Received: September 8, 2021. Revised: October 20, 2021.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Adaptive Lévy Flower Pollination Algorithm for Multiple Objective Optimization

Gemilang Santiyuda[1] Retantyo Wardoyo[1]* Afiahayati[1]

[1]*Department of Computer Science and Electronics, Faculty of Mathematics and Natural Sciences, Universitas Gadjah Mada, Indonesia*
* Corresponding author's Email: rw@ ugm.ac.id

Abstract: The adaptive Lévy flower pollination algorithm (ALFPA) is a recent addition to variants of flower pollination algorithm (FPA). Despite its excellent performance on single objective problem instances, it has shown inefficiency in the number of function evaluation (FE). Inspired by this, this paper proposed two algorithms extending ALFPA to solve multiple objective problem while also improving FE efficiency. The first algorithm proposed is ALFPA with non-dominated sorting denoted as MO-ALFPA alongside two variants which are proposed to improve its FE efficiency denoted as MO-ALFPAT and MO-ALFPAB. The second proposed algorithm, MOEA/D-ALFPA, uses decomposition strategy instead of non-dominated sorting on MO-ALFPA. The empirical study on two benchmark suits shows that MO-ALFPAT and MOEA/D-ALFPA performed better than other methods. Furthermore, MO-ALFPAT and MOEA/D-ALFPA produced the best results in three benchmark instances, based on inverted generational distance indicator, and two and three best results based on the hypervolume indicator, respectively.

Keywords: Adaptive Lévy mutation, Adaptive operator selection, Flower pollination algorithm, Multiple objective problem.

1. Introduction

Many design optimization problems in various fields are formulated into multiple objective problem (MOP) to address the existence of multiple and often conflicting objectives in real-world situation. For instances in engineering design [1], logistic [2, 3], economics [4, 5], and bioinformatics [6].

Multiple objective problem (MOP) can be formally described as follows. There are K objective functions to consider simultaneously in a MOP. The objectives are formulated as $F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \dots, f_K(\vec{x}))$ with $\vec{x} \in \Omega$ and Ω is the set of all feasible solutions. The formulation of MOP is given in Eq. (1).

$$\begin{gathered} \text{minimize } F(\vec{x}) \\ \text{subject to} \\ g_j(\vec{x}) \leq 0, for\ 1 \leq j \leq H_1 \\ h_l(\vec{x}) = 0, for\ 1 \leq l \leq H_2 \end{gathered} \tag{1}$$

There are H_1 inequality constraints and H_2 equality constraints. The optimal solutions of MOP can be defined in terms of Pareto optimality [7].

A solution $\vec{x}^*$ dominates another solution $\vec{x}$ ($\vec{x}^* \prec \vec{x}$), if and only if $f_i(\vec{x}^*) \leq f_i(\vec{x})$ for every index i, and $f_j(\vec{x}^*) < f_j(\vec{x})$ for at least one index j with $1 \leq i, j \leq K$. A solution $\vec{x}^*$ is nondominated regarding to a set $X' \subseteq X$ if and only if $\nexists \vec{x} \in X' : \vec{x} \prec \vec{x}^*$. A set $X' \subseteq \Omega$ is called Pareto optimal set if and only if there is not any solution $\forall \vec{x}' \in X' : \nexists x \in \Omega : \vec{x} \prec \vec{x}'$. The corresponding set of objective values from a Pareto optimal set is called Pareto optimal front (POF) [8]. The main task of solving MOP is to find a good set of solutions in terms of quality and spread that can best represent the POF.

A significant number of multiple objective evolutionary algorithms (MOEA) have been developed. A common challenge faced by MOEAs is balancing the rate of exploration and exploitation,

which determines the quality and the spread of the resulted POF [9].

Several methods are proposed to address this challenge. Firstly, one can specify the search neighborhood in each iteration by selecting potential parent solutions, e.g., by tournament selection based on dominance and crowdedness of the solutions in nondominated sorted genetic algorithm II (NSGA-II) [10] or by selecting a potential subproblem in decomposition-based MOEA (MOEA/D)[11] as done in MOEA/D-DRA [12]. The neighborhood size also affects the exploitation and exploration capability as empirically shown in [13] by varying the tournament size of NSGA-II, or by varying the neighborhood size in MOEA/D as shown in [14] and improving the performance by using an ensemble of neighborhood sizes as shown in ENS-MOEA/D [15].

Other studies have also been conducted on recombination and mutation operators to improve the performance of MOEA. In [16], a more general equation for DE operator is proposed, which can be derived into up to 57 DE operators. An SOP benchmarking study showed that each operator performs differently in different problem characteristics. Eight best performing operators at the end are selected based on the benchmarking study. A similar development can be observed in several other studies, e.g. considering Gaussian and Cauchy random walk in Cuckoo Search (CS) [17] and flower pollination algorithm (FPA) [18], using DE operators [19] or CS operators [20] in MOEA/D.

The issue of varying operators is choosing the appropriate operator to use in a particular situation. Although using multiple operators at once and choosing the best result could perform well, as empirically shown in [18], this strategy is inefficient when the function evaluation (FE) is limited. The most common strategy is to use a controlling parameter of operator utilization. It is shown that the performance of MOEAs depends on the parameterization and the problem characteristic [20, 21]. Therefore, parameters of MOEAs must be tuned for each specific problem to get the best result. However, for some problems, especially a dynamic problem, parameter tuning can be expensive and time-consuming [21]. This leads to the problem of finding algorithms that are robust to problem formulations so that more efforts can be made on problem modelling and analysis instead of parameter tuning or algorithm development [23].

Subsequently, several parameter selection methods are proposed for AOS. The most common is probability matching (PM) [24], with probability based on the normalized credits. However, an improvement of PM called adaptive pursuit (AP) [25] is proposed to help PM adapt faster to change of operator preference. Other deterministic alternatives to adapt to fast rewards dynamic are by using bandit based operator selector [26]. A recent AOS strategy based on the current solution status is called bicriteria assisted AOS (BAOS) [27]. By using the current solution state instead of the operator performance, BAOS does not need credit assignment, and therefore straightforward to be adopted into any MOEAs.

Inspired by the above findings, this paper proposed to extend adaptive Lévy flower pollination algorithm (ALFPA) [18] to solve MOP. Based on the experimental study conducted, ALFPA showed superior performance on benchmark SOPs than several well-known algorithms. The reason for the better performance of ALFPA can be attributed to its three significant additions to the original FPA.

There are three significant additions to the flower pollination algorithm (FPA) [28] proposed in ALFPA. Firstly, ALFPA enhances the exploration by using four recombination operators in the global random walk with step sizes drawn from symmetrical stable Lévy distribution with four different shapes. Secondly, the number of offspring generated by local random walk is increased to four to enhance the exploitation, thus balancing the exploration. Lastly, a dynamic strategy is applied to adjust the switching probability p to balance exploration and exploitation. Despite being well-performing, a clear drawback of ALFPA is the inefficiency of the number of FE due to the number of candidate offspring generated. This drawback becomes prominent when the FE is expensive or limited.

In this paper, two variants of multiple objective extensions of ALFPA are proposed. The first variant, denoted as MO-ALFPA, is a Pareto dominance based extension of ALFPA by incorporating survival selection mechanism employed NSGA-II [10] and replacing the NSGA-II crowdedness measure by harmonic average distance (HAD)[29]. Furthermore, two variants of MO-ALFPA other than the straightforward extension are proposed to solve the FE inefficiency of ALFPA. One variant employs a tournament parent selection mechanism similar to NSGA-II but with HAD as the crowdedness measure, denoted as MO-ALFPAT. The other variant, MO-ALFPAB, with bicriteria adaptive operator selection (BAOS) [27] to choose the appropriate operator based on the current status of the solution. Both variants improve the performance of MO-ALFPA, especially MO-ALFPAT.

An ALFPA extension as a variant of MOEA/D is proposed by incorporating the adaptive Lévy mutation operators and BAOS into MOEA/D,

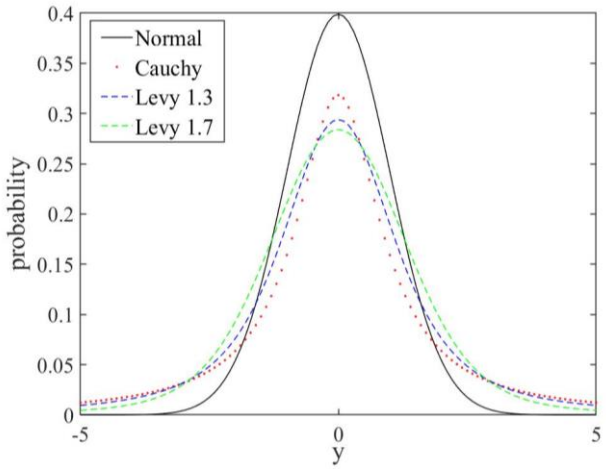

Figure. 1 Comparison of the lévy probability distributions with $\lambda = 1.3$ and $\lambda = 1.7$, Cauchy distribution and Gaussian distribution. The value of $\gamma = 1$ for all λ

denoted as MOEA/D-ALFPA. The experimental study showed that MOEA/D-ALFPA and MO-ALFPAT performed better than the compared methods.

