D^2MOPSO: MOPSO based on Decomposition and Dominance with Archiving using Crowding Distance in Objective and Solution Spaces

N. Al Moubayed n.al-moubayed@rgu.ac.uk
Robert Gordon University, Aberdeen, AB25 1HG, UK

A. Pertovski a.petrovski@rgu.ac.uk
Robert Gordon University, Aberdeen, AB25 1HG, UK

J. McCall j.mccall@rgu.ac.uk
Robert Gordon University, Aberdeen, AB25 1HG, UK

Abstract

This paper improves a recently developed multi-objective particle swarm optimizer (D^2MOPSO) that incorporates dominance with decomposition used in the context of multi-objective optimisation. Decomposition simplifies a multi-objective problem (MOP) by transforming it to a set of aggregation problems, whereas dominance plays a major role in building the leaders' archive. D^2MOPSO introduces a new archiving technique that facilitates attaining better diversity and coverage in both objective and solution spaces. The improved method is evaluated on standard benchmarks including both constrained and unconstrained test problems, by comparing it with three state-of-the-art multi-objective evolutionary algorithms: MOEA/D, OMOPSO and dMOPSO. The comparison and analysis of the experimental results, supported by statistical tests, indicate that the proposed algorithm is highly competitive, efficient and applicable to a wide range of multi-objective optimisation problems.

Keywords

Multi-Objective Particle Swarm Optimisation, decomposition-based Evolutionary Algorithms, Dominance, Archiving Technique, $OMOPSO$, $MOEA/D$.

1 Introduction

Particle Swarm Optimisation (PSO) is a population-based metaheuristic (Kennedy and Eberhart (1995)) that simulates the behavior of a flock of birds in nature. The particles in the swarm move in the solution space searching for the regions where promising solutions are located. The particles communicate with each other to discover the social and personal information that direct their movement.

Many real-world applications often involve optimisation of multiple, competing objectives in large search spaces (Talbi (2009)). It is therefore an important task to effectively and simultaneously address multiple optimisation objectives by identifying a set of well-distributed Pareto optimal solutions that yield good values for each objective.

Population-based metaheuristics (e.g. PSO) have being developed to facilitate an efficient search in multi-dimensional solution spaces, the feasible regions within which are determined by a set of (often non-linear) constraints. However, instead of obtaining infinite number of Pareto optimal solutions, which is a time consuming and resource

©200X by the Massachusetts Institute of Technology

demanding task, it is often preferable to search for a set of representative solutions that closely approximate the true Pareto front being uniformly distributed along its length (Coello Coello et al. (2007)).

Designing effective measures for diversification of solutions to a Multi-Objective Problem (MOP) and for their uniform distribution along the Pareto optimal front is a challenging research problem (Reyes-Sierra and Coello Coello (2006)). Multi-objective metaheuristics can be classified into four categories: decomposition-based (scalar), criterion-based, dominance-based, and indicator-based approaches (discussed in detail by Talbi (2009)). It would be interesting therefore to ascertain whether/how these approaches can be combined or enhanced to achieve a better preservation of solution diversity, and as a consequence, a closer approximation of the Pareto optimal front. Hybridising different search approaches have been reported in (Zhou et al. (2011)).

D^2MOPSO, originally proposed in Al Moubayed et al. (2012), utilizes a hybrid approach of dominance (e.g., Reyes-Sierra and Coello Coello (2005)) and decomposition (e.g., Zhang and Li (2007)). This approach achieves fast convergence to the true Pareto Front without resorting to the use of genetic operators (e.g., mutation). Also, a better exploitation of the information discovered during the search enables the suggested multi-objective PSO approach to be applied to problems that necessitate complex system optimisation. The version we proposed in (Al Moubayed et al. (2012)) only presented tentative ideas on how to achieve this hybrid approach. The work presented here differs in several major points: 1) The mechanism for leaders' selection, 2) the archiving technique 3) the objectives are no longer normalized using a Sigmoid function 4) the current paper also provides comprehensive experiments and analysis of the performance of the algorithms. From now on D^2MOPSO will refer to the version presented in this work only.

D^2MOPSO introduces a bounded leaders' archive based on the crowding distance in both objective and solution spaces to store the non-dominated particles. The leaders are then selected from the archive using the aggregation value as the selection criterion.

The rest of the paper is organized as follow: Section 2 surveys the related work. Section 3 describes and details the methods. The experimental setup and benchmarks used for testing the proposed algorithm are discussed in Section 4. Results, statistical and complexity analysis, and discussion are presented in Section 5. Section 6 concludes the paper.

2 Background and Related Work

2.1 Multi-Objective Optimisation Problems

Solving a multi-objective optimisation problem is challenging because an improvement in one objective often happens at the expense of deterioration in other objective(s). The optimisation challenge therefore is to find the entire set of trade-off solutions that satisfy all conflicting objectives.

Let $F(x) \in \Delta \subset R^m$ be a vector of objectives:

$$F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n) \in \Omega \subset R^n$ is the vector of decision variables, n is the dimension of solution space, and $m \geq 2$ is the number of objectives. The search space (also called the solution space) refers to the space of decision variables, whereas the objective space is the space where the objective vectors lie.

When minimizing $F(x)$, for example, a domination relationship is defined between the solutions as follows: let $x, y \in \Omega$, $x \prec y$ if and only if $f_i(x) \leq f_i(y)$ for all $i = \{1, 2, ..., m\}$, and there is at least one j for which $f_j(x) < f_j(y)$. x^* is a Pareto optimal solution if there is no other solution $s \in \Omega$ such that $s \prec x^*$. Therefore the Pareto optimality of a solution guarantees that any enhancement of one objective would result in the worsening of at least one other objective. The concept of x^* gives a set of solutions called the Pareto optimal set P. The image of the Pareto optimal set in the objective space (i.e. $F(P)$) is called the Pareto Front (PF) (Reyes-Sierra and Coello Coello (2006)).

Solving MOPs is highly dependent on the structure of the PF, in addition to the number of the objectives as the number of optimal solutions necessary to find a good approximation of the PF tends to grow with an increase in the number of objectives. A multi-objective evolutionary algorithm aims at producing an approximated PF with uniform diversity that fully covers the PF.

2.2 Multi-Objective Particle Swarm Optimisation

PSO is a population-based metaheuristic yielding competitive solutions in many application domains (Wang et al. (2004); Jaishia and Ren (2007)). Several multi-objective PSO (MOPSO) methods have recently been developed and demonstrated their performance on real-life problems and standard benchmarks (Reyes-Sierra and Coello Coello (2006); Baltar and Fontane (2006)). In MOPSO, each particle in a swarm represents a potential solution in the solution space.

A particle is characterized by its position and velocity. The position is the location in the solution space, whereas the velocity represents the positional change. The particle uses the positions of the selected global leader, and its own personal movement trajectory to update the velocity and position values using Eq. 2 and Eq. 3 (Reyes-Sierra and Coello Coello (2006); Kennedy et al. (2001)).

$$\begin{aligned} v_i(t+1) &= w * v_i(t) + C_1.r_1.(x_{pbest_i} - x_i(t)) \\ &\quad + C_2.r_2.(x_{lbest_i} - x_i(t)) \end{aligned} \tag{2}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{3}$$

where $pbest_i$ and $lbest_i$ are the best personal performance and the best local performance of $particle_i$ respectively; r_1, r_2 are vectors of normally distributed random values, w is the inertia weight, C_1, C_2 are the learning factors, and . is the element by element product.

2.3 Decomposition-based Evolutionary Algorithms

Decomposition-based evolutionary approaches rely mainly on an aggregation function that converts the MOP into a single-objective problem by assigning a weight to each objective (i.e. objectives are not necessarily equally important). Different weight assignments yield different aggregation functions, which are used to transform the MOP into a set of distinct single-objective problems. The original MOP is then addressed by simultaneously solving these sub-problems.

MOEA/D (Zhang and Li (2007); Li and Zhang (2009)) discovers Pareto optimal solutions of a MOP by solving single-objective sub-problems using a Genetic Algorithm. MOEA/D defines a number of distinct evenly distributed weighting vectors (λ) equal to the size of the population.

$$\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_m\} : \sum_{i=1}^{m} \lambda_i = 1 \tag{4}$$

where m is the number of objectives.

Each individual in MOEA/D has a fixed size neighborhood throughout the optimisation process. The neighbours are the T individuals that have the smallest distance between their own λ and the corresponding individual's λ. The population is evolved by mating each individual with a randomly selected member of its neighborhood. The resulted solution replaces a neighbor only when it has a better aggregation value calculated using the neighbor's λ. As only the fittest individuals survive, the last population of the evolution process presents the approximation of the PF. The advantages of this approach in terms of mathematical soundness, algorithmic structure and computational cost are explained in Li and Zhang (2009). Following is a brief description of some decomposition-based MOEA using PSO:

MOPSO/D (Peng and Zhang (2008)) is a multi-objective optimization method that uses the MOEA/D framework to solve continuous MOPs. MOPSO/D substitutes the genetic algorithm in MOEA/D with PSO. It relies fully on decomposition to update the personal and global information. Each particle is associated with one local best, so an update of a particle position can trigger position update in its neighbors' local best(s) resulting in duplications and making the algorithm prone to falling into local optima. Hence, mutation is employed.

SDMOPSO: In SDMOPSO (Al Moubayed et al. (2010)), the particle's global best is found among the solutions located within a certain neighborhood. SDMOPSO tackles the drawback of MOPSO/D by allowing the particle position update only if it leads to a better aggregation value (i.e. the value of the aggregation function). Duplicated global bests are avoided by restricting the number of updates to a predefined small number (e.g. two). Although SDMOPSO shows significant improvement over MOPSO/D, the particles may still fall into a local optima if they were unable to find better locations to move to.

dMOPSO: dMOPSO (Martínez and Coello Coello (2011)) uses decomposition to update the leaders' archive and to select the swarm leader(s). The archive stores the particles with the best aggregation values for each particle in the swam, whereas the particles' personal memory store the position with the best aggregated value found so far. To maintain the diversity of the swarm and to avoid local optima, dMOPSO re-initializes the particles' memory using a Gaussian normal distribution when the particle exceeds a certain age (i.e. number of iterations with no update). This may lead to losing all the experience gained throughout the exploration process, as well as adding more complexity to the algorithm. Besides, it uses decomposition as a way to substitute dominance. With the absence of dominance, the decomposition strategy is confined to leading the swarm into a limited number of destinations equal to the swarm size (the number of λ vectors). With complicated Pareto fronts (i.e. disconnected) and the limited size of the swarm, dMOPSO might fail to cover the entire PF.

