function, the best values of Spacing-VGD, MS, HVR, ACC alt, and Stability have been achieved by HPSO, FIPSO, CLPSO, and CSO respectively. Simulation of estimating the POF of function $dMOP_{2iso}$ shows the quite satisfactory results of using CSO in comparison with other PSO-based algorithms. Table 14 shows that this algorithm achieves the best values for all metrics. Both CSO and HPSO is the best candidate for function HE_2 where the best values of {Spacing–HVR–Stability} and {VGD–ACC alt} are obtained by CSO and HPSO respectively. Finally, this table shows that CSO obtains the best values of maximum spread for FDA_1 and FDA_3 while achieves the best HVR for FDA_3. By summarizing the simulation results of all functions, it is clearly that CSO is the best candidate based on measures of Spacing, MS, and Stability while CLPSO obtains the best value of ACC alt. Also {HPSO, CSO} and {CLPSO, CSO} obtain the best performance for measures of VGD, and HVR respectively.

4.4 Dynamic multi-objective heating optimization

The main goal of a residential heating model is usually to keep the indoor temperature within comfortable limits and find the optimal value of the indoor temperature by taking account of the heating costs. So this problem is investigated as a multi-objective problem where the optimizers try to minimize the consumed heating energy, cost of energy and deviation of the current temperature from ideal temperature as follows [32]:

Table 12 Comparison of using MR: 0.8, MR: 1 with traditional approaches

Function	n_t–τ_t	Algorithm	Spacing	Stab	VGD	MS
HE_2	10–10	DNSGAII(a)	0.00062	0.00213	0.20337	0.9193
		DNSGAII(b)	0.00061	0.00206	0.20331	**0.9208**
		dCOEA	0.0045	0.0159	0.2345	0.6925
		Just Global (MR = 1)	0.00010	0.00031	0.0456	0.8623
		Hybrid (MR = 0.8)	**0.000090**	**0.00018**	**0.04246**	0.8613
	10–50	DNSGAII(a)	0.00063	0.00048	0.1913	**0.91808**
		DNSGAII(b)	0.00062	0.00048	0.1753	0.9179
		dCOEA	0.0014	0.00346	0.2538	0.8177
		Just Global (MR = 1)	**0.00007**	0.00042	0.0380	0.8459
		Hybrid (MR = 0.8)	0.00081	**0.00023**	**0.03360**	0.8435
$dMOP_1$	10–10	DNSGAII(a)	0.00577	**0.000036**	0.1521	0.9834
		DNSGAII(b)	0.00497	0.000059	0.1535	0.9397
		dCOEA	0.00045	0.00253	0.0389	0.8623
		Just Global (MR = 1)	0.00164	0.0134	0.2941	0.9868
		Hybrid (MR = 0.8)	**0.00039**	0.0008	**0.03157**	**0.9950**
	10–50	DNSGAII(a)	0.00034	0.000013	0.1178	0.9832
		DNSGAII(b)	0.00032	**0.000012**	0.121	0.9833
		dCOEA	0.00023	0.00021	0.0957	0.9783
		Just Global (MR = 1)	**0.00010**	0.0029	0.0085	0.9955
		Hybrid (MR = 0.8)	0.00089	0.0009	**0.0055**	**0.999**

Best found answers in each table are indicated in Bold

$$
\begin{aligned}
&\min f = (f_1, f_2, f_3)\\
&f_1 = \sum_{i=0}^{n-1} p_i \times q_i\\
&f_2 = \sum_{i=0}^{n-1} q_i\\
&f_3 = \sum_{i=0}^{n-1} |T_i - T_i^{ideal}|\\
&\text{S.t:}\begin{cases} T_0 = T_n \\ l_i \le T_i \le u_i \\ 0 \le q_i \le q \\ T_{i+1} = T_i + \beta \Delta t q_i - \alpha\beta\Delta t(T_i - T_i^{out}) \end{cases}
\end{aligned}
\tag{22}
$$

