Dynamic Vector-evaluated PSO with Guaranteed Convergence in the Sub-swarms

Mardé Helbig
Department of Computer Science
University of Pretoria
Hatfield, South Africa
Email: mhelbig@cs.up.ac.za

Andries Engelbrecht
Department of Computer Science
University of Pretoria
Hatfield, South Africa
Email: engel@cs.up.ac.za

***Abstract*—Optimisation problems with more than one objective, of which at least at least one changes over time and at least two are in conflict with one another, are referred to as dynamic multi-objective optimisation problems (DMOOPs). The dynamic vector evaluated particle swarm optimisation (DVEPSO) algorithm is a co-operative particle swarm optimisation (PSO)-based algorithm and each of its sub-swarms solves only one objective function. The sub-swarms then share knowledge with one another through the particles' velocity update. The default DVEPSO algorithm uses global best (gbest) PSOs as its sub-swarms. The guaranteed convergence PSO (GCPSO) algorithm prevents stagnation by forcing the global best particle to search within a defined region for a better solution. Using GCPSO results in proven convergence to at least a local optimum. Therefore, it is guaranteed that DVEPSO will converge to at least a local Pareto-optimal front (POF). This study investigates the effect of using GCPSOs as sub-swarms of DVEPSO. The results indicate that the GCPSO version of DVEPSO outperforms the gbest PSO DVEPSO on type I DMOOPs and in slowly changing environments.**

I. Introduction

Optimisation problems with more than one objective where at least two objectives are in conflict with one another, are referred to as multi-objective optimisation problems (MOOPs). If at least one of the objectives of the MOOP changes over time, it is called a dynamic multi-objective optimisation problem (DMOOP). A multi-objective algorithm (MOA) solving a static MOOP has to find a set of trade-off solutions that are as close as possible to the optimal set of trade-off solutions, and that contains a diverse set solutions. In addition to these two goals, when a dynamic MOA (DMOA) is solving a DMOOP it has to track the changing set of optimal solutions over time. Therefore, the ability of a DMOA to converge to the set of optimal solutions is important.

The dynamic vector evaluated particle swarm optimisation (DVEPSO) algorithm is a co-operative particle swarm optimisation (PSO)-based algorithm. Each sub-swarm solves only one objective and then the sub-swarms share knowledge amongst each other. Each sub-swarm's search is guided by a global guide and local guides. The global guide is a global best (gbest) position, i.e. the best position found so far by either the sub-swarm itself or another sub-swarm. A local guide is normally the personal best (pbest) position, i.e. the best position found so far by a particle.

One of the issues faced by PSO is that stagnation can occur if all particles' position, pbest position and gbest position are the same for a number of iterations [1]. In this situation, the particles' velocity update depends entirely on the value of inertia component (momentum) term of the velocity update. If the value of the inertia weight is less than one, this will cause this term to eventually become zero, causing stagnation. However, stagnation can be prevented by forcing the gbest to change if this situation occurs. The guaranteed convergence PSO (GCPSO) [2], [3] forces the gbest to search for a better position within a confined region, consequently preventing stagnation. The goal of this study is to investigate whether using GCPSO as the sub-swarms of DVEPSO will improve the performance of DVEPSO when solving DMOOPs.

The rest of the paper's layout is as follows: Background information that is required for the rest of the paper, namely GCPSO and multi-objective optimisation (MOO), is discussed in Section II. Section III discusses the default DVEPSO configuration. The experimental setup for this study is discussed in Section IV. The DMOAs used in this study, the benchmark functions that the algorithms are evaluated on, the performance measures used to measure the algorithms' performance and the statistical analysis approach used to analyse the data are discussed. Section V presents and discusses the results of the study. Finally, the conclusions are presented in Section VI.

II. Background

This section discusses GCPSO and MOO.

A. Guranteed Convergence Particle Swarm Optimisation Algorithm

GCPSO [2], [3] forces the gbest to search for a better position within a confined region. Let τ represent the index of the gbest ($\hat{y}$) such that $y_\tau = \hat{y}$. Then the velocity update of $\hat{y}$ is defined as:

$$\mathbf{v}_\tau(t+1) = -\mathbf{x}_\tau(t) + \omega\mathbf{v}_\tau(t) + \rho(t)(1 - 2\mathbf{r}_2(t)) \quad (1)$$

where $\mathbf{x}$ is the position of the particle, $\mathbf{v}$ is the velocity of the particle, and ρ is a scaling factor, defined as follows:

$$\rho(t+1) = \begin{cases} 2\rho(t), & \text{if \#successes(t)} > \epsilon_s \\ 0.5\rho(t), & \text{if \#failures(t)} > \epsilon_f \\ \rho(t), & \text{otherwise.} \end{cases} \quad (2)$$

Only the gbest's velocity is updated using Equation (1). All other particles use the standard PSO velocity update. The term

