Chapter 8
Dynamic Multi-Objective Optimization Using PSO

Mardé Helbig and Andries P. Engelbrecht

Abstract. Dynamic multi-objective optimization problems occur in many situations in the real world. These optimization problems do not have a single goal to solve, but many goals that are in conflict with one another - improvement in one goal leads to deterioration of another. Therefore, when solving dynamic multi-objective optimization problem, an algorithm attempts to find the set of optimal solutions, referred to as the Pareto-optimal front. Each dynamic multi-objective optimization problem also has a number of boundary constraints that limits the search space. When the particles of a particle swarm optimization (PSO) algorithm move outside the search space, an approach should be followed to manage violation of the boundary constraints. This chapter investigates the effect of various approaches to manage boundary constraint violations on the performance of the dynamic Vector Evaluated Particle Swarm optimization (DVEPSO) algorithm when solving DMOOP. Furthermore, the performance of DVEPSO is compared against the performance of three other state-of-the-art dynamic multi-objective optimization algorithms.

8.1 Introduction

Many problems in the real-world change over time and require more than one goal to be optimized. However, these goals, or objectives, are normally in conflict with one another, where an improvement in one objective results in deterioration of another objective. Therefore, a single solution does not exist and the goal becomes

Mardé Helbig
CSIR Meraka Institute, Scientia, Meiring Naude Road, 0184,
Brummeria, South Africa
e-mail: mhelbig@csir.co.za

Mardé Helbig · Andries P. Engelbrecht
Department of Computer Science, University of Pretoria, 0002,
Pretoria, South Africa
e-mail: engel@cs.up.ac.za

E. Alba et al. (Eds.): Metaheuristics for Dynamic Optimization, SCI 433, pp. 147–188.
springerlink.com

to find the set of optimal trade-off solutions. These problems are called dynamic multi-objective optimization problem (DMOOP). Each DMOOP has a number of objective functions to optimize and each variable within an objective function has a range of values that are valid, referred to as the search space. These bounds of valid values of the decision variable are called boundary constraints.

A multi-swarm algorithm, called Dynamic Vector Evaluated Particle Swarm optimization (DVEPSO) [10], is presented. The effect that various approaches to manage boundary constraints have on the performance of DVEPSO is investigated. Furthermore, DVEPSO is compared against three other state-of-the-art dynamic multi-objective optimization (DMOO) algorithms.

The rest of the chapter's layout is as follows: Section 8.2 presents theory and background information with regard to particle swarm optimization (PSO) and DMOO. The DVEPSO algorithm is presented in Section 8.3, as well as the approaches that can be used to manage boundary constraints. Section 8.4 provides information about the experiments that were run, including the benchmark functions, performance metrics and statistical analysis that were used to measure the performance of the various algorithms. The results that were obtained from the experiments are discussed in Section 8.5. Conclusions about this research are presented in Section 8.6.

8.2 Background

This section presents background information on PSO, as well as the theory on multi-objective optimization (MOO) and DMOO. Furthermore, some issues when solving DMOOP are presented.

8.2.1 Particle Swarm Optimization

Inspired by the social behaviour of bird flocks, Eberhart and Kennedy introduced PSO [15]. The PSO algorithm maintains a swarm of particles, where each particle represents a solution of the optimisation problem. Each particle moves through the search space and its position is updated based on its own experience (cognitive information), as well as the experience of the its neighbours (social information). The particle's position that produced the best solution so far is referred to as its personal best or *pbest*. The position that leads to the best overall solution by all particles in a pre-defined neighbourhood, is called the neighbourhood best or *nbest*. If the neighbourhood is defined as the whole swarm, the neighbourhood best is referred to as the global best or *gbest*.

In general, the PSO algorithm can be described as indicated in Algorithm 8.1.

Every optimisation problem has boundary constraints and therefore a particle should be prevented from drifting outside the boundary constraints of the problem. In some cases it may be beneficiary to allow a particle to move somewhat outside

Algorithm 8.1. PSO Algorithm

```
1.  Create and initialise a swarm
2.      while stopping condition has not been reached
3.          for each particle in swarm do
4.              set pbest
5.          set gbest
6.          for each particle in swarm do
7.              calculate new velocity
8.              calculate new position
```

the bounds when the solution is in close proximity of the bounds. However, once a particle has moved outside the bounds, it should not be allowed to roam outside the boundary constraints indefinitely and should be pulled back within the valid bounds of the decision space. Furthermore, if a particle's position is outside the bounds, the position should not be used as the particle's *pbest*.

According to Chu *et al*, there are three basic boundary handling techniques that are widely used, namely [4]:

- Random, where if a particle moves outside the search space, a random value from a uniform distribution between the lower and upper boundaries of the violating dimension is assigned to the violating dimension of the particle's position.
- Absorbing, where if a particle moves outside the search space, the dimension that is violating the bounds are set to the boundary of that dimension, so that it seems as though the particle has been absorbed by the boundary.
- Reflection, where if a particle moves outside the search space, the boundary acts like a mirror that reflects the projection of the particle's displacement.

Recently, studies have been done on the effect of boundary constraint violation approaches on the performance of PSO. Helwig and Wanka investigated four approaches for managing boundary constraints when solving high-dimensional single-objective optimization problem (SOOP) [13]. Chu *et al.* investigated the effect of the three boundary handling techniques mentioned above for high dimensional SOOP and high dimensional composite SOOP. However, in this chapter various boundary handling approaches are investigated to determine their effect on the performance of VEPSO when solving DMOOP.

8.2.2 Multi-Objective Optimization Theory

When dealing with a MOOP, the various objectives are normally in conflict with one another, i.e. improvement in one objective leads to a worse solution for another objective. Therefore, for MOOP, the definition of optimality has to be adjusted from the one that is used for SOOP. When solving a MOOP the goal is to find a set of trade-off solutions where for each of these solutions no objective can be improved

without causing a worse solution for at least one of the other objectives. These solutions are referred to as *non-dominated solutions* and the set of such solutions is called the *non-dominated set* or *Pareto-optimal set (POS)*. The corresponding objective vectors in the objective space that lead to the non-dominated solutions are referred to as the *Pareto-optimal front (POF)* or *Pareto-front*.

For MOOP, when one decision vector dominates another, the dominating decision vector is considered as a better decision vector. Therefore, only non-dominated decision vectors are included in the POS. Decision vector domination is defined as follows:

Definition 8.1. Decision Vector Domination: A decision vector $\mathbf{x}_1$ dominates another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \prec \mathbf{x}_2$, if and only if

- $\mathbf{x}_1$ is at least as good as $\mathbf{x}_2$ for all the objectives, i.e. $f_m(\mathbf{x_1}) \leq f_m(\mathbf{x_2})$, $\forall m = 1,\ldots,n_m$; and
- $\mathbf{x}_1$ is strictly better than $\mathbf{x}_2$ for at least one objective, i.e. $\exists i = 1,\ldots,n_m : f_m(\mathbf{x_1}) < f_m(\mathbf{x_2})$.

The best decision vectors are called Pareto-optimal, defined as follows:

Definition 8.2. Pareto-optimal: A decision vector $\mathbf{x}^*$ is Pareto-optimal if there does not exist a decision vector $\mathbf{x} \neq \mathbf{x}^* \in F$ that dominates it, i.e. $\nexists m : f_m(\mathbf{x}) < f_m(\mathbf{x}^*)$. If $\mathbf{x}^*$ is Pareto-optimal, the objective vector, $\mathbf{f}(\mathbf{x}^*)$, is also Pareto-optimal.

Together, all the Pareto-optimal decision vectors form the POS, defined as:

Definition 8.3. Pareto-optimal Set: The POS, P^*, is formed by the set of all Pareto-optimal decision vectors, i.e.

$$P^* = \{\mathbf{x}^* \in F \mid \nexists \mathbf{x} \in F : \mathbf{x} \prec \mathbf{x}^*\}, \tag{8.1}$$

The POS contains the best trade-off solutions for the MOOP. The corresponding objective vectors form the POF, which is defined as follows:

Definition 8.4. Pareto-optimal Front: For the objective vector $\mathbf{f}(\mathbf{x})$ and the POS P, the POF, $PF^* \subseteq O$ is defined as

$$PF^* = \{\mathbf{f} = (f_1(\mathbf{x}^*), f_2(\mathbf{x}^*), \ldots, f_{n_m}(\mathbf{x}^*)) \mid \mathbf{x}^* \in P\}, \tag{8.2}$$

Therefore, the POF contains the set of objective vectors that corresponds to the POS, i.e. the set of decision vectors that are non-dominated. The POF can have various shapes, e.g. a convex POF or a concave POF.

8.2.3 Dynamic Multi-Objective Optimisation Theory

Let the n_x-dimensional search space (also referred to as the *decision space*) be represented by $S \subseteq \mathbb{R}^{n_x}$ and the feasible space represented by $F \subseteq S$, where $F = S$ for

unconstrained optimisation problems. Let $\mathbf{x} = (x_1, x_2, \ldots, x_{n_x}) \in S$ represent a vector of the decision variables, i.e. the *decision vector*, and let a single objective function be defined as $f_m \colon \mathbb{R}^{n_x} \to \mathbb{R}$. Then $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_{n_m}(\mathbf{x})) \in O \subseteq \mathbb{R}^{n_m}$ represents an *objective vector* containing n_m objective function evaluations, and O is the *objective space*.

Using the notation above, mathematically, a DMOOP can be defined as:

$$\begin{aligned} &\textit{minimise } \mathbf{f}(\mathbf{x}, \mathbf{W}(t)),\ \mathbf{x} = (x_1, \ldots, x_{n_x}), \mathbf{W}(t) = (\mathbf{w}_1(t), \ldots, \mathbf{w}_{n_m}(t)) \\ &\textit{subject to } \ g_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, n_g \\ &\qquad\qquad h_j(\mathbf{x}) = 0,\ j = n_g + 1, \ldots, n_h \\ &\qquad\qquad \mathbf{x} \in \quad [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x}, \end{aligned} \tag{8.3}$$

where $\mathbf{W}(t)$ is a vector of time-dependent control parameters of an objective function at time t, n_x is the number of decision variables, $\mathbf{x} \in \mathbb{R}^{n_x}$, n_g is the number of inequality constraints, $\mathbf{g}$, n_h is the number of equality constraints, $\mathbf{h}$, and $\mathbf{x} \in [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x}$ refers to the boundary constraints.

Unlike DSOOP with only one objective function, DMOOP has many objective functions. Therefore, in order to solve the DMOOP the goal is to track the POF over time, i.e.

$$PF^*(t) = \{\mathbf{f}(t) = (f_1(\mathbf{x}^*, \mathbf{w}_1(t)), f_2(\mathbf{x}^*, \mathbf{w}_2(t)), \ldots, f_{n_m}(\mathbf{x}^*, \mathbf{w}_{n_m}(t))) \,|\, \mathbf{x}^* \in P\}, \tag{8.4}$$

Farina *et al.* [7] classified dynamic environments for DMOOP into four types, namely:

- **Type I environment** where the POS (optimal set of decision variables) changes, but the POF (corresponding objective function values) remains unchanged.
- **Type II environment** where both the POS and the POF change.
- **Type III environment** where the POS remains unchanged, but the POF changes.
- **Type IV environment** where both the POS and the POF remain unchanged, even though the problem can change.

8.2.4 Dynamic Multi-Objective Optimization Issues

In order to solve a DMOOP, an algorithm has to be able to detect when a change in the environment has occurred and then respond to the change. A change in the environment can be detected through the use of sentry particles [2] where a random number of sentry particles are selected after each iteration. Just before the next iteration is performed, these particles are re-evaluated, and if their current fitness value differs more than a specified value from their fitness value just after the previous iteration, the swarm is alerted that a change has occurred in the environment.

In order to test whether an algorithm can solve DMOOPs, benchmark functions are developed that test an algorithm's ability to manage certain difficulties, such as

local POF and a POF that changes shape (such as from convex to concave) over time. Benchmark functions are representative of typical real-world problems. An approach to reformulate a three-objective optimisation test function to define a dynamic two-objective optimisation problem was presented by Jin and Sendhof [14]. Guan *et al.* [11] presented an approach to create DMOOPs by replacing objective functions with new ones at specific times. DMOOPs based on the static MOO two-objective ZDT functions [26] and the scalable DTLZ functions [5] was presented by Farina *et al.* [7]. Some adaptions to these test functions were proposed in [19, 25].

However, when algorithms' performances are compared against each other, performance measures are required [1, 7, 8, 18]. Two main categories of performance metrics for DMOOP exist, namely metrics that require knowledge about the true POF and metrics that do not require any prior knowledge about the DMOOP. Various performance metrics were developed to measure the performance of an algorithm with regard to two main goals when solving a DMOOP, namely finding solutions that are as close as possible to the true POF and finding a diverse set of solutions.

One of the problems when working with DMOOP is that there are no standard benchmark functions or performance metrics that are used when research on an algorithm's performance is presented.

8.3 Dynamic Vector Evaluated Particle Swarm Optimization Approach

This section discusses the Vector Evaluated Particle Swarm Optimization (VEPSO) algorithm and how it has been adapted to solve DMOOPs. One type of constraint that forms part of a DMOOP is the bounds for each decision variable, also referred to as boundary constraints. This section presents approaches that can be used to manage boundary constraint violations when solving DMOOPs.

8.3.1 Vector Evaluated Particle Swarm Optimization

The Vector Evaluated Particle Swarm Optimization (VEPSO) algorithm, inspired by the Vector Evaluated Genetic Algorithm (VEGA) [21], was introduced by Parsopoulos *et al.* [22]. With VEPSO, each swarm solves only one objective function and then shares its knowledge with the other swarms.

$$v_i^j(t+1) = w^j v_i^j(t) + c_1^j r_1 (y_i^j(t) - x_i^j(t)) + c_2^j r_2 (\hat{y}_i^s(t) - x_i^j(t)) \tag{8.5}$$

$$x_i^j(t+1) = x_i^j(t) + v_i^j(t+1), \tag{8.6}$$

where n represents the dimension with $i = 1, \ldots, n$; m represents the number of swarms with $j = 1, \ldots, m$ as the swarm index; $\hat{y}_i^s$ is the global best of the s-th

swarm with $s \neq j$; c_1^j and c_2^j are the cognitive and social parameters of the j-th swarm respectively; $r_1, r_2 \in [0,1]$; w^j is the inertia weight of the j-th swarm; and $s \in [1,\ldots,j-1,j+1,\ldots,M]$ represents the index of a respective swarm. The index s can be set up in various ways, affecting the topology of the swarms in VEPSO.