Lastly, this paper is organized as follows. A brief explanation of ALFPA is given in Chapter 2. The proposed multiple objective extension of ALFPA, namely MO-ALFPA, MO-ALFPAT, MO-ALFPAB and MOEA/D-ALFPA are given in Chapter 3. The details of the experimental study are given in Chapter 4. The results of the experimental study are discussed in Chapter 5. Finally, the paper is concluded in Chapter 6.

2. Adaptive lévy flower pollination algorithm

Adaptive Lévy FPA (ALFPA) is a recent addition to the variants of FPA [28] proposed by Salgotra [18]. The ALFPA algorithm follows the same framework of FPA in which the algorithm starts with a population P of n randomly initialized solutions. The population is then updated for several generations until the terminal condition is met. In every generation, every solution has a chance to be updated. Based on the switching probability p, each solution is used to generate a candidate solution via global pollination or via local pollination update scheme. The global pollination update scheme generates a candidate solution which is the current solution shifted towards the current best solution in the population with step sizes drawn from Lévy distribution as given in Eq. (2). On the other hand, local pollination generates a candidate solution from the current population shifted by a directional vector

Algorithm 1: ALFPA algorithm

Initialize population P of $n = \frac{N}{4}$ solutions
Evaluate and set $\vec{g}_* = \text{argmin}_{\vec{x} \in P} f(\vec{x})$
Set initial switch probability $p \in [0, 1]$
while $t < t_{max}$
  Compute current p by Eq. (11)
  for $i \coloneqq 1 : n$
    if $rand < p$
      Generate $\vec{x}_{i,j}^{t+1}, \forall\, j \in$ [1,4] with Eq. (3-6)
    else
      Generate $\vec{x}_{i,j}^{t+1}, \forall\, j \in$ [1,4] with Eq. (3-6)
    end if
    Evaluate new solutions
    $\vec{x}_i^{t+1} =$
$\arg\min\{f(\vec{x}_{i,1}^{t+1}), f(\vec{x}_{i,2}^{t+1}), f(\vec{x}_{i,3}^{t+1}), f(\vec{x}_{i,4}^{t+1})\}$
  end for
  Update current best solution as $\vec{g}_*$
  $t{+}{+}$
end while
Return $\vec{g}_*$

made of two random solutions in the current population. If the candidate solution improves the current solution, then the candidate solution will replace the current solution in the next generation. Finally, the best solution found is returned after the algorithm is terminated.

In ALFPA, Salgotra [18] proposed using a candidate pool of four different distributions to generate new solutions in the global pollination instead of using the standard Lévy distribution only. This strategy is adopted from adaptive Lévy mutation strategy first proposed by Yao and Lee in 2001[30]. In ALFPA, there are four offspring generated for each solution in a generation. Each offspring is generated by probability density function which is derived from probability density function of symmetrical stable Lévy distribution with different value of λ.

$$f_{\text{Lévy}}(y; \lambda, \gamma) = \frac{1}{\pi} \int_0^\infty e^{-\gamma\, q^\lambda} \cos(qy) dq \qquad (2)$$

The distribution has two parameters, scaling factor γ and λ which controls the shape of the probability distribution especially in the tail region as can be seen in Fig. 1. The scaling factor γ can be set to 1 without the loss of generality[30]. Therefore, the probability distribution with a fixed $\gamma = 1$ will be denoted as $f_{Lévy}(y; \lambda, 1) = f_{Lévy}(y; \lambda)$.

For $\lambda = 1$, the distribution reduces to Cauchy distribution, and the distribution reduces to Gaussian

distribution for $\lambda = 2$. As shown in Fig. 1, Cauchy distribution has a very heavy tail indicating that that long step size has a high probability to be drawn, meanwhile Gaussian distribution gives higher probability to small step sizes. The other two distributions are Lévy distributions with $\lambda = 1.3$ and $\lambda = 1.7$. The Lévy distributions are employed to balance the extreme between Cauchy and Gaussian distributions. The four new solutions generated via global pollination are given as:

$$\vec{x}_{i,1}^{t+1} = \vec{x}_i^t + \alpha\, C(\lambda)(\vec{g}_* - \vec{x}_i^t) \quad (3)$$

$$\vec{x}_{i,2}^{t+1} = \vec{x}_i^t + \alpha\, L_1(\lambda)(\vec{g}_* - \vec{x}_i^t) \quad (4)$$

$$\vec{x}_{i,3}^{t+1} = \vec{x}_i^t + \alpha\, L_2(\lambda)(\vec{g}_* - \vec{x}_i^t) \quad (5)$$

$$\vec{x}_{i,4}^{t+1} = \vec{x}_i^t + \alpha\, G(\lambda)(\vec{g}_* - \vec{x}_i^t) \quad (6)$$

With α is the learning rate, g_* is the current best solution, $G(\lambda), C(\lambda), L_1(\lambda)$ and $L_2(\lambda)$ corresponds to random step size drawn from Gaussian distribution, Cauchy distribution, Lévy with $\lambda = 1.3$, and Lévy with $\lambda = 1.7$ respectively. Four new solutions are also generated in the local pollination to even out the number of search equations in the global pollination.

$$\vec{x}_{i,1}^{t+1} = \vec{x}_i^t + \epsilon\,(\lambda)(\vec{x}_{j_1}^t - \vec{x}_{j_2}^t) \quad (7)$$

$$\vec{x}_{i,2}^{t+1} = \vec{x}_i^t + \epsilon\,(\lambda)(\vec{x}_{j_3}^t - \vec{x}_{j_4}^t) \quad (8)$$

$$\vec{x}_{i,3}^{t+1} = \vec{x}_i^t + \epsilon\,(\lambda)(\vec{x}_{j_5}^t - \vec{x}_{j_6}^t) \quad (9)$$

$$\vec{x}_{i,4}^{t+1} = \vec{x}_i^t + \epsilon\,(\lambda)(\vec{x}_{j_7}^t - \vec{x}_{j_8}^t) \quad (10)$$

An obvious drawback from this strategy is the number of objective function evaluations (FE) which is four times the number of FE done in the standard FPA. If the same population size is employed, then ALFPA will either take longer computational time or terminate four times earlier than the FPA. Therefore, the population size employed is be reduced to $\frac{N}{4}$ so that the number of FE will be the same. The limit on number of FE then will also directly limit the population size which directly affects the performance of the algorithm as it has been shown that larger *N* shown better performance as shown in the experiments conducted [18].

Besides employing adaptive Lévy mutation in the global pollination, Salgotra [18] also proposed to use dynamic value of switching probability p instead of using a predetermined constant value. The general equation for the dynamic switch probability is given in Eq.(11).

$$p = p_0 - \frac{t_{max} - t}{t_{max}} \quad (11)$$

Here t_{max} is the maximum generation until termination, and t is the current generation. The initial value of the switching probability, p_0 is set as 0.8. The pseudocode for ALFPA is given in Algorithm 1.

3. Multiple objective ALFPA

3.1 MO-ALFPA

The first proposed method, MO-ALFPA, is a straightforward extension of ALFPA to solve MOP by incorporating non-dominated as in NSGA-II as the survival selection mechanism. However, the crowding distance calculation in NSGA-II is replaced by archive truncation method based on HAD[29]. In addition, because there can be multiple optimal solutions in the population P_t of the current generation t, therefore the best solution g_* in Eq. (3-6) is replaced by a random nondominated solutions regarding to P_t. The generated offspring from population P_t will be stored in the offspring set O_t. Afterwards, the solutions for the next generation P_{t+1} will be selected from $P_t \cup O_t$.

The survival selection is done by first partitioning the set of solutions into several fronts of nondominated solutions A_i with A_1 is the non-dominated solutions of $P_t \cup O_t$,and A_{i+1} is the non-dominated solutions of $(P_t \cup O_t) \setminus \left(\cup_{j=1}^{i} A_j\right)$. After the solutions are partitioned into fronts, the solutions are inserted into P_{t+1} starting from A_1 until $|P_{t+1}| = N$. Suppose that the last front to be inserted into P_{t+1} is A_m with $m \geq 1$. If $|A_m \cup P_{t+1}| >$ N, then the solutions in A_m will be truncated starting with the one with the lowest HAD will be removed from A_m until $|A_m \cup P_{t+1}| \leq N$ as in Algorithm 2.

The calculation of HAD is given in Eq. (12) with $B(\vec{x})$ is set of k-nearest solutions of $\vec{x}$ in the objective space and $d(\vec{x}, \vec{y})$ is the Euclidean distance between solution $\vec{x}$ and $\vec{y}$ in the objective space. Note that $\vec{x} \notin B(\vec{x})$. Similar to crowding distance in NSGA-II, the solution with lower value of HAD is more crowded than the solution with a higher value of HAD.