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.184

$-\mathbf{x}_\tau(t)$ in Equation (2), resets the gbest's position to $\hat{\mathbf{y}}(t)$. The the current search direction is represented by the term $\omega\mathbf{v}_\tau(t)$ and is therefore added to the velocity. A random sample from the sampling space with lengths $2\rho(t)$ is generated by the term $\rho(t)(1 - 2\mathbf{r}_2(t))$. GCPSO is forced to perform a random search in an area that surrounds $\hat{\mathbf{y}}(t)$ by the scaling term, $\rho(t)(1 - 2\mathbf{r}_2(t))$. The diameter of the search area is controlled by $\rho(t)$ that is defined as specified in Equation (2). In Equation (2), #successes and #failures refer to the number of consecutive successes or failures respectively. A failure is defined as the case where the gbest does not improve, i.e. $f(\hat{\mathbf{y}}(t)) \leq f(\hat{\mathbf{y}}(t+1))$. The threshold parameters, ϵ_f and ϵ_s, adhere to the following conditions:

$$\begin{aligned} \#successes(t+1) > \#successes(t), \texttt{then} \\ \#failures(t+1) = 0 \\ \#failures(t+1) > \#failures(t), \texttt{then} \\ \#successes(t+1) = 0 \end{aligned} \tag{3}$$

B. Multi-objective Optimisation

The conflicts between a MOOP's objectives result in a MOOP not having a single optimum, since an improvement in one objective leads to worse values for at least one other objective. Therefore, when an algorithm solves MOOPs, it has to find a set of trade-off solutions thare are as close as possible to the set of optimal trade-off solutions and that contains a diverse set of solutions. The set of optimal trade-off solutions is referred to as the Pareto-optimal set (POS) in the decision variable space and as the Pareto-optimal front (POF) in the objective space.

III. Dynamic Vector-evaluated Particle Swarm Optimisation Algorithm

The vector evaluated particle swarm optimisation (VEPSO) algorithm, a co-operative MOO PSO-based algorithm, was introduced by Parsopoulos *et al.* [4] to solve static MOOPs. Each sub-swarm solves only one objective function and the gained knowledge of the various sub-swarms is shared with one another through the global best position used in the velocity update of the particles.

VEPSO was extended for dynamic multi-objective optimisation (DMOO) by Greeff and Engelbrecht [5] and called DVEPSO. The default configuration of DVEPSO used in this study works as follows:

- Each sub-swarm is a gbest PSO. Each particle has a local guide and a global guide that guide its search through the search space. The particle's local guide is its pbest (best position found so far by the particle) and the global guide is selected through a knowledge transfer strategy [6], [7] from one of the sub-swarms called the random knowledge sharing topology [6]. First, a sub-swarm is selected (selected sub-swarm can be another sub-swarm or the sub-swarm itself). The gbest of the selected sub-swarm is then selected from the sub-swarm using tournament selection.
- The particle's new position is selected as the particle's new pbest if its new position leads to a better objective function value than its current pbest position. When comparing positions for the pbest update, only the objective function being optimised by the sub-swarm is taken into consideration and no Pareto-dominance relations are used.
- The particle's new position is selected as the new gbest if the particle's new position dominates the current gbest position. However, if a particle's new position is non-dominated with regards to the sub-swarm's current gbest position, one of these two positions is randomly selected as the sub-swarm's new gbest position.
- A specified number of particles (called sentry particles) [8] are randomly selected and re-evaluated after the algorithm performed the specific iteration, but before the next iteration starts. If the sentry particle's fitness value differs after re-evaluation with more than a specificed value, the sub-swarm is notified that a change in the environment has occurred. Once a change has been detected, the algorithm needs to respond appropriately to the change to address diversity loss and outdated memory.
- After a change in the environment has occurred, 30% of the particles of the sub-swarm whose objective function changed is randomly re-initialised. After re-initialisation, each particle's objective function value and its pbest's objective function value are re-evaluated. If the particle's current position results in a better objective function value than the current pbest, the pbest position is set to the particle's current position. Once all the particles have been re-evaluated, the sub-swarm's gbest is re-evaluated using a similar approach. In addition, the non-dominated solutions in the archive are also re-evaluated and the solutions that have become dominated are removed from the archive.
- When the archive is has reached its maximum capacity, a solution is removed from a crowded region in the archive. Crowded regions in the archive are determined by calculating each archive solution's minimum distance to the other solutions in the archive. The archive solution with the smallest minimum distance is then selected for removal from the archive.

IV. Experimental Setup

This section discusses the experimental setup of the experiments conducted for this study. The DVEPSO configurations that were evaluated on the selected benchmark functions are discussed in Section IV-A. Sections IV-B and IV-C discuss the benchmark functions and performance measures used to evaluate the performance of the algorithms respectively. The approach that was followed to analyse the performance of the various DVEPSO configurations is discussed in Section IV-D.

A. Dynamic Vector Evaluated Particle Swarm Optimisation Algorithm Configurations

The following DVEPSO configurations were used in the study:

- Default DVEPSO using gbest PSO as discussed in Section III.
- DVEPSO using GCPSO for its sub-swarms. Van den Bergh and Engelbrecht [3] recommend to use $\epsilon_s = 15$ and $\epsilon_f = 5$ in high-dimensional search spaces, since the algorithm is then quicker to punish poor ρ values than it is to reward successful ρ values. Therefore, for this study the following ρ-ϵ_s-ϵ_f were used:
  - 0.5-3-1, 1.0-3-1, 1.5-3-1, 2.0-3-1
  - 0.5-6-2, 1.0-6-2, 1.5-6-2, 2.0-6-2
  - 0.5-9-3, 1.0-9-3, 1.5-9-3, 2.0-9-3

  Therefore, 12 GCPSO DVEPSO configurations were used in this study.