In Equation (8.5) the global best of another swarm (indexed by s) is used to update the velocity of the particles of the j-th swarm. In this way the knowledge of the s-th swarm is shared with the j-th swarm.

8.3.2 Dynamic Vector Evaluated Particle Swarm Optimization

When solving DMOOPs, in order to track the changing POF, an algorithm must be able to detect that a change has occurred in the environment and then respond to the change appropriately. The VEPSO algorithm adapted to solve DMOOPs (DVEPSO) is presented in Algorithm 8.2.

Algorithm 8.2. VEPSO for DMOO problems

```
1.  for number of iterations do
2.      check whether a change has occurred
3.      if change has occurred
4.          respond to change
5.          remove dominated solutions from archive
6.      perform iteration
7.      if new solutions are non-dominated
8.          if space in archive
9.              add new solutions to archive
10.         else
11.             remove solutions from archive
12.             add new solutions to archive
13.     select sentry particles
```

The default configuration of DVEPSO algorithm that is used for this research is as follows:

- Each swarm has 20 particles.
- The non-dominated solutions found so far is stored in an archive and the archive size is set to 100.
- If a particle's new position is non-dominant with regard to its current *pbest*, one of these two positions is randomly selected as the particle's new *pbest*.
- If a particle's new position is non-dominant with regard to the swarm's current *gbest*, one of these two positions is randomly selected as the swarm's new *gbest*.
- Sentry particles are used for change detection (refer to lines 2 and 13 in Algorithm 8.2).
- If a change has been detected, 30% of the particles of the swarm(s) whose objective function changed is re-initialised (refer to line 4 in Algorithm 8.2).

The non-dominated solutions in the archive is re-evaluated and the solutions that have become dominated are removed from the archive (refer to line 5 in Algorithm 8.2).
- If the archive is full, the distance between the solution and the other non-dominated solutions in the archive is calculated, and the one with the lowest average distance is removed. This ensures that a solution from a crowded region in the found POF is removed (refer to line 11 in Algorithm 8.2).
- For knowledge sharing between the various swarms, a ring topology is used. Therefore, s in Equation (8.5) is selected using

$$s = \begin{cases} M & \text{for } j = 1 \\ j-1 & \text{for } j = 2,\ldots,M, \end{cases} \tag{8.7}$$

The next section discusses approaches that can be followed to appropriately respond to a violation of the boundary constraints.

8.3.3 Management of Boundary Constraints

This section presents the various approaches that are used in the experiments to manage boundary constraint violations. Below, $\mathbf{x}_{max}$ and $\mathbf{x}_{min}$ refer to the upper bounds and lower bounds of the decision variables of the DMOOP respectively.

The following approaches to handle boundary constraints are investigated to determine their effect on the performance of DVEPSO when solving DMOOPs:

8.3.3.1 Clamping Approach

With the *clamping* approach, any particle that violates a specific boundary of the search space is placed on or close to the violated boundary of the search space [20]. This approach is used for the default configuration of DVEPSO as discussed in Section 8.3.2. Mathematically, clamping is defined as:

$$\begin{aligned} &\text{if } \mathbf{x}(t+1) > \mathbf{x}_{max} \text{ then } \mathbf{x}(t+1) = \mathbf{x}_{max} - \varepsilon \\ &\text{if } \mathbf{x}(t+1) < \mathbf{x}_{min} \text{ then } \mathbf{x}(t+1) = \mathbf{x}_{min} \end{aligned} \tag{8.8}$$

with ε a very small positive number.

8.3.3.2 Deflection Approach

With the deflection approach, if a particle moves outside the bounds of the search space, the velocity's direction of the violated dimension is inverted, thereby causing a bouncing effect of the bounds. Mathematically, the deflection approach is defined as:

$$\begin{aligned}
&\text{if } x_i(t+1) > x_{max^i} \text{ then } x_i(t+1) = x^i_{max} - (x_i(t+1) - x^i_{max})\%(x^i_{max} - x^i_{min}) \text{ and} \\
&\qquad\qquad v_i(t+1) = -v_i(t) \\
&\text{if } x_i(t+1) < x_{min^i} \text{ then } x_i(t+1) = x^i_{min} + (x^i_{min} - x_i(t+1))\%(x^i_{max} - x^i_{min}) \text{ and} \\
&\qquad\qquad v_i(t+1) = -v_i(t), \qquad (8.9)
\end{aligned}$$

where x_i, x^i_{min} and x^i_{max} are the i-th dimension of $\mathbf{x}$, $\mathbf{x}_{max}$ and $\mathbf{x}_{min}$ respectively.

8.3.3.3 Per Element Re-initialization Approach

With *per element re-initialisation*, if a particle moves outside the search space, each dimension of the particle's position that violates the boundary constraint is re-initialized to a random valid value [20]. Therefore, the dimensions of the position that is valid remain the same. Mathematically, per element re-initialization is defined as:

$$\begin{aligned}
&\text{if } x_i(t+1) > x^i_{max} \text{ then } x_i(t+1) = rand(x^i_{min}, x^i_{max}) \\
&\text{if } x_i(t+1) < x^i_{min} \text{ then } x_i(t+1) = rand(x^i_{min}, x^i_{max}), \qquad (8.10)
\end{aligned}$$

8.3.3.4 Periodic Approach

The *periodic approach* is similar to the deflection approach. However, if a particle's position violates the upper boundary for a specific dimension, it is placed near the lower boundary for that dimension and vice versa [24]. Mathematically, the periodic approach is defined as:

$$\begin{aligned}
&\text{if } x_i(t+1) > x^i_{max} \text{ then } x_i(t+1) = x^i_{min} - (x_i(t+1) - x^i_{max})\%(x^i_{max} - x^i_{min}) \\
&\text{if } x_i(t+1) < x^i_{min} \text{ then } x_i(t+1) = x^i_{max} - (x^i_{min} - x_i(t+1))\%(x^i_{max} - x^i_{min}). \qquad (8.11)
\end{aligned}$$

8.3.3.5 Random Approach

The *random* approach re-initializes a particle's position to a valid position within the search space if it violates the boundaries of the search space [13, 24]. Therefore, in contrast to the per element re-initialization approach, all dimensions are re-initialized and not only the violating dimensions. Mathematically, it is defined as:

$$\begin{aligned}
&\text{if } \mathbf{x}(t+1) > \mathbf{x}_{max} \text{ then } \mathbf{x}(t+1) = rand(\mathbf{x}_{min}, \mathbf{x}_{max}) \\
&\text{if } \mathbf{x}(t+1) < \mathbf{x}_{min} \text{ then } \mathbf{x}(t+1) = rand(\mathbf{x}_{min}, \mathbf{x}_{max}). \qquad (8.12)
\end{aligned}$$

8.3.3.6 Re-initialization Approach

With the *re-initialisation approach*, a particle that violates the bounds of the search space has its position re-initialised to a valid position within the search space, its velocity set to zero and its *pbest* set to the particle's new position [20].

8.3.3.7 Unconstrained Approach

With the *unconstrained* approach, no clamping is performed and particles are free to move outside the search space. However, only valid positions are selected as the *pbest* of a particle.

8.4 Experiments

This section describes experiments that were conducted, using benchmark functions and performance metrics discussed in Sections 8.4.1 and 8.4.2, respectively, to test:

- the effect of various approaches to manage boundary constraints on the performance of DVEPSO (refer to Section 8.3.3); and
- the performance of DVEPSO compared to three other state-of-the-art DMOO algorithms (refer to Section 8.4.3).

All experiments consisted of 30 independent runs and each run consisted of 1,000 iterations. For all benchmark functions the severity of change (n_t) is set to 10 and the frequency of change (τ_t) is set to either 5, 25 or 50. This will cause the DMOOP to change every τ_t iteration with n_t distinct steps in time t.

The PSO parameters were set to values that lead to convergent behaviour [23], namely $w = 0.72$ and $c_1 = c_2 = 1.49$.

All codes are implemented in the Computational Intelligence Library (CIlib) [20]. All simulations were run on the Sun Hybrid System's Nehalem System of the Center for High Performance Computing [3]. The SUN Nehalem system has an Intel Nehalem Processor of 2.93 GHz, 2304 CPU cores, 3465 GB of Memory and produces 24 TFlops at peak performance.

8.4.1 Benchmark Functions

This section presents the benchmark functions that were used to test whether the algorithms can track a POF that changes over time. Three functions presented by Farina *et al.* [7] and three functions of Goh and Tan [8] were used. Additionally, two functions that are based on the ZDT3 function of Deb [26] that were adapted to become DMOOPs were used [12]. Below, τ is the generation counter, τ_t is the number of iterations for which t remains fixed, and n_t is the number of distinct steps in t.

$$\mathsf{FDA1} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_\mathsf{I}) = x_i \\ g(\mathbf{x_{II}}) = 1 + \sum_{x_i \in \mathbf{x_{II}}} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \sin(0.5\pi t), \;\; t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathsf{x}_\mathsf{I} \in [0,1]; \;\; \mathbf{x_{II}} = (x_2, \ldots, x_n) \in [-1,1] \end{cases}, \tag{8.13}$$

As suggested by [7], the dimension, n, was set to 20. Function FDA1's values in the decision variable space change over time, but its values in the objective space remain the same. Therefore, it is a Type I DMOOP. It has a convex POF with $POF = 1 - \sqrt{f_1}$.

$$\mathsf{FDA2} = \begin{cases} Minimize: f(\mathsf{x},t) = (f_1(\mathbf{x_I},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathbf{x_I},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathbf{x_I}) = x_i \\ g(\mathbf{x_{II}}) = 1 + \sum_{x_i \in \mathbf{x_{II}}} x_i^2 \\ h(f_1,g) = 1 - \frac{f_1}{g}^{(H(t) + \sum_{x_i \in \mathbf{x_{III}}} (x_i - H(t))^2)^{-1}} \\ where: \\ H(t) = 0.75 + 0.75\sin(0.5\pi t), t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x_I} \in [0,1]; \mathbf{x_{II}}, \mathbf{x_{III}} \in [-1,1] \end{cases}. \tag{8.14}$$

For FDA2 the parameters $|\mathsf{X}_\mathsf{II}|$ and $|\mathsf{X}_\mathsf{III}|$ were set to: $|\mathsf{X}_\mathsf{II}| = |\mathsf{X}_\mathsf{III}| = 15$ (as suggested by [7]). Function FDA2 has a POF that changes from a convex to a non-convex shape. It is a Type III DMOOP, since the values in the objective space change while the values in the decision variable space remain the same. For FDA2, $POF = 1 - f_1^{H(t)^{-1}}$.

$$\mathsf{FDA3} = \begin{cases} Minimize: f(\mathsf{x},t) = (f_1(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathbf{x_I},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathbf{x_I}) = \sum_{x_i \in \mathbf{x_I}} x_i^{F(t)} \\ g(\mathbf{x_{II}}) = 1 + G(t) + \sum_{x_i \in \mathbf{x_{II}}} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ G(t) = |sin(0.5\pi t)| \\ F(t) = 10^{2\sin(0.5\pi t)}, t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x_I} \in [0,1]; \mathbf{x_{II}} \in [-1,1] \end{cases}. \tag{8.15}$$

As suggested by [7], the function parameters $|\mathsf{X}_{\mathsf{II}}|$ and $|\mathsf{X}_{\mathsf{III}}|$ were set to: $|\mathsf{X}_{\mathsf{I}}| = 5$ and $|\mathsf{X}_{\mathsf{II}}| = 25$. Function FDA3 has a convex shaped POF and both the values in the decision variable space, as well as the objective space, change. Therefore, it is called a Type II DMOOP. For FDA3, $POF = (1+G(t))(1-\sqrt{\frac{f_1}{1+G(t)}})$.

$$\mathsf{dMOP1} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_{\mathsf{I}}) = x_i \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{x_i \in \mathbf{x_{II}}} (x_i)^2 \\ h(f_1,g) = 1 - \frac{f_1}{g}^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x_I} = (x_1); \;\; \mathbf{x_{II}} = (x_2, \ldots, x_n) \end{cases} . \tag{8.16}$$

As suggested by [8], the dimension was set to $n = 10$. Function dMOP1 has a convex POF where the values in the objective space change, but the values in the decision space remain the same. Therefore, it is a Type III problem, with $POF = 1 - f_1^{H(t)}$.

$$\mathsf{dMOP2} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_{\mathsf{I}}) = x_i \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{x_i \in \mathbf{x_{II}}} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \frac{f_1}{g}^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \\ G(t) = \sin(0.5\pi t) t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x_I} = (x_1); \;\; \mathbf{x_{II}} = (x_2, \ldots, x_n) \end{cases} . \tag{8.17}$$

The dimension, n, was set 10 (as suggested by [8]). Function dMOP2 has a convex POF where the values in both the decision space and objective space change. Therefore, dMOP2 is a Type II problem, with $POF = 1 - f_1^{H(t)}$.

$$\mathsf{dMOP3} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_{\mathsf{I}},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_{\mathsf{I}}) = x_r \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{x_i \in \mathbf{x_{II}} \setminus \mathsf{x}_r} (x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \sin(0.5\pi t), \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; r = \bigcup(1,2,\ldots,n) \end{cases} . \tag{8.18}$$

As suggested by [8], the dimension, n, was set to 10. Function dMOP3 has a convex POF where the values in the objective space change, but the values in the decision space remain the same, and is therefore a Type I DMOOP, but the spread of the *POF* changes over time. For dMOP3, $POF = 1 - \sqrt{f_1}$.