Algorithm 2: Archive truncation algorithm

Given A_m
while $\left|\cup_{j=1}^{m} A_j\right| > N$
  $\vec{x}' = \arg\min_{\forall \vec{x} \in A_m} \text{HAD}(\vec{x})$
  $A_m = A_m \setminus \vec{x}'$
end while

Algorithm 3: MO-ALFPA algorithm

Initialize population P of N solutions
Evaluate initial solutions
Set initial switch probability $p_0 \in [0,1]$
while $t < t_{max}$
  Update p by Eq. (11)
  $O_t = \{\}$
  $Q_t = TournamentSelection(P_t)$
  for $x_i^t \in Q_t$
    if $rand < p$
      Get a random non-dominated solution $\vec{g}_* \in P_t$
      Generate $\vec{x}_{i,j}^{t+1}, \forall j \in [1,4]$ with Eq. (3)-(6)
      $\text{PolynomialMutation}\left(x_{i,j}^{t+1}\right), \forall j \in [1,4]$
    else
      Generate $x_{i,j}^{t+1}, \forall j \in [1,4]$ with Eq. (3)-(6)
    end if
    Evaluate new solutions
    Insert new solutions to O_t
  end for
  $P_{t+1} = SurvivalSelection(P_t \cup O_t)$
  $t = t + 1$
end while
Return P_t

$$\text{HAD}(\vec{x}) = \frac{|B(\vec{x})|}{\sum_{\vec{y} \in B(\vec{x})} \frac{1}{d(\vec{x},\vec{y})}} \tag{12}$$

Lin *et al.* [27] argued that HAD can reflect the crowdedness of each solution in the local search space. In addition, it is also shown in [29] that crowding distance in NSGA-II is sensitive to outlier and therefore may not accurately reflect the actual crowdedness of solutions in the population. On the other hand, the influence of outlier existence is overcome by using HAD.

Besides that, two variants of MO-ALFPA are proposed to handle the FE inefficiency of ALFPA. The first variant, MO-ALFPAT, only chooses $\frac{N}{4}$ parent solutions from P_t to be explored in every generation. In this variant, the parent is chosen by binary tournament selection based on Pareto crowded comparison as in NSGA-II with HAD replacing the original crowding distance. The crowdedness comparison operator ($\prec_n$) is given as follows. Let $\vec{x}_1 \in A_i$ and $\vec{x}_2 \in A_j$ with $A_i, A_j \subseteq P_t, \vec{x}_1 \prec_n \vec{x}_2$ if and only if $i < j$ or $i = j$ and $HAD(x_1) > HAD(x_2)$. The pseudocode of MO-ALFPAT is given in Algorithm 4.

Algorithm 4: MO-ALFPAT algorithm

Initialize population P of $n = \frac{N}{4}$ solutions
Evaluate initial solutions
Set initial switch probability $p_0 \in [0,1]$
while $t < t_{max}$
  Update p by Eq. (11)
  $O_t = \{\}$
  for $i = 1$ to n
    if $rand < p$
      Get a random non-dominated solution $\vec{g}_* \in P_t$
      Generate $\vec{x}_{i,j}^{t+1}, \forall j \in [1,4]$ with Eq. (3)-(6)
      $\text{PolynomialMutation}\left(\vec{x}_{i,j}^{t+1}\right), \forall j \in [1,4]$
    else
      Generate $\vec{x}_{i,j}^{t+1}, \forall j \in [1,4]$ with Eq. (3)-(6)
    end if
    Evaluate new solutions
    Insert new solutions to O_t
  end for
  $P_{t+1} = SurvivalSelection(P_t \cup O_t)$
  $t = t + 1$
end while
Return P_t

The second variant, MO-ALFPAB, select an appropriate operator to explore a solution instead of selecting the appropriate solution to explore as in MO-ALFPA. An adaptive operator selection called bi-criteria adaptive operator selection (BAOS) proposed by Lin *et al.* [27] is used in MO-ALFPAB.

The two criteria in selecting the appropriate operator are whether the solution is a non-dominated solution and the solution's crowdedness compared to another randomly selected solution in the population. HAD[29] is used as the crowdedness measure as given in Eq.(12).

To use BAOS, the adaptive Lévy mutation operators are arranged into two operator pools AL^1 and AL^2. Each operator pool has two operators. The first operator pool AL^1 is selected when the solution is a non-dominated solution, otherwise AL^2 is selected. If x is a non-dominated solution, then optimal solutions near x should be found by exploitation. Otherwise, if x is a dominated solution, then more exploration is needed to find non-dominated solutions in unexplored area.

Algorithm 5: BAOS algorithm

Compute $B(x)$
Select a solution $\vec{y} \in P_t$ randomly
if $\vec{x}$ is non-dominated in regard to P_t
  $Pool = AL^1$
else
  $Pool = AL^2$
end if
if $HAD(\vec{x}) > HAD(\vec{y})$
  Return operator $Pool_1$
else if $HAD(\vec{x}) > HAD(\vec{y})$
  Return operator $Pool_2$
else
  Return random operator from $Pool$
end if

Algorithm 6: MO-ALFPAB algorithm

Initialize population P of N solutions
Evaluate initial solutions
while $t < t_{max}$
  $O_t = \{\}$
  for $i = 1$ **to** N
    Operator $op = BAOS(\vec{x}_i^t)$
    Get a random non-dominated solution $\vec{g}_* \in P_t$
    Generate $\vec{x}_i^{t+1}$ by op
    PolynomialMutation$(\vec{x}_i^{t+1})$
    Evaluate $\vec{x}_i^{t+1}$
    Insert $\vec{x}_i^{t+1}$ to O_t
  end for
  $P_{t+1} = \text{SurvivalSelection}(P_t \cup O_t)$
  $t = t + 1$
end while

Based on this, Gaussian operator Eq. (6)is in AL^1 because it has a higher probability to search with small step sizes as shown in Fig. 1 and has been shown to perform well for local search. On the other hand, the Cauchy operator Eq. (3)is well suited to search at large area of search space and therefore will be included in AL^2. Compared to L_2 operator Eq. (5) L_1 operator Eq. (4) has a heavier tail as shown in Fig. 1, and therefore L_2 operator is included in AL_1 and L_1 operator is in AL^2.

$$AL^1 = \{\text{Gaussian operator}, L_2 \text{ operator}\} \quad (13)$$

$$AL^2 = \{L_1\text{operator}, \text{Cauchy operator}\} \quad (14)$$

Afterwards, an operator will be picked from the chosen operator pool based on the crowdedness of solution x. The pseudocode of the BAOS is given in Algorithm 5.

Algorithm 7: MOEA/D-ALFPA algorithm

Initialize weight vectors $W = \{\vec{w}^1, \dots, \vec{w}^N\}$
Initialize population P of N solutions
Initialize neighbourhood $B(\vec{x}_i), \forall\, \vec{x}_i \in P$
Evaluate initial solutions
Initialize $\vec{z}, z_m = \min_{\forall\, \vec{x}_i \in P} f_m(\vec{x}_i) \;\; \forall\, m \in \{1, \dots, K\}$
while $t < t_{max}$
  for $i = 1$ **to** n
    $E = P$
    if $rand < p$
      $E = B(\vec{x}_i)$
    end if
    Operator $op = BAOS(\vec{x}_i)$
    Get a random solution $\vec{g}_* \in E$
    Generate $\vec{x}'$ by op
    PolynomialMutation$(\vec{x}')$
    Evaluate $\vec{x}'$
    $z_m = \min\{f_m(\vec{x}'), z_m\}, \forall m \in \{1, \dots, K\}$
    $c_r = 0$
    while $c_r < n_r$ **and** E is not empty
      Select solution $\vec{x}_j \in E$
      Calculate $g^{te}(\vec{x}_j|\vec{w}^j, \vec{z})$ and $g^{\text{te}}(\vec{x}'|\vec{w}^j, \vec{z})$
      if $g^{te}(\vec{x}'|\vec{w}^j, \vec{z}) < g^{te}(\vec{x}_j|\vec{w}^j, \vec{z})$
        Replace $\vec{x}_j = \vec{x}'$ in the population P
        $c_r = c_r + 1$
      end if
      $E = E \setminus \vec{x}_j$
    end while
  end for
  $t = t + 1$
end while
Return P

By using BAOS, each solution in the population P_t will be explored and will generate one offspring with the appropriate operator based on its Pareto dominance and crowdedness. Besides that, the exploitation done by local random walk will be replaced by operator pool AL^1, and therefore the switching probability p is not needed anymore. The generated offspring will be perturbed by polynomial mutation before evaluated and collected in O_t until the end of generation and the population for the next generation will be selected by survival selection used in NSGA-II from $P_t \cup O_t$, similar to MO-ALFPA. The pseudocode for MO-ALFPAB is given in Algorithm 6.