Each algorithm was executed on each benchmark function for 20 environments or changes for 30 independent runs each. For all DVEPSO configurations sub-swarm sizes of 20 particles were used. The inertia weight and c_1, c_2 were set to values that lead to convergent behaviour [2], namely 0.72, 1.49, and 1.49 respectively. The source code of DVEPSO is available in the opensource library CIlib [9].

B. Benchmark Functions

Based on the analysis of DMOOPs in [10], nine benchmark functions were selected to compare the performance of the DVEPSO configurations. These benchmark functions are of various DMOOP types [11] and were selected based on the characteristics of their POF and POS and the specific difficulties they present to a DMOA. These functions include:

- a modified version of DIMP2 [12] with a concave POF (referred to as DIMP2 in the rest of the paper). DIMP2 is a type I DMOOP and each decision variable has its own rate of change.
- dMOP2$_{dec}$ [13], a type II DMOOP where both the POF and POS change over time and the POF changes from concave to convex and vice versa. Furthermore, the POF is deceptive, i.e. dMOP2$_{dec}$ has at least two optima, but the search space favours the deceptive local POF.
- dMOP3 [14], a type I DMOOP where the POS changes over time and the POF remains static. The variable that controls the spread of the solutions also changes over time.
- FDA4 [11], a type I DMOOP with 3 objective functions.
- FDA5$_{iso}$ [13], a 3-objective type II DMOOP where the spread of solutions in the POF changes over time. In addition, FDA5$_{iso}$ has an isolated POF, i.e. the majority of the fitness landscape is fairly flat and no useful information is provided with regards to the location of the POF.
- FDA5$_{dec}$ [13], a type II DMOOP with 3 objectives, where the spread of solutions in the POF varies over time. Furthermore, its POF is deceptive.
- HE2 [15], a type III problem where the POF changes over time, but the POS remains static. It addition, it has a discontinuous or disconnected POF, i.e. the POF has disconnected pieces, but each piece is continuous.
- HE7 [16], [10], a type III DMOOP with a non-linear POS and each decision variable has a different POS.

C. Performance Measures

An analysis of performance measures in [16], [17], [18] lead to the selection of the following two performance measures for this study:

- the alternative accuracy measure (acc_{alt}) [19] that measures the difference between the hypervolume [20], [21] of the found or approximated POF and the hypervolume of the true POF. A low acc value indicates good performance. This measure is referred to as acc in this paper.
- stability ($stab$) [19] that quantifies the effect of changes in the environment on acc of the DMOA. It measures the difference in acc of two consecutive executive environments, i.e. whether the acc remains relatively stable for the various environments. A low $stab$ value indicates good performance. It is referred to as $stab$ in this paper.

The hypervolume value required for these measures was calculated according to [22]. The reference vector used for the hypervolume calculations was the worst objective function value for each dimension.

D. Statistical Analysis

For each DMOOP, for each environment, and for each performance measure, the following process was followed for the statistical analysis of the results [23]:

1) For each time step just before a change in the environment occurred, the average performance measure value over 30 runs was calculated for each algorithm. Therefore, for each algorithm and for each environment there were 30 performance measure values.
2) A Kruskal-Wallis test was performed on these performance measure values obtained by all the DVEPSO configurations to determine whether there was a statistical significant difference between the performance of these algorithms.
3) If the Kruskal-Wallis test indicated that there was a statistical significant difference between the performance measure values, a pair-wise Mann-Whitney U test was performed for each pair of DMOAs.
4) If the Mann-Whitney U test indicated a statistical significant difference, wins and losses were awarded based on the average performance measure value of each time step just before a change in the environment occurred as follows:

- At each time step just before a change in the environment occurred, the average performance measure values of the two DMOAs were compared.
- The DMOA with the best performance measure value was awarded a win and the other DMOA was awarded a loss.
- In order to prevent skewed results, the number of wins and losses were normalised for each DMOOP.

A confidence level of 95% was used for all statistical tests.

V. RESULTS

This section presents the results obtained by the various DVEPSO configurations. It should be noted that DVEPSO has been extensively evaluated against other DMOAs on a wide range of benchmark functions and environment types [23], [16]. Therefore, in this paper the effect of using GCPSOs as sub-swarms is investigated by comparing various DVEPSO configurations' (refer to Section IV-A) performances against one another.

A. Overall Performance

This section discusses the results that were obtained over all the environments, all the performance measures and all the benchmark functions. The overall wins and losses are presented in Table I. In all tables in this section, D_d refers to the default DVEPSO configuration and $GC_{r,s}$ refers to a GCPSO configuration that uses GCPSOs for the sub-swarms with $\rho = r$, $\epsilon_s = s$, $\epsilon_f = \epsilon_s/3$.

The results indicate that D_d performed the best, obtaining many more wins than losses. $GC_{1.0,9}$ obtained the second best rank and the worst performing algorithm was $GC_{2.0,9}$.