The following two functions, HE1 and HE2, are based on the function ZDT3 [26], and adapted to be dynamic.

$$\mathsf{HE1} = \begin{cases} Minimize : f(\mathbf{x},t) = (f_1(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_\mathsf{I}) = x_i \\ g(\mathbf{x_{II}}) = 1 + \frac{9}{n-1}\sum_{x_i \in \mathbf{x_{II}}} x_i \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi t f_1) \\ where : \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor ; \ \mathsf{x}_\mathsf{i} \in [0,1] \\ \mathbf{x_I} = (x_1); \ \mathbf{x_{II}} = (x_2,\ldots,x_n) \end{cases} . \tag{8.19}$$

$$\mathsf{HE2} = \begin{cases} Minimize : f(\mathbf{x},t) = (f_1(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t) \cdot h(\mathbf{x_{III}}, f_i(\mathsf{x}_\mathsf{I},t), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathsf{x}_\mathsf{I}) = x_i \\ g(\mathbf{x_{II}}) = 1 + \frac{9}{n-1}\sum_{x_i \in \mathbf{x_{II}}} x_i \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}}^{H(t)} - \frac{f_1}{g}^{H(t)}\sin(10\pi f_1) \\ where : \\ H(t) = 0.75\sin(0.5\pi t) + 1.25; \ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathsf{x}_\mathsf{i} \in [0,1]; \ \mathbf{x_I} = (x_1); \ \mathbf{x_{II}} = (x_2,\ldots,x_n) \end{cases} . \tag{8.20}$$

The dimension, n, was set to 30 (as suggested by [26]) for both HE1 and HE. Both functions have a discontinuous POF. For HE1, $POF = 1 - \sqrt{f_1} - f_1\sin(10\pi t f_1)$, and, for HE2, $POF = 1 - \sqrt{f1}^{H(t)} - f_1^{H(t)}\sin(0.5\pi f_1)$.

8.4.2 Performance Metrics

This section discusses the performance metrics that were used to measure the performance of the various algorithms. Each metric is calculated every time just before a change occurs in the environment. The average of all these values is then calculated for each of the runs. However, if it is unknown when a change will occur, the performance metrics can be calculated over all iterations instead of only the iterations just before a change occurs in the environment.

To determine the algorithm with the best performance for a specific function, the algorithm's overall rank is calculated. For each of the performance metrics the algorithm is ranked according to its performance with regards to the specific metric. The algorithm's average rank value is calculated and then the algorithm is ranked accordingly. Two average ranks are calculated, namely: (a) the sum of all ranks divided by the number of performance metrics (indicated in Tables 8.1-8.8 as R_1); and (b) the sum of all ranks (but the ranks of performance metrics that rely on the true POF, namely HV R, VD and MS counted double) divided by the adjusted number of performance metrics (indicated in Tables 8.1-8.8 as R_2).

8.4.2.1 Spacing

Measuring how evenly the non-dominated solutions are distributed along the found POF (POF^*) can be done using the metric of spacing [9], defined as:

$$\overline{S}^i = \frac{1}{n_c}\sum_{j=1}^{n_c} S_j^i,\ \ S = \frac{1}{n_{PF}}\left[\frac{1}{n_{PF}}\sum_{i=1}^{n_{PF}}(d_i - \overline{d})^2\right]^{\frac{1}{2}},\ \ \overline{d} = \frac{1}{n_{PF}}\sum_{i=1}^{n_{PF}} d_i, \tag{8.21}$$

where n_c is the number of changes that occurred in the environment, n_{PF} is the number of non-dominated solutions found at time t and d_i is the Euclidean distance, in the objective space, between non-dominated solution i and its nearest solution in POF^*.

8.4.2.2 Hypervolume Ratio

The hypervolume (HV) or S-metric [26] computes the size of the region that is dominated by a set of non-dominated solutions, based on a reference vector. According to Li *et al.*, comparing the HV averaged over a number of runs may not be as meaningful when dealing with dynamic environments [17]. Therefore, they suggest using the HV ratio (HV R) to overcome this problem, since the HV of the found POF (POF^*) is computed in relation to the HV of the true POF (POF) [17]. Mathematically, HVR is defined as:

$$HVR = \frac{1}{n_c}\sum_{i=1}^{n_c} HVR(t),\ \ HVR(t) = \frac{HV(POF^*(t))}{HV(POF(t)}. \tag{8.22}$$

Prior knowledge about POF is required to calculate the HVR, POF and the value of the metric will depend on the distribution of sampling points on POF and the selection of the reference vector. For this research the reference vector is selected as the maximum value for each objective.

8.4.2.3 Accuracy

A measure of accuracy that measures the quality of the solutions as a relation between the HV of POF^* and the maximum HV that has been found so far was introduced by Cámara *et al.* [1]. Mathematically, it is defined as:

$$acc = \frac{1}{n_c}\sum_{i=1}^{n_c} acc(t), \;\; acc(t) = \frac{HV(POF^*(t))}{HV_{max}(POF^*(t))}, \tag{8.23}$$

8.4.2.4 Stability

The effect of the changes in the environment on the accuracy (*acc* defined above) of the algorithm can be measured by the measure of stability that was introduced by Cámara *et al.* [1]. Mathematically, stability is defined as:

$$stab = \frac{1}{n_c}\sum_{i=1}^{n_c} stab(t), \;\; stab(t) = max\{0, acc(t-1) - acc(t)\} \tag{8.24}$$

8.4.2.5 Variable Space Generational Distance

The static generational distance (GD) metric was adapted for dynamic environments by Goh and Tan [8]. It measures the distance between POF^* and POF, i.e. the proximity of POF^* to POF. The variable space GD (VGD) metric calculates the GD just before a change occurs in the environment, and is mathematically expressed as:

$$\begin{aligned} VD(t) &= \frac{1}{\tau}\sum_{t=0}^{\tau} VD(t)I(t) \\ VD(t) &= \frac{1}{n_{POF^*(t)}}\sqrt{n_{POF^*(t)}\sum_{i=1}^{n_{POF^*(t)}} d_i(t)^2} \\ \mathrm{I(t)} &= \begin{cases} 1, \text{ if } t\%\tau_t = 0 \\ 0, \text{ otherwise} \end{cases}, \end{aligned} \tag{8.25}$$

where $n_{POF(t)^*}$ is the number of non-dominated solutions in POF^* at time t and d_i is the Euclidean distance between the i-th solution of POF^* and the nearest solution solution of POF. Goh and Tan calculate d_i in the decision space [8]. However, for this research it is calculated in the objective space.

8.4.2.6 Maximum Spread

Goh and Tan adapted the maximum spread (MS) metric for dynamic environments [8]. MS measures how well POF^* covers the POF, i.e. how well the

non-dominated solutions of POF^* are spread along POF. MS for dynamic environments calculates the MS just before a change occurs in the environment, and is defined mathematically as:

$$
\begin{aligned}
MS(t) &= \frac{1}{\tau}\sum_{t=0}^{\tau} MS(t)I(t) \\
MS(t) &= \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left[\frac{\min\left[\overline{POF_i^*}(t),\overline{POF_i}(t)\right]-\max\left[\underline{POF_i^*}(t),\underline{POF_i}(t)\right]}{\overline{POF_i}(t)-\underline{POF_i}(t)}\right]} \\
\mathrm{I(t)} &= \begin{cases} 1, \text{ if } t\%\tau_t = 0 \\ 0, \text{ otherwise} \end{cases}
\end{aligned}
\tag{8.26}
$$

where M is the number of objectives, $n_{POF(t)^*}$ is the number of non-dominated solutions in POF^* at time t, $\overline{POF_i^*}$ and $\underline{POF_i^*}$ refer to the maximum and minimum of the i-th objective of non-dominated solutions in POF^* and $\overline{POF_i}$ and $\underline{POF_i}$ refer to the maximum and minimum of the i-th objective of non-dominated solutions in POF respectively.

8.4.3 Comparison

The performance of DVEPSO is compared against three those of the other state-of-the-art DMOO algorithms, namely:

- DNSGA-II-A algorithm, an NSGA-II algorithm adapted for DMOO and proposed by Deb *et al.* [6]. If a change in the environment is detected, a percentage of individuals are randomly selected and replaced with newly created individuals.
- DNSGA-II-B algorithm, an NSGA-II algorithm that selects a percentage of individuals randomly and replaces them with individuals that are mutated from existing individuals when a change is detected. DNSGA-II-B was proposed by Deb *et al.* [6].
- dCOEA algorithm, a dynamic competitive-cooperative coevolutionary algorithm proposed by Goh and Tan [8].

The source code of the dCOEA algorithm was obtained from the first author of [8]. The source code of the static NSGA-II algorithm was obtained from [16] and was adapted for DMOO according to [6].

8.4.4 Statistical Analysis

A Kruskal-Wallis test was performed for each function for each τ_t to determine whether there is a difference in performance with respect to the performance metrics. If this test indicated that there was a difference, pairwise Mann-Whitney U tests were performed.

8.5 Results

This section discusses the results that were obtained from the experiments. The values of the performance metrics that were obtained, are presented in Tables 8.1- 8.8. In all tables, $DVEPSO_c$, $DVEPSO_d$, $DVEPSO_{pe}$, $DVEPSO_p$, $DVEPSO_r$, $DVEPSO_{re}$ and $DVEPSO_u$ refer to the clamping, deflection, per element re-initialisation, periodic, random, re-initialisation and unconstrained approaches respectively (refer to Section 8.3.3 for the definitions of these approaches).

8.5.1 Managing Boundary Constraints

This section discusses the results that were obtained by the various boundary constraint management approaches. The values of the performance metrics are presented in Tables 8.1- 8.8.

When comparing the POF that was found by the various approaches to the true POF, the VD and MS metrics provide a good indication of the algorithms' performance. These tables show that for a change frequency of 10, $DVEPSO_c$, $DVEPSO_{pe}$ and $DVEPSO_d$ obtained the best overall VD value for two, one and one function(s) respectively, $DVEPSO_p$, $DVEPSO_u$ and $DVEPSO_{re}$ each obtained the best MS value for one function and $DVEPSO_r$, $DVEPSO_d$ and $DVEPSO_u$ obtained the best rank over all performance measures for one, two and two function(s) respectively.

For a change frequency of 25, $DVEPSO_r$ obtained the best overall VD value for three functions, $DVEPSO_p$, $DVEPSO_d$ and $DVEPSO_{pe}$ each obtained the best MS value for one function, and $DVEPSO_{pe}$, $DVEPSO_c$, $DVEPSO_u$ and $DVEPSO_{re}$ each obtained the best overall rank for one function.

For a change frequency of 50, $DVEPSO_{pe}$, $DVEPSO_u$ and $DVEPSO_r$ obtained the best VD value for two, one and one function(s) respectively, $DVEPSO_u$ and $DVEPSO_r$ obtained the best MS value for one function each and $DVEPSO_u$ obtained the best overall rank for three functions.

Figure 8.1 illustrates the found POF of the various boundary handling approaches for FDA2. Figure 8.1 shows that good results were obtained by $DVEPSO_c$, $DVEPSO_r$, $DVEPSO_u$ and $DVEPSO_{pe}$, but $DVEPSO_d$ and $DVEPSO_p$ struggled to find the POF.

The results obtained by the various boundary handling techniques for dMOP2 can be seen in Figure 8.2. Good results were obtained by all approaches, but the approximated POFs of $DVEPSO_p$ and $DVEPSO_{pe}$ had a worse spread or coverage than the other DVEPSO approaches.

Table 8.9 presents the overall rank that the various algorithms obtained for each performance measure, as well as their overall rank for the various frequencies of change. Table 8.9 shows that with regard to the various boundary constraint management approaches, for a change frequency of 10 the best overall rank for VD was obtained by $DVEPSO_r$ and $DVEPSO_c$, the best MS rank was obtained by $DVEPSO_r$ and the best overall rank for all DVEPSO approaches was obtained by $DVEPSO_c$. For a change frequency of 25 the best overall rank for VD was obtained by $DVEPSO_{cl}$ and $DVEPSO_r$, the best overall rank for MS was obtained by

Table 8.1 Performance Measure Values for FDA1

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	99.4	**0.00043**	0.99658	0.9967	0.00154	**0.06593**	0.9761	2	2
10	DVEPSO$_d$	99.4	0.00074	0.99361	0.99373	0.00217	0.12731	0.92471	6	6
10	DVEPSO$_{pe}$	99.2	0.00051	0.99589	0.99601	0.00133	0.08515	0.92806	3.5	3.5
10	DVEPSO$_p$	**99.5**	0.00053	0.99538	0.9955	0.00163	0.08932	0.95041	3.5	3.5
10	**DVEPSO$_r$**	**99.5**	0.00043	**0.99701**	**0.99713**	**0.00116**	0.07035	0.94377	**1**	**1**
10	DVEPSO$_{re}$	99.4	0.00053	0.9953	0.99541	0.00143	0.07855	0.89446	5	5
10	DVEPSO$_u$	99.3	0.00077	0.99391	0.99403	0.00191	0.14115	0.91986	7	7
10	DNSGAII-A	22.8	0.00494	0.97425	0.97436	0.00339	0.83219	0.78693	10	10
10	DNSGAII-B	21.1	0.00612	0.95019	0.9503	0.00543	1.13392	1.19478	9	9
10	dCOEA	33.7	0.00132	0.90528	0.90538	0.01328	1.13184	**2.48561**	8	8
25	DVEPSO$_c$	**99.9**	0.0008	0.99857	**0.99858**	0.00034	0.18913	0.91448	3	4
25	DVEPSO$_d$	**99.9**	**0.00042**	0.98439	0.9763	0.00397	0.12891	0.86929	6	8
25	**DVEPSO$_{pe}$**	**99.9**	0.00046	0.99928	0.99016	0.00032	0.12982	0.90767	**1**	**1**
25	DVEPSO$_p$	**99.9**	0.00045	0.98084	0.97189	0.00485	0.10817	0.89605	9	9
25	DVEPSO$_r$	99.8	0.00047	0.99856	0.98944	0.00049	**0.10446**	0.90257	4.5	3
25	DVEPSO$_{re}$	**99.9**	0.00057	0.99922	0.9901	0.00035	0.13211	0.86428	4.5	5
25	DVEPSO$_u$	98.5	0.00068	**1.00377**	0.99409	0.0013	0.24299	0.88969	7	6
25	DNSGAII-A	37.8	0.00056	0.99903	0.98891	**0.00014**	0.29491	0.9446	8	7
25	DNSGAII-B	38.3	0.00046	0.99913	0.98901	**0.00014**	0.28079	0.94903	2	2
25	dCOEA	39.8	0.00053	0.96001	0.95028	0.00428	1.32408	**2.93453**	10	10
50	DVEPSO$_c$	**100.0**	0.00039	0.99865	**0.99866**	0.00035	0.19331	0.93334	6	7
50	DVEPSO$_d$	99.8	0.00048	0.96771	0.96395	0.00616	0.17621	0.87048	10	10
50	DVEPSO$_{pe}$	99.9	0.00044	0.99915	0.99456	0.0004	**0.09639**	0.86153	7	6
50	DVEPSO$_p$	99.9	0.00037	0.97749	0.97285	0.00541	0.14417	0.83086	9	9
50	DVEPSO$_r$	**100.0**	0.00046	0.99888	0.9941	0.00038	0.24311	0.89013	8	8
50	DVEPSO$_{re}$	**100.0**	0.00033	0.99917	0.99439	0.00041	0.1331	0.87969	4	4
50	**DVEPSO$_u$**	99.9	0.00033	**1.00125**	0.9957	0.00126	0.15148	0.91074	**2**	**1.5**
50	**DNSGAII-A**	40.0	0.00032	0.99985	0.99419	$3.016\text{x}10^{-05}$	0.1716	**0.98858**	**2**	**1.5**
50	**DNSGAII-B**	40.0	0.00033	0.99986	0.9942	$\mathbf{2.245\text{x}10^{-05}}$	0.17261	0.98778	**2**	3
50	dCOEA	39.9	**0.00026**	0.99965	0.994	0.00017	0.1515	0.95904	5	5

$DVEPSO_r$, and the approach that ranked the best over all performance measures was $DVEPSO_{cl}$. For a change frequency of 50 the best overall rank for VD was obtained by $DVEPSO_{pe}$, the best overall rank for MS was obtained by $DVEPSO_r$ and the approach that ranked the best over all performance measures was $DVEPSO_u$. It is interesting to note that $DVEPSO_r$ consistently provided the best overall MS value. Furthermore, $DVEPSO_c$ and $DEVPSO_r$ obtained the best rank for VD for change frequencies of 10 and 25. Therefore, for the lower change frequencies of 10 and 25, $DVEPSO_c$ and $DVEPSO_r$ outperformed the other approaches and for a change frequency of 50 $DVEPSO_u$ performed the best of the DVEPSO approaches.