In addition to the discussed methods, Sigma-MOPSO (Mostaghim and Teich (2003)) uses a decomposition-like approach to select the local guide (i.e. $lbest$). Each particle p_i is assigned a value, σ_i, based on its location in the objective space:

$$\sigma_i = \frac{(f_1^2 - f_2^2)}{f_1^2 + f_2^2} \tag{5}$$

for a bi-objective problem, where f_1, f_2 are the objective values of p_i. Using this

definition: all the particles where $f_1 = af_2$, i.e. are located in the objective space on a line with slope a, would have the same σ_i. $lbest$ for the corresponding particle p_i is the one that has σ_{lbest} with the closest distance to σ_i. The clustered particles in the swarm have similar σ_i making them move in the same direction, as a result of selecting a set of clustered leaders. This might reduce the coverage and diversity of the PF. Hence Sigma-MOPSO requires a large swarm (Parsopoulos and Vrahatis (2008)). The particles in a decomposition approach, on another hand, are guided to distinct directions using unique and evenly distributed λ values.

3 Methods

3.1 Archiving based on Crowding Distance in Objective and Solution Spaces

Dominance-based approaches to multi-objective optimisation use the concept of dominance and Pareto optimality to guide the search process. The majority of dominance-based MOPSOs use a fixed-size leaders' archive to store trade-off solutions found through the optimisation process (Coello Coello et al. (2007)). Thus, the selected leaders influence significantly the optimisation process; maintaining the archive and selecting the leaders is, therefore a major challenge for a MOPSO.

MOPSO aims at minimizing the distance between the solutions in the archive and the true PF, whilst maximizing the diversity of these solutions in the objective space. Several density estimators are employed to tackle these challenges. Some commonly used techniques are listed below (Talbi (2009)):

Kernel (Fonseca and Fleming (1993)): Kernel methods define the neighborhood of a solution using a kernel function that takes the distance between two solutions as the argument. The density estimator of a solution is represented by the sum of the kernel function values (usually referred to as crowding distance). The individuals with the lowest crowding distance are preferred.

Adaptive grid (Knowles and Corne (2000)): This method divides the objective space recursively when the front bounds grow/shrink beyond a certain amount to reduce computational overhead. The objective space is divided using a grid so that the crowding of the solutions is measured by the crowding of their images in the objective space within the grid. This allows the system to remove or replace solutions at the highly populated cells.

Niche count (Deb and Goldberg (1989)): The neighbourhoods are defined using a niche, i.e. a circular space with a predefined radius around the particle. The neighbours are the ones located within its niche. Particles/individuals with less populated niche are preferred.

ϵ-dominance (Laumanns et al. (2002)): determines how much better a solution should be to replace another which requires dividing locally each dimension in the objective space into small cells of size ϵ. ϵ loosly defines the resolution of the approximated PF produced using MOPSO.

Nearest neighbour (Deb et al. (2002)): For each solution, the nearest neighbour density estimator calculates the average distance between two individuals of the Pareto front on either side of the current solution along each of the objectives. The non-dominated individuals with highest distance are favored.

Most archiving techniques maintain the quantity and diversity of the solutions in the objective space without taking into account the diversity of these solutions in the solution space, which might result in discarding potentially important regions there. In earlier work (Al Moubayed et al. (2011)), we tackled this issue using an approach based on clustering both in objective and solution spaces. The major drawback of this

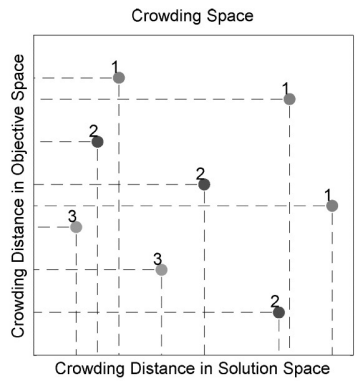


Figure 1: Dominance-based ranking for the non-dominated solutions of the leaders' archive using the crowding distance values in both solution and objective spaces. X-axis is the crowding distance in the solution space, Y-axis is the crowding distance in the objective space. The number next to each particle represents its rank. In this example the particles ranked with 3 are the best.

approach is its computational complexity. The archiving technique suggested in this paper provides a relatively simple solution that uses a density estimator in both the solution and the objective spaces.

Each particle has two crowding distance coordinates one in each space. Therefore, the crowding distance is a two-dimensional vector where the first dimension characterizes crowding in the objective space, and the second in the solution space. We use crowding distance (kernel density estimator) defined as follows:

$$CD(p_i) = \Big(\sum_{j=1}^{AS} \parallel p_i, p_j \parallel_{\Omega}, \sum_{j=1}^{AS} \parallel F(p_i), F(p_j) \parallel_{\Delta} \Big) \tag{6}$$

where AS is the size of the archive, p_i is the particle i's decision variable vector. $CD(p_i)$ is a vector of the crowding distances in the solution and objective spaces.

The crowding distance is only calculated when the maximum archive size is exceeded, and a replacement of some particles is needed. The elimination process starts by crowding the particles in both spaces. The elimination then considers the particles' two crowding distances in order to decide on the particle to be removed or substituted.

A domination relationship and dominance-based ranking are applied to the created crowding space. The particle with the worst rank is then replaced, with one selected randomly in the case of a tie. This is used in many MOEAs to sort the solutions in the objective space (Zitzler et al. (2003a)). Fig. 1 demonstrates an example of the dominance-depth ranking used. The mutually non-dominated solutions of the leaders' archive are ranked in the crowding space using their crowding values.

Algorithm 1 outlines the proposed archiving algorithm, where the operator $r(A)$ assigns a ranking value $rank$ to the set A, CD is defined in Eq. 6, and Φ is the empty set.

Algorithm 1 Dominance-depth ranking in the crowding space

```
if Size(leadersArchive) ≥ MaxSize then
    for p_i in leadersArchive do
        CD(p_i)
    end for
    temporaryArchive = leadersArchive
    rank = 0
    while temporaryArchive ≠ Φ do
        r(a) = rank
        rank = rank + 1
        temporayArchive = temporaryArchive \ a
    end while
    replace the particle with the worst rank.
end if
```

3.2 D^2MOPSO

Decomposition assists the optimisation process to find potential solutions that are evenly distributed along the PF (Zhang and Li (2007)). By associating each particle with a distinct aggregation problem (i.e. λ value), the direction of exploration activity of each particle is focused on a specific region in the objective space and is aimed at reducing the distance to the reference point.

Substituting entirely the dominance approach with decomposition in MOPSO (i.e. using the aggregation value instead of dominance as the leaders' selection criterion) might lead to premature convergence as each particle is strictly directed to one destination. At some point during the optimisation process, the particles would be unable to update their positions and personal best memory as the local best and neighborhood information become static. In addition, solving a MOP with complicated PF raises a serious challenge as some λ vectors direct the corresponding particles to unattainable areas. In such cases, part of the swarm would be exploring undesirable regions in the objective space for a considerable number of evaluations. Fig 2 demonstrates this problem where only eight out of twenty particles are directed towards the true PF. One may suggest adjusting the initialization of λ vectors to cover only attainable regions. This solution, however, only works if the true PF is known a-priori, which is not the case for most, if not all, real-life problems.

Another limitation of decomposition relates to how it operates in high-dimensional objective spaces. It struggles to produce a sufficient number of non-dominated solutions that cover the entire PF as the space to be covered by the swarm/population using λ vectors grows exponentially with the number of dimensions. This requires the decomposition-based approaches to use a large swarm/population in order to offer a good PF coverage, increasing therefore the number of necessary function evaluations, which can be a disadvantage for real-life problems with expensive or difficult to obtain evaluations.

To overcome all these drawbacks within MOPSO framework, D^2MOPSO integrates both dominance and decomposition. The bounded leaders' archive, Section 3.1, uses dominance to store only non-dominated particles. The personal best values are updated, and the leaders are selected using the decomposition's aggregation function.

Many aggregation functions can be used with decomposition. Recently, the

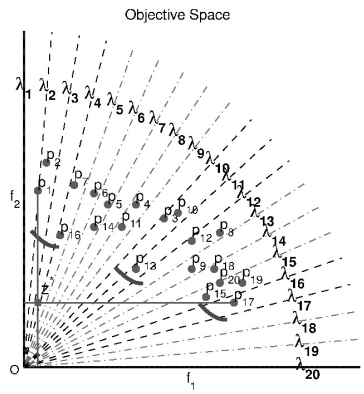


Figure 2: Swarm of 20 particles in a sample objective space. When only decomposition is used 8 particles are directed to promising regions in the space, the remaining 12 are directed to unpromising ones, i.e. 60% of the swarm is wasting the search effort.

weighted penalty-based boundary intersection (PBI) method has been used (Zhang and Li (2007); Martínez and Coello Coello (2011)), and is adopted in this paper. PBI uses a weighted vector λ and a penalty value θ to minimize the distance to the utopia vector (i.e. a hypothetical vector between the reference point ($z^* = \min\{f_i(x)|x \in \Psi \subseteq \Omega\}$) and the center of the PF (Zhang and Li (2007)), where Ψ the area investigated so far in the solution space). In addition it minimizes d_1 and the direction error of the weighted vector d_2 from the solution in the objective space $F(x)$, defined as:

$$minimize_{x\in\Omega} \quad g(x|\lambda, z^*) = d_1 + \theta d_2 \tag{7}$$

where

$$\begin{aligned} d_1 &= \frac{\| (F(x) - z^*)^T \lambda \|}{\| \lambda \|} \\ d_2 &= \| (F(x) - z^*) - d_1 \frac{\lambda}{\| \lambda \|} \| \end{aligned} \tag{8}$$

D^2MOPSO uses PBI to transform the optimisation objective defined by Eq. 1 into N scalar optimisation problems, where N is the swarm size. By changing the weights and using the reference point defined above, Pareto optimal solutions may be approximated. The following steps summarize D^2MOPSO:

Initialization: D^2MOPSO starts by initializing the swarm with N particles and N λ vectors. Every particle is assigned a unique λ vector that gives the best aggregated fitness value (e.g., minimum in case of a minimization problem) for the initialized particle. The initial value of the particle's memory $pbest$ is its own information ($pbest_i = x_i$) as it lacks any exploration experience at the beginning of the search process. The initial velocity of the particle is set to zero ($V_{i_0} = 0$). The leaders' archive is set to a fixed size, and is initialized by the non-dominated particles in the swarm. The reference point z^* is the vector in the objective space with the best objective values found so far.