Table 13 Ranks obtained by the Table 11

Function	n_t–τ_t	Rank(R1)		
		MR:0	MR:0.5	MR:1
$dMOP_1$	10-10	3	**1**	2
	10-50	3	**1**	2
$dMOP_2$	10-10	3	**1**	2
	10-50	3	**1**	2
$dMOP_{2iso}$	10-10	3	2	**1**
	10-50	3	2	**1**
HE_2	10-10	3	2	**1**
	10-50	3	2	**1**
FDA_1	10-10	3	**1**	2
	10-50	3	**1**	2

Best found answers in each table are indicated in Bold

f_1, f_2, and f_3 represent heating costs, heating energy, and deviation from the ideal temperature respectively. T_i represents indoor temperature; q_i represents the heating power at time i; The initial indoor temperature is indicated as T_0 and the lower and upper bounds of T_i are indicated as l_i and u_i respectively; q is the maximum heating capacity of the heating system; p_i is the hourly price of electricity at time i; constants α, β are 0.17 and 0.038 respectively; and T_i^{ideal} is the hourly ideal indoor temperature.

4.4.1 Simulation results of the heating optimization

The main goal in heating optimization problem is to find an optimum value of heating energy consumption during a day. So the algorithm should be able to find these optimum values with according to satisfy three criteria included of heating cost, heating energy, and deviation from ideal point to determine that how much energy should be given to system to reach to desired temperature. The characteristics of the heating optimization problem for 24 h of a day are as

Table 14 Comparison of CSO with pure PSO, FIPSO, CLPSO, DAPSO, and HPSO in main core of optimization phase in approach 'H' for 1000 iterations and population size of 100

Function	n_t–τ_t	Algorithm	Spacing	VGD	MS	HVR	Acc alt	Stab
$dMOP_1$	10–10	Pure PSO	0.00229	0.0195	0.86721	0.96979	0.0604	0.12636
		FIPSO	0.00190	0.0110	0.9585	0.95005	0.0801	0.01254
		CLPSO	0.00048	0.0368	**0.9980**	**0.9990**	**0.0091**	0.00156
		DAPSO	0.00152	0.0856	0.9910	0.9812	0.0236	0.01760
		HPSO	0.00045	0.0405	0.9945	0.9989	0.0185	0.00458
		CSO	**0.0003994**	**0.0351**	0.9957	0.98811	0.0149	**0.000823**
$dMOP_2$	10–10	Pure PSO	0.00642	1.5432	1.3953	0.4193	1.796	0.0497
		FIPSO	0.000698	1.0015	**1.4102**	0.8817	1.0010	0.00980
		CLPSO	0.000310	0.07105	0.9989	**0.9652**	**0.00946**	0.0167
		DAPSO	0.0098	1.0002	1.2592	0.9069	0.09521	0.00523
		HPSO	**0.000119**	**0.0125**	0.9986	0.9454	0.0240	0.0198
		CSO	0.000322	0.06779	0.98768	0.91348	0.0524	**0.00496**
$dMOP_{2iso}$	10–10	Pure PSO	0.01286	0.20046	0.83678	0.64063	0.30177	0.11988
		FIPSO	0.00269	0.10009	0.9696	0.70521	0.12058	0.0169
		CLPSO	0.000185	0.0235	0.9971	0.9800	0.00690	0.00812
		DAPSO	0.00980	0.09915	0.9815	0.95412	0.01701	0.0529
		HPSO	0.00098	0.0349	0.9909	0.98132	0.00909	0.01003
		CSO	**0.000151**	**0.01397**	**0.99775**	**0.98939**	**0.00643**	**0.00576**
HE_2	10–10	Pure PSO	0.00067	0.02966	0.60204	0.58960	0.3757	0.0520
		FIPSO	0.00012	0.01980	0.7958	0.6971	0.35201	0.00159
		CLPSO	0.00009	0.00230	**0.91020**	0.95102	0.05402	0.000201
		DAPSO	0.00086	0.02300	0.8508	0.89021	0.4109	0.00187
		HPSO	0.00010	**0.00127**	0.8081	0.95006	**0.0298**	0.000419
		CSO	**0.000089**	0.04126	0.86135	0.**95840**	0.05188	**0.00014**
FDA_1	10–10	Pure PSO	0.01609	0.33650	0.7725	0.6422	0.49387	0.0595
		FIPSO	0.00065	0.31025	0.79651	0.70205	0.4140	0.1502
		CLPSO	0.000546	**0.01962**	0.93265	**0.96080**	**0.01298**	0.0098
		DAPSO	0.000657	0.4850	0.81025	0.9174	0.39620	0.1985
		HPSO	**0.0000518**	0.1094	0.93690	0.90369	0.1960	**0.00910**
		CSO	0.000496	0.5052	**0.9483**	0.9097	0.2067	0.0103
FDA_3	10–10	Pure PSO	0.02085	0.64912	0.57863	2.0577	2.997	0.0502
		FIPSO	0.000854	0.2982	0.8525	2.0001	2.0125	0.0196
		CLPSO	**0.000012**	0.39850	0.99850	0.9989	1.257	0.00851
		DAPSO	0.000680	**0.10129**	0.95215	0.1951	2.851	0.00527
		HPSO	0.000196	0.8505	0.98598	1.125	**1.118**	**0.00493**
		CSO	0.000696	0.7769	**0.9990**	**2.443**	2.660	0.00908