Table 8.2 Performance Measure Values for FDA2

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	63.3	0.00367	0.99525	**0.99191**	0.00049	0.43937	0.87783	7.5	7
10	DVEPSO$_d$	**73.4**	0.00118	0.99533	0.97848	0.00049	0.45824	0.90878	7.5	8
10	DVEPSO$_{pe}$	63.0	0.00391	0.99905	0.98157	0.00029	**0.43234**	0.88916	5	3
10	DVEPSO$_p$	68.5	0.002	0.99846	0.98098	0.00034	0.45147	**0.91258**	3	2
10	DVEPSO$_r$	68.6	0.00372	0.99634	0.9789	0.00043	0.44453	0.90914	6	5
10	DVEPSO$_{re}$	63.3	0.00297	0.99554	0.97812	0.00037	0.45008	0.87382	9.5	9
10	**DVEPSO$_u$**	71.5	0.00283	1.00171	0.98418	0.00019	0.44998	0.90757	**1**	**1**
10	DNSGAII-A	39.4	0.00044	1.0044	0.98681	$9.565\text{x}10^{-06}$	0.71581	0.77096	4	6
10	DNSGAII-B	39.6	**0.00042**	**1.00441**	0.98683	**$9.206\text{x}10^{-06}$**	0.71681	0.77866	2	4
10	dCOEA	38.4	0.00051	1.00209	0.98454	0.00122	0.70453	0.61923	9.5	10
25	DVEPSO$_c$	78.5	0.0023	0.99644	0.99421	0.00037	0.43181	0.86647	7	7
25	DVEPSO$_d$	77.2	0.00204	0.99354	0.98997	0.00058	0.43196	0.86884	9.5	8.5
25	DVEPSO$_{pe}$	76.7	0.00221	0.99882	**0.99493**	0.00024	0.43695	0.85983	4	4
25	DVEPSO$_p$	**79.3**	0.00166	0.99701	0.9893	0.0004	0.4421	**0.89688**	6	4
25	DVEPSO$_r$	78.0	0.00114	0.9968	0.98855	0.00036	**0.42211**	0.87893	2.5	**1**
25	DVEPSO$_{re}$	78.5	0.00251	0.99684	0.98859	0.00028	0.42642	0.82876	9.5	8.5
25	DVEPSO$_u$	76.0	0.00145	1.00077	0.99249	0.00021	0.43903	0.86418	5.0	4
25	DNSGAII-A	39.7	**0.00043**	**1.00314**	0.99484	$7.579\text{x}10^{-06}$	0.72841	0.78969	2.5	6
25	**DNSGAII-B**	39.7	0.00051	**1.00314**	0.99484	**$6.707\text{x}10^{-06}$**	0.7268	0.83159	**1**	2
25	dCOEA	39.9	0.00099	1.00265	0.99436	0.00017	0.74606	0.78319	8	10
50	DVEPSO$_c$	93.7	0.00031	0.99961	0.9979	0.00017	0.50599	0.95397	3	3
50	DVEPSO$_d$	93.3	**0.00028**	0.99491	0.99166	0.00173	0.49882	0.94	4.5	5
50	DVEPSO$_{pe}$	93.1	0.00031	1.001	0.99732	$7.344\text{x}10^{-05}$	0.4994	0.95325	2	2
50	DVEPSO$_p$	**94.0**	0.00031	0.99524	0.99158	0.00161	0.51161	0.93862	9	9
50	DVEPSO$_r$	93.0	0.00032	1.00035	0.99668	0.00012	0.50096	0.92995	8	7
50	DVEPSO$_{re}$	93.7	0.00036	0.99904	0.99537	0.00012	0.49984	0.95716	6.5	4
50	**DVEPSO$_u$**	91.4	0.00031	1.00155	0.99787	$9.68\text{x}10^{-05}$	**0.49669**	**0.95937**	**1**	**1**
50	DNSGAII-A	40.0	0.0005	1.00287	**0.99918**	$2.804\text{x}10^{-06}$	0.67584	0.75404	4.5	6
50	DNSGAII-B	40.0	0.00039	**1.00287**	**0.99918**	**$2.778\text{x}10^{-06}$**	0.67736	0.74332	6.5	8
50	dCOEA	40.0	0.00207	1.00268	0.999	$4.575\text{x}10^{-05}$	0.69043	0.86612	10	10

8.5.2 Comparison

This section discusses the results that were obtained by the various DMOO algorithms. The results are presented in Tables 8.1- 8.8. These tables show that for a change frequency of 10, *dCOEA* and *DNSGAII-A* each obtained the best overall *VD* value for 2 functions and with regard to the *MS* value, *DNSGAII-A* and *dCOEA* obtained the best overall value for two and three functions respectively a change frequency of 25, *dCOEA* obtained the best overall *VD* value for two functions and *DNSGAII-A* and *DNSGAII-B* each obtained the best overall *VD* value for one function; *dCOEA* and *DNSGAII-A* obtained the best *MS* value for two and three

Table 8.3 Performance Measure Values for FDA3

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	**100.0**	0.00109	1.00221	**0.99889**	0.00013	**0.95943**	0.83848	4	**1.5**
10	**DVEPSO$_d$**	**100.0**	0.00084	1.963x10^{+44}	0.00773	0.00334	0.98365	0.82782	**2**	3
10	DVEPSO$_{pe}$	**100.0**	0.00095	1.00169	5.124x10^{-48}	7.62x10^{-52}	0.99045	0.80762	7.5	8
10	**DVEPSO$_p$**	**100.0**	0.00084	5.822x10^{+40}	2.977x10^{-07}	2.96x10^{-07}	1.0308	0.86383	**2**	4
10	DVEPSO$_r$	**100.0**	0.00087	1.00164	5.124x10^{-48}	8.014x10^{-52}	0.99326	0.82882	7.5	7
10	DVEPSO$_{re}$	**100.0**	0.00091	1.0017	5.124x10^{-48}	7.63x10^{-52}	0.99818	0.8472	6	6
10	**DVEPSO$_u$**	**100.0**	0.00081	**4.767x10^{+45}**	0.00068	0.00068	0.97488	0.81679	**2**	**1.5**
10	DNSGAII-A	32.8	0.00318	0.99967	7.171x10^{-50}	2.796x10^{-53}	1.32639	1.09947	9	9.5
10	DNSGAII-B	27.3	0.00498	0.99796	7.158x10^{-50}	5.576x10^{-53}	1.31649	1.18386	10	9.5
10	dCOEA	39.3	**0.00076**	1.00182	7.186x10^{-50}	**1.503x10^{-53}**	1.08503	**1.30535**	5	5
25	**DVEPSO$_c$**	**100.0**	0.00076	1.00045	**0.99981**	2.746x10^{-05}	1.0931	0.95493	**1.5**	2
25	DVEPSO$_d$	**100.0**	0.00087	**7.955x10^{+41}**	0.01251	0.0025	1.14336	1.02693	5	4
25	DVEPSO$_{pe}$	**100.0**	0.00069	1.00037	1.053x10^{-45}	**2.484x10^{-50}**	**1.08436**	0.91634	6.5	6
25	DVEPSO$_p$	**100.0**	0.00071	4.334x10^{+41}	0.00046	0.00045	1.10933	0.96636	4	5
25	DVEPSO$_r$	**100.0**	0.00066	1.00036	1.053x10^{-45}	2.646x10^{-50}	1.11311	0.99296	6.5	7
25	DVEPSO$_{re}$	**100.0**	0.00069	1.00037	1.053x10^{-45}	2.5x10^{-50}	1.10671	0.95784	8	8
25	**DVEPSO$_u$**	**100.0**	0.0008	1.508x10^{+35}	1.588x10^{-10}	1.586x10^{-10}	1.10233	0.97723	**1.5**	**1**
25	DNSGAII-A	38.2	0.00124	1.00039	1.053x10^{-45}	4.373x10^{-50}	1.27408	1.1752	10	9.5
25	DNSGAII-B	39.1	0.0011	1.00041	1.053x10^{-45}	3.612x10^{-50}	1.27814	1.17337	9	9.5
25	dCOEA	39.9	**0.00052**	1.00044	1.053x10^{-45}	3.221x10^{-50}	1.22933	**1.37518**	3	3
50	DVEPSO$_c$	**100.0**	0.00103	1.01768	**0.98517**	0.00231	0.70117	0.98572	6	6
50	DVEPSO$_d$	**100.0**	0.00098	5.573x10^{+41}	0.02758	0.00885	0.68577	0.97358	5	5
50	DVEPSO$_{pe}$	**100.0**	0.00076	1.00645	6.998x10^{-45}	1.587x10^{-47}	0.67082	0.98334	2.5	4
50	DVEPSO$_p$	**100.0**	0.00115	**1.969x10^{+43}**	0.00167	0.00167	0.68958	0.98313	4	2
50	DVEPSO$_r$	**100.0**	0.00077	1.00532	8.548x10^{-47}	2.067x10^{-49}	**0.66911**	0.9844	2.5	3
50	DVEPSO$_{re}$	**100.0**	0.00092	1.00664	8.559x10^{-47}	1.935x10^{-49}	0.70841	0.97215	10	10
50	**DVEPSO$_u$**	**100.0**	0.00088	4.341x10^{+41}	3.674x10^{-05}	3.674x10^{-05}	0.68476	0.98049	**1**	**1**
50	DNSGAII-A	40.0	0.00137	1.02952	8.753x10^{-47}	3.248x10^{-50}	1.15409	**0.99744**	7	7
50	DNSGAII-B	40.0	0.00141	1.02976	8.755x10^{-47}	**2.781x10^{-50}**	1.16742	0.99743	8.5	8
50	dCOEA	40.0	**0.00065**	1.01787	8.654x10^{-47}	2.083x10^{-49}	0.75373	0.9469	8.5	9

functions respectively; and *dCOEA*, *DNSGAII-A* and *DNSGAII-B* obtained the best overall rank for one, two and two functions respectively.

For a change frequency of 50, *dCOEA* and *DNSGAII-B* obtained the best *VD* value for two and three functions respectively; *DNSGAII-A* and *DNSGAII-B* obtained the best *MS* value for four and one function(s) respectively; and *DNSGAII-A* and *DNSGAII-B* obtained the best overall rank for four and three functions respectively.

Figure 8.3 illustrates the found POF of the various DMOO algorithms for FDA2. Figure 8.3 shows that *DVEPSO* was tracking the changing POF well over time, but *DNSGAII-A* and *dCOEA* struggled to track the changing POF once it changed from convex to concave.

Table 8.4 Performance Measure Values for dMOP1

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	**99.9**	0.00407	0.99962	**0.99962**	0.00035	0.26344	0.87907	2	3
10	DVEPSO$_d$	**99.9**	0.00452	0.99821	0.99796	7.232x10^{-05}	0.29477	0.89326	8	9.5
10	DVEPSO$_{pe}$	**99.9**	0.00484	0.9991	0.99885	0.00046	0.29445	0.89736	7	6
10	DVEPSO$_p$	**99.9**	0.00405	0.9983	0.99805	6.998x10^{-05}	0.28384	0.89884	5	5
10	DVEPSO$_r$	**99.9**	0.00431	0.99841	0.99816	0.00083	0.29631	0.90964	9	8
10	DVEPSO$_{re}$	**99.9**	0.00365	0.99921	0.99896	0.00041	0.23362	0.88294	4	4
10	DVEPSO$_u$	**99.9**	0.00386	0.99866	0.99817	0.00086	0.23642	0.86045	6	7
10	**DNSGAII-A**	38.8	0.00577	**0.99991**	0.99933	**3.603x10^{-05}**	0.15212	**0.9834**	**1**	**1**
10	DNSGAII-B	38.7	0.00497	**0.99991**	0.99933	5.904x10^{-05}	0.15351	0.93976	3	2
10	dCOEA	39.8	**0.00045**	0.99582	0.99524	0.00253	**0.03892**	0.86235	10	9.5
25	DVEPSO$_c$	**100.0**	0.00361	0.9936	**0.99343**	0.00148	0.68678	0.76746	4	4
25	DVEPSO$_d$	**100.0**	0.00352	0.99097	0.97202	0.00091	0.77566	0.75222	8	8
25	DVEPSO$_{pe}$	**100.0**	0.00395	0.99877	0.96826	0.00055	0.71365	0.73278	6	6.5
25	DVEPSO$_p$	**100.0**	0.00351	0.99056	0.96029	0.00105	0.70929	0.74939	9	9
25	DVEPSO$_r$	**100.0**	0.00358	0.99347	0.96311	0.00177	0.80396	0.76349	10	10
25	DVEPSO$_{re}$	**100.0**	0.00386	0.99892	0.9684	0.00049	0.72382	0.72943	6	6.5
25	DVEPSO$_u$	**100.0**	0.00361	**1.0082**	0.94919	0.00276	0.72882	0.75882	6	5
25	**DNSGAII-A**	39.3	0.0004	0.9998	0.93468	**7.896x10^{-06}**	0.15351	**0.97874**	**1**	**1**
25	DNSGAII-B	39.3	0.0004	0.99976	0.93464	1.998x10^{-05}	0.13231	0.9755	2	2
25	dCOEA	40.0	**0.0003**	0.99887	0.93381	0.00064	**0.0686**	0.95086	3	3
50	DVEPSO$_c$	**100.0**	0.00136	0.97142	**0.9714**	0.00117	1.43242	0.56964	7	9.5
50	DVEPSO$_d$	**100.0**	0.00146	0.97285	0.91286	0.00368	1.48468	0.58482	9	9.5
50	DVEPSO$_{pe}$	**100.0**	0.00164	0.9977	0.90593	0.00074	1.27847	0.5732	6	6
50	DVEPSO$_p$	**100.0**	0.00112	0.97275	0.88327	0.00248	1.40793	0.6043	8	7
50	DVEPSO$_r$	**100.0**	0.00148	0.97523	0.88553	0.00208	1.4258	0.60255	10	8
50	DVEPSO$_{re}$	**100.0**	0.00194	0.99825	0.90643	0.00068	1.26249	0.56443	5	5
50	DVEPSO$_u$	**100.0**	0.00126	**1.01387**	0.91444	0.00885	1.60305	0.66245	4	4
50	DNSGAII-A	40.0	0.00034	0.99967	0.89645	1.349x10^{-05}	0.11787	0.98323	2	2
50	**DNSGAII-B**	40.0	0.00032	0.99967	0.89646	**1.246x10^{-05}**	0.121	**0.98338**	**1**	**1**
50	dCOEA	40.0	**0.00023**	0.99942	0.89624	0.00021	**0.09572**	0.97838	3	3