Evolution: During this phase D^2MOPSO goes through a pre-set number of iterations. At iteration (t), the particle determines the next move by calculating the new

velocity and new position using Eq. 9 and Eq. 10, which involve $pbest$ and the information about a global leader selected from the leaders' archive.

$$\begin{aligned} v_i(t+1) &= w * v_i(t) + C_1.r_1.(x_{pbest_i} - x_i(t)) \\ &+ C_2.r_2.(x_{lbest_i} - x_i(t)) \end{aligned} \tag{9}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{10}$$

where $pbest_i$ is the personal best performance of $particle_i$, $lbest_i$ is a leader selected from the archive, $r_1, r_2 \in [0,1]$ are uniformly distributed random variables, $w \in [0.1, 0.5]$ is the inertia weight, and $C_1 = C_2 = 2.0$ are the learning factors. These parameters are defined following other recent MOPSOs (Reyes-Sierra and Coello Coello (2005); Al Moubayed et al. (2010); Martínez and Coello Coello (2011); Peng and Zhang (2008)).

In order to ensure that the decision variables fall into the predefined boundaries in the solution space, after each update their values are checked as follows:

$$(x_i^d, v_i^d) = \begin{cases} (min_d, -v_i^d) & if\ x_i^d < min_d \\ (max_d, -v_i^d) & if\ x_i^d > max_d \end{cases} \tag{11}$$

where i is the particle index, d is the index of the decision variable within the decision variables vector. min_d and max_d are the lower and upper boundaries of decision variable d respectively.

During leader selection (Algorithm 2 , where $lbest_i$ is the selected leader for the corresponding $particle_i$) each particle selects the leader that gives the best aggregation value using the particle's λ and the aggregation function in Eq. 7.

Algorithm 2 Leaders' Selection

```
for i = 1 to N do
  lbest_i = lbest_1
  for j = 2 to Size(leaders' archive) do
    if g(lbest_i|λ_i, z*) >  g(lbest_j|λ_i, z*) then
      lbest_i = lbest_j
    end if
  end for
  Select lbest_i as leader for particle i
end for
```

After the particle updates its position and velocity, it has to update its $pbest_i$ as well. $pbest_i$ is replaced only if the new aggregation value is better:

$$\begin{aligned} if \quad & g(pbest_i|\lambda_i, z^*) > g(x_i|\lambda_i, z^*) \\ \text{then} \quad & pbest_i = x_i \end{aligned} \tag{12}$$

The leaders' archive is then updated with any new non-dominated particles subject to the crowding restriction explained in Section 3.1. The reference point is updated

when a better objective value is found. When a particle updates its position, the new position is checked against z^* and updates it if necessary (Eq. 13).

$$if \quad z_j^* < f_j(x_i) \text{ then } z_j^* = f_j(x_i) : j \in [1, .., m] \tag{13}$$

Finally, the external archive, which contains all the non-dominated solutions found during the optimization process, is updated to contain the new non-dominated particles. The use of the external archive is optional as it is not involved in the evolution process. However, it is recommended as it may contain solutions with better PF coverage and enhanced distribution in the solution space than the leaders' archive.

Termination: The algorithm terminates when the maximum number of iterations is reached. The content of the external archive is used to approximate the PF. If the external archive is not used, then the leaders' archive is considered.

Algorithm 3 lists a pseudo-code for D^2MOPSO, where $CheckBoundaries$ validates the decision variables and adjust them when necessary.

Algorithm 3 D^2MOPSO

```
Initialize the swarm with N particles and a set Λ of N λ vectors
for i = 1 to N do
   assign the particle i to the λ vector that gives the best aggregation value
   remove the selected λ from Λ
   initialize velocities V = {v_1, ..., v_N} and pbest_i
   Initialize leaders' archive, external archive and z*
end for
for t = 1 to MaxIterations do
   for j = 1 to N do
      Select lbest_j (Algorithm 2)
      update Velocity, v_j(t + 1) (Eq. 9)
      update position, x_j(t + 1) (Eq. 10)
      CheckBoundaries(x_j(t + 1)) (Eq. 11)
      evaluate the new position (The corresponding problem fitness function)
      update pbest_j (Eq. 12)
      update leaders archive (Algorithm 4)
      update z* (Eq. 13)
      update external archive
   end for
end for
Return the final result in the external archive
```

D^2MOPSO can solve both constrained and unconstrained continuous MOPs. An additional step is required when creating and updating the leaders' archive to accommodate constrained problems. The constraints are evaluated for each particle so that the leaders' archive update process is biased towards particles which do not violate the constraints (or breach the constraints to a lesser degree).

Algorithm 4 outlines the update of the leaders' archive with a new particle S, where $Size$ is the size of leaders' archive, $breachConst$ checks if the particle has violated the constraints, $constraints$ evaluates the constraints; $valid(S)$ is correct if S has caused the removal of at least one particle from the archive or if it was not dominated by any other particle.

3.3 Novelty of D^2MOPSO

Dominance and decomposition are commonly used approaches in multi-objective evolutionary algorithms (Coello Coello et al. (2007); Li and Zhang (2009); Deb et al. (2002); Reyes-Sierra and Coello Coello (2005)), but, up to our knowledge, they have mostly been used separately. Nasir et al. (2011) introduced the concept of fuzzy dominance and only used decomposition when one solution fails to dominate the other in terms of fuzzy dominance level. D^2MOPSO is designed to take advantage of both concepts so that decomposition is used to select the leaders from a dominance-based archive. D^2MOPSO maintains the algorithmic simplicity of MOPSO by not utilizing any genetic or sampling operators. D^2MOPSO also uses a novel archiving technique that maintains diversity in both the objective and the solution spaces. Table 1 compares among five state-of-the-art decomposition-based MOEAs.

4 Experiments

4.1 Selected Test Problem

D^2MOPSO is tested on 27 (5 constrained and 22 unconstrained) standard MOPs. The selected test problems cover diverse MOPs with convex, concave, connected and disconnected PFs, with two and three optimisation objectives. These problems were frequently used to verify the performance of several algorithms in the field of multi-objective optimisation (Nebro et al. (2008); Coello Coello et al. (2007); Li and Zhang (2009); Deb et al. (2002); Reyes-Sierra and Coello Coello (2005); Al Moubayed et al. (2011, 2010); Martínez and Coello Coello (2011)).

The following unconstrained bi-objective problems are selected: Shaffer (Deb and Agrawal (1994)), Fonseca (Fonseca and Fleming (1998)), Kursawe (Kursawe (1991)) in addition to the bi-objective version of WFG toolkit (WFG1-8 and WFG9) proposed in (Huband et al. (2005)). For three-objective problems, the following MOPs are used: Viennet2 and Viennet3 (Vlennet et al. (1996)), in addition to the DTLZ family (DTLZ1-6 and DTLZ7) proposed in (Deb et al. (2005)), which cover scalable MOPs with the number of decision variables of 7, 12, 12, 12, 12, 12, and 22 respectively.

To cover constrained bi-objective MOPs: three bi-constraints problems(Srinivas (Srinivas and Deb (1994)), Constr_Ex (Deb et al. (2002)) and Tanaka (Tanaka et al. (1995))) are used in addition to the six- and eleven-constraint problems Osyczka2 (Osyczka and Kundu (1995)) and Golinski (Kurpati et al. (2002)) respectively. A three-objectives three-constraint problem (Viennet4 (Vlennet et al. (1996))) is also examined.

4.2 Experimental Setup

D^2MOPSO is compared to MOEA/D (Li and Zhang (2009)), dMOPSO (Martínez and Coello Coello (2011)) and OMOPSO (Reyes-Sierra and Coello Coello (2005)) [1].

Thirty independent runs are performed for each test problem. For the bi-objective problems, 300 iterations per run and 150 particles per generation are used for all algorithms. For the three-objective problems, 600 iterations and 600 individuals are used. All algorithms under comparison adopt real encoding, perform the same number of objective function evaluations and use the same aggregation function with $\theta = 5$.

MOEA/D uses the differential evolution crossover (DE) (probability = 1.0 and differential weight = 0.5), polynomial mutation (probability = 1/number of decision variables), the mutation distribution index is equal to 20, and the neighbourhood size is set

[1] jMetal Framework (Durillo and Nebro (2011)) is used to implement MOEA/D and OMOPSO. dMOPSO implementation was provided by the authors.

to 30.

dMOPSO sets the age threshold to 2; C_1, C_2 are assigned random values in the range [1.2, 2.0]. It uses a global set of size N, where N is the swarm size (the number of λ vectors): $N = 150$ for bi-objective problems, and $N = 600$ for three-objective ones.

OMOPSO uses turbulence probability of 0.5. C_1, C_2 were set to random values in the range [1.5, 2.0], ϵ-crowding archive with ϵ=0.0075 and leaders' archive of size N.

Both OMOPSO and dMOPSO set r_1, r_2 to random values in $[0, 1]$, and w to a random value in [0.1, 0.5]. [2]

D^2MOPSO uses the parameters explained in the previous section with AS equals to 100 for the bi-objective problems and to 300 for the three-objective problems.