Best found answers in each table are indicated in Bold

Re-initialization rate is 30% and results are indicated by mean value

follows: P is randomly distributed between [0, 10], T^{ideal} is between [17, 22], T^{out} is [−8, −6, −1, −3], and the number of heating energy variables as decision makers are 24 and between [0, 10]. Experimental results of solving heating optimization problem using the proposed approach indicate that the average, maximum, and minimum error (deviation of current temperature from ideal temperature) are 41.808, 61.13, and 29.56 respectively while these values are 42.80, 61.13, and 29.52 for $DNSGA_{II}$ (b). It is notable that by dividing the mean value of deviation from ideal temperature during 24 h of a day into 24, the average deviation reaches to 1.74 and 1.78 in an hour by Borda and $DNSGA_{II}$ (b) respectively. Figure 4 compares the obtained POF of Borda method with $DNSGA_{II}$ (b) where horizontal and vertical axes are the cost of heating energy and deviation from ideal temperature respectively. It is clear that minimum deviation from ideal value can be obtained by using maximum energy. Comparing the obtained POF and considering the average deviation declares that the proposed method achieves quite satisfactory results in comparison with DNSGA.

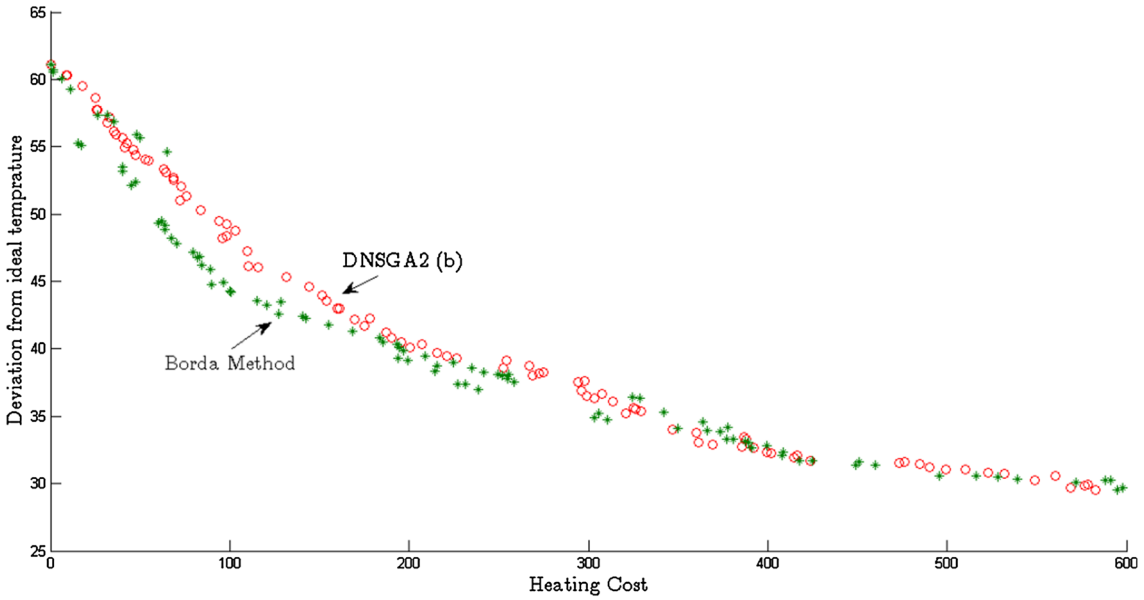

Fig. 4 Simulation of heating optimization: obtained Pareto front by Borda method VS $DNSGA_{II}$ (b)

4.5 Regulation of a lake–river system