Although all DMOO algorithms tracked the changing POF of dMOP1 very well over time, Figure 8.4 shows that *DNSGAII-A* and *dCOEA* struggled to track the changing POF of dMOP2 over time. However, *DVEPSO* had no problem tracking the changing POF of dMOP2. The *VD* value that is obtained by the *DVEPSO* approaches for dMOP1 is high compared to the evolutionary algorithms. The *DVEPSO* approaches find much more solutions than the evolutionary algorithms, and most of the these solutions are on or very close to the true POF. However, a few outlier solutions in the archive of the *DVEPSO* approaches lead to the high *VD* values, even though they have tracked the changing POF.

Table 8.9 presents the overall rank that the various algorithms obtained for each performance measure, as well as their overall rank for the various frequencies of

Table 8.5 Performance Measure Values for dMOP2

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	**99.9**	0.00073	0.99962	**0.99951**	**0.00027**	0.07904	0.97647	2	2
10	**DVEPSO$_d$**	**99.9**	**0.00062**	1.00667	0.98227	0.00042	**0.07402**	0.97937	**1**	**1**
10	DVEPSO$_{pe}$	**99.9**	0.00083	0.99915	0.9732	0.00039	0.09291	0.97288	5	7
10	DVEPSO$_p$	**99.9**	0.00067	1.00603	0.97887	0.00045	0.07467	0.9744	3	3
10	DVEPSO$_r$	**99.9**	0.00076	0.99891	0.97127	0.00047	0.08269	0.97518	6.5	6
10	DVEPSO$_{re}$	**99.9**	0.00067	0.99903	0.97138	0.00046	0.0855	0.97567	4	4
10	DVEPSO$_u$	**99.9**	0.0008	**1.00709**	0.95911	0.00197	0.08911	0.9689	6.5	5
10	DNSGAII-A	33.5	0.00095	0.99321	0.93715	0.00064	0.90415	**1.45643**	8	8
10	DNSGAII-B	28.7	0.00212	0.99216	0.93616	0.00068	1.03746	1.43973	9.5	9.5
10	dCOEA	33.7	0.00112	0.98988	0.93401	0.00213	0.81297	1.40996	9.5	9.5
25	DVEPSO$_c$	**100.0**	0.00078	0.998	**0.9978**	0.00097	0.17631	0.91634	3	4
25	DVEPSO$_d$	**100.0**	0.00085	0.99396	0.96988	0.00188	0.1772	0.93172	6	5
25	DVEPSO$_{pe}$	**100.0**	0.00076	0.99874	0.96992	**0.00056**	0.18783	0.94799	2	2
25	DVEPSO$_p$	**100.0**	0.00085	0.99719	0.96842	0.00163	0.17535	0.91477	6	6.5
25	DVEPSO$_r$	**100.0**	0.00054	0.99767	0.96888	0.00096	**0.17112**	0.93158	4	3
25	**DVEPSO$_{re}$**	**100.0**	0.00079	0.99867	0.96986	0.00064	0.17207	0.93278	**1**	**1**
25	DVEPSO$_u$	**100.0**	0.00099	**1.0045**	0.96771	0.00241	0.18725	0.91586	9	9
25	DNSGAII-A	39.9	0.00043	0.98884	0.95201	0.00101	0.93768	**1.63537**	8	6.5
25	DNSGAII-B	39.9	0.00041	0.98885	0.95203	0.001	0.94214	1.63414	6	8
25	dCOEA	39.8	**0.0004**	0.98775	0.95096	0.00144	0.93822	1.61199	10	10
50	DVEPSO$_c$	**100.0**	**0.00016**	0.97296	**0.97124**	0.00629	0.19285	0.84654	9	10
50	DVEPSO$_d$	**100.0**	0.00017	1.05717	0.71632	0.01254	0.1688	0.85856	2.5	4
50	DVEPSO$_{pe}$	**100.0**	**0.00016**	0.99637	0.62876	0.00077	0.16929	0.85865	4	6.5
50	DVEPSO$_p$	**100.0**	0.00017	1.13016	0.61318	0.01325	0.20012	0.88857	6	5
50	DVEPSO$_r$	**100.0**	**0.00016**	0.98452	0.49138	0.0024	0.16467	0.87944	6	8
50	DVEPSO$_{re}$	**100.0**	0.00018	0.99619	0.4972	0.00065	0.14661	0.85065	6	6.5
50	DVEPSO$_u$	**100.0**	**0.00016**	**1.23115**	0.61117	0.04407	0.15933	0.87335	2.5	2
50	DNSGAII-A	40.0	0.00032	0.99845	0.45172	0.00014	0.15645	0.9955	8	3
50	**DNSGAII-B**	40.0	0.00032	0.99863	0.45181	**0.00012**	**0.14069**	**0.99639**	**1**	**1**
50	dCOEA	39.8	0.00027	0.98953	0.44769	0.00229	0.15248	0.95434	10	9

change. Table 8.9 shows that for a change frequency of 10 the best overall rank for *VD* was obtained by $DVEPSO_c$ and $DVEPSO_r$ and the best overall rank for *MS* was obtained by *DNSGAII-A*. *DNSGAII-B* obtained the best rank over all performance measures and $DVEPSO_c$ obtained the best rank over all performance measures when the measures that use the true POF count more towards the overall rank average.

For a change frequency of 25 the best overall rank for *VD* was obtained by $DVEPSO_{cl}$ and $DVEPSO_r$, the best overall rank for *MS* was obtained by *DNSGAII-A* and the approach that ranked the best over all performance measures was *DNSGAII-B*. For a change frequency of 50 the best overall rank for *VD* was obtained by *DNSGAII-B* and *dCOEA*. The best overall rank for *MS* was obtained by *DNSGAII-A* and the approach that ranked the best over all performance measures was *DNSGAII-A*.

Table 8.6 Performance Measure Values for dMOP3

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	5.1	0.07368	0.9973	0.99735	**0.00045**	1.35206	1.91012	2.5	2
10	DVEPSO$_d$	5.1	0.0789	0.91942	0.91945	0.01367	1.38248	1.93216	10	10
10	DVEPSO$_{pe}$	5.2	0.07884	0.99545	0.99548	0.00108	1.38861	1.93614	4	4
10	DVEPSO$_p$	5.1	0.07866	0.91629	0.91632	0.01296	1.37957	1.92784	9	9
10	DVEPSO$_r$	5.0	0.07815	0.99515	0.99519	0.00125	1.3581	1.90401	8	7
10	DVEPSO$_{re}$	5.1	0.08445	0.99547	0.9955	0.00107	1.38452	1.93409	5	5.5
10	DVEPSO$_u$	5.2	0.07905	0.99403	0.99406	0.00232	1.38136	**1.94846**	6.5	5.5
10	**DNSGAII-A**	36.7	**0.00075**	**0.99744**	**0.99745**	0.00086	0.84372	1.54286	**1**	**1**
10	DNSGAII-B	28.2	0.00146	0.97038	0.97039	0.00558	0.94898	1.27303	6.5	8
10	dCOEA	**36.9**	0.00078	0.99611	0.99612	0.00159	**0.74777**	1.34982	2.5	3
25	DVEPSO$_c$	5.5	0.07858	0.98458	0.98515	0.00178	1.43742	2.08363	7	5
25	DVEPSO$_d$	5.5	0.07596	0.88596	0.88644	0.01572	1.45526	2.12036	9	10
25	DVEPSO$_{pe}$	5.5	0.08177	0.98197	0.9825	0.002	1.48467	**2.14814**	8	8
25	DVEPSO$_p$	5.5	0.07608	0.89117	0.89164	0.01617	1.45777	2.1446	10	9
25	DVEPSO$_r$	5.8	0.07085	0.98163	0.98215	0.00206	1.42254	2.10335	4	4
25	DVEPSO$_{re}$	5.5	0.07123	0.98211	0.98263	0.00191	1.44345	2.12769	3	3
25	DVEPSO$_u$	5.5	0.07527	0.98564	0.98583	0.00289	1.46414	2.10228	6	6
25	DNSGAII-A	**40.0**	0.00038	0.99017	0.99018	0.0014	0.90791	1.6252	2	2
25	DNSGAII-B	**40.0**	**0.00035**	0.9741	0.97411	0.00812	0.89235	1.49494	5	7
25	**dCOEA**	39.9	0.00042	**0.99104**	**0.99106**	**0.00116**	**0.88038**	1.52265	**1**	**1**
50	DVEPSO$_c$	8.9	0.01957	0.99372	0.99612	0.00105	0.60082	0.83513	5	6
50	DVEPSO$_d$	9.6	0.01643	0.86685	0.86849	0.0281	0.5848	0.86028	6	5
50	DVEPSO$_{pe}$	9.3	0.01741	0.99347	0.99535	0.00123	0.59723	0.84383	4	4
50	DVEPSO$_p$	9.0	0.01997	0.86521	0.86588	0.03018	0.60588	0.83987	9	9
50	DVEPSO$_r$	9.2	0.01932	0.99301	0.99365	0.00154	0.58804	0.83986	7	7
50	DVEPSO$_{re}$	8.9	0.02124	0.99359	0.99423	0.00114	0.61199	0.84182	8	8
50	DVEPSO$_u$	8.7	0.02248	0.98684	0.98649	0.00439	0.61941	0.83323	10	10
50	**DNSGAII-A**	**40.0**	0.00032	**0.99982**	**0.99912**	$3.1\text{x}10^{-05}$	0.11419	**0.99401**	**1**	**1**
50	DNSGAII-B	**40.0**	**0.00029**	0.99561	0.99491	0.00422	**0.09471**	0.97185	3	3
50	dCOEA	39.9	**0.00025**	0.99942	0.99871	0.00028	0.12694	0.968	2	2

With regard to the overall rank presented in Table 8.9 over all frequencies of change, the DVEPSO approaches performed the best with regards to *VD* and the dynamic NSGA-II approaches performed the best with regards to *MS* and the overall rank. The DVEPSO approaches obtained the best overall rank for *VD* on eleven occasions, and the dynamic NSGA-II approaches and *dCOEA* on six occasions each. With regards to *MS*, the DVEPSO approaches obtained the highest rank on 8 occasions, the dynamic NSGA-II approaches on eleven occasions and *dCOEA* on 4 occasions. The dynamic NSGA-II approaches obtained the best overall rank on 15 occasions, the DVEPSO approaches on 12 occasions and *dCOEA* on no occasion.

