Vector Evaluated Particle Swarm Optimization Archive Management: Pareto Optimal Front Diversity Sensitivity Analysis

Christiaan Scheepers
Department of Computer Science
University of Pretoria
South Africa
Email: cscheepers@acm.org

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

***Abstract*—Vector evaluated particle swarm optimization (VEPSO) extends the particle swarm optimization (PSO) algorithm to deal with multi-objective optimization problems (MOPs). VEPSO stores the found non-dominated solutions in an archive. Management of the VEPSO archive is, however, not described in detail. In this paper, a Pareto optimal front (POF) diversity sensitivity analysis on the choice of the VEPSO's archive size and deletion approach is presented. The results indicate that the well-known spread, solution distribution, maximum spread, and spacing metrics are all sensitive to the choice of the archive size and deletion approach. It is concluded that care must be taken when selecting the archive size and deletion approach as the impact on the diversity of the POF is notable.**

I. Introduction

Particle swarm optimization (PSO) is a well-known stochastic optimization technique for single objective optimization with its roots in the simulation of the social behavior of birds within a flock. PSO was first introduced by Kennedy and Eberhart [1] in 1995. For a variety of complex problems, the PSO algorithm has been shown to outperform traditional evolutionary computation (EC) algorithms [2]. PSO has since been extended to deal with multi-objective optimization problems (MOPs) [3]–[7]. Vector evaluated particle swarm optimization (VEPSO) is one such extension that extends the PSO algorithm to deal with MOPs. VEPSO was first introduced by Parsopoulos and Vrahatis [3]–[5] in 2002, inspired by and based on the vector evaluated genetic algorithm (VEGA) developed by Schaffer in 1985 [8].

Various studies have been conducted to analyze the performance of the VEPSO algorithm [9], [10]. These studies use quantitative performance metrics to compare the performance of the VEPSO algorithm against other