Table 1. Properties of the benchmark problems

Problem	Properties
WFG1	separable, deceptive, mixed
WFG2	nonseparable, unimodal, discontinuous
WFG3	nonseparable, unimodal, degenerate
WFG4	separable, multimodal, concave
WFG5	separable, deceptive, concave
WFG6	nonseparable, unimodal, concave
WFG7	separable, unimodal, concave
WFG8	nonseparable, unimodal, concave
WFG9	nonseparable, multimodal, concave
ZDT1	separable, unimodal, convex
ZDT2	separable, unimodal, conceive
ZDT3	separable, multimodal, discontinuous
ZDT4	separable, multimodal, convex
ZDT6	separable, multimodal, convex

3.2 MOEAD-ALFPA

The second proposed method, MOEA/D-ALFPA, differs from the MO-ALFPA in following the well-known multiple objective evolutionary algorithm based on decomposition framework MOEA/D-DE [19]. MOEA/D-DE is a variant of MOEA/D [11] framework which uses DE operators and polynomial mutation.

The bottom line of MOEA/D framework is to solve several single objective subproblems decomposed from the original MOP. In this paper, the scalarization method used is Tchebycheff method[31] as given in Eq. (15). The utopia point, denoted by $\vec{z}^*$, is the best value of all objectives, $z_m^* = \min_{\forall x \in \Omega} f_m(x), \forall m \in \{1, \dots, K\}$. However, the utopia point might not be known beforehand and it is often the case that finding the utopia point will be time consuming. Therefore, an approximation of the utopian point can be used in the calculation by using the current best-known values of all objectives, denoted by $\vec{z}, z_m = \min_{\forall \vec{x} \in P_t} \{z_m, f_m(\vec{x})\}$ with P_t is the population in the current generation t and the initial $z_m = \min_{\forall \vec{x} \in P_1} f_m(\vec{x})$.

$$\text{minimize } \mathrm{g}^{\mathrm{te}}(\vec{x}|\vec{w}, \vec{z}^*) = \max_{1 \le i \le K} w_i \left| f_{i(\vec{x})} - z_i^* \right| \quad (15)$$

An optimal solution for Eq. (15) is also a member of Pareto optimal set, and for each solution x^* in the Pareto optimal set, there is at least one weight vector w so that x^* is the optimal solution for Eq. (15)(15). Therefore, one can solve MOP by solving Eq. (15) with different weight vectors and thus finding different Pareto optimal solutions to approximate POF.

In MOEA/D, each solution in the population is assigned a different weight vector and solve a different subproblem. New solutions are generated each generation by means of crossover and mutation operator. However, the mating pool of each solution $\vec{x}_i$, with associated weight vector $\vec{w}^i$, is limited to a neighbourhood $B(\vec{x}_i)$, with $B(\vec{x}_i)$ is set of k solutions in the current population which weight vectors are the k-nearest weight vectors to $\vec{w}^i$.

This mating restriction is employed as an exploitation-exploration rate. Small k makes the solution only consider similar subproblems, therefore emphasizing exploitation. On the other hand, large k will emphasize more on the exploration. To adopt the mating restriction in the MOEA/D, the adaptive Lévy mutation operators in Eqs. (3-6) are modified so that $\vec{g}_*$ is not a non-dominated solution in regards to P_t anymore, but a random solution selected from the mating pool.

The complete algorithm of MOEA/D-ALFPA is given in Algorithm 7. Firstly, a uniformly distributed set of weight vectors $W = \{\vec{w}^1, \dots, \vec{w}^N\}$ are initialized. The chosen method to generate the weight vectors are Das and Dennis method [32]. Afterwards, initial population is initialized and each solution $\vec{x}_i$ is associated with a neighbourhood $B(\vec{x}_i)$ of size k. In every generation t, every solution $\vec{x}_i^t$ will generate a new offspring $\vec{x}_i^{t+1}$ by an adaptive Lévy mutation operator selected with BAOS explained in subsection MO-ALFPAB. A mating pool E will be selected randomly from $B(x_i^t)$ with probability p or from the whole population P_t with probability $1-p$. A random solution $\vec{g}_*$ will be selected from E as the parent to create a new solution by the selected operator.

Afterwards, the new solution is evaluated and the utopian point estimation is updated $z_m = \min\{f_m(\vec{x}_i^{t+1}), z_m\}$. All solutions' quality, based on their associated subproblem, in the chosen mating pool E are recalculated with the new $\vec{z}$ value and compared to the new solution. Random solution from E will be compared to the new solution one by one, and will be replaced with the new solution if the new solution is more optimal in regard to the associated subproblem. This will continue until at most n_r solutions from E is replaced by the new solution or until all solutions from E have been compared.

4. Experimental study

The first experimental study is conducted to examine the performance of the variants of the first proposed method, MO-ALFPA. The second experimental study is conducted to compare the three proposed methods with several well-known multiple objective evolutionary algorithms. The performance

of the proposed algorithms are assessed based on their respective inverted generational distance (IGD) [33] value and hypervolume indicator (HV) [34] value on benchmark instances.

4.1 Test instances

Two sets of continuous MOPs are used in the conducted experimental studies. The two sets of MOPs are from WFG [35], and ZDT [8]. These problem sets are widely used in experimental studies to compare the performance of MOEAS [20], [23], [36], [37]. The properties of the problem instances provided by Chen *et al.* [20] is given in Table 1.

The number of objectives is two for all of the problems. The numbers of position-related and distance-related decision variables for WFG problems were set to eight and two respectively. The number of decision variables is 30 for ZDT1-ZDT3 and ZDT6, and 10 for ZDT4.

4.2 Experimental settings

For all the employed algorithms, the population size N is set to 100, and the algorithm is terminated if maximum number of FE is done, with $FE_{max} = 25 \times 10^3$. The value of the maximum iteration $t_{\max}$, which is needed for MO-ALFPA's dynamic switching probability, can be calculated by Eq. (16) with FE_{iter} is the number of FE done per generation which equals to the population size in MO-ALFPA.

$$t_{max} = \left\lceil \frac{FE_{max}}{FE_{iter}} \right\rceil \tag{16}$$

Other parameter settings of the employed algorithms are given in Table 2. The parameter settings are set with values recommended by the corresponding literatures. The values of η_c and η_m are the distribution index of SBX and polynomial mutation respectively. The probability of polynomial mutation is set to $p_m = \frac{1}{d}$, with d is the size of decision variables. F and K are both scaling step sizes used in DE operators, while CR is the crossover rate in DE. k denotes the neighbourhood size in MOEA/D variants, and the number of closest solutions to be considered in HAD calculation in MO-ALFPAB. The probability that controls the neighbourhood choice in MOEA/D variants is set to $p = 0.9$. The value of LP is the number of generations to update the selection probability in ENS-MOEA/D, and the possible neighbourhood sizes are 30, 60, 90 and 120. The value of C, W and D are parameters for bandit-based AOS in MOEA/D-FRRMAB and MOEA/D-CS. The value of α and λ are the scaling step size and the parameter for Lévy distribution in MOEA/D-CS and MOFPA. Lastly, the initial switching probability of MO-ALFPA is denoted by p_0.

The experiments are done with PlatEMO[38], a MATLAB-based evolutionary multiple objective platform. Besides MOFPA, MOEA/D-CS and the proposed methods, the other methods are available in PlatEMO.

Each algorithm is run for thirty independent runs, and the mean and standard deviation of IGD and HV are collected. The best average results of IGD and HV are highlighted in boldface. Wilcoxon's rank sum test at a 0.05 significance level is applied to further compare the differences between the compared algorithms.

5. Results and discussion

Several well-known MOEAs are considered in the performance comparison with the proposed algorithms. Seven competitive and relatively new MOEA/D variants, MOEA/D [11], MOEA/D-DE [19], MOEA/D-DRA [12], ENS-MOEA/D [15], MOEA/D-FRRMAB[26], and MOEA/D-CS without the angle based selection [20] are considered alongside NSGA-II [10], and MOFPA [39]. The variant of MOEA/D-CS without angular selection is selected because the average results produced outperform MOEA/D-CS with angular selection in the original paper. All MOEA/D variants considered use Tchebycheff scalarization method given in Eq. (15).

Table 3 shows the average ranks of the compared algorithms across problem instances based on IGD and HV values. For IGD values, MO-ALFPAT has the best average rank followed by MOEA/D-ALFPA and MOEA/D-CS. However, regarding HV values, MOEA/D-ALFPA has the best average rank followed by MOEA/D-CS and MO-ALFPAT. Afterwards, based on the average rank on both IGD and HV values, the next best performing algorithms are a tie between ENS-MOEA/D and MOEA/D-FRRMAB, MO-ALFPAB, MO-ALFPA, NSGA-II, MOFPA, MOEA/D-DE, followed by MOEA/D, and the last is MOEA/D-DRA.