Table 8.7 Performance Measure Values for HE1

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	13.5	0.01173	0.66388	**0.85439**	0.01399	1.55763	0.78202	7	9
10	DVEPSO$_d$	12.6	0.01359	0.73684	0.67688	0.01324	1.51808	0.77148	8	6
10	DVEPSO$_{pe}$	13.1	0.01518	0.68747	0.61666	0.00955	1.522	0.76341	10.0	10
10	DVEPSO$_p$	13.3	0.01196	0.78513	0.70427	0.01191	1.53101	0.77698	4.5	4.5
10	DVEPSO$_r$	14.4	0.0108	0.67781	0.608	0.00835	1.53891	0.76821	6	7.5
10	DVEPSO$_{re}$	13.5	0.01411	0.70286	0.63047	0.01085	1.54639	0.78341	9	7.5
10	DVEPSO$_u$	13.8	0.01429	0.89687	0.7716	0.01334	1.56948	0.78503	4.5	4.5
10	DNSGAII-A	**40.0**	0.00058	**0.96747**	0.80366	0.00827	**0.13607**	**0.8044**	**1**	**1**
10	DNSGAII-B	**40.0**	**0.00033**	0.90325	0.75031	**0.00317**	0.13917	0.42739	2	2
10	dCOEA	28.6	0.00326	0.92802	0.77089	0.01347	0.18269	0.60639	3	3
25	DVEPSO$_c$	19.9	0.0069	0.66264	**0.88732**	0.01299	1.54781	0.77076	6.5	5.5
25	DVEPSO$_d$	24.2	0.00558	0.8044	0.75121	0.02017	1.55831	0.76052	9.5	10
25	DVEPSO$_{pe}$	18.3	0.00837	0.69068	0.63844	0.00961	1.51748	0.75548	9.5	9
25	DVEPSO$_p$	21.1	0.0075	0.83401	0.74319	0.01696	1.53058	0.76397	8	7
25	DVEPSO$_r$	25.0	0.00543	0.68311	0.60386	0.00915	1.57137	0.76668	4.5	5.5
25	DVEPSO$_{re}$	19.9	0.00543	0.71364	0.63085	0.00988	1.57552	0.76542	6.5	8
25	DVEPSO$_u$	17.3	0.009	0.87798	0.77612	0.01192	1.52789	0.77961	4.5	4
25	**DNSGAII-A**	**40.0**	0.00058	**0.96607**	0.85399	0.01143	**0.15803**	**0.79656**	**1**	**1**
25	DNSGAII-B	**40.0**	**0.00038**	0.90938	0.80388	**0.00648**	0.16232	0.47371	2	2
25	dCOEA	39.7	0.0011	0.94994	0.83974	0.0125	0.18167	0.72581	3	3
50	DVEPSO$_c$	34.2	0.00367	0.71493	**0.91246**	0.01146	1.59845	0.76024	5	8.5
50	DVEPSO$_d$	34.7	0.00401	0.87724	0.79065	0.01461	1.56689	0.7584	9	8.5
50	DVEPSO$_{pe}$	29.0	0.00409	0.72615	0.64561	0.00899	1.56256	0.76148	6.5	5
50	DVEPSO$_p$	29.4	0.00489	0.89416	0.79499	0.01309	1.54536	0.75448	10	6.5
50	DVEPSO$_r$	33.4	0.00357	0.71771	0.63811	0.00863	1.5742	0.75946	6.5	10
50	DVEPSO$_{re}$	34.2	0.00342	0.75603	0.67218	0.00934	1.56573	0.76762	4	4
50	DVEPSO$_u$	32.4	0.00416	0.89113	0.7923	0.01105	1.56451	0.75721	8	6.5
50	**DNSGAII-A**	**40.0**	0.00058	**0.96827**	0.86088	0.01083	0.19626	**0.78734**	**1**	**1**
50	DNSGAII-B	**40.0**	**0.00041**	0.91908	0.81715	**0.00778**	0.19108	0.46374	3	3
50	dCOEA	**40.0**	0.00072	0.96564	0.85854	0.00961	**0.18173**	0.78317	2	2

8.5.3 Statistical Analysis

This section discusses the statistical analysis that was done on the performance metrics values. Kruskal-Wallis tests were performed to determine whether there was a statistically significant difference between the values obtained by the various DMOO algorithms for a performance metric for a specific function at a specific τ_t. The p-values that were obtained from the Kruskal-Wallis tests are presented in Tables 8.10- 8.17. In these tables, p-values that are statistically significant are displayed in bold.

When the p-value of the Kruskal-Wallis test indicated that there was a statistically significant difference, Mann-Whitney U tests were performed to determine

Table 8.8 Performance Measure Values for HE2

τ_t	Algorithm	NS	S	HV R	Acc	Stab	VD	MS	R_1	R_2
10	DVEPSO$_c$	24.4	0.01986	0.48235	0.54748	0.01305	1.52451	0.98783	4.5	3.5
10	DVEPSO$_d$	29.8	0.0115	0.46427	0.51349	0.0134	1.55614	1.02734	6.5	7
10	DVEPSO$_{pe}$	27.8	0.0132	0.47364	0.52271	0.01315	1.52684	1.01328	4.5	5
10	DVEPSO$_p$	28.1	0.01271	0.46458	0.51271	0.01325	1.56867	1.01558	10	10
10	DVEPSO$_r$	23.3	0.02176	0.46844	0.51697	0.01302	1.53622	1.024	8.5	8
10	DVEPSO$_{re}$	24.4	0.01208	0.46198	0.50984	0.01338	1.55797	**1.03365**	8.5	9
10	DVEPSO$_u$	23.9	0.01354	0.47324	0.52226	0.01305	1.5348	1.01992	6.5	6
10	DNSGAII-A	**40.0**	0.00062	0.99071	0.94744	0.00213	0.20337	0.91933	2	2
10	**DNSGAII-B**	**40.0**	**0.00061**	**0.99095**	**0.9474**	**0.00206**	**0.20331**	0.92084	**1**	**1**
10	dCOEA	27.4	0.00452	0.9062	0.89176	0.01591	0.23457	0.6925	3	3.5
25	DVEPSO$_c$	**43.5**	0.00478	0.6976	**0.7411**	0.00911	1.50652	0.89599	5	5
25	DVEPSO$_d$	42.7	0.01078	0.98233	0.45845	0.00316	1.5213	**0.90232**	4	4
25	DVEPSO$_{pe}$	34.1	0.01208	0.71057	0.18521	0.00214	1.48753	0.88314	9	9
25	DVEPSO$_p$	39.5	0.00823	0.98768	0.25744	0.00207	1.49003	0.89328	3	3
25	DVEPSO$_r$	32.0	0.01914	0.69406	0.18091	0.00235	1.50608	0.88741	10	10
25	DVEPSO$_{re}$	**43.5**	0.01553	0.74847	0.19509	0.00181	1.51314	0.89228	8	8
25	DVEPSO$_u$	33.6	0.01342	0.9345	0.24358	0.00187	1.52096	0.89767	7	7
25	DNSGAII-A	**40.0**	0.00065	**0.9896**	0.25794	**0.00022**	0.26994	0.88562	2	2
25	**DNSGAII-B**	**40.0**	**0.00061**	**0.9896**	0.25794	**0.00022**	**0.26682**	0.88501	**1**	**1**
25	dCOEA	39.4	0.00131	0.95101	0.24789	0.00204	0.2783	0.74454	6	6
50	DVEPSO$_c$	33.2	0.0145	0.66815	**0.89051**	0.00519	1.75131	1.16669	9	10
50	DVEPSO$_d$	24.4	0.02218	0.9366	0.72392	0.00342	1.71371	1.14846	7	7
50	DVEPSO$_{pe}$	28.0	0.02009	0.68452	0.48548	0.00278	1.71787	1.18569	8	8
50	DVEPSO$_p$	32.3	0.01708	0.94995	0.67374	0.00244	1.73754	1.1905	4	3.5
50	DVEPSO$_r$	20.3	0.01837	0.67025	0.47536	0.00298	1.74609	**1.1989**	10	9
50	DVEPSO$_{re}$	33.2	0.01405	0.72528	0.51439	0.00193	1.72075	1.1981	5	5
50	DVEPSO$_u$	25.9	0.01376	0.86189	0.61128	0.00301	1.71865	1.17589	6	6
50	**DNSGAII-A**	**40.0**	0.00063	**0.9985**	0.70817	**0.00048**	0.19138	0.91808	**1**	**1**
50	DNSGAII-B	**40.0**	**0.00062**	0.99847	0.70815	**0.00048**	**0.17538**	0.91793	2	2
50	dCOEA	**40.0**	0.00146	0.97275	0.68991	0.00346	0.25389	0.81778	3	3

between which DMOO algorithms' performance metric values there were a statistically significant difference. Both the Kruskal-Wallis tests and the Mann-Whitney U tests were performed using the statistical software package R and testing for a confidence level of 95%. Due to a lack of space all results of the Mann-Whitney U tests are not presented. However, Tables 8.18- 8.25 in the appendix present the results of the Mann-Whitney U tests for the VD performance metric. In all these tables "-" indicates that there was no statistically significant difference and "x" indicates that according to the Mann-Whitney U test, there was a statistically significant difference between the specific performance metric values.

Table 8.10 shows that for FDA1 there is a statistically significant difference between almost all of the algorithms for a change frequency of 10 and for almost half

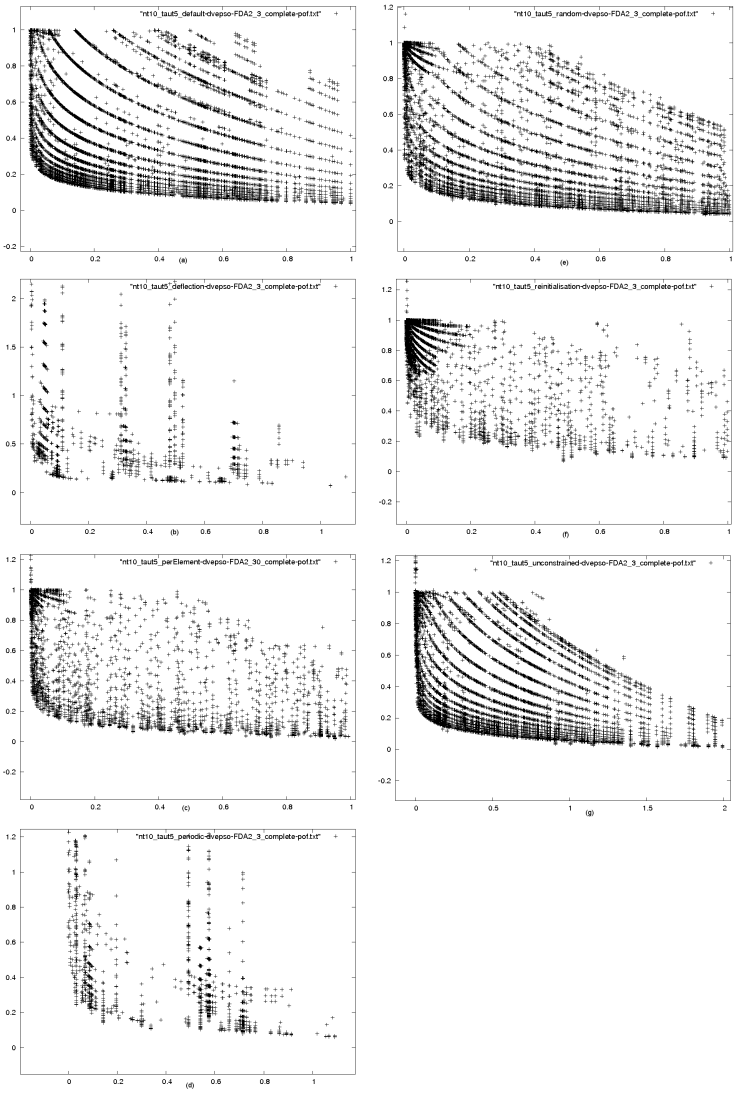

Fig. 8.1 Results of various boundary constraint management approaches solving FDA2, with (a) DVEPSO$_c$, (b) DVEPSO$_d$, (c) DVEPSO$_{pe}$, (d) DVEPSO$_p$, (e) DVEPSO$_r$, (f) DVEPSO$_{re}$ and (g) DVEPSO$_u$. The numbering is from top to bottom on the left, and then from top to bottom on the right.

of the algorithm combinations for a change frequency of 50. However, for a change frequency of 25 there is no statistically significant difference when comparing the evolutionary algorithms against each other, but there is a statistically significant difference for almost all combinations when comparing the evolutionary algorithms against the DVEPSO approaches.

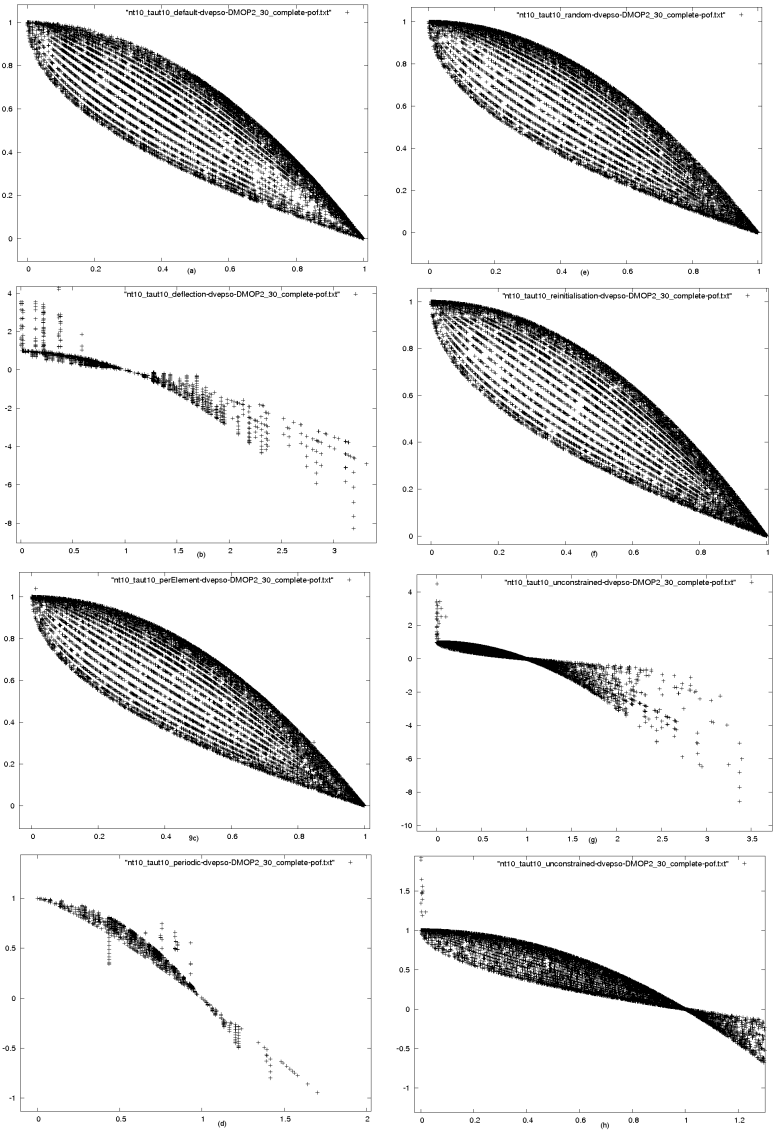

Fig. 8.2 Results of various boundary constraint management approaches solving dMOP2, with (a) DVEPSO$_c$, (b) DVEPSO$_d$, (c) DVEPSO$_{pe}$, (d) DVEPSO$_p$, (e) DVEPSO$_r$, (f) DVEPSO$_{re}$, (g) DVEPSO$_u$. The numbering is from top to bottom on the left, and then from top to bottom on the right.