Algorithm 4 Leaders' Archive Update

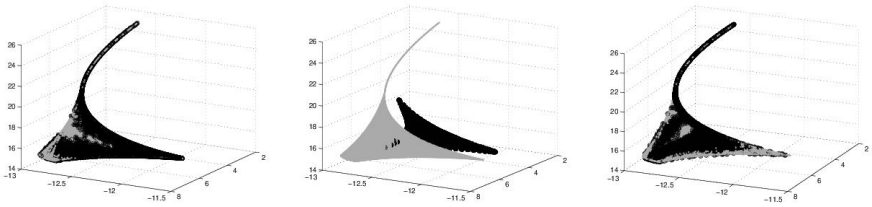
```
for i = 1 to Size do
  if breachConst(S) & breachConst(particle_i) then
    if constraints(S) > constraints(particle_i) then
      remove particle_i
    else
      if constraints(S) < constraints(particle_i) then
        break
      else
        if constraints(S) > constraints(particle_i) then
          remove particle_i
        else
          break
        end if
      end if
    end if
  else
    if !breachConst(S) & breachConst(particle_i) then
      remove particle_i
    else
      break
    end if
  end if
end for
if valid(S) then
  add S to archive
end if
```

4.3 Performance Metrics

To validate our approach, three indicators (Talbi (2009)), which estimate the convergence and diversity of the solutions, are used.

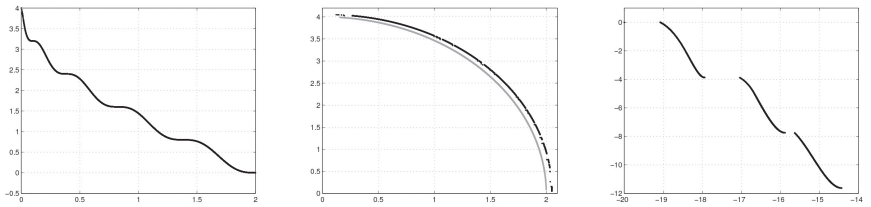
The inverted generational distance, I_{IGD}, (Van Veldhuizen and Lamont (1998)) measures the uniformity of distribution of the obtained solutions in terms of dispersion and extension. The average distance is calculated for each point of the actual PF,

[2]The values are chosen according to recommendations by the algorithms' authors.

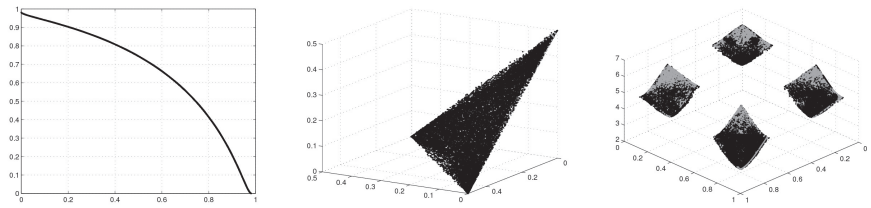


(a) Viennet4: D^2MOPSO (b) MOEA/D (c) OMOPSO

Figure 3: Plot of the non-dominated solutions with the lowest IGD values in 30 runs of D^2MOPSO, MOEA/D and OMOPSO for solving Viennet4.

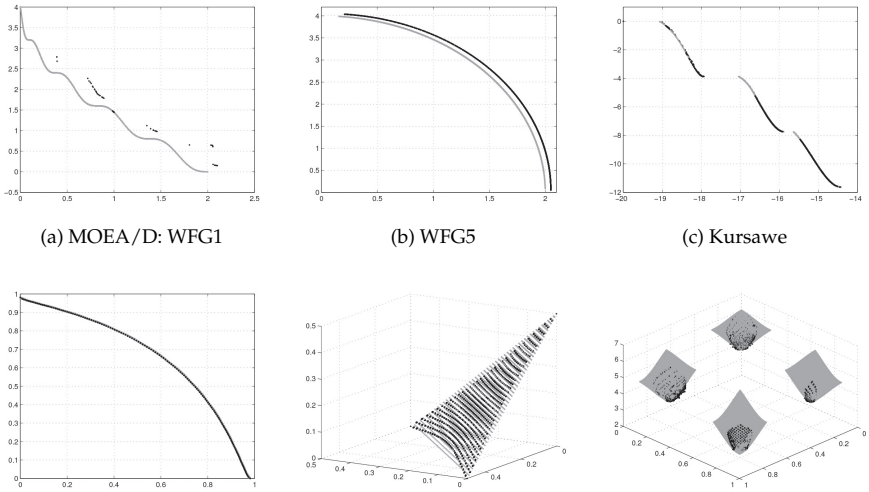


(a) D^2MOPSO: WFG1 (b) WFG5 (c) Kursawe



(d) Fonseca2 (e) DTLZ1 (f) DTLZ7

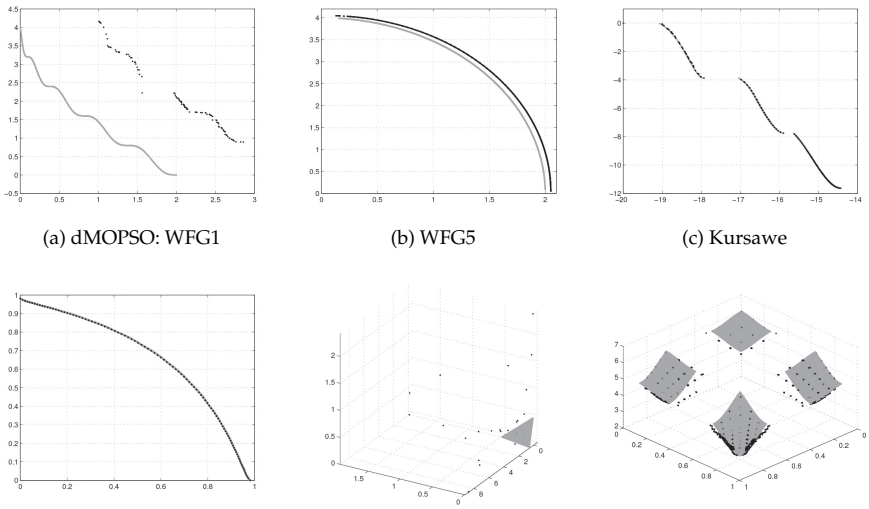
Figure 4: Plot of the non-dominated solutions with the lowest IGD values in 30 runs of D^2MOPSO.



(a) MOEA/D: WFG1 (b) WFG5 (c) Kursawe

(d) Fonseca2 (e) DTLZ1 (f) DTLZ7

Figure 5: Plot of the non-dominated solutions with the lowest IGD values in 30 runs of $MOEA/D$.



(a) dMOPSO: WFG1 (b) WFG5 (c) Kursawe

(d) Fonseca2 (e) DTLZ1 (f) DTLZ7

Figure 6: Plot of the non-dominated solutions with the lowest IGD values in 30 runs of $dMOPSO$.

Table 1: A comparison among the decomposition-based MOEA under study

	MOEA/D	MOPSO/D	SDMOPSO	dMOPSO	D^2MOPSO
Decomposition	x	x	x	x	x
Dominance	-	-	x	-	x
Mutation	x	x	-	-	-
Memory reinit.	-	-	-	x	-
nbest	x	x	x	-	-
lbest	-	-	-	x	x
Leaders' archive	-	x	x	x	x

denoted as A, and the nearest point of the approximated PF, denoted as B.

$$I_{IGD(A,B)} = \frac{(\sum_{a \in A}(\min_{b \in B} \parallel F(a) - F(b) \parallel^2))^{1/2}}{|A|} \tag{14}$$

The hypervolume indicator, I_{hv}, (Zitzler and Thiele (1998)) measures the volume of the objective space that is dominated by a PF approximation (B). I_{hv} uses a reference point v^* which denotes an upper bound over all objectives. v^* is defined as the worst objective values found in the true PF A (i.e. v^* is dominated by all solutions in A). Using the Lebesgue measure (Λ), I_{hv} is defined as:

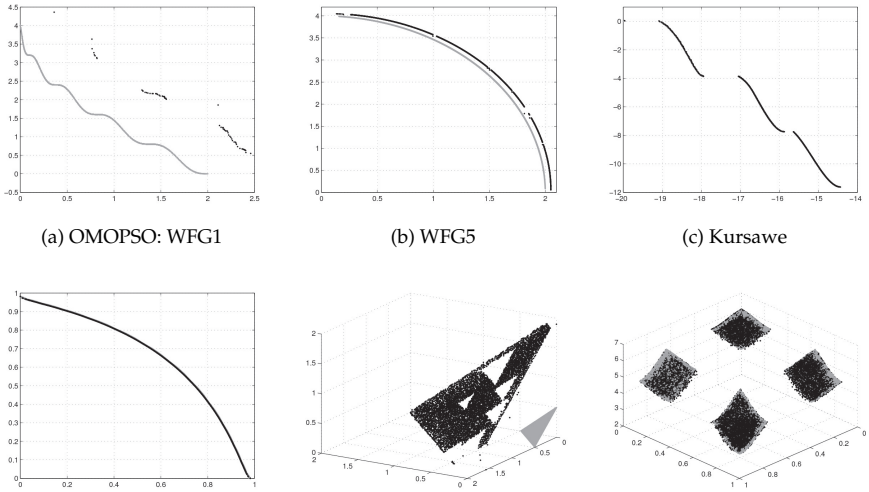
$$I_{hv}(B) = \Lambda\Big(\bigcup_{b \in B}\{x \,|b \prec x \prec v^*|\}\Big). \tag{15}$$

where x is the volume between the origin and b.

The ϵ indicator, I_ϵ, (Zitzler et al. (2003b)) measures the minimum distance which a PF approximation (A) has to be translated in the objective space to weakly dominate the actual PF B. The ϵ-Indicator is defined as:

$$I_\epsilon(A, B) = \min_{\epsilon \in \mathbb{R}}\{\forall b \in B, \exists b'_i - \epsilon \leq b_i, \forall 1 \leq i \leq n\} \tag{16}$$

Table 2 summarizes the main features of the performance measures used in this paper. In order to calculate accurate measures and produce informative plots the objective values are normalized by the true PF, i.e. the minimum and maximum of each objective value of the true PF are used to normalise the objective values of the approximated PF.