Based solely on the average ranking, the proposed MO-ALFPA variants show a competitive performance for the given problem instances, performing better than the other algorithms including other MOEA/D variants i.e., MOEA/D-DE, MOEA/D and MOEA/D-DRA. Moreover, MO-ALFPAT outperforms all other algorithms in term of average ranking of IGD values and only perform

Table 2. Parameter settings

Algorithm	Parameters
MOEA/D	$\eta_c = 20, n_r = \infty$
MOEA/D-DE	$F = 0.5,\ CR = 1,\ k = \frac{N}{10},\ n_r = 2$
MOEA/D-DRA	$F = 0.5,\ CR = 1,\ k = \frac{N}{10},\ n_r = \frac{N}{100}$
ENS-MOEA/D	$F = 0.5,\ CR = 1,\ n_r = \frac{N}{100},\ Lp = 50$
MOEA/D-FRRMAB	$F = 0.5, K = 0.5,\ CR = 1, k = \frac{N}{2},\ n_r = 2,\ C = 5, W = \frac{N}{2}, D = 1$
MOEA/D-CS	$F = 0.5,\ K = 0.5,\ CR = 1,\ k = \frac{N}{2},\ n_r = 2,\ C = 5,\ W = \frac{N}{2}, D = 1,\ \alpha = 1,\ \lambda = 1.5$
NSGA-II	$\eta_m = 20$
MOFPA	$\lambda = 1.5,\ p_a = 0.8,\ \alpha = 0.01$
MO-ALFPA	$k = 5,\ \alpha = 0.05$
MO-ALFPAT	$k = 5,\ \alpha = 0.05$
MO-ALFPAB	$k = 3,\ \alpha = 0.05$
MOEA/D-ALFPA	$nr = 10,\ k = \frac{N}{10},\ \alpha = 1$

worse than MOEA/D-ALFPA and MOEA/D-CS in term of average ranking of HV values. It can also be seen that incorporating tournament selection in MO-ALFPAT as well as BAOS operator selection in BAOS not only improve the FE inefficiency of MO-ALFPA but also improve the performance of MO-ALFPA. Similarly, MOEA/D-ALFPA outperforms other algorithms in term of average ranking of HV values and only worse than MO-ALFPAT in term of average ranking of IGD values.

Wilcoxon's rank sum test is done to further determine the performance difference between the compared algorithms. As shown by Table 4, MOEA/D-ALFPA performs better in 7 problem instances compared to MO-ALFPAT while performing worse in 3 problem instances in regard to IGD values. A similar case can be seen in regard to HV values as shown in Table 5, MOEA/D-ALFPA performs better in 6 problem instances compared to MO-ALFPAT while perform worse in 4 problem instances. In addition, compared to the pareto dominance-based algorithms, MOEA/D-ALFPA performs better in 9 and 10 problem instances compared to NSGA-II and MOFPA respectively. In regard to HV values, MOEA/D-ALFPA performs better in 8 and 10 problem instances compared to NSGA-II and MOFPA respectively.

Table 3. The average rank of the compared algorithms based on the average IGD and HV values

	IGD	HV
MO-ALFPAT	**4.0**	4.8
MOEA/D-ALFPA	4.6	**4.5**
MOEA/D-CS	5.6	4.7
MOEA/D-FRRMAB	5.9	6.5
ENS-MOEA/D	5.9	6.5
MO-ALFPAB	6.2	6.4
MO-ALFPA	6.8	6.2
NSGAII	6.9	6.6
MOEA/D-DE	7.4	7.8
MOFPA	8.1	7.4
MOEA/D	8.2	8.1
MOEA/D-DRA	8.3	8.3

6. Conclusion and future work

This paper proposed two methods to extend ALFPA to solve MOP. The first proposed method denoted as MO-ALFPA is a simple extension of ALFPA by incorporating a similar framework as NSGA-II. To solve the FE inefficiency of ALFPA, a tournament selection is used to select the solutions to update every generation. However, the crowdedness measure in NSGA-II is replaced by HAD in both the tournament selection and the non-dominated sorting which is shown to improve the performance of MO-ALFPA. In addition, another proposed method denoted as MO-ALFPAB is similar to MO-ALFPA but uses AOS to solve ALFPA FE inefficiency. The last proposed method denoted as MOEA/D-ALFPA is a decomposition based multi objective extension of ALFPA with BAOS to choose the appropriate λ of the adaptive Lévy mutation operator based on the solution's dominance and crowdedness. Based on the obtained results for the bi-objective WFG and ZDT problem instances, the proposed ALFPA extension to solve MOP, MO-ALFPAT and MOEA/D-ALFPA, perform better than the compared algorithms. The results also showed that MO-ALFPAT and MO-ALFPAB performed better than MO-ALFPA while improving the FE inefficiency of ALFPA. However, the proposed methods perform only slightly better compared the relatively novel MOEA/D-CS with FRRMAB adaptive operator selection.

In our future work, we intend to extend the study to exploring other AOS strategies to choose the appropriate λ for the adaptive Lévy mutation operator, including a strategy that does not discretize

Table 4. Comparative results of the compared algo regarding the mean (std) of IGD values. "+", "-", and "=" respectively shows that the corresponding algorithm performs better, worse, and similar compared to MOEA/D-ALFPA according to wilcoxon's rank sum test with 0.05 significance level. The best mean and standard deviation results are highlighted in boldface