Table II presents the average acc values obtained by the various DVEPSO algorithms for each benchmark function. The results indicate that D_d performed the best for HE2, HE7 and FDA5$_{iso}$. $GC_{1.5,3}$ performed the best for DIMP2 and $GC_{1.5,6}$ was the best performing algorithm for dMOP2$_{dec}$. $GC_{0.5,6}$ performed the best forfor $dMOP3$. For FDA4 $GC_{1.5,6}$ obtained the best acc average. Therefore, D_d obtained the best acc average for 3 benchmarks, $GC_{1.5,6}$ for 2 benchmarks, and $GC_{0.5,6}$ and $GC_{1.5,3}$ for 1 benchmark.

B. Performance per Measure

The wins and losses that were obtained per measure, over all benchmarks and all environments, are discussed in this section. Table III presents the wins and losses obtained for each performance measure. For acc a similar trend was observed as for the overall performance. Once again, the D_d obtained the best rank for acc and $GC_{1.0,9}$ obtained the second best rank. D_d performed the best for $stab$ and was the only algorithm that obtained more wins than losses for $stab$.

C. Performance per Environment

Table IV presents the wins and losses over all benchmark functions and all performance measures for each of the environments.

D_d performed the best in all environments, except slowly changing environments ($n_t = 10, \tau_t = 25$) where it ranked 11th out of 13 algorithms. For slowly changing environments $GC_{1.0,9}$ performed the best. Furthermore, from the GCPSO configurations, $GC_{1.0,9}$ performed the best overall, since it obtained a top 3 rank in three of the four environments and a rank of 5 out of 13 for the fourth environment. $GC_{1.0,9}$ was also the only algorithm that obtained more wins than losses in all environments.

D. Performance per DMOOP Type

This section discusses the performance of the algorithms for each DMOOP type.

The wins and losses obtained for type I DMOOPs are presented in Table VI. The type I DMOOPs are dMOP3, FDA4 and DIMP2. $GC_{1.0,9}$ performed the best acc. D_d performed poorly for acc, obtaining the worst rank and being award much more losses than wins. However, the opposite is true for these two algorithms for $stab$. $GC_{0.5,6}$ obtained good results for both acc and $stab$, being ranked third and fourth respectively for these measures.

Table VIII presents the wins and losses for type II DMOOPs. The type II DMOOPs are dMOP2$_{dec}$, FDA5$_{iso}$ and FDA5$_{dec}$. For type II DMOOPs D_d performed the best for both acc and $stab$. However, there is not a huge difference in the number of wins between the various algorithms for type II DMOOPs.

The wins and losses for type III DMOOPs are presented in Table IX. The type III DMOOPs are HE2 and HE7. D_d outperformed the other algorithms for acc, obtaining many more wins than losses. The worst performing algorithm was $GC_{1.0,6}$. $GC_{1.0,9}$ obtained the second best rank for acc and also obtained more wins than losses. Only three algorithms were awarded more wins than losses for acc, namley D_d, $GC_{1.0,9}$ and $GC_{0.5,9}$.

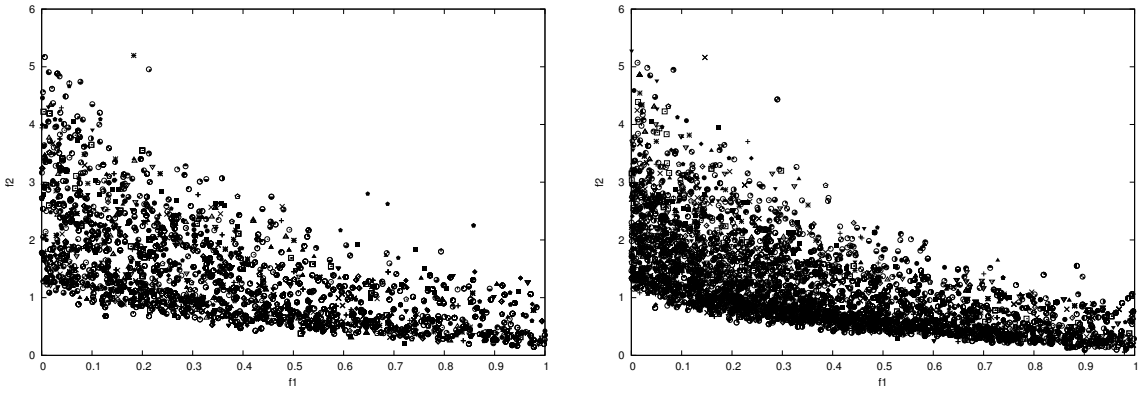
E. Discussion of Results

Figures 1 and 2 illustrate the approximated POFs found by the algorithms. The GCPSO approaches found a much better spread of solutions for these functions, which resulted in better hypervolume and acc values.

VI. CONCLUSIONS

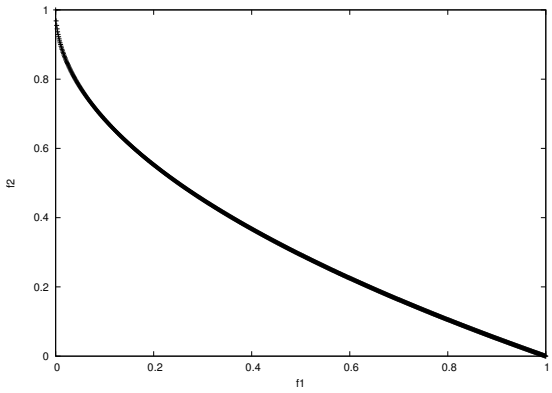
This study investigated the effect of using guaranteed convergence PSO (GCPSO) for the sub-swarms of dynamic vector evaluated particle swarm optimisation (DVEPSO). GCPSO forces the gbest to search for a better solution within a confined region and thereby avoid stagnation. The results indicated that the GCPSO configurations of DVEPSO outperformed the default DVEPSO configuration in slowly changing environments and on type I dynamic multi-objective optimisation problems (DMOOPs), where the Pareto-optimal front (POF) remains static but the Pareto-optimal set (POS) changes over time.