The main goal of a regulating a lake–river system is to control the rate of inflow and outflow of water [31]. The main dynamic component is the water level, x_i that is driven by the difference between the inflow and outflow. Also this variable depends on the surface area of the lake, A(x), the previous water level and time interval. The area of the lake is approximated by a piecewise linear function. The modelling of the change in the water level shown in Fig. 5, given the inflow and outflow difference, takes place in two phases, as the surface area of the lake depends on the water level. First, an initial change using the current area of the lake as follows:

$$\begin{aligned} \Delta\tilde{x}_i &= \frac{\Delta q_i \Delta t_i}{A(\tilde{x}_{i-1})} \\ \tilde{x}_i &= \tilde{x}_{i-1} + \Delta\tilde{x}_i. \end{aligned} \tag{23}$$

Actual change is obtained by the following equation:

$$\begin{aligned} \Delta x_i &= \frac{\Delta q_i \Delta t_i}{A\left(\frac{x_{i-1}+\tilde{x}_i}{2}\right)} \\ x_i &= x_{i-1} + \Delta x_i, \end{aligned} \tag{24}$$

where k is the set of goal observation indexes of the planning period; c_1 and c_2 are adjustable parameters; g_k is the deviation from the goal point; x_k^{goal} is the goal; x_k is the true water level; I_k is the deviation from the goal set; the lower and upper bounds of the goal x_k, is indicated as l_k and u_k respectively; p_i is a penalty function; q_i is the outflow from lake; q_{max} is the upper limit of the change in flow rate; and i refers to the discretized time interval.

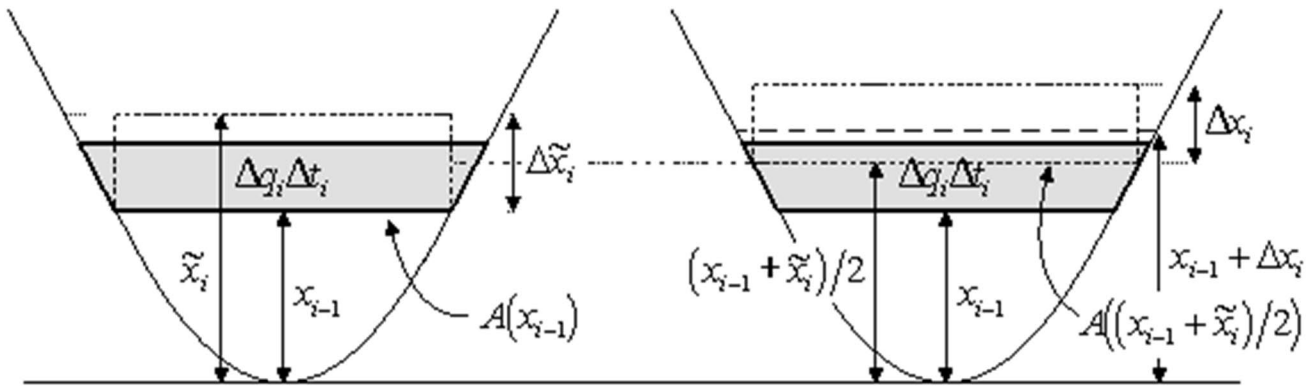

Fig. 5 Model of change in the water level [31]