Problem	**NSGA II**	**MOFPA**	**MOEA/D**	**MOEA/D-DE**	**MOEA/D-DRA**	**ENS-MOEA/D**	**MOEA/D-FRRMAB**	**MOEA/D-CS**	**MO-ALFPA**	**MO-ALFPAB**	**MO-ALFPAT**	**MOEA/D-ALFPA**
WFG1	1.858×10^{-1} (3.09×10^{-1})-	1.606×10^{-2} (8.49×10^{-4})-	5.416×10^{-1} (4.25×10^{-1})-	2.903×10^{-2} (2.74×10^{-1})-	3.398×10^{-2} (3.20×10^{-2})-	2.286×10^{-2} (2.18×10^{-2})-	3.721×10^{-2} (3.54×10^{-2})-	8.572×10^{-2} (4.03×10^{-1})-	1.484×10^{-2} (8.13×10^{-4})-	1.490×10^{-2} (6.35×10^{-4})-	1.256×10^{-2} (1.36×10^{-4})-	**1.203×10^{-2} (4.29×10^{-5})**
WFG2	4.954×10^{-1} (2.19×10^{-1})-	8.580×10^{-2} (9.76×10^{-2})+	6.308×10^{-1} (2.43×10^{-1})-	2.657×10^{-1} (1.93×10^{-1})-	3.926×10^{-1} (2.27×10^{-1})-	2.746×10^{-1} (2.11×10^{-1})-	1.733×10^{-1} (1.61×10^{-1})-	1.778×10^{-1} (1.27×10^{-1})=	1.361×10^{-2} (3.39×10^{-3}) +	**1.263×10^{-2} (1.58×10^{-3})+**	1.740×10^{-1} (7.68×10^{-2})-	1.412×10^{-1} (9.52×10^{-2})
WFG3	1.524×10^{-2} (1.11×10^{-3})-	1.530×10^{-2} (8.99×10^{-4})-	3.879×10^{-2} (2.19×10^{-2})-	1.324×10^{-2} (5.66×10^{-5})-	1.349×10^{-2} (5.93×10^{-4})-	**1.129×10^{-2} (9.37×10^{-4})+**	1.364×10^{-2} (1.58×10^{-3})-	1.141×10^{-2} (5.10×10^{-5})+	1.521×10^{-2} (5.61×10^{-4}) -	1.525×10^{-2} (7.74×10^{-4})-	1.264×10^{-2} (2.60×10^{-4})-	1.145×10^{-2} (1.16×10^{-4})
WFG4	1.666×10^{-2} (9.36×10^{-4})=	2.453×10^{-2} (3.29×10^{-3})-	1.74×10^{-2} (3.92×10^{-3})=	2.353×10^{-2} (3.29×10^{-3})-	2.664×10^{-2} (7.99×10^{-3})-	2.262×10^{-2} (4.31×10^{-3})-	3.047×10^{-2} (7.33×10^{-3})-	**1.419×10^{-2} (1.17×10^{-3})+**	3.052×10^{-2} (5.94×10^{-3})-	2.852×10^{-2} (6.00×10^{-3})-	1.997×10^{-2} (2.43×10^{-3}) -	1.762×10^{-2} (2.87×10^{-3})
WFG5	7.163×10^{-2} (6.50×10^{-4})-	8.223×10^{-2} (1.39×10^{-2})-	7.480×10^{-2} (4.27×10^{-3})-	7.022×10^{-2} (2.86×10^{-4})+	6.990×10^{-2} (1.46×10^{-4})+	7.061×10^{-2} (8.70×10^{-4})=	7.039×10^{-2} (3.17×10^{-4})+	7.067×10^{-2} (4.11×10^{-3})+	7.294×10^{-2} (9.69×10^{-3})-	7.787×10^{-2} (3.29×10^{-3})-	**6.838×10^{-2} (8.11×10^{-3})=**	7.077×10^{-2} (7.48×10^{-4})
WFG6	2.385×10^{-2} (8.73×10^{-3})+	1.634×10^{-2} (8.91×10^{-4})+	4.238×10^{-2} (1.68×10^{-2})-	4.847×10^{-2} (1.93×10^{-2})-	3.937×10^{-2} (1.59×10^{-2})-	4.476×10^{-2} (2.25×10^{-2})=	3.128×10^{-2} (3.16×10^{-2})+	2.707×10^{-2} (2.87×10^{-2}) =	1.610×10^{-2} (7.90×10^{-4})+	1.627×10^{-2} (7.70×10^{-4})+	**1.396×10^{-2} (2.42×10^{-3})+**	3.680×10^{-2} (4.20×10^{-2})
WFG7	1.097×10^{-1} (2.97×10^{-2})-	1.862×10^{-2} (8.41×10^{-3})-	9.624×10^{-2} (3.38×10^{-2})-	6.062×10^{-2} (3.39×10^{-2})-	6.666×10^{-2} (2.28×10^{-2})-	4.240×10^{-2} (1.79×10^{-2})-	4.085×10^{-2} (2.52×10^{-2})-	1.823×10^{-2} (1.37×10^{-2})-	1.610×10^{-2} (7.32×10^{-4})-	1.567×10^{-2} (7.82×10^{-4})-	3.266×10^{-2} (1.73×10^{-2})-	**1.233×10^{-2} (1.19×10^{-4})**
WFG8	9.116×10^{-2} (1.32×10^{-2})-	9.653×10^{-2} (3.47×10^{-2})-	2.648×10^{-1} (2.04×10^{-1})-	**5.062×10^{-2} (5.21×10^{-2})+**	5.679×10^{-2} (5.04×10^{-2})-	5.967×10^{-2} (5.28×10^{-2}) -	6.156×10^{-2} (5.29×10^{-2})=	6.781×10^{-2} (4.84×10^{-2})-	1.419×10^{-1} (2.83×10^{-2}) -	9.094×10^{-2} (1.96×10^{-2})-	5.187×10^{-2} (1.02×10^{-2})=	5.355×10^{-2} (9.75×10^{-3})
WFG9	2.503×10^{-2} (2.03×10^{-3}) -	2.999×10^{-2} (3.30×10^{-3})-	4.430×10^{-2} (2.03×10^{-2})-	2.461×10^{-2} (2.55×10^{-3})-	2.556×10^{-2} (5.41×10^{-3})-	2.311×10^{-2} (2.51×10^{-3})=	2.414×10^{-2} (1.98×10^{-3})-	**2.254×10^{-2} (1.86×10^{-3}) =**	3.127×10^{-2} (3.67×10^{-3})-	3.168×10^{-2} (3.30×10^{-3})-	2.346×10^{-2} (2.57×10^{-3})=	2.302×10^{-2} (2.69×10^{-3})
ZDT1	4.816×10^{-3} (1.86×10^{-4})-	5.866×10^{-3} (5.20×10^{-4})-	4.364×10^{-3} (5.75×10^{-4})-	2.343×10^{-2} (7.45×10^{-3})-	8.512×10^{-3} (6.02×10^{-3})-	3.962×10^{-3} (1.74×10^{-3})-	4.154×10^{-3} (4.20×10^{-4})-	3.930×10^{-3} (2.75×10^{-5})-	5.178×10^{-3} (2.14×10^{-4})-	5.316×10^{-3} (2.30×10^{-4})-	4.906×10^{-3} (3.05×10^{-4})-	**3.915×10^{-3} (2.57×10^{-5})**
ZDT2	4.956×10^{-3} (2.45×10^{-4})=	5.881×10^{-3} (4.61×10^{-4})=	4.214×10^{-3} (5.77×10^{-4})=	1.494×10^{-2} (4.69×10^{-3})=	2.852×10^{-2} (2.48×10^{-2})=	5.839×10^{-3} (6.44×10^{-3})+	**4.152×10^{-3} (3.52×10^{-4})=**	4.482×10^{-1} (2.73×10^{-1})-	5.039×10^{-3} (2.21×10^{-4})=	5.172×10^{-3} (2.10×10^{-4})=	4.744×10^{-3} (2.60×10^{-4}) =	2.461×10^{-1} (3.02×10^{-1})
ZDT3	6.442×10^{-3} (5.33×10^{-3})+	7.644×10^{-3} (9.02×10^{-4})+	1.300×10^{-2} (1.01×10^{-2})+	3.957×10^{-2} (1.77×10^{-2})-	1.046×10^{-1} (4.69×10^{-2})-	2.854×10^{-2} (1.69×10^{-2})+	1.950×10^{-2} (8.19×10^{-3})+	7.002×10^{-3} (8.69×10^{-5})=	5.915×10^{-3} (6.41×10^{-4})+	5.725×10^{-3} (2.50×10^{-4})+	**5.686×10^{-3} (2.77×10^{-4})+**	3.408×10^{-2} (1.48×10^{-1})
ZDT4	**6.504×10^{-3} (1.64×10^{-3})+**	6.4327 (6.95)-	9.094×10^{-3} (3.11×10^{-3})+	3.542×10^{-2} (1.90×10^{-1})+	2.015 (2.24)=	3.7616 (2.05)-	8.904×10^{-1} (8.67×10^{-1})=	5.4668 (6.92) -	8.040 (5.25)-	1.229 (7.86×10^{-1})=	3.460×10^{-1} (2.14×10^{-1})+	7.939×10^{-1} (2.62×10^{-1})
ZDT6	1.228×10^{-1} (2.10×10^{-2})-	7.862×10^{-1} (7.45×10^{-1})-	3.018×10^{-2} (6.27×10^{-3}) -	5.688×10^{-1} (3.94×10^{-1})-	1.879×10^{-1} (1.83×10^{-1})-	**5.123×10^{-3} (8.19×10^{-3})+**	6.174×10^{-3} (1.54×10^{-2})+	3.945×10^{-2} (1.38×10^{-1})-	7.259×10^{-2} (1.17×10^{-1})-	9.734×10^{-3} (1.84×10^{-2})+	6.675×10^{-1} (1.70×10^{-1})-	2.048×10^{-2} (9.45×10^{-2})
+/-/=	3/9/2	3/10/1	2/10/2	3/10/1	1/11/2	4/7/3	4/7/3	3/7/4	3/10/1	4/8/2	3/7/4	

Table 5. Comparative results of the compared algo regarding the mean (std) of HV values. "+", "-", and "=" respectively shows that the corresponding algorithm performs better, worse, and similar compared to MOEA/D-ALFPA according to wilcoxon's rank sum test with 0.05 significance level. The best mean and standard deviation results are highlighted in boldface