Future work will include investigating the effect of the parameters that determine the search area of the gbest on a larger set of benchmarks and a larger set of environments. In addition, a study will be conducted where a heterogoneous DVEPSO will be evaluated on a wide range of DMOOPs.



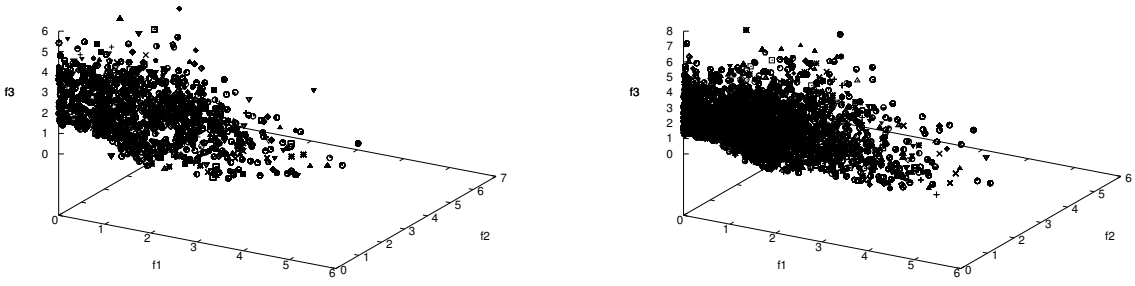
(a) POF of D_d

(b) POF of $GC_{0.5,6}$



(c) True POF of dMOP3

Fig. 1: Approximated POFs found by D_d and $GC_{0.5,6}$ and true POF for dMOP3 with $n_t = 10$ and $\tau_t = 25$



(a) POF of D_d

(b) POF of $GC_{0.5,6}$

Fig. 2: Approximated POFs found by D_d and $GC_{1.5,6}$ for FDA4 with $n_t = 10$ and $\tau_t = 25$

TABLE I: Overall wins for the various DVEPSO configurations

Results	DMOO Algorithm												
	$\mathbf{D}_d$	$\mathbf{GC_{0.5;3}}$	$\mathbf{GC_{0.5;6}}$	$\mathbf{GC_{0.5;9}}$	$\mathbf{GC_{1.0;3}}$	$\mathbf{GC_{1.0;6}}$	$\mathbf{GC_{1.0;9}}$	$\mathbf{GC_{1.5;3}}$	$\mathbf{GC_{1.5;6}}$	$\mathbf{GC_{1.5;9}}$	$\mathbf{GC_{2.0;3}}$	$\mathbf{GC_{2.0;6}}$	$\mathbf{GC_{2.0;9}}$
Wins	197.5	20.65	26.45	26.9	21.45	15.2	44.6	24.55	26.1	24.7	28.25	26.4	30.35
Losses	110.7	32.25	34.5	27.0	31.45	36.5	22.0	34.3	38.75	31.15	30.65	26.5	54.55
Diff	86.8	-11.6	-8.05	-0.1	-10.0	-21.3	22.6	-9.75	-12.65	-6.45	-2.4	-0.1	-24.2
Rank	**1.0**	10.0	7.0	3.0	9.0	12.0	2.0	8.0	11.0	6.0	5.0	3.0	13.0

TABLE II: Average acc value for various DVEPSO configurations

DMOOP	DMOO Algorithm												
	$\mathbf{D}_d$	$\mathbf{GC_{0.5;3}}$	$\mathbf{GC_{0.5;6}}$	$\mathbf{GC_{0.5;9}}$	$\mathbf{GC_{1.0;3}}$	$\mathbf{GC_{1.0;6}}$	$\mathbf{GC_{1.0;9}}$	$\mathbf{GC_{1.5;3}}$	$\mathbf{GC_{1.5;6}}$	$\mathbf{GC_{1.5;9}}$	$\mathbf{GC_{2.0;3}}$	$\mathbf{GC_{2.0;6}}$	$\mathbf{GC_{2.0;9}}$
DIMP2	0.493	0.436	0.431	0.450	0.443	0.459	0.449	**0.425**	0.433	0.430	0.435	0.443	0.431
dMOP2$_{dec}$	0.067	0.066	0.067	0.068	0.067	0.067	0.066	0.066	**0.065**	0.067	0.068	0.067	0.067
dMOP3	1.480	1.249	**1.180**	1.232	1.242	1.289	1.232	1.179	1.233	1.216	1.206	1.245	1.279
HE2	**4.325**	4.405	4.527	4.522	4.470	4.498	4.435	4.493	4.413	4.454	4.486	4.491	4.362
HE7	**2.786**	4.813	4.648	4.078	4.938	4.148	4.613	4.590	4.590	4.872	4.692	4.085	5.061
FDA4	34.969	17.666	16.456	16.877	16.100	17.318	16.689	17.915	**15.846**	17.723	17.899	15.866	17.260
FDA5$_{iso}$	**39.054**	45.221	45.655	46.774	45.409	45.648	46.137	46.509	46.217	45.818	45.537	46.016	46.019