For FDA2 with a change frequency of 10 and 25, only a few of the DVEPSO approaches have statistically significant differences when compared to other DVEPSO approaches, but almost all combinations of comparisons between DVEPSO

Table 8.9 Overall Ranking of Algorithms

τ_t	Algorithm	R_{NS}	R_S	R_{HVR}	R_{Acc}	R_{Stab}	R_{VD}	R_{MS}	R_{O_1}	R_{O_2}
10	DVEPSO$_c$	**1**	9	8	**1**	8	**1.5**	4	3	**1**
10	DVEPSO$_d$	3	6	9	2	10	9.5	3	10	9
10	DVEPSO$_{pe}$	4.5	7.5	6	3	4	3.5	9	6	6
10	DVEPSO$_p$	4.5	5	7	8	9	3.5	7	9	4
10	DVEPSO$_r$	6	4	10	10	6	**1.5**	2	6	7
10	DVEPSO$_{re}$	2	7.5	5	6.5	3	6.5	10	8	8
10	DVEPSO$_u$	7	10	**1**	5	7	9.5	5.5	6	2
10	DNSGAII-A	10	3	2	4	**1**	6.5	**1**	2	3
10	**DNSGAII-B**	9	**1.5**	3	2	2	5	5.5	**1**	5
10	dCOEA	8	**1.5**	4	9	5	8	8	4	10
25	DVEPSO$_c$	**1**	9	8	**1**	8	**1.5**	4	3	3
25	DVEPSO$_d$	3	6	9	2	10	9.5	3	10	10
25	DVEPSO$_{pe}$	4.5	7.5	6	3	4	3.5	9	6	6
25	DVEPSO$_p$	4.5	5	7	8	9	3.5	7	9	9
25	DVEPSO$_r$	6	4	10	10	6	**1.5**	2	6	5
25	DVEPSO$_{re}$	2	7.5	5	6.5	3	6.5	10	8	8
25	DVEPSO$_u$	7	10	**1**	5	7	9.5	5.5	6	4
25	DNSGAII-A	10	3	2	4	**1**	6.5	**1.0**	2	2
25	DNSGAII-B	9	**1.5**	3	6.5	2	5.0	5.5	**1.0**	**1**
25	dCOEA	8	**1.5**	4	9	5	8	8	4	7
50	DVEPSO$_c$	**1.5**	5	10	**1**	7	10	8	7	9.5
50	DVEPSO$_d$	**1.5**	9	7	4.5	10	7	10	8	8
50	DVEPSO$_{pe}$	5.5	6	8	6.5	5	3	5.5	4	4
50	DVEPSO$_p$	3.5	9	5	9	9	8	9	10	7
50	DVEPSO$_r$	3.5	7	9	10	6	9	4	9	9.5
50	DVEPSO$_{re}$	5.5	9	6	8	4	5	7	6	6
50	DVEPSO$_u$	9	3	3	2	8	6	5.5	3	3
50	DNSGAII-A	7.5	4	**1**	3	2	4	**1**	**1**	**1**
50	DNSGAII-B	7.5	2	2	4.5	**1**	**1.5**	2	2	2
50	dCOEA	10	**1**	4	6.5	3	**1.5**	3	5	5

approaches and the evolutionary algorithms resulted in statistically significant differences. This is shown in Table 8.11.

Table 8.12 shows that for FDA3 for a change frequency of 10 and 25 there is no statistically significant difference between the VD values of the DVEPSO approaches, but almost all comparisons of DVEPSO approaches with an evolutionary algorithm resulted in a statistically significant difference in VD values. For a change frequency of 50, almost all comparisons resulted in statistically significant differences.

For dMOP1 with a change frequency of 10 most DVEPSO approaches compared against each other resulted in statistically significant differences and for a change frequency of 25 and 50 almost half of the DVEPSO approaches comparisons resulted in statistically significant differences. However, for all three change frequencies all comparisons between the evolutionary algorithms and the DVEPSO approaches resulted in statistically significant differences and the values obtained by

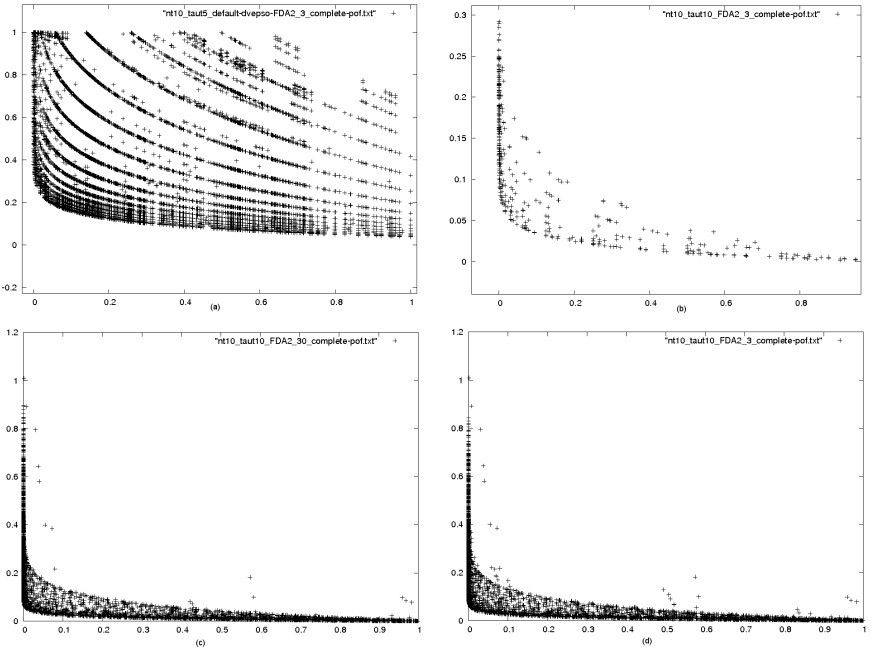

Fig. 8.3 Results of various algorithms solving FDA2, with (a) DVEPSO, (b) dCOEA, (c) DNSGAII-A and (d) DNSGAII-B.

DNSGAII-A and *DNSGAII-B* was statistically significantly different, but the comparison between *DNSGAII-B* and *dCOEA* was not statistically significantly different. This is shown in Table 8.13. Table 8.14 shows that for dMOP2 for a change frequency of 10 all comparisons lead to statistically significant differences and for a change frequency of 50 only the comparison between *DNSGAII-A* and *DNSGAII-B* indicated a statistically significant difference. For a change frequency of 25 all comparisons amongst the evolutionary algorithms, and all comparisons between the evolutionary algorithms and the DVEPSO approaches, resulted in statistically significant differences. However, only a few comparisons amongst the DVEPSO approaches resulted in a statistically significant difference.

For dMOP3 no statistically significant difference was found for any comparisons amongst the DVEPSO approaches for all frequencies of change. For a change frequency of 10, all comparisons amongst the evolutionary algorithms indicated a statistically significant difference, but not for the change frequencies of 25 and 50. All comparisons between the evolutionary algorithms and the DVEPSO approaches indicated a statistically significant difference for all frequencies of change for dMOP3. This is shown in Table 8.15.

Table 8.16 shows that for HE1 for a change frequency of 10, all comparisons lead to a statistically significant difference, except the comparison of *DNSGAII-B*

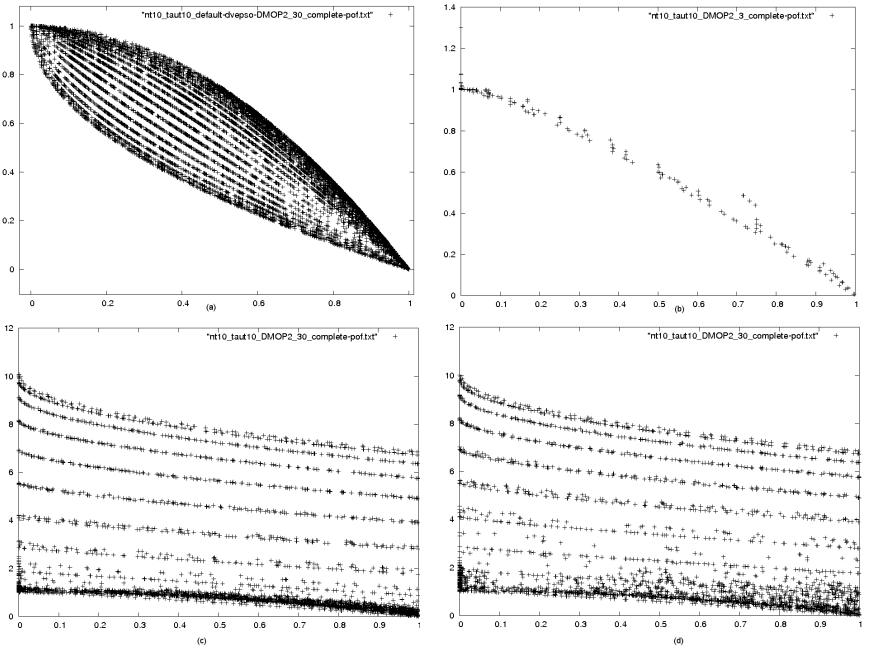

Fig. 8.4 Results of various algorithms solving dMOP2, with (a) DVEPSO, (b) dCOEA, (c) DNSGAII-A and (d) DNSGAII-B.

and *dCOEA*. For a change frequency of 25, almost all DVEPSO comparisons and all comparisons between DVEPSO and evolutionary computation algorithms lead to a statistically significant difference and amongst the evolutionary algorithms only the comparison between *DNSGAII-B* and *dCOEA* indicated VD values that were not statistically significantly different. For a change frequency of 50, all comparisons between DVEPSO and the evolutionary algorithms, and a few of the comparisons amongst the DVEPSO approaches, indicated statistically significant different values.

For HE2 for a change frequency of 10 and 50, amongst the evolutionary algorithms only the comparison between *DNSGAII-B* and *dCOEA* indicated VD values that were not statistically significantly different and for a change frequency of 25 none of the comparisons amongst the evolutionary algorithms indicated a statistically significant difference. From the comparisons amongst the DVEPSO approaches, approximately half indicated a statistically significant difference for change frequencies of 10 and 50. For a change frequency of 25, none of the comparisons amongst the DVEPSO approaches indicated a statistically significant difference. However, for all frequencies of change, the comparisons between the evolutionary algorithms and the DVEPSO approaches indicated a statistically significant difference.

Table 8.10 *p*-values of Kruskal-Wallis test for FDA1

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
25	**0.00045**	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
50	**0.01745**	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	**0.0001837**	$< \mathbf{2.2x10^{-16}}$

Table 8.11 *p*-values of Kruskal-Wallis test for FDA2

τ_t	S	HV R	Acc	Stab	VD	MS
10	$\mathbf{3.509x10^{-14}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
25	**0.003196**	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{5.444x10^{-12}}$
50	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$

Table 8.12 *p*-values of Kruskal-Wallis test for FDA3

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{9.382x10^{-14}}$	**0.01549**
25	$\mathbf{4.898x10^{-08}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	**0.0182**	0.08228
50	$\mathbf{1.864x10^{-07}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{1.96x10^{-08}}$

Table 8.13 *p*-values of Kruskal-Wallis test for dMOP1

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
25	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
50	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$

Table 8.14 *p*-values of Kruskal-Wallis test for dMOP2

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{7.723x10^{-16}}$
25	0.9811	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{3.127x10^{-16}}$	$\mathbf{2.888x10^{-08}}$	$\mathbf{7.932x10^{-08}}$
50	$\mathbf{2.564x10^{-15}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	0.9032	$< \mathbf{2.2x10^{-16}}$

Table 8.15 p-values of Kruskal-Wallis test for dMOP3

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{1.07x10^{-12}}$
25	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{1.573x10^{-07}}$	0.1925
50	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$

Table 8.16 p-values of Kruskal-Wallis test for HE1

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
25	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{1.231x10^{-15}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$
50	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	**0.009434**	$< \mathbf{2.2x10^{-16}}$	$\mathbf{1.158x10^{-06}}$

Table 8.17 p-values of Kruskal-Wallis test for HE2

τ_t	S	HV R	Acc	Stab	VD	MS
10	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{2.682x10^{-10}}$
25	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{3.902x10^{-15}}$	$< \mathbf{2.2x10^{-16}}$	**0.003448**
50	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$	$\mathbf{2.815x10^{-05}}$	$< \mathbf{2.2x10^{-16}}$	$< \mathbf{2.2x10^{-16}}$

8.6 Conclusions

This chapter discussed DMOO and issues that should be addressed when solving DMOOP. The DVEPSO algorithm was presented and the effect that various boundary handling approaches have on the performance of DVEPSO was investigated. It could clearly be seen that the deflection and periodic boundary handling approaches lead to bad performance with especially the FDA2 problem.

The performance of DVEPSO were compared against those of three other state-of-the-art DMOO algorithms. DVEPSO performed quite well with regards to the VD metric that measures the closeness of the approximated POF to the true POF and the MS metric that measures the spread of the found non-dominated solutions. The DNSGAII approaches and dCOEA struggled to track the changing POF of the FDA2 and dMOP2 problems, but DVEPSO had no problem to track the changing POF for these problems. However, the DNSGAII approaches and dCOEA outperformed DVEPSO with the problems that have a discontiuous POF.

Acknowledgement. The authors would like to thank the Centre for High Performance Computing (CHPC) for the use of their infrastructure for this research. Furthermore, they would like to thank C.-K. Goh for sharing his source code of the dCOEA algorithm, and Kalyanmoy Deb for making the code of the static NSGA-II available on his website.

Appendix

Tables 8.18- 8.25 present the results that were obtained with the Mann-Whitney U tests that were performed on the performance metric values. In all tables below, D_d, D_{pe}, D_p, D_r, D_{re}, D_u, *N-A*, *N-B* and *C* refers to $DVEPSO_d$, $DVEPSO_{pe}$, $DVEPSO_p$, $DVEPSO_r$, $DVEPSO_{re}$, $DVEPSO_u$, *DNSGAII-A* and *DNSGAII-B* respectively. In all tables "-" indicates that there was no statistically significant difference and "x" indicates that according to the Mann-Whitney U test, there was a statistically significant difference between the specific performance metric values.