(a) OMOPSO: WFG1 (b) WFG5 (c) Kursawe

(d) Fonseca2 (e) DTLZ1 (f) DTLZ7

Figure 7: Plot of the non-dominated solutions with the lowest IGD values in 30 runs of $OMOPSO$.

5 Results and Discussion

5.1 Numeric Comparison

Tables 4, 5, and 6 contain the results of applying I_{hv}, I_{IGD} and I_ϵ respectively to the bi-objective problems, whereas Tables 7, 8, and 9 and Tables 10, 11, and 12 show the results for the three objective and constrained problems respectively. Tables 10, 11, and 12 include results produced using D^2MOPSO, MOEA/D and OMOPSO. The rest of the tables presents results from the four discussed methods: D^2MOPSO, MOEA/D, dMOPSO, and OMOPSO. [3] The results of each problem contain three pieces of information: Med., the median value of the indicator over 30 runs; Iqr., the inter quartile ranges of the indicator value over 30 runs; p., the p-value of a Wilcoxon signed-rank test applied to 30 runs of D^2MOPSO and the corresponding algorithm. A non-parametric statistical test is applied as the values are not guaranteed to follow the Gaussian normal distribution (Shapiro-Wilk normality test shows that some values do follow a Gaussian distribution but others do not).

5.2 Visual Comparison

To visually demonstrate the performance of the different algorithms seven problems were selected: Four bi-objective (Schaffer, Fonesca2, WFG1, and WFG5); two three-objective (DTLZ1, and DTLZ7); and a constrained problem (Viennet4). These problems are selected to demonstrate the output of D^2MOPSO in both cases where it outperforms and under performs (although slightly) the other methods. The approximated Pareto fronts found by D^2MOPSO (PF_{approx} in black with PF_{true} in gray) are plot-

[3] dMOPSO has not been applied to the constrained problems because it is specially designed for non-constrained continuous problems, as stated by the authors, so the comparison would not be fair.

Table 2: Main Features of the Performance Measures

	I_{IGD}	I_{hv}	I_ϵ
Goal	Hybrid	Hybrid	Diversity
Monotone	No	Strict	Mon
Parameter	Ref set	Ref point	Ref set
Min/Max	Min	Max	min

ted in Fig. 4. The results from MOEA/D, dMOPSO, and OMOPSO experiments are illustrated in Fig. 5, Fig. 6 and Fig. 7 respectively.

Although different methods might perform similarly in terms of finding the approximated Pareto front, the number of iterations each algorithm requires to reach this PF may vary. To visually check the convergence of the different methods when solving various problems, the convergence of the four algorithms on the previously selected subgroup of problems is presented. Fig. 8 shows the change of IGD per iteration for each method on the seven selected problems. Fig. 9 depicts similar plots for the change in the hyper-volume indicator, whereas Fig. 10 plots the changes of IGD and hyper-volume for Viennet4.

The Kruskal-Wallis test, which is a nonparametric version of the classical one-way ANOVA and an extension of the Wilcoxon rank sum test to more than two groups, is applied to the unconstrained problems and yielding a value of $p = 0.0092 < 0.05$ (among the four methods), and $p = 0.0066 < 0.05$ when applied on all the problems [4].

There are some anomalies in the presented tables that should be noted. In Table 8, the values of hyper-volume for D^2MOPSO, dMOPSO, and OMOPSO applied to problem $DTLZ3$ are all zero. This is due to the failure of the algorithms to produce a reasonable approximation of the PF. This results in an invalid rank sum test, which is indicated as $--$ in the table. In Table 11, MOEA/D has not succeeded to approximate a reasonable PF for $Osyczka2$ resulting in a zero hyper-volume. Finally, Tanaka has a hyper-volume of 1 for MOEA/D (Table 11) and a negative ϵ value (Table 12), which is impossible because it means the approximated PF dominates the true PF, hence these values are omitted. This can be explained by the fact that MOEA/D could not find any solution that satisfies the problem constraints as it converges to an infeasible solution. For $DTLZ3$, the only method able to approximate the PF is MOEA/D.

5.3 Analysis of Computational Complexity

D^2MOPSO combines the advantages of both decomposition (used by MOEA/D) and dominance (adopted in OMOPSO). By doing so, it capitalizes on the benefits of both techniques. In order for D^2MOPSO to be a viable alternative for the state-of-the-art methods, it should have a similar (or better) computational complexity. In this section we compare the computational complexity of D^2MOPSO to that of MOEA/D, MOPSO/D, SDMOPSO, dMOPSO, and OMOPSO.

MOEA/D updates its population using a set of T neighbors. The newly produced solutions replace one or more individuals in the neighborhood based on the aggregation values. Therefore, for a population of size N the complexity is of the order $O(NT) \sim O(N)$. When MOEA/D uses an archive of size $K \geq N$, then the complexity becomes $O(KN + NT) \sim O(KN)$ as each individual will be compared to all the

[4] dMOPSO is excluded as it does not solve constrained problems

Table 3: A comparison of computational complexity

	MOEA/D	MOPSO/D	SDMOPSO	dMOPSO	OMOPSO	D^2MOPSO
No Arch.	O(N)	-	-	O(N^2)	-	O(N)
Arch.	O(KN)	O(N^2)	O(N^2)	O(KN)	O(KN)	O(KN)

particles in the archive. Similarly, MOPSO/D and SDMOPSO have the complexity of $O(N^2 + NT) \sim O(N^2)$ as $K = N$. The global best set, of size N, in dMOPSO is updated at each iteration using a newly formed set of size $2N$ (as it results from the merge of the global best set with the swarm); hence the computational complexity is $O(2N^2) \sim O(N^2)$ as the aggregation value for each individual must be evaluated against the possible $N\lambda$ vectors. OMOPSO uses the leaders' archive of size N, therefore it requires an algorithm of complexity $O(N^2)$ to be updated. In addition, it uses an $\epsilon-$dominance archive with a size depending on ϵ and the range of objectives. However, assumption can be made that it is of size $K > N$ making the total computational complexity of OMOPSO $O(KN + N^2) \sim O(KN)$.

D^2MOPSO uses the leaders' archive (of size $L \leq N$) which is updated on each iteration. In order to select the global leader for each particle, all solutions in the leaders' archive are checked for the best aggregation value. The complexity would then be $O(2LN) \sim O(N)$. When an external archive (of size $K > N$) is used, the complexity becomes $O(KN+2LN) \sim O(KN)$. The external archive is only used when the method is expected to generate a very large number of non-dominated solutions, as shown in Table 3.

We can conclude from this analysis that D^2MOPSO has similar computational complexity to the other state-of-the-art algorithms.

6 Conclusion

D^2MOPSO is presented as a novel multi-objective particle swarm optimisation algorithm that combines decomposition and dominance. The decomposition simplifies the optimisation problem by transforming it to a set of single-objective problems, whereas dominance facilitates the leaders' archiving process. Decomposition is used to update the personal information and to select the global leaders.

A new archiving technique is also presented, which considers the diversity in both the search and objective spaces. By doing so, the archive helps covering promising regions in both spaces. Crowding distance is used to implement the new archive in this paper, but it can be substituted by any of the other techniques explained in Section 3.1.

An extensive experimentation is carried out covering the different types of PFs. To quantify the performance of D^2MOPSO, three distinct quality measures are used to compare its performance with three state-of-the-art algorithms: a) MOEA/D, a genetic algorithm based decomposition algorithm. b) dMOPSO, a decomposition-based MOPSO. c) OMOPSO, a dominance-based MOPSO. The results are supported by several statistical tests that count for direct and multiple comparison conditions. For unconstrained bi-dimensional problems, D^2MOPSO outperforms the other methods (except for $WFG8$) with respect to I_{IGD}, I_{hv} and I_ϵ. For unconstrained three-dimensional problems, D^2MOPSO performs better in terms of I_{IGD}, I_{hv}, and I_ϵ in all problems except for $DTLZ1$, and $DTLZ3$. For constrained problems, D^2MOPSO outperforms the other algorithms in terms of I_{IGD}. According to I_{hv}, D^2MOPSO under-performs in only one problem: $ConstrEx$. With respect to I_ϵ, D^2MOPSO yields similar results -

outperforming in the case of $Osyczka2$, and $Srinivas$.

In general, D^2MOPSO is demonstrated to be highly competitive to the other algorithms with the advantage of no requirement of parameter tuning and a comparable computational overhead (Section 5.3).

References

Al Moubayed, N., Petrovski, A., and McCall, J. (2010). A novel smart particle swarm optimisation using decomposition. In *PPSN XI*, Lecture Notes in Computer Science. Springer.

Al Moubayed, N., Petrovski, A., and McCall, J. (2011). Clustering-based leaders selection in multi-objective particle swarm optimisation. In *Intelligent Data Engineering and Automated Learning - IDEAL 2011*, volume 6936 of *Lecture Notes in Computer Science*, pages 100–107. Springer Berlin / Heidelberg.

Al Moubayed, N., Petrovski, A., and McCall, J. (2012). d^2mopso: multi-objective particle swarm optimizer based on decomposition and dominance. *Evolutionary Computation in Combinatorial Optimization*, pages 75–86.

Baltar, A. M. and Fontane, D. G. (2006). A generalized multiobjective particle swarm optimization solver for spreadsheet models: application to water quality. In *the twenty sixth annual American geophysical union hydrology days*.

Coello Coello, C. A., Lamont, G., and Veldhuizen, D. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation. Springer, New York, 2nd edition.

Deb, K. and Agrawal, R. B. (1994). Simulated binary crossover for continuous search space. *Complex Systems*, 1(9):115–148.

Deb, K. and Goldberg, D. E. (1989). An investigation of niche and species formation in genetic function optimization. In *3rd Int. Conf. Gen. Alg.*, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: Nsga-ii. *IEEE Trans. on Evolutionary Computation*, 6(2):181–197.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable test problems for evolutionary multiobjective optimization. *Evolutionary Multiobjective Optimization*, pages 105–145.