TABLE III: Overall wins and losses per performance measure for various DVEPSO configurations

Measure	Results	DMOO Algorithm												
		$\mathbf{D}_d$	$\mathbf{GC_{0.5;3}}$	$\mathbf{GC_{0.5;6}}$	$\mathbf{GC_{0.5;9}}$	$\mathbf{GC_{1.0;3}}$	$\mathbf{GC_{1.0;6}}$	$\mathbf{GC_{1.0;9}}$	$\mathbf{GC_{1.5;3}}$	$\mathbf{GC_{1.5;6}}$	$\mathbf{GC_{1.5;9}}$	$\mathbf{GC_{2.0;3}}$	$\mathbf{GC_{2.0;6}}$	$\mathbf{GC_{2.0;9}}$
acc	Wins	140.25	20.55	26.45	26.75	21.2	15.2	44.3	23.6	25.75	24.35	28.1	26.3	30.1
acc	Losses	107.75	27.45	30.55	22.25	26.8	32.8	16.7	29.4	33.25	25.65	25.9	21.7	49.9
acc	Diff	32.5	-6.9	-4.1	4.5	-5.6	-17.6	27.6	-5.8	-7.5	-1.3	2.2	4.6	-19.8
acc	Rank	**1.0**	10.0	7.0	4.0	8.0	12.0	2.0	9.0	11.0	6.0	5.0	3.0	13.0
$stab$	Wins	57.25	0.1	0.0	0.15	0.25	0.0	0.3	0.95	0.35	0.35	0.15	0.1	0.25
$stab$	Losses	2.95	4.8	3.95	4.75	4.65	3.7	5.3	4.9	5.5	5.5	4.75	4.8	4.65
$stab$	Diff	54.3	-4.7	-3.95	-4.6	-4.4	-3.7	-5.0	-3.95	-5.15	-5.15	-4.6	-4.7	-4.4
$stab$	Rank	**1.0**	9.0	3.0	7.0	5.0	2.0	11.0	4.0	12.0	12.0	7.0	9.0	5.0

TABLE IV: Overall wins and losses per environment for various DVEPSO configurations

n_t-τ_t	Results	DMOO Algorithm												
		$\mathbf{D}_d$	$\mathbf{GC_{0.5;3}}$	$\mathbf{GC_{0.5;6}}$	$\mathbf{GC_{0.5;9}}$	$\mathbf{GC_{1.0;3}}$	$\mathbf{GC_{1.0;6}}$	$\mathbf{GC_{1.0;9}}$	$\mathbf{GC_{1.5;3}}$	$\mathbf{GC_{1.5;6}}$	$\mathbf{GC_{1.5;9}}$	$\mathbf{GC_{2.0;3}}$	$\mathbf{GC_{2.0;6}}$	$\mathbf{GC_{2.0;9}}$
10-10	Wins	38.9	4.35	6.55	5.05	5.85	3.35	5.55	5.1	11.85	4.25	3.2	8.25	14.65
10-10	Losses	30.0	5.25	11.05	6.55	7.75	8.2	4.95	10.45	4.7	5.3	8.4	4.35	9.95
10-10	Diff	8.9	-0.9	-4.5	-1.5	-1.9	-4.85	0.6	-5.35	7.15	-1.05	-5.2	3.9	4.7
10-10	Rank	**1.0**	6.0	10.0	8.0	9.0	11.0	5.0	13.0	2.0	7.0	12.0	4.0	3.0
10-25	Wins	33.35	4.3	3.3	8.1	3.95	1.95	12.2	1.3	4.9	4.85	7.7	4.85	5.25
10-25	Losses	36.75	5.3	5.3	4.5	3.65	5.6	4.35	3.3	4.7	3.75	3.9	3.75	10.35
10-25	Diff	-3.4	-1.0	-2.0	3.6	0.3	-3.65	7.85	-2.0	0.2	1.1	3.8	1.1	-5.1
10-25	Rank	11.0	8.0	9.0	3.0	6.0	12.0	**1.0**	9.0	7.0	4.0	2.0	4.0	13.0
10-5	Wins	47.7	2.15	14.7	10.1	4.45	6.95	15.1	1.6	6.95	6.85	10.95	6.45	5.7
10-5	Losses	24.95	14.4	4.9	5.45	11.15	5.6	4.4	13.95	9.6	10.7	5.6	9.1	18.85
10-5	Diff	22.75	-12.25	9.8	4.65	-6.7	1.35	10.7	-12.35	-2.65	-3.85	5.35	-2.65	-13.15
10-5	Rank	**1.0**	11.0	3.0	5.0	10.0	6.0	2.0	12.0	7.0	9.0	4.0	7.0	13.0
1-10	Wins	77.55	9.85	1.9	3.65	7.2	2.95	11.75	16.55	2.4	8.75	6.4	6.85	4.75
1-10	Losses	19.0	7.3	13.25	10.5	8.9	17.1	8.3	6.6	19.75	11.4	12.75	9.3	15.4
1-10	Diff	58.55	2.55	-11.35	-6.85	-1.7	-14.15	3.45	9.95	-17.35	-2.65	-6.35	-2.45	-10.65
1-10	Rank	**1.0**	4.0	11.0	9.0	5.0	12.0	3.0	2.0	13.0	7.0	8.0	6.0	10.0