Table 8.18 Results of Mann-Whitney U test for VD metric for FDA1

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	x	x	n/a							
10	D_p	x	x	–	n/a						
10	D_r	x	x	–	x	n/a					
10	D_{re}	x	x	–	x	–	n/a				
10	D_u	x	–	x	x	x	x	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	x	n/a
25	D_c	n/a									
25	D_d	x	n/a								
25	D_{pe}	x	–	n/a							
25	D_p	x	–	–	n/a						
25	D_r	x	x	–	–	n/a					
25	D_{re}	x	–	–	–	x	n/a				
25	D_u	–	x	x	x	x	x	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	–	n/a	
25	C	x	x	x	x	x	x	–	–	–	n/a
50	D_c	n/a									
50	D_d	–	n/a								
50	D_{pe}	x	x	xn/a							
50	D_p	–	–	x	n/a						
50	D_r	–	x	–	x	n/a					
50	D_{re}	–	–	–	–	x	n/a				
50	D_u	–	–	x	–	x	–	n/a			
50	N-A	–	–	x	–	x	–	–	n/a		
50	N-B	–	–	x	x	x	x	–	x	n/a	
50	C	–	–	x	x	–	x	–	x	–	n/a

Table 8.19 Results of Mann-Whitney U test for VD metric for FDA2

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	–	x	n/a							
10	D_p	–	–	x	n/a						
10	D_r	–	–	–	–	n/a					
10	D_{re}	–	–	x	–	–	n/a				
10	D_u	–	–	–	–	–	–	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	–	n/a	
10	C	x	x	x	x	x	x	x	–	–	n/a
25	D_c	n/a									
25	D_d	–	n/a								
25	D_{pe}	–	–	n/a							
25	D_p	–	–	–	n/a						
25	D_r	–	–	–	x	n/a					
25	D_{re}	–	–	–	x	–	n/a				
25	D_u	–	–	–	–	–	–	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	–	n/a	
25	C	x	x	x	x	x	x	x	–	–	n/a
50	D_c	n/a									
50	D_d	x	n/a								
50	D_{pe}	x	–	n/a							
50	D_p	x	x	x	n/a						
50	D_r	x	–	–	x	n/a					
50	D_{re}	–	–	–	x	–	n/a				
50	D_u	x	–	–	x	–	–	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	x	n/a	
50	C	x	x	x	x	x	x	x	–	–	n/a

Table 8.20 Results of Mann-Whitney U test for VD metric for FDA3

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	–	n/a								
10	D_{pe}	–	–	n/a							
10	D_p	–	–	–	n/a						
10	D_r	–	–	–	–	n/a					
10	D_{re}	–	–	–	–	–	n/a				
10	D_u	–	–	–	–	–	–	n/a			
10	N-A	–	–	–	–	–	–	–	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	–	n/a
25	D_c	n/a									
25	D_d	–	n/a								
25	D_{pe}	–	–	n/a							
25	D_p	–	–	–	n/a						
25	D_r	–	–	–	–	n/a					
25	D_{re}	–	–	–	–	–	n/a				
25	D_u	–	–	–	–	–	–	n/a			
25	N-A	–	–	–	–	–	–	–	n/a		
25	N-B	x	x	x	x	x	x	x	–	n/a	
25	C	x	x	x	x	x	x	x	–	–	n/a
50	D_c	n/a									
50	D_d	x	n/a								
50	D_{pe}	x	–	n/a							
50	D_p	x	–	–	n/a						
50	D_r	x	–	x	–	n/a					
50	D_{re}	x	x	x	x	x	n/a				
50	D_u	x	x	x	x	x	–	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	–	n/a	
50	C	x	x	x	x	x	x	x	x	–	n/a

Table 8.21 Results of Mann-Whitney U test for VD metric for dMOP1

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	x	–	n/a							
10	D_p	x	–	–	n/a						
10	D_r	x	–	–	x	n/a					
10	D_{re}	x	x	x	x	x	n/a				
10	D_u	x	x	x	x	x	–	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	–	n/a
25	D_c	n/a									
25	D_d	x	n/a								
25	D_{pe}	–	–	n/a							
25	D_p	–	x	–	n/a						
25	D_r	x	–	x	x	n/a					
25	D_{re}	–	x	–	–	x	n/a				
25	D_u	–	–	–	–	x	–	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	x	n/a	
25	C	x	x	x	x	x	x	x	x	–	n/a
50	D_c	n/a									
50	D_d	–	n/a								
50	D_{pe}	x	x	n/a							
50	D_p	–	x	x	n/a						
50	D_r	–	–	x	–	n/a					
50	D_{re}	x	x	–	x	x	n/a				
50	D_u	x	x	x	x	x	x	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	x	n/a	
50	C	x	x	x	x	x	x	x	x	–	n/a

Table 8.22 Results of Mann-Whitney U test for VD metric for dMOP2

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	x	x	n/a							
10	D_p	x	x	x	n/a						
10	D_r	x	x	x	x	n/a					
10	D_{re}	x	x	x	x	x	n/a				
10	D_u	x	x	x	x	x	x	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	x	n/a
25	D_c	n/a									
25	D_d	–	n/a								
25	D_{pe}	–	x	n/a							
25	D_p	–	–	x	n/a						
25	D_r	–	–	x	–	n/a					
25	D_{re}	–	–	–	–	–	n/a				
25	D_u	x	–	–	–	–	–	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	x	n/a	
25	C	x	x	x	x	x	x	x	x	x	n/a
50	D_c	n/a									
50	D_d	–	n/a								
50	D_{pe}	–	–	n/a							
50	D_p	–	–	–	n/a						
50	D_r	–	–	–	–	n/a					
50	D_{re}	–	–	–	–		n/a				
50	D_u	–	–	–	–	–	–	n/a			
50	N-A	–	–	–	–	–	–	–	n/a		
50	N-B	–	–	–	–	–	–	–	x	n/a	
50	C	–	–	–	–	–	–	–	–	–	n/a

Table 8.23 Results of Mann-Whitney U test for VD metric for dMOP3

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	–	n/a								
10	D_{pe}	–	–	n/a							
10	D_p	–	–	–	n/a						
10	D_r	–	–	–	–	n/a					
10	D_{re}	–	–	–	–	–	n/a				
10	D_u	–	–	–	–	–	–	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	x	n/a
25	D_c	n/a									
25	D_d	–	n/a								
25	D_{pe}	–	–	n/a							
25	D_p	–	–	–	n/a						
25	D_r	–	–	–	–	n/a					
25	D_{re}	–	–	–	–	–	n/a				
25	D_u	–	–	–	–	–	–	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	–	n/a	
25	C	x	x	x	x	x	x	x	–	–	n/a
50	D_c	n/a									
50	D_d	–	n/a								
50	D_{pe}	–	–	n/a							
50	D_p	–	–	–	n/a						
50	D_r	–	–	–	–	n/a					
50	D_{re}	–	–	–	–	–	n/a				
50	D_u	–	–	–	–	–	–	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	–	n/a	
50	C	x	x	x	x	x	x	x	–	–	n/a

Table 8.24 Results of Mann-Whitney U test for VD metric for HE1

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	x	x	n/a							
10	D_p	x	x	x	n/a						
10	D_r	x	x	x	x	n/a					
10	D_{re}	x	x	x	x	x	n/a				
10	D_u	x	x	x	x	x	x	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	–	n/a
25	D_c	n/a									
25	D_d	x	n/a								
25	D_{pe}	x	x	n/a							
25	D_p	x	x	x	n/a						
25	D_r	x	x	x	x	n/a					
25	D_{re}	x	x	x	x	–	n/a				
25	D_u	x	x	–	–	x	x	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	x	n/a	
25	C	x	x	x	x	x	x	x	x	–	n/a
50	D_c	n/a									
50	D_d	x	n/a								
50	D_{pe}	x	–	n/a							
50	D_p	x	x	x	n/a						
50	D_r	x	–	–	x	n/a					
50	D_{re}	x	–	–	x	–	n/a				
50	D_u	x	–	–	x	–	–	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	x	n/a	
50	C	x	x	x	x	x	x	x	–	–	n/a

Table 8.25 Results of Mann-Whitney U test for VD metric for HE2

τ_t	Algorithm	Algorithm									
		D_c	D_d	D_{pe}	D_p	D_r	D_{re}	D_u	N-A	N-B	C
10	D_c	n/a									
10	D_d	x	n/a								
10	D_{pe}	–	x	n/a							
10	D_p	x	–	x	n/a						
10	D_r	–	x	–	x	n/a					
10	D_{re}	x	–	x	–	x	n/a				
10	D_u	–	x	–	x	–	x	n/a			
10	N-A	x	x	x	x	x	x	x	n/a		
10	N-B	x	x	x	x	x	x	x	x	n/a	
10	C	x	x	x	x	x	x	x	x	–	n/a
25	D_c	n/a									
25	D_d	–	n/a								
25	D_{pe}	–	–	n/a							
25	D_p	–	–	–	n/a						
25	D_r	–	–	–	–	n/a					
25	D_{re}	–	–	–	–	–	n/a				
25	D_u	–	–	–	–	–	–	n/a			
25	N-A	x	x	x	x	x	x	x	n/a		
25	N-B	x	x	x	x	x	x	x	–	n/a	
25	C	x	x	x	x	x	x	x	–	–	n/a
50	D_c	n/a									
50	D_d	x	n/a								
50	D_{pe}	x	–	n/a							
50	D_p	–	x	–	n/a						
50	D_r	–	x	x	–	n/a					
50	D_{re}	x	–	–	–	–	n/a				
50	D_u	x	–	–	–	x	–	n/a			
50	N-A	x	x	x	x	x	x	x	n/a		
50	N-B	x	x	x	x	x	x	x	x	n/a	
50	C	x	x	x	x	x	x	x	x	–	n/a

References

[1] Cámara, M., Ortega, J., Toro, J.: Parallel Processing for Multi-objective Optimization in Dynamic Environments. In: Proc. of IEEE International Parallel and Distributed Processing Symposium, p. 243 (2007)

[2] Carlisle, A., Dozier, G.: Adapting Particle Swarm Optimization to Dynamic Environments. In: Proc. of International Conference on Artificial Intelligence (ICAI 2000), pp. 429–434 (2000)

[3] CHPC. Sun hybrid system, `http://www.chpc.ac.za/sun` (last accessed online on March 15, 2011)

[4] Chu, W., Gao, X., Sorooshian, S.: Handling boundary constraints for particle swarm optimization in high-dimensional search space. Information Sciences (2010) (in press)

[5] Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proc. of Congress on Evolutionary Computation (CEC 2002), vol. 1, pp. 825–830 (2002)

[6] Deb, K., Udaya Bhaskara Rao, N., Karthik, S.: Dynamic Multi-objective Optimization and Decision-Making Using Modified NSGA-II: A Case Study on Hydro-thermal Power Scheduling. In: Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., Murata, T. (eds.) EMO 2007. LNCS, vol. 4403, pp. 803–817. Springer, Heidelberg (2007)

[7] Farina, M., Deb, K., Amato, P.: Dynamic multiobjective optimization problems: test cases, approximations, and applications. IEEE Transactions on Evolutionary Computation 8(5), 425–442 (2004)

[8] Goh, C.-K., Tan, K.C.: A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. IEEE Transactions on Evolutionary Computation 13(1), 103–127 (2009)

[9] Goh, C.K., Tan, K.C.: An Investigation on Noisy Environments in Evolutionary Multiobjective Optimization. IEEE Transactions on Evolutionary Computation 11(3), 354–381 (2007)

[10] Greeff, M., Engelbrecht, A.P.: Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation. In: Proc. of IEEE World Congress on Evolutionary Computation: IEEE Congress on Evolutionary Computation, Hong Kong, pp. 2917–2924 (June 2008)

[11] Guan, S.-U., Chen, Q., Mo, W.: Evolving Dynamic Multi-Objective Optimization Problems with Objective Replacement. Artificial Intelligence Review 23(3), 267–293 (2005)

[12] Helbig, M., Engelbrecht, A.P.: Archive management for dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation. Submitted for Review

[13] Helwig, S., Wanka, R.: Particle swarm optimization in high-dimensional bounded search spaces. In: Proc. of IEEE Swarm Intelligence Symposium, Honululu (HI), pp. 198–205 (2007)

[14] Jin, Y., Sendhoff, B.: Constructing Dynamic Optimization Test Problems Using the Multi-objective Optimization Concept. In: Raidl, G.R., Cagnoni, S., Branke, J., Corne, D.W., Drechsler, R., Jin, Y., Johnson, C.G., Machado, P., Marchiori, E., Rothlauf, F., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2004. LNCS, vol. 3005, pp. 525–536. Springer, Heidelberg (2004)

[15] Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. of IEEE International Conference on Neural Networks, vol. IV, pp. 1942–1948 (1995)

[16] Deb, K.: Kanpur Genetic Algorithms Laboratory (2011), http://www.iitk.ac.in/kangal/codes.shtml (last accessed online on March 6, 2011)
[17] Li, X., Branke, J., Blackwell, T.: Particle Swarm with Speciation and Adaptation in a Dynamic Environment. In: Proc. of 8th Conference on Genetic and Evolutionary Computation (GECCO 2006), pp. 51–58 (2006)
[18] Li, X., Branke, J., Kirley, M.: On Performance Metrics and Particle Swarm Methods for Dynamic Multiobjective Optimization Problems. In: Proc. of Congress of Evolutionary Computation (CEC 2007), pp. 1635–1643 (2007)
[19] Mehnen, J., Wagner, T., Rudolph, G.: Evolutionary Optimization of Dynamic Muli-Objective Test Functions. In: Proc. of 2nd Italian Workshop on Evolutionary Computation and 3rd Italian Workshop on Artificial Life (2006)
[20] Pampara, G., Engelbrecht, A.P., Cloete, T.: Cilib: A collaborative framework for computational intelligence algorithms - part i. In: Proc. of IEEE World Congress on Computational Intelligence (WCCI), Hong Kong, June 1-8, pp. 1750–1757 (2011), Source code available at, http://www.cilib.net (last accessed on March 6, 2011)
[21] Parsopoulos, K.E., Tasoulis, D.K., Vrahatis, M.N.: Multiobjective Optimization using Parallel Vector Evaluated Particle Swarm Optimization. In: Proc. of IASTED International Conference on Artificial Intelligence and Applications, Innsbruck Austria (2004)
[22] Parsopoulos, K.E., Vrahatis, M.N.: Recent Approaches to Global Optimization Problems through Particle Swarm Optimization. Natural Computing 1(2-3), 235–306 (2002)
[23] Bergh, F.V.D.: An analysis of particle swarm optimizers. PhD thesis, Department of Computer Science, University of Pretoria (2002)
[24] Zhang, W.-J., Xie, X.-F., Bi, D.-C.: Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space. In: IEEE Congress on Evolutionary Computation, vol. 2, pp. 2307–2311 (June 2004)
[25] Zheng, B.: A New Dynamic Multi-Objective Optimization Evolutionary Algorithm. In: Proc. of third International Conference on Natural Computation (ICNC 2007), vol. V, pp. 565–570 (2007)
[26] Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Emperical Results. Evolutionary Computation 8(2), 173–195 (2000)