Problem	NSGA II	MOFPA	MOEA/D	MOEA/D-DE	MOEA/D-DRA	ENS-MOEA/D	MOEA/D-FRRMAB	MOEA/D-CS	MO-ALFPA	MO-ALFPAB	MO-ALFPAT	MOEA/D-ALFPA
WFG1	6.138×10^{-1} (1.43×10^{-1})-	6.954×10^{-1} (2.84×10^{-4})-	4.533×10^{-1} (1.89×10^{-1})-	6.882×10^{-1} (1.13×10^{-2})-	6.864×10^{-1} (1.37×10^{-2})-	6.904×10^{-1} (9.53×10^{-3})-	6.837×10^{-1} (1.83×10^{-2})-	6.729×10^{-1} (1.27×10^{-1})=	6.958×10^{-1} (2.17×10^{-4})-	6.958×10^{-1} (1.80×10^{-4})-	**6.964×10^{-1} (4.82×10^{-5})+**	6.962×10^{-1} (1.72×10^{-5})
WFG2	5.669×10^{-1} (3.51×10^{-2})-	6.260×10^{-1} (9.89×10^{-3})+	5.416×10^{-1} (4.22×10^{-2})-	6.018×10^{-1} (3.01×10^{-2})-	5.831×10^{-1} (3.59×10^{-2})-	6.005×10^{-1} (3.19×10^{-2})-	6.143×10^{-1} (2.24×10^{-2})-	6.151×10^{-1} (1.69×10^{-2})=	6.330×10^{-1} (3.89×10^{-4})+	**6.332×10^{-1} (3.73×10^{-4})+**	6.167×10^{-1} (7.60×10^{-3})-	6.200×10^{-1} (9.40×10^{-3})
WFG3	5.793×10^{-1} (1.30×10^{-3})-	5.804×10^{-1} (3.13×10^{-4})-	5.679×10^{-1} (7.54×10^{-3})-	5.817×10^{-1} (1.75×10^{-4})-	5.812×10^{-1} (7.61×10^{-4})-	5.821×10^{-1} (1.07×10^{-3})+	5.812×10^{-1} (1.13×10^{-3})-	**5.822×10^{-1} (1.21×10^{-4})+**	5.799×10^{-1} (3.67×10^{-4})-	5.798×10^{-1} (4.11×10^{-4})-	5.808×10^{-1} (2.99×10^{-4})-	5.821×10^{-1} (1.31×10^{-4})
WFG4	3.429×10^{-1} (9.60×10^{-4})+	3.373×10^{-1} (1.55×10^{-3})-	3.407×10^{-1} (2.32×10^{-3})=	3.369×10^{-1} (1.62×10^{-3})-	3.356×10^{-1} (3.82×10^{-3})-	3.375×10^{-1} (1.82×10^{-3})-	3.340×10^{-1} (2.79×10^{-3})-	**3.431×10^{-1} (1.29×10^{-3})+**	3.35×10^{-1} (2.23×10^{-3})-	3.358×10^{-1} (2.62×10^{-3})-	3.389×10^{-1} (1.49×10^{-3})-	3.405×10^{-1} (1.96×10^{-3})
WFG5	3.064×10^{-1} (4.43×10^{-4})=	3.040×10^{-1} (5.09×10^{-3})-	3.045×10^{-1} (1.66×10^{-3})-	3.063×10^{-1} (2.17×10^{-4})-	3.066×10^{-1} (1.21×10^{-4})=	3.065×10^{-1} (3.75×10^{-4})=	3.062×10^{-1} (2.33×10^{-4})-	3.066×10^{-1} (2.32×10^{-3})+	3.068×10^{-1} (5.28×10^{-3})+	3.037×10^{-1} (9.77×10^{-4})-	**3.079×10^{-1} (4.44×10^{-3})=**	3.065×10^{-1} (4.18×10^{-4})
WFG6	3.386×10^{-1} (4.17×10^{-3})-	**3.462×10^{-1} (3.14×10^{-4})+**	3.301×10^{-1} (4.94×10^{-3})-	3.292×10^{-1} (5.64×10^{-0})-	3.322×10^{-1} (6.76×10^{-3})-	3.313×10^{-1} (8.80×10^{-3})-	3.387×10^{-1} (1.14×10^{-2})+	3.406×10^{-1} (1.01×10^{-2})=	3.46×10^{-1} (2.47×10^{-4})+	3.457×10^{-1} (2.92×10^{-4})+	3.454×10^{-1} (2.32×10^{-3})+	3.386×10^{-1} (1.34×10^{-2})
WFG7	3.143×10^{-1} (4.99×10^{-3})-	3.446×10^{-1} (4.69×10^{-3})-	3.161×10^{-1} (6.03×10^{-3})-	3.286×10^{-1} (1.09×10^{-2})-	3.256×10^{-1} (5.77×10^{-3})-	3.333×10^{-1} (7.79×10^{-3})-	3.344×10^{-1} (9.88×10^{-3})-	3.443×10^{-1} (6.40×10^{-3})-	3.460×10^{-1} (2.42×10^{-4})-	3.460×10^{-1} (2.42×10^{-4})-	3.367×10^{-1} (7.95×10^{-3})-	**3.470×10^{-1} (1.49×10^{-4})**
WFG8	3.119×10^{-1} (3.59×10^{-3})-	3.099×10^{-1} (8.71×10^{-3})-	2.720×10^{-1} (4.20×10^{-2})-	3.255×10^{-1} (2.11×10^{-2})-	3.228×10^{-1} (2.01×10^{-2})=	3.214×10^{-1} (2.07×10^{-2})=	3.211×10^{-1} (2.04×10^{-2})=	3.244×10^{-1} (7.90×10^{-3})=	2.934×10^{-1} (7.84×10^{-3})-	3.100×10^{-1} (6.03×10^{-3})-	3.248×10^{-1} (3.26×10^{-3})=	**3.260×10^{-1} (3.16×10^{-3})**
WFG9	3.386×10^{-1} (6.48×10^{-4})=	3.36×10^{-1} (1.15×10^{-3})-	3.283×10^{-1} (9.28×10^{-3})-	3.378×10^{-1} (9.74×10^{-4})-	3.369×10^{-1} (2.32×10^{-3})-	3.386×10^{-1} (7.43×10^{-4})=	3.378×10^{-1} (6.08×10^{-4})-	**3.389×10^{-1} (5.90×10^{-4})=**	3.362×10^{-1} (9.88×10^{-4})-	3.363×10^{-1} (8.33×10^{-4})-	3.387×10^{-1} (7.69×10^{-4})=	3.384×10^{-1} (1.03×10^{-3})
ZDT1	7.188×10^{-1} (2.62×10^{-4})-	7.164×10^{-1} (9.97×10^{-4})-	7.194×10^{-1} (8.31×10^{-4})-	6.917×10^{-1} (9.58×10^{-3})-	7.094×10^{-1} (1.37×10^{-2})-	7.190×10^{-1} (3.55×10^{-3})=	7.192×10^{-1} (9.58×10^{-4})-	7.200×10^{-1} (1.12×10^{-4})=	7.178×10^{-1} (4.21×10^{-4})-	7.174×10^{-1} (4.27×10^{-4})-	7.178×10^{-1} (5.98×10^{-4})-	**7.200×10^{-1} (1.34×10^{-4})**
ZDT2	4.435×10^{-1} (3.18×10^{-4})=	4.408×10^{-1} (8.29×10^{-4})=	**4.439×10^{-1} (1.06×10^{-3})=**	4.214×10^{-1} (8.43×10^{-3})=	4.201×10^{-1} (2.29×10^{-2})=	4.421×10^{-1} (7.45×10^{-3})+	4.438×10^{-1} (7.06×10^{-4})=	1.849×10^{-1} (1.59×10^{-1})-	4.429×10^{-1} (3.44×10^{-4})=	4.423×10^{-1} (4.23×10^{-4})=	4.424×10^{-1} (5.75×10^{-4})=	3.033×10^{-1} (1.76×10^{-1})
ZDT3	**6.022×10^{-1} (1.62×10^{-2})+**	5.968×10^{-1} (1.59×10^{-3})+	5.974×10^{-1} (7.57×10^{-3})+	5.853×10^{-1} (1.51×10^{-2})+	5.030×10^{-1} (4.86×10^{-2})-	5.752×10^{-1} (1.63×10^{-2})-	5.837×10^{-1} (9.29×10^{-3})+	5.992×10^{-1} (2.20×10^{-4})=	5.982×10^{-1} (7.71×10^{-4})+	5.987×10^{-1} (5.38×10^{-4})+	5.980×10^{-1} (7.16×10^{-4})+	5.822×10^{-1} (9.28×10^{-2})
ZDT4	**7.151×10^{-1} (2.72×10^{-3})+**	4.057×10^{-3} (1.67×10^{-2})-	7.109×10^{-1} (4.67×10^{-3})+	3.258×10^{-1} (1.60×10^{-1})+	1.116×10^{-1} (1.53×10^{-1})=	1.038×10^{-2} (4.41×10^{-2})-	1.222×10^{-1} (1.37×10^{-1})=	7.389×10^{-2} (1.76×10^{-1})-	0.000 (0.00)-	5.289×10^{-2} (9.78×10^{-2})-	3.401×10^{-1} (1.92×10^{-1})+	1.253×10^{-1} (1.49×10^{-1})
ZDT6	2.471×10^{-1} (2.11×10^{-2})-	9.867×10^{-2} (1.54×10^{-1})-	3.498×10^{-1} (7.65×10^{-3})-	8.758×10^{-2} (1.40×10^{-1})-	2.563×10^{-1} (1.09×10^{-1})-	**3.865×10^{-1} (8.84×10^{-3})=**	3.855×10^{-1} (1.66×10^{-2})+	3.655×10^{-1} (8.80×10^{-2})=	3.187×10^{-1} (1.12×10^{-1})-	3.803×10^{-1} (2.45×10^{-2})+	9.384×10^{-3} (1.47×10^{-2})-	3.780×10^{-1} (5.86×10^{-2})
+/-/=	3/8/3	3/10/1	2/10/2	2/11/1	0/10/4	2/7/5	3/8/3	3/3/8	4/9/1	4/9/1	4/6/4	

the choice of λ. Moreover, based on the promising performances of the proposed methods which are verified by the empirical findings, we will extend the study to solve many-objective optimization problems (MaOP).

Conflicts of interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Gemilang Santiyuda, Retantyo Wardoyo and Afiahayati; methodology, investigation, original draft preparation, Gemilang Santiyuda and Afiahayati; software, Gemilang Santiyuda; validation, resources, writing—review and editing, Retantyo Wardoyo and Afiahayati; supervision, project administration, Retantyo Wardoyo.

Acknowledgments

Researchers acknowledge the Directorate of Research and Community Service, Deputy for Strengthening Research and Development, Ministry of Research, Technology/National Research and Innovation Agency of the Republic of Indonesia in the PMDSU program.

References

[1] Z. Liu, J. Wang, S. Tan, S. Qiao, and H. Ding, "Multi-objective optimal design of the nuclear reactor pressurizer", *Int. J. Adv. Nucl. React. Des. Technol.*, Vol. 1, pp. 1–9, 2019, doi: 10.1016/j.jandt.2019.09.001.