TABLE V: Overall wins and losses per performance measure for various DVEPSO configurations solving type I DMOOPs

TABLE VI: Overall wins and losses per performance measure for various DVEPSO configurations

Measure	Results	DMOO Algorithm												
		$\mathbf{D}_d$	$\mathbf{GC_{0.5;3}}$	$\mathbf{GC_{0.5;6}}$	$\mathbf{GC_{0.5;9}}$	$\mathbf{GC_{1.0;3}}$	$\mathbf{GC_{1.0;6}}$	$\mathbf{GC_{1.0;9}}$	$\mathbf{GC_{1.5;3}}$	$\mathbf{GC_{1.5;6}}$	$\mathbf{GC_{1.5;9}}$	$\mathbf{GC_{2.0;3}}$	$\mathbf{GC_{2.0;6}}$	$\mathbf{GC_{2.0;9}}$
acc	Wins	14.8	15.65	17.65	12.65	18.1	12.7	20.35	17.8	18.15	17.6	21.0	19.85	16.7
acc	Losses	97.2	11.35	8.35	5.35	5.9	11.3	2.65	12.2	16.85	11.4	9.0	5.15	23.3
acc	Diff	-82.4	4.3	9.3	7.3	12.2	1.4	17.7	5.6	1.3	6.2	12.0	14.7	-6.6
acc	Rank	13.0	9.0	5.0	6.0	3.0	10.0	**1.0**	8.0	11.0	7.0	4.0	2.0	12.0
$stab$	Wins	21.65	0.1	0.0	0.15	0.25	0.0	0.3	0.1	0.35	0.35	0.15	0.1	0.25
$stab$	Losses	2.1	1.8	0.95	1.75	1.65	0.95	2.55	1.8	2.5	2.5	1.75	1.8	1.65
$stab$	Diff	19.55	-1.7	-0.95	-1.6	-1.4	-0.95	-2.25	-1.7	-2.15	-2.15	-1.6	-1.7	-1.4
$stab$	Rank	**1.0**	8.0	2.0	6.0	4.0	2.0	13.0	8.0	11.0	11.0	6.0	8.0	4.0

TABLE VII: Overall wins and losses per performance measure for various DVEPSO configurations solving type II DMOOPs

TABLE VIII: Overall wins and losses per performance measure for various DVEPSO configurations

Measure	Results	DMOO Algorithm												
		$\mathbf{D}_d$	$\mathbf{GC}_{0.5;3}$	$\mathbf{GC}_{0.5;6}$	$\mathbf{GC}_{0.5;9}$	$\mathbf{GC}_{1.0;3}$	$\mathbf{GC}_{1.0;6}$	$\mathbf{GC}_{1.0;9}$	$\mathbf{GC}_{1.5;3}$	$\mathbf{GC}_{1.5;6}$	$\mathbf{GC}_{1.5;9}$	$\mathbf{GC}_{2.0;3}$	$\mathbf{GC}_{2.0;6}$	$\mathbf{GC}_{2.0;9}$
acc	Wins	60.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
acc	Losses	0.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0
acc	Diff	60.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0	-5.0
acc	Rank	**1.0**	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
$stab$	Wins	35.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$stab$	Losses	0.0	3.0	3.0	3.0	3.0	2.75	2.75	3.0	3.0	3.0	3.0	3.0	3.0
$stab$	Diff	35.5	-3.0	-3.0	-3.0	-3.0	-2.75	-2.75	-3.0	-3.0	-3.0	-3.0	-3.0	-3.0
$stab$	Rank	**1.0**	4.0	4.0	4.0	4.0	2.0	2.0	4.0	4.0	4.0	4.0	4.0	4.0

TABLE IX: Overall wins and losses per performance measure for various DVEPSO configurations solving type III DMOOPs

Measure	Results	DMOO Algorithm												
		$\mathbf{D}_d$	$\mathbf{GC}_{0.5;3}$	$\mathbf{GC}_{0.5;6}$	$\mathbf{GC}_{0.5;9}$	$\mathbf{GC}_{1.0;3}$	$\mathbf{GC}_{1.0;6}$	$\mathbf{GC}_{1.0;9}$	$\mathbf{GC}_{1.5;3}$	$\mathbf{GC}_{1.5;6}$	$\mathbf{GC}_{1.5;9}$	$\mathbf{GC}_{2.0;3}$	$\mathbf{GC}_{2.0;6}$	$\mathbf{GC}_{2.0;9}$
acc	Wins	65.45	4.9	8.8	14.1	3.1	2.5	23.95	5.8	7.6	6.75	7.1	6.45	13.4
acc	Losses	10.55	11.1	17.2	11.9	15.9	16.5	9.05	12.2	11.4	9.25	11.9	11.55	21.6
acc	Diff	54.9	-6.2	-8.4	2.2	-12.8	-14.0	14.9	-6.4	-3.8	-2.5	-4.8	-5.1	-8.2
acc	Rank	**1.0**	8.0	11.0	3.0	12.0	13.0	2.0	9.0	5.0	4.0	6.0	7.0	10.0
$stab$	Wins	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.85	0.0	0.0	0.0	0.0	0.0
$stab$	Losses	0.85	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0
$stab$	Diff	-0.75	0.0	0.0	0.0	0.0	0.0	0.0	0.75	0.0	0.0	0.0	0.0	0.0
$stab$	Rank	13.0	2.0	2.0	2.0	2.0	2.0	2.0	**1.0**	2.0	2.0	2.0	2.0	2.0

The heterogeneous DVEPSO will contain particles with different behaviours within one sub-swarm, of which one of the behaviours will be using the GCPSO position and velocity updates.