$$\min f = (f_1, f_2)$$

$$f_1 = \sum_{k\in K} c_g g_k + \sum_{i=1}^{n} p_i, f_2 = \sum_{k\in K} c_I I_k + \sum_{i=1}^{n} p_i$$

$$g_k = (x_k^{goal} - g_k)^2,\ I_k = \begin{cases} (l_k - x_k)^2 & \text{if} \quad x_k < l_k \\ (x_k - u_k)^2 & \text{if} \quad x_k > u_k \\ 0 \quad \text{Otherwise} \end{cases} \tag{25}$$

$$p_i = \begin{cases} c_1 \times \left(|q_i - q_{i-1}| - \Delta q_{max}\right)^2 \\ +c_2 \times (q_i - q_{i-1})^2, \text{if} \quad |q_i - q_{i-1}| > \Delta q_{max} \\ c_2 \times (q_i - q_{i-1})^2, \ \text{Otherwise} \end{cases}$$

4.5.1 Simulation results of the lake–river system

The main goal in a lake–river system is to control the flow of water to obtain to desired water level. In this study both inflow and outflow of water are investigated as decision maker variables. So the optimization algorithm should find optimum values for 48 variables: 24 for inflow and 24 for outflow (one variable for an hour of a day). Here inflow and outflow are bounded between [10, 30] and desired water level is between [2, 3] meter. Also in order to calculate the area, an approximated linear function is used as follows: (water level $\times 12 + 200$) L/m^2. Also $c_g = 10$ L/m^2, $c_I = 100$ L/m^2, $c_1 = 100$ s^2/m^6, and $c_2 = 0.00001$ s^2/m^6, $\Delta q_{max} = 300$, $l_k = 2.1$, and

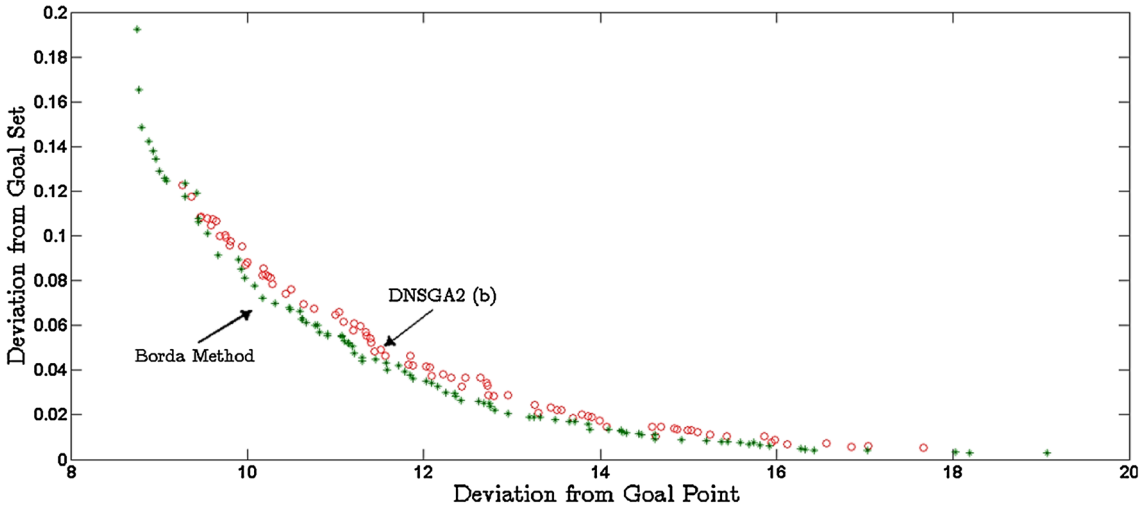

Fig. 6 Simulation of lake–river system: obtained Pareto front by Borda method vs DNSGA$_{II}$ (b)