Durillo, J. and Nebro, A. (2011). jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, 42:760–771.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization (s. forrest, ed.). In *Proceedings of the fifth international conference on genetic algorithms*, volume 1, pages 416–423, San Mateo, CA. Morgan Kaufmann.

Fonseca, C. M. and Fleming, P. J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation. *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans*, 28(1):26–37.

Huband, S., Barone, L., While, L., and Hingston, P. (2005). A scalable multi-objective test problem toolkit. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 280–295. Springer.

Jaishia, B. and Ren, W. (2007). Finite element model updating based on eigenvalue and strain. *Mechanical Systems and Signal Processing*, 21(5):2295–2317.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE.

Kennedy, J., Eberhart, R., and Shi, Y. (2001). *Swarm intelligence*. Morgan Kaufmann, San Francisco.

Knowles, J. and Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8:149–172.

Kurpati, A., Azarm, S., and Wu, J. (2002). Constraint handling improvements for multiobjective genetic algorithms. *Structural and Multidisciplinary Optimization*, 23(3):204–213.

Kursawe, F. (1991). A variant of evolution strategies for vector optimization. *Parallel Problem Solving from Nature*, pages 193–197.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evol. Comput.*, 10(3):263–282.

Li, H. and Zhang, Q. (2009). Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-ii. *IEEE Trans. on Evolutionary Computation*, 13(2):284 –302.

Martínez, S. and Coello Coello, C. (2011). A multi-objective particle swarm optimizer based on decomposition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11. ACM.

Mostaghim, S. and Teich, J. (2003). Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO). In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 26–33. IEEE.

Nasir, M., Mondal, A., Sengupta, S., Das, S., and Abraham, A. (2011). An improved multiobjective evolutionary algorithm based on decomposition with fuzzy dominance. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 765–772. IEEE.

Nebro, A. J., Luna, F., Alba, E., B, A. B., and Dorronsoro, B. (2008). Abyss: Adapting scatter search for multiobjective optimization. *IEEE Trans. on Evolutionary Computation*, 12:439–457.

Osyczka, A. and Kundu, S. (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural and Multidisciplinary Optimization*, 10(2):94–99.

Parsopoulos, K. and Vrahatis, M. (2008). Multi-objective particles swarm optimization approaches. *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*, pages 20–42.

Peng, W. and Zhang, Q. (2008). A decomposition-based multi-objective particle swarm optimization algorithm for continuous optimization problems. In *IEEE International Conference on Granular Computing*, Hangzhou.

Reyes-Sierra, M. and Coello Coello, C. (2005). Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance. In *Evolutionary Multi-Criterion Optimization*, volume 3410 of *Lecture Notes in Computer Science*, pages 505–519. Springer.

Reyes-Sierra, M. and Coello Coello, C. (2006). Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2(3):287–308.

Srinivas, N. and Deb, K. (1994). Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248.

Talbi, E. (2009). *Metaheuristics - From Design to Implementation*. Wiley.

Tanaka, M., Watanabe, H., Furukawa, Y., and Tanino, T. (1995). GA-based decision support system for multicriteria optimization. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1556–1561. IEEE.

Van Veldhuizen, D. A. and Lamont, G. B. (1998). Multiobjective evolutionary algorithm research: A history and analysis. *Air Force Inst. Technol., Dayton, OH, Tech. Rep. TR-98-03*.

Vlennet, R., Fonteix, C., and Marc, I. (1996). Multicriteria optimization using a genetic algorithm for determining a Pareto set. *International Journal of Systems Science*, 27(2):255–260.

Wang, Z., Durst, G., Eberhart, R., Boyd, D., and Miled, Z. B. (2004). Particle swarm optimization and neural network application for QSAR. *Parallel and Distributed Processing Symposium, International*, 10:194.

Zhang, Q. and Li, H. (2007). MOEA/D: A multi-objective evolutionary algorithm based on decomposition. *IEEE Trans. on Evolutionary Computation*, 11(6):712–731.

Zhou, A., Qu, B. Y., Li, H., Zhao, S. Z., Suganthan, P. N., and Zhang, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state-of-the-art. *Swarm and Evolutionary Computation*.

Zitzler, E., Laumanns, M., and Bleuler, S. (2003a). A tutorial on evolutionary multiobjective optimization. In *In Metaheuristics for Multiobjective Optimisation*, pages 3–38. Springer-Verlag.

Zitzler, E. and Thiele, L. (1998). Multiobjective optimization using evolutionary algorithmsa comparative case study. In *Parallel problem solving from naturePPSN V*, Lecture Notes in Computer Science, pages 292–301. Springer.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. G. (2003b). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. on Evolutionary Computation*, 7(2):117–132.

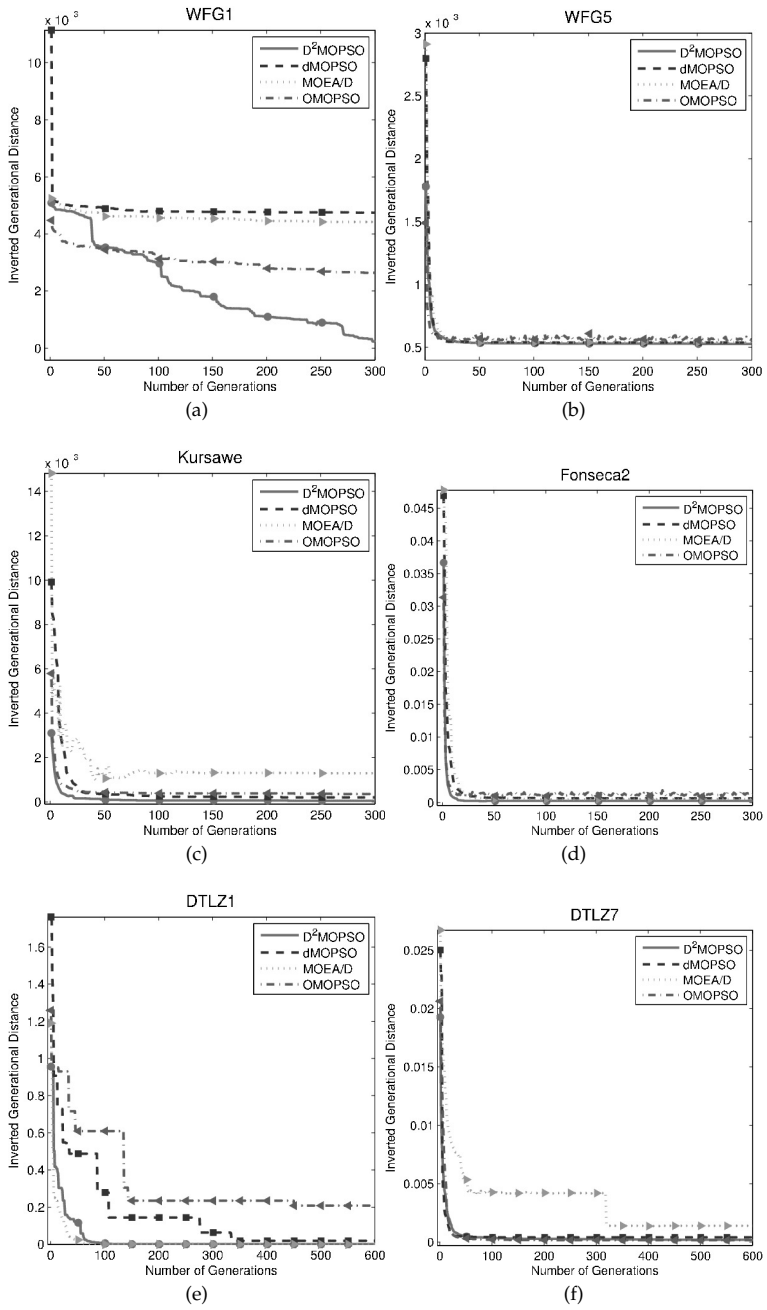


Figure 8: The evaluation of IGD for the four algorithms.

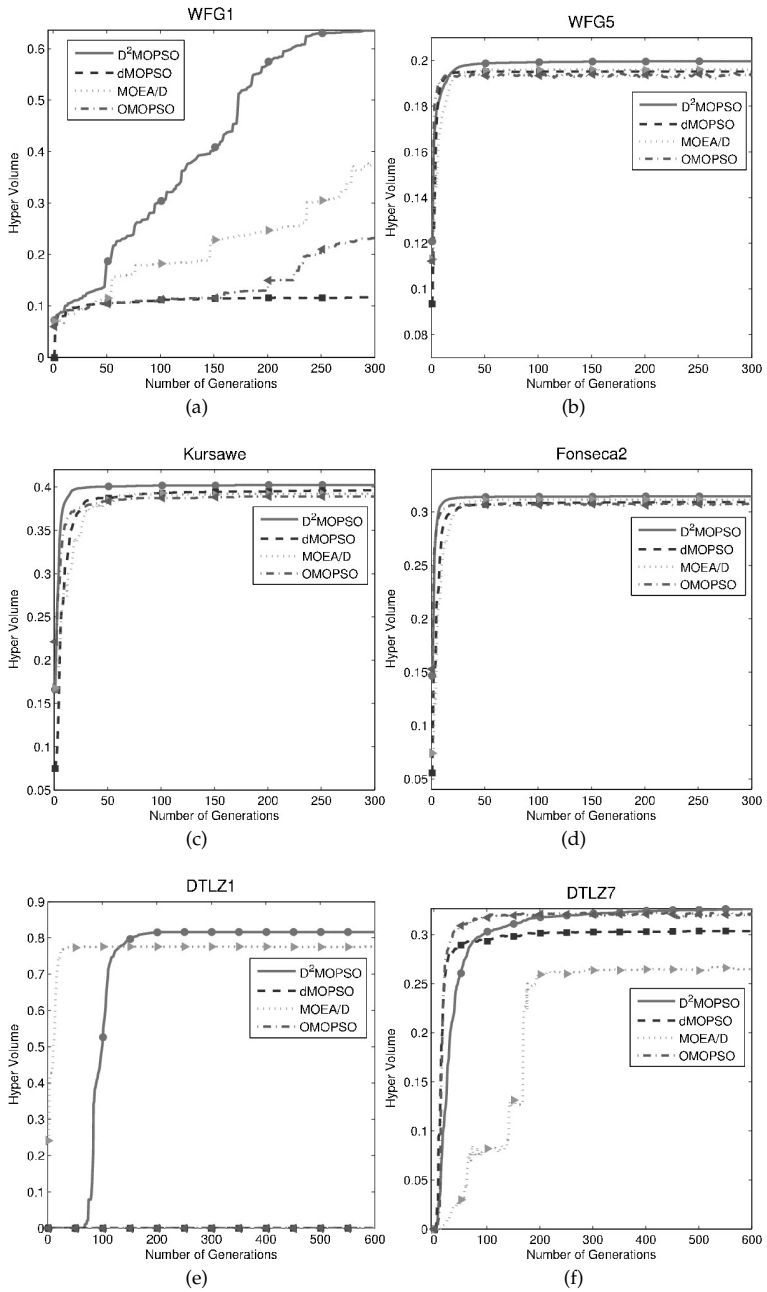


Figure 9: The evaluation of Hyper Volume for the four algorithms.