[2] N. Jozefowiez, F. Semet, and E. G. Talbi, "Multi-objective vehicle routing problems", *Eur. J. Oper. Res.*, Vol. 189, No. 2, pp. 293–309, 2008, doi: 10.1016/j.ejor.2007.05.055.

[3] Z. Jingwei and M. Zujun, "Fuzzy Multi-objective Location-Routing-Inventory Problem in Recycling Infectious Medical Waste", in *2010 International Conference on E-Business and E-Government*, pp. 4069–4073, 2010, doi: 10.1109/ICEE.2010.1021.

[4] A. Ponsich, A. L. Jaimes, and C. A. C. Coello, "A Survey on Multiobjective Evolutionary Algorithms for the Solution of the Portfolio Optimization Problem and Other Finance and Economics Applications," *IEEE Trans. Evol. Comput.*, Vol. 17, No. 3, pp. 321–344, 2013, doi: 10.1109/TEVC.2012.2196800.

[5] M. G. C. Tapia and C. A. C. Coello, "Applications of multi-objective evolutionary algorithms in economics and finance: A survey", in *2007 IEEE Congress on Evolutionary Computation*, pp. 532–539, 2007, doi: 10.1109/CEC.2007.4424516.

[6] J. Handl, D. B. Kell, and J. Knowles, "Multiobjective Optimization in Bioinformatics and Computational Biology", *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, Vol. 4, No. 2, pp. 279–292, 2007, doi: 10.1109/TCBB.2007.070203.

[7] K. Miettinen, "Concepts", in *Nonlinear Multiobjective Optimization*, Boston, MA: Springer US, pp. 5–36, 1998.

[8] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results", *Evol. Comput.*, Vol. 8, No. 2, pp. 173–195, 2000, doi: 10.1162/106365600568202.

[9] J. Sun, H. Zhang, Q. Zhang, and H. Chen, "Balancing Exploration and Exploitation in Multiobjective Evolutionary Optimization", in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 199–200, 2018, doi: 10.1145/3205651.3205708.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Trans. Evol. Comput.*, Vol. 6, No. 2, pp. 182–197, 2002, doi: 10.1109/4235.996017.

[11] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition", *IEEE Trans. Evol. Comput.*, Vol. 11, No. 6, pp. 712–731, 2007, doi: 10.1109/TEVC.2007.892759.

[12] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances", in *2009 IEEE Congress on Evolutionary Computation*, pp. 203–208, 2009, doi: 10.1109/CEC.2009.4982949.

[13] M. Laszczyk and P. B. Myszkowski, "Improved selection in evolutionary multi–objective optimization of multi–skill resource–constrained project scheduling problem", *Inf. Sci. (Ny).*, Vol. 481, pp. 412–431, 2019, doi: https://doi.org/10.1016/j.ins.2019.01.002.

[14] H. Ishibuchi, N. Akedo, and Y. Nojima, "Relation between Neighborhood Size and MOEA/D Performance on Many-Objective Problems", in *Evolutionary Multi-Criterion Optimization*, pp. 459–474, 2013.

[15] S. Z. Zhao, P. N. Suganthan, and Q. Zhang, "Decomposition-Based Multiobjective Evolutionary Algorithm With an Ensemble of Neighborhood Sizes", *IEEE Trans. Evol. Comput.*, Vol. 16, No. 3, pp. 442–446, 2012, doi:

10.1109/TEVC.2011.2166159.

[16] H. S. Noghabi, H. R. Mashhadi, and K. Shojaei, "Differential Evolution with Generalized Mutation Operator for Parameters Optimization in Gene Selection for Cancer Classification", 2016.

[17] Q. Guo, Y. Gao, L. Cui, and J. Zhang, "Cuckoo search algorithm based on three random walks", in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pp. 2180–2186, 2017, doi: 10.1109/CompComm.2017.8322923.

[18] R. Salgotra and U. Singh, "Application of mutation operators to flower pollination algorithm", *Expert Syst. Appl.*, Vol. 79, pp. 112–129, 2017, doi: 10.1016/j.eswa.2017.02.035.

[19] H. Li and Q. Zhang, "Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II", *IEEE Trans. Evol. Comput.*, Vol. 13, No. 2, pp. 284–302, 2009, doi: 10.1109/TEVC.2008.925798.

[20] L. Chen, W. Gan, H. Li, K. Cheng, D. Pan, L. Chen, and Z. Zhang, "Solving multi-objective optimization problem using cuckoo search algorithm based on decomposition", *Appl. Intell.*, Vol. 51, No. 1, pp. 143–160,2020, doi: 10.1007/s10489-020-01816-y.

[21] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms", *IEEE Trans. Evol. Comput.*, Vol. 3, No. 2, pp. 124–141, 1999, doi: 10.1109/4235.771166.

[22] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter Control in Evolutionary Algorithms: Trends and Challenges", *IEEE Trans. Evol. Comput.*, Vol. 19, No. 2, pp. 167–187, Apr. 2015, doi: 10.1109/TEVC.2014.2308294.

[23] N. Hitomi and D. Selva, "A Classification and Comparison of Credit Assignment Strategies in Multiobjective Adaptive Operator Selection", *IEEE Trans. Evol. Comput.*, Vol. 21, No. 2, pp. 294–314, 2017, doi: 10.1109/TEVC.2016.2602348.

[24] D. E. Goldberg, "Probability matching, the magnitude of reinforcement, and classifier system bidding", *Mach. Learn.*, Vol. 5, No. 4, pp. 407–425, 1990, doi: 10.1007/BF00116878.

[25] D. Thierens, "Adaptive Strategies for Operator Allocation", in *Parameter Setting in Evolutionary Algorithms*, F. G. Lobo, C. F. Lima, and Z. Michalewicz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 77–90, 2007.

[26] K. Li, Á. Fialho, S. Kwong, and Q. Zhang, "Adaptive Operator Selection With Bandits for a Multiobjective Evolutionary Algorithm Based on Decomposition", *IEEE Trans. Evol. Comput.*, Vol. 18, No. 1, pp. 114–130, 2014, doi: 10.1109/TEVC.2013.2239648.

[27] W. Lin, Q. Lin, J. Ji, Z. Zhu, C. A. C. Coello, and K. C. Wong, "Decomposition-based multiobjective optimization with bicriteria assisted adaptive operator selection", *Swarm Evol. Comput.*, Vol. 60, p. 100790, 2021, doi: 10.1016/j.swevo.2020.100790.

[28] X. S. Yang, "Flower Pollination Algorithm for Global Optimization", in *Unconventional Computation and Natural Computation*, pp. 240–249, 2012.

[29] V. L. Huang, P. N. Suganthan, A. K. Qin, and S. Baskar, "Multiobjective Differential Evolution with External Archive and Harmonic Distance-Based Diversity Measure", *Singapore*, 2005.

[30] C. Y. Lee and X. Yao, "Evolutionary algorithms with adaptive Levy mutations", in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, Vol. 1, pp. 568–575, 2001, doi: 10.1109/CEC.2001.934442.

[31] K. Miettinen, "A Posteriori Methods", in *Nonlinear Multiobjective Optimization*, Boston, MA: Springer US, pp. 77–113, 1998.

[32] I. Das and J. E. Dennis, "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems", *SIAM J. Optim.*, Vol. 8, No. 3, pp. 631–657, 1998, doi: 10.1137/S1052623496307510.

[33] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review", *IEEE Trans. Evol. Comput.*, Vol. 7, No. 2, pp. 117–132, 2003, doi: 10.1109/TEVC.2003.810758.

[34] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach", *IEEE Trans. Evol. Comput.*, Vol. 3, No. 4, pp. 257–271, 1999, doi: 10.1109/4235.797969.

[35] S. Huband, L. Barone, L. While, and P. Hingston, "A Scalable Multi-objective Test Problem Toolkit", in *Evolutionary Multi-Criterion Optimization*, 2005, pp. 280–295.

[36] S. Jiang, S. Yang, Y. Wang, and X. Liu, "Scalarizing Functions in Decomposition-Based Multiobjective Evolutionary Algorithms", *IEEE Trans. Evol. Comput.*, Vol. 22, No. 2, pp. 296–313, 2018, doi: 10.1109/TEVC.2017.2707980.

[37] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu, "Moea/d with Adaptive Weight Adjustment", *Evol. Comput.*, Vol. 22, No. 2, pp. 231–264, 2014.

[38] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization [Educational Forum]", *IEEE Comput. Intell. Mag.*, Vol. 12, No. 4, pp. 73–87, 2017, doi: 10.1109/MCI.2017.2742868.

[39] X. S. Yang, M. Karamanoglu, and X. He, "Flower pollination algorithm: A novel approach for multiobjective optimization", *Eng. Optim.*, Vol. 46, No. 9, pp. 1222–1237, 2014, doi: 10.1080/0305215X.2013.832237.