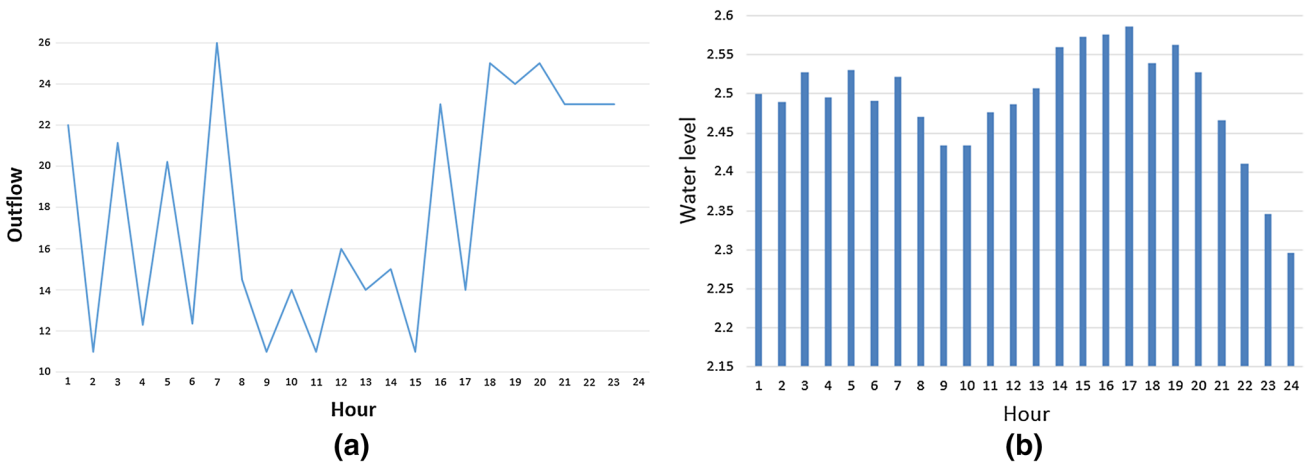

Fig. 7 **a** Outflow during a day, **b** water level during a day

$u_k = 2.9$. Figure 6 shows the obtained Pareto front by Borda method and DNSGA2 (b) where the horizontal axis is deviation from the goal point and vertical axis is deviation from goal set. The found extreme solutions is (8.375, 0.1921) and (19.07, 0.0029) for Borda method while (9.26, 0.1226) and (17.66, 0.0054) is obtained by DNSGA2 (b). Also the spacing measure of the obtained POF by the Borda method is 0.001125 while 0.002244 is for DNSGA2 (b). So the comparison clears that the better POF is found by Borda method. Also, the optimal water outflow and optimal water level of lake during 24 h of a day are shown in Fig. 7a, b respectively. For example, the figure shows that the outflow of Lake should be 10 in order to reach to water level of 2.5 meters at 3 PM.

5 Conclusion

This paper proposed a dynamic multi-objective optimization algorithm based on Borda count ranking method and cat swarm algorithm. Experimental results indicate that the proposed algorithm gets the best rank for measures of MS, HVR, and ACC in comparison with traditional approaches. Moreover, the results investigate the optimal rate of re-initialization. It is concluded that the proposed algorithm achieves the best performance by re-initializing 15 percent of population. Also, the comparison between CSO with pure and some modified versions of PSO shows the quite satisfactory results of CSO. Moreover, the effect of MR on the performance of the algorithm was investigated, and the results revealed that mixture ratio of 50% (using both local and global search simultaneously) is the better candidate to get the best performance. Finally, two real-world dynamic multi-objective problems are solved by the proposed approach and compared by $DNSGA_{II}$.