Table 4: Results of I_{IGD} on unconstrained bi-objective test problems

Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
Fonseca2	Med.	2.41e-004	5.03e-004	6.49e-004	1.20e-003
	Iqr.	1.38e-005	1.89e-006	5.55e-006	1.28e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
Kursawe	Med.	6.74e-005	1.30e-003	2.02e-004	3.78e-004
	Iqr.	1.76e-005	1.51e-005	8.75e-006	1.98e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
Schaffer	Med.	9.88e-005	1.27e-002	6.26e-003	1.81e-004
	Iqr.	1.89e-005	6.73e-003	2.16e-006	1.22e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG1	Med.	1.45e-004	1.86e-003	4.73e-003	3.77e-003
	Iqr.	2.96e-004	3.65e-004	4.75e-005	8.92e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG2	Med.	1.82e-005	1.16e-003	7.94e-004	1.13e-004
	Iqr.	9.42e-006	3.32e-005	9.78e-005	2.42e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG3	Med.	6.84e-004	6.84e-004	1.52e-003	6.84e-004
	Iqr.	1.42e-007	2.51e-008	1.11e-006	7.58e-008
	p	–	4.20e-010	3.02e-011	3.02e-011
WFG4	Med.	4.87e-005	1.95e-004	2.85e-004	2.71e-004
	Iqr.	1.98e-005	4.55e-005	3.91e-005	6.67e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG5	Med.	5.31e-004	5.39e-004	5.39e-004	5.70e-004
	Iqr.	1.48e-006	2.05e-007	2.23e-006	1.16e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG6	Med.	1.53e-005	8.55e-005	1.86e-004	1.98e-004
	Iqr.	1.14e-006	6.44e-007	2.32e-005	3.65e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG7	Med.	1.48e-005	9.24e-005	1.79e-004	1.60e-004
	Iqr.	1.01e-006	3.30e-007	1.45e-005	1.95e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG8	Med.	1.03e-003	8.70e-004	6.80e-004	1.04e-003
	Iqr.	1.37e-004	1.50e-004	1.65e-004	1.23e-005
	p	–	2.88e-006	8.89e-010	3.03e-002
WFG9	Med.	6.26e-005	1.16e-004	1.85e-004	2.22e-004
	Iqr.	9.63e-006	2.52e-005	8.82e-006	3.04e-005
	p	–	3.02e-011	3.02e-011	3.02e-011

Table 5: Results of I_{hv} on unconstrained bi-objective test problems

Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
Fonseca2	Med.	3.14e-001	3.12e-001	3.09e-001	3.07e-001
	Iqr.	1.93e-005	4.01e-007	1.08e-004	5.22e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
Kursawe	Med.	4.04e-001	3.92e-001	3.96e-001	3.90e-001
	Iqr.	4.91e-004	3.44e-004	7.25e-004	9.11e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
Schaffer	Med.	8.33e-001	7.09e-001	8.22e-001	8.32e-001
	Iqr.	2.94e-005	9.82e-002	6.75e-006	7.99e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG1	Med.	6.31e-001	3.81e-001	1.19e-001	1.57e-001
	Iqr.	2.71e-002	5.41e-002	2.56e-003	5.57e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG2	Med.	5.65e-001	5.53e-001	5.55e-001	5.61e-001
	Iqr.	1.64e-004	2.32e-003	1.25e-003	8.64e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG3	Med.	4.44e-001	4.42e-001	2.77e-001	4.42e-001
	Iqr.	5.39e-005	6.79e-006	2.32e-004	1.65e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG4	Med.	2.20e-001	2.10e-001	2.01e-001	2.07e-001
	Iqr.	1.20e-003	3.41e-003	2.38e-003	1.03e-003
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG5	Med.	1.99e-001	1.96e-001	1.95e-001	1.93e-001
	Iqr.	4.50e-005	1.80e-005	8.42e-005	6.89e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG6	Med.	2.13e-001	2.11e-001	2.01e-001	2.07e-001
	Iqr.	8.21e-005	1.44e-005	1.52e-003	6.16e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG7	Med.	2.14e-001	2.11e-001	2.01e-001	2.07e-001
	Iqr.	6.64e-005	5.73e-006	1.47e-003	7.03e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
WFG8	Med.	1.48e-001	1.52e-001	1.65e-001	1.46e-001
	Iqr.	2.67e-003	1.44e-003	7.46e-003	1.07e-003
	p	–	6.53e-008	2.23e-009	3.52e-007
WFG9	Med.	2.41e-001	2.39e-001	2.31e-001	2.32e-001
	Iqr.	9.93e-004	1.99e-003	6.12e-004	9.57e-004
	p	–	4.20e-010	3.02e-011	3.02e-011

Table 6: Results of I_ϵ on unconstrained bi-objective test problems

Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
	Med.	1.88e-003	4.12e-003	6.41e-003	1.05e-002
Fonseca2	Iqr.	1.96e-003	1.47e-005	2.77e-004	3.20e-003
	p	–	9.51e-006	8.48e-009	1.46e-010
	Med.	6.42e-002	3.58e-001	1.18e-001	1.50e-001
Kursawe	Iqr.	2.40e-002	1.58e-002	1.42e-002	1.35e-002
	p	–	3.02e-011	7.39e-011	3.02e-011
	Med.	4.69e-003	7.29e-001	9.03e-002	1.37e-002
Schaffer	Iqr.	1.37e-003	3.43e-001	5.50e-005	2.33e-003
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	8.31e-002	5.85e-001	1.13e+000	1.12e+000
WFG1	Iqr.	1.22e-001	1.14e-001	4.12e-002	1.16e-001
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	3.71e-003	1.14e-001	9.39e-002	2.80e-002
WFG2	Iqr.	3.51e-003	6.12e-001	6.68e-003	6.53e-003
	p	–	3.02e-011	3.02e-011	6.01e-008
	Med.	2.00e+000	2.00e+000	3.00e+000	2.00e+000
WFG3	Iqr.	4.84e-004	7.17e-005	1.89e-004	2.14e-004
	p	–	1.07e-009	3.02e-011	1.34e-005
	Med.	1.45e-002	5.75e-002	6.73e-002	5.67e-002
WFG4	Iqr.	7.28e-003	2.14e-002	1.05e-002	1.09e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	5.20e-002	6.96e-002	7.20e-002	9.00e-002
WFG5	Iqr.	2.53e-004	3.58e-004	4.95e-004	5.80e-003
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	4.05e-003	1.79e-002	5.41e-002	4.22e-002
WFG6	Iqr.	8.72e-004	1.27e-003	1.14e-002	1.13e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	3.63e-003	2.09e-002	4.31e-002	4.57e-002
WFG7	Iqr.	3.62e-004	1.08e-003	3.68e-003	1.07e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
	Med.	5.08e-001	3.93e-001	5.06e-001	5.31e-001
WFG8	Iqr.	1.11e-002	2.01e-001	8.85e-002	1.71e-002
	p	–	4.73e-001	7.62e-001	3.09e-006
	Med.	1.28e-002	3.50e-002	3.93e-002	4.99e-002
WFG9	Iqr.	1.40e-003	1.22e-002	2.52e-003	8.76e-003
	p	–	3.02e-011	3.02e-011	3.02e-011