Once a robust DVEPSO algorithm has been developed taking the results of the aforementioned studies into account, DVEPSO will be evaluated against state-of-the-art dynamic MOAs (DMOAs) that were recently proposed [24].

ACKNOWLEDGMENT

The authors would like to thank the Center for High Performance Computing in Cape Town, South Africa, for the usage of their infrastructure to obtain the data for this study.

REFERENCES

[1] A. Engelbrecht, *Computational intelligence: an introduction*. Wiley, 2007.

[2] F.v.d.Bergh, "An analysis of particle swarm optimizers," Ph.D. dissertation, Department of Computer Science University of Pretoria, 2002.

[3] F. Van den Bergh and A. Engelbrecht, "A new locally convergent particle swarm optimiser," in *Proceedings of Conference on Systems, Man and Cybernetics*, vol. 3, 2002, pp. 96–101.

[4] K. Parsopoulos and M. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2-3, pp. 235–306, 2002.

[5] M. Greeff and A. Engelbrecht, "Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation," in *Proceedings of World Congress on Computational Intelligence (WCCI): Congress on Evoluationary Computation*, Hong Kong, June 2008, pp. 2917–2924.

[6] ——, "Dynamic multi-objective optimisation using PSO," in *Multi-Objective Swarm Intelligent Systems*, ser. Studies in Computational Intelligence, N. Nedjah, L. dos Santos Coelho, and L. de Macedo Mourelle, Eds. Springer Berlin/Heidelberg, 2010, vol. 261, pp. 105–123.

[7] K. Harrison, B. Ombuki-Berman, and A. Engelbrecht, *Evolutionary Multi-Criterion Optimization*. Springer Berlin/Heidelberg, 2013, vol. 7811, ch. Knowledge transfer strategies for vector evaluated particle swarm optimization, pp. 171–184.

[8] A. Carlisle and G. Dozier, "Tracking changing extrema with adaptive particle swarm optimizer," in *Proceedings of World Automation Congress*, vol. 13, Orlando, Florida, U.S.A., June 2002, pp. 265–270.

[9] G. Pampará, A. Engelbrecht, and T. Cloete, "CIlib: A collaborative framework for computational intelligence algorithms - Part I," in *Proceedings of World Congress on Computational Intelligence*, Hong Kong, 1-8 June 2008, pp. 1750–1757, source code available at: http://www.cilib.net. Last accessed on: 10 December 2013.

[10] M. Helbig and A. Engelbrecht, "Dynamic multi-objective optimisation problems," *ACM Computing Surveys*, 2014, in Press.

[11] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: test cases, approximations, and applications," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, October 2004.

[12] W. Koo, C. Goh, and K. Tan, "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment," *Memetic Computing*, vol. 2, no. 2, pp. 87–110, 2010.

[13] M. Helbig and A. Engelbrecht, "Benchmarks for dynamic multi-objective optimisation," in *Proc. of IEEE Symposium Series on Computational Intelligence*, Singapore, April 2013, pp. 84–91.

[14] C.-K. Goh and K. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 103–127, February 2009.

[15] M. Helbig and A. Engelbrecht, "Archive management for dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation," in *Proceedings of Congress on Evolutionary Computation*, New Orleans, U.S.A., June 2011, pp. 2047–2054.

[16] M. Helbig, "Solving dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation," Ph.D. dissertation, University of Pretoria, 2012.

[17] M. Helbig and A. Engelbrecht, "Issues with performance measures for dynamic multi-objective optimisation," in *Proc. of IEEE Symposium Series on Computational Intelligence*, Singapore, April 2013, pp. 17–24.

[18] ——, "Performance measures for dynamic multi-objective optimisation algorithms," *Information Sciences*, vol. 250, pp. 61–81, November 2013.

[19] M. Cámara, J. Ortega, and F. de Toro, "A single front genetic algorithm for parallel multi-objective optimization in dynamic environments," *Neurocomputing*, vol. 72, no. 16–18, pp. 3570–3579, 2007.

[20] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms a comparative case study," vol. 1498, pp. 292–301, 1998.

[21] ——, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, nov 1999.

[22] C. Fonseca, L. Paquete, and M. Lopez-Ibanez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *Proceedings of Congress on Evolutionary Computation*, july 2006, pp. 1157–1163.

[23] M. Helbig and A. Engelbrecht, "Analysing the performance of dynamic multi-objective optimisation algorithms," in *Proceedings of the IEEE Congress on Evolutionary Computation*, Cancún, Mexico, June 2013, pp. 1531–1539.

[24] ——, "Results of the IEEE CEC 2015 DMOO Competition," To be published. [Online]. Available: https://sites.google.com/site/cec2015dmoocomp/