References

1. Avdagic Z, Konijicija K, Omanovic S (2009) Evolutionary approach to solving nonstationary dynamic multi-objective problems. Found Comput Intell 3:267–289
2. Camara M, Ortega J, Toto F (2009) Single front genetic algorithm for parallel multi-objective optimization in dynamic environments. Neurocomputing 72:3570–3579
3. Camara M, Ortega J, Toto J (2007) Parallel processing for multi-objective optimization in dynamic environments. IEEE international parallel and distributed processing symposium
4. Chu S, Tsai P, Pan J (2006) Cat swarm optimization. Lecture note in artificial intelligence, 4099. Springer, Berlin, pp 854–858
5. Coello CA, Lamont GB, Veldhuisen DA (2007) Evolutionary algorithms for solving multi-objective problems, 2nd edn. Springer, New York
6. Dang Y, Wang C (2008) An evolutionary algorithm for dynamic multi-objective optimization. Appl Math Comput 25:6–18
7. Deb K (2002) Multiobjective optimization using evolutionary algorithms. Wiley, Oxford
8. Deb K, Rao N, Karthik S (2007). Dynamic mylti-objective optimization and decision making using modified NSGA2: a case study on hydro thermal power scheduling. LNCS, pp 803–817
9 Engelbrecht A (2010) Heterogeneous particle swarm optimization. In: Proceeding of the 7th international conference on swarm intelligence, pp 191–202
10 Farina M (2001) A minimal cost hybrid strategy for pareto optimal front approximation. Evol Optim 3(1):41–52
11 Farina M, Deb K, Amato P (2004) Dynamic multiobjective optimization problems: test cases, approximations, and applications. IEEE Trans Evolut Comput 8:425–442
12 Goh C, Tan K (2007) An investigation on noisy environments in evolutionary multiobjective optimization. IEEE Trans Evolut Comput 11:354–381
13 Goh C, Tan K (2009) A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. IEEE Trans Evolut Comput 13:103–127
14 Hatzakis IW (2006) Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. In: Proceedings of the 8th annual conference on genetic and evolutionary computation, Washington, pp 1201–1208
15 Helbig M (2012) Solving dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation. University of Pretoria, Pretoria
16 Helbig M, Engelbrecht A (2011) Archive management for dynamic multi-objective optimisation problems using vector evaluated particle swarm optimization. In: IEEE congress of evolutionary computation (CEC), New Orleans, pp 2047–2054
17 Helbig M, Engelbrecht A (2012) Analyses of guide update approaches for vector evaluated particle swarm optimisation on dynamic multi-objective optimisation problems. WCCI 2012 IEEE world congress on computational intelligence, Australia
18 Helbig M, Engelbrecht A (2013) Dynamic multi-objective optimization using PSO. In: Alba E (ed) Methaheuristic for dynamic optimization. Springer, Berlin, pp 147–188
19 Helbig M, Engelbrecht A (2013) Performance measures for dynamic multi-objective optimisation algorithms. Inform Sci 250:61–81
20 Helbig M, Engelbrecht A (2014) Heterogeneous dynamic vector evaluated particle swarm optimisation for dynamic multi-objective optimisation. IEEE congress on evolutionary computation (CEC), China
21 Helbig M, Engelbrecht A (2015) Using headless chicken crossover for local guide selection when solving dynamic multi-objective optimization. In: Proceedings of the 7th world congress on nature and biologically inspired computing (NaBIC2015) in Pietermaritzburg, South Africa, pp 381–392
22 Helbig M, Deb K, Engelbrecht A (2016) Key challenges and future directions of dynamic multi-objective optimization. IEEE world congress on computational intelligence, Canada
23 Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, pp 1942–1948
24 Koo W, Goh C, Tan K (2010) A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment. Memetic Comp:87–110
25 Li X, Branke J, Blackwel T (2006) Particle swarm with speciation and adaptation in a dynamic environment. In: Proceedings of 8th conference on genetic and evolutionary computation (GECCO 2006), pp 51–58
26 Liang J, Qin A, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE trans evol comput 10(3):281–295