Table 7: Results of I_{IGD} on unconstrained three-objective test problems

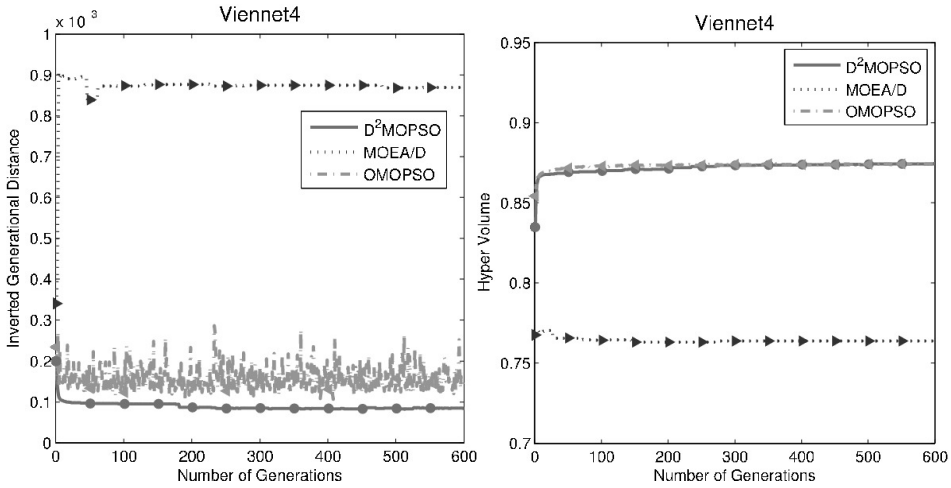
Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
DTLZ1	Med.	4.72e-002	4.75e-004	4.72e-002	1.88e-001
	Iqr.	6.65e-002	1.20e-006	6.65e-002	1.34e-001
	p	–	3.02e-011	1.00e+000	2.03e-007
DTLZ2	Med.	4.19e-005	1.09e-004	1.18e-004	9.25e-005
	Iqr.	3.61e-007	2.94e-008	8.17e-007	7.23e-006
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ3	Med.	3.54e-001	3.87e-004	6.14e-001	1.76e+000
	Iqr.	3.78e-001	7.17e-007	5.07e-001	8.46e-001
	p	–	3.02e-011	4.43e-003	9.92e-011
DTLZ4	Med.	2.09e-004	3.88e-004	4.39e-004	2.71e-004
	Iqr.	1.82e-006	1.03e-006	5.32e-006	5.52e-006
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ5	Med.	1.08e-005	1.80e-004	1.06e-004	1.68e-004
	Iqr.	9.91e-006	9.63e-008	6.55e-006	5.49e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ6	Med.	2.90e-005	1.81e-004	1.80e-004	1.72e-004
	Iqr.	1.20e-005	9.01e-009	9.28e-008	3.83e-005
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ7	Med.	1.95e-004	1.37e-003	4.11e-004	1.47e-004
	Iqr.	1.75e-005	1.52e-005	6.35e-007	3.46e-006
	p	–	3.02e-011	3.02e-011	3.02e-011
Viennet2	Med.	6.91e-005	2.23e-003	1.56e-003	1.08e-003
	Iqr.	1.33e-005	1.24e-006	7.02e-006	4.29e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
Viennet3	Med.	2.02e-003	4.98e-003	4.12e-003	6.85e-004
	Iqr.	2.26e-003	1.39e-006	2.86e-006	5.75e-004
	p	–	3.02e-011	3.02e-011	6.53e-007

Table 8: Results of I_{hv} on unconstrained three-objective test problems

Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
DTLZ1	Med.	8.16e-001	7.76e-001	0.00e+000	0.00e+000
	Iqr.	9.96e-003	3.10e-004	0.00e+000	0.00e+000
	p	–	7.88e-012	1.00e+000	5.58e-003
DTLZ2	Med.	4.63e-001	4.53e-001	4.42e-001	4.61e-001
	Iqr.	1.70e-004	1.09e-005	7.52e-004	2.46e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ3	Med.	0.00e+000	4.49e-001	0.00e+000	0.00e+000
	Iqr.	0.00e+000	4.06e-005	0.00e+000	0.00e+000
	p	–	1.21e-012	–	–
DTLZ4	Med.	4.61e-001	4.49e-001	4.38e-001	4.59e-001
	Iqr.	1.57e-004	3.03e-005	8.09e-004	3.99e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ5	Med.	9.56e-002	8.78e-002	9.11e-002	9.13e-002
	Iqr.	8.36e-005	6.03e-006	3.08e-004	7.40e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ6	Med.	9.46e-002	8.78e-002	8.78e-002	9.08e-002
	Iqr.	1.91e-004	1.32e-007	7.17e-006	5.46e-004
	p	–	3.02e-011	3.02e-011	3.02e-011
DTLZ7	Med.	3.27e-001	2.64e-001	3.04e-001	3.21e-001
	Iqr.	6.88e-004	1.49e-003	4.13e-004	2.14e-003
	p	–	3.02e-011	3.02e-011	3.02e-011
Viennet2	Med.	9.31e-001	8.45e-001	9.03e-001	8.81e-001
	Iqr.	1.24e-004	1.38e-004	3.47e-004	1.41e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
Viennet3	Med.	8.40e-001	8.18e-001	8.31e-001	8.09e-001
	Iqr.	2.95e-004	4.02e-005	4.13e-005	9.01e-003
	p	–	3.02e-011	3.02e-011	3.02e-011

Table 9: Results of I_ϵ on unconstrained three-objective test problems

Problem		D^2MOPSO	**MOEA/D**	**dMOPSO**	**OMOPSO**
DTLZ1	Med.	1.18e+000	3.28e-002	1.18e+000	3.81e+000
	Iqr.	1.27e+000	4.14e-004	1.27e+000	2.09e+000
	p	–	3.02e-011	1.00e+000	3.52e-007
DTLZ2	Med.	1.85e-002	3.31e-002	3.75e-002	1.96e-002
	Iqr.	1.99e-003	8.01e-004	1.65e-003	2.52e-003
	p	–	3.02e-011	3.02e-011	1.68e-004
DTLZ3	Med.	1.48e+001	4.07e-002	2.84e+001	8.91e+001
	Iqr.	1.45e+001	1.55e-003	2.90e+001	4.24e+001
	p	–	3.02e-011	3.77e-004	3.02e-011
DTLZ4	Med.	2.73e-002	4.10e-002	4.48e-002	2.43e-002
	Iqr.	2.27e-003	2.06e-003	1.40e-003	1.89e-003
	p	–	3.02e-011	3.02e-011	3.83e-006
DTLZ5	Med.	2.85e-003	1.56e-002	1.25e-002	1.08e-002
	Iqr.	3.91e-003	2.12e-005	1.11e-003	3.09e-003
	p	–	3.02e-011	2.67e-009	1.56e-008
DTLZ6	Med.	7.54e-003	1.56e-002	1.56e-002	1.14e-002
	Iqr.	9.46e-003	5.03e-009	2.60e-005	2.56e-003
	p	–	1.11e-006	1.11e-006	1.63e-002
DTLZ7	Med.	5.20e-002	1.46e-001	7.31e-002	4.02e-002
	Iqr.	1.00e-002	3.66e-003	1.18e-003	1.33e-002
	p	–	3.02e-011	5.57e-010	7.70e-004
Viennet2	Med.	5.26e-003	6.03e-002	3.52e-002	4.83e-002
	Iqr.	7.28e-004	1.62e-004	4.58e-004	1.99e-002
	p	–	3.02e-011	3.02e-011	3.02e-011
Viennet3	Med.	2.66e-002	1.06e-001	5.22e-002	1.39e-001
	Iqr.	7.39e-003	1.68e-004	1.40e-004	4.38e-002
	p	–	3.02e-011	3.02e-011	3.02e-011



(a) IGD (b) Hyper Volume

Figure 10: The evaluation of the four algorithms for Viennet4.

Table 10: Results of I_{IGD} on constrained test problems

Problem		D^2MOPSO	**MOEA/D**	**OMOPSO**
ConstrEx	Med.	2.42e-003	1.02e-002	2.92e-004
	Iqr.	1.04e-003	1.76e-007	2.40e-005
	p	–	3.02e-011	3.02e-011
Golinski	Med.	9.65e-003	2.65e-002	9.65e-003
	Iqr.	9.90e-003	3.74e-008	3.61e-003
	p	–	3.02e-011	9.82e-001
Osyczka2	Med.	3.98e-003	2.57e-001	4.49e-003
	Iqr.	7.56e-004	2.57e-003	5.63e-003
	p	–	3.02e-011	7.48e-002
Srinivas	Med.	1.05e-005	1.42e-004	1.11e-005
	Iqr.	3.47e-006	1.14e-007	5.52e-006
	p	–	3.02e-011	3.04e-001
Tanaka	Med.	3.36e-004	4.71e-002	3.95e-004
	Iqr.	8.25e-005	0.00e+000	5.27e-005
	p	–	1.21e-012	6.55e-004
Viennet4	Med.	1.74e-004	8.72e-004	1.44e-004
	Iqr.	3.07e-005	4.44e-006	5.16e-005
	p	–	3.02e-011	1.76e-003

Table 11: Results of I_{hv} on constrained test problems

Problem		D^2MOPSO	**MOEA/D**	**OMOPSO**
ConstrEx	Med.	7.12e-001	9.02e-001	7.74e-001
	Iqr.	2.49e-002	2.82e-005	5.02e-004
	p	–	3.02e-011	3.02e-011
Golinski	Med.	9.68e-001	9.96e-001	9.62e-001
	Iqr.	1.45e-003	0.00e+000	1.72e-003
	p	–	5.22e-012	3.02e-011
Osyczka2	Med.	6.34e-001	0.00e+000	7.09e-001
	Iqr.	3.78e-002	0.00e+000	9.66e-003
	p	–	1.21e-012	3.02e-011
Srinivas	Med.	5.45e-001	5.36e-001	5.45e-001
	Iqr.	1.66e-004	1.64e-005	7.42e-005
	p	–	3.02e-011	2.23e-001
Tanaka	Med.	3.04e-001	–	3.00e-001
	Iqr.	4.21e-004	–	2.45e-003
	p	–	–	3.02e-011
Viennet4	Med.	8.70e-001	7.64e-001	8.74e-001
	Iqr.	5.45e-004	6.90e-004	5.09e-004
	p	–	3.02e-011	2.99e-011

Table 12: Results of I_ϵ on Constrained test problems

Problem		D^2MOPSO	**MOEA/D**	**OMOPSO**
ConstrEx	Med.	1.14e-001	2.20e-002	1.51e-002
	Iqr.	5.32e-002	2.09e-005	3.05e-003
	p	–	3.02e-011	3.02e-011
Golinski	Med.	7.24e+000	2.58e+000	3.78e+001
	Iqr.	3.09e+000	0.00e+000	1.05e+001
	p	–	5.22e-012	3.02e-011
Osyczka2	Med.	1.56e+001	9.69e+001	2.58e+001
	Iqr.	3.93e+000	7.02e-001	1.41e+001
	p	–	3.02e-011	3.83e-005
Srinivas	Med.	8.74e-001	2.51e+000	1.28e+000
	Iqr.	8.73e-001	3.25e-002	4.30e-001
	p	–	4.98e-011	5.83e-003
Tanaka	Med.	1.53e-002	–	1.35e-002
	Iqr.	4.59e-003	–	2.37e-003
	p	–	–	1.33e-002
Viennet4	Med.	1.31e-001	3.49e-001	9.78e-002
	Iqr.	2.16e-002	1.21e-003	1.63e-002
	p	–	3.02e-011	3.79e-010