27 Liu M, Liu Y (2016) A dynamic evolutionary multi-objective optimization algorithm based on decomposition and adaptive diversity introduction. 12th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD)

28 Liu R, Chen Y, Ma W, Mu C, Jiao L (2013) A novel cooperative coevolutionary dynamic multi-objective optimization algorithm using a new predictive model. Soft Comput 18(10):1913–1929

29 Liu R, Fan J, Jiao L (2015) Integration of improved predictive model and adaptive differential evolution based dynamic multi-objective evolutionary optimization algorithm. Appl Intell 43:192–207

30 Liu R, Nui X, Fan J, Mu C, Jiao L (2014) An orthogonal predictive model-based dynamic multi-objective optimization algorithm. Soft Comput 19:3083–3107

31 Mantysaari J, Hamalainen R (2001) A dynamic interval goal programming approach to the regulation of a lake-river system. J Multi-Criteria Decis Anal 10(2):75–86

32 Mantysaari J, Hamalainen R (2002) Dynamic multi-objective heating optimization. Eur J Oper Res 142:1–15

33 Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler, maybe better. IEEE Trans Evolut Comput 8:204–210

34 Muruganantham A, Zhao Y, Gee S, Qiu X (2013) Dynamic multiobjective optimization using evolutionary algorithm with Kalman Filter. 17th Asia Pacific symposium on intelligent and evolutionary systems, IES2013

35 Qiao J, Zhang W (2016) Dynamic multi-objective optimization control for wastewater treatment process. J Neural Comput Appl. doi:10.1007/s00521-016-2642-8

36 Saari D (1985) The optimal ranking method is the Borda Count. International institute for applied systems analysis, IIASA collaborative paper

37 Shang R, Jiao L, Ren Y, Li L, Wang L (2014) Quantum immune clonal coevolutionary algorithm for dynamic multiobjective optimization. Soft Comput 18:743–756

38 Tizhoosh H (2005) Opposition-based learning: a new scheme for machine intelligence. CIMCA/IAWTIC

39 Wu Y, Jin Y, Liu X (2014) A directed search strategy for evolutionary dynamic multiobjective optimization. Soft Comput 19(11):3221–3235

40 Xu B, Zhang Y, Gong D, Rong M (2016) Cooperative co-evolutionary algorithm for dynamic multi-objective optimization based on environmental variable grouping. 7th international conference, ICSI 2016, Bali, pp 564–570

41 Yang X, Yuan J, Mao H (2007) A modified particle swarm optimizer with dynamic adaptation. Appl Math Comput 189:1205–1213

42 Zheng Y, Ling H, Xue J, Chen S (2014) Population classification in fire evacuation: a multi-objective particle swarm optimization approach. IEEE Trans Evol Comput 18(1):70–81

43 Zheng Y, Song Q, Chen S (2013) Multi-objective fireworks optimization for variable-rate fertilization in oil crop production. Appl Soft Comput 13(11):4253–4263

44 Zhou X, Liu Y, Li B, Sun G (2015) Multi-objective biogeography based optimization algorithm with decomposition for community detection in dynamic networks. Phys A Stat Mech Appl 436:430–442

45 Zhou A, Zhang Q (2014) A population prediction strategy for evolutionary dynamic multiobjective optimization. IEEE Trans Cybern 44(1):40–53

46 Zhou A, Zhang Q, Sendhoff B, Tsang E (2007) Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization. The 4th international conference on evolutionary multi-criterion optimization. Springer, Berlin

47 Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and applications. Swiss Federal Institute of Technology (ETH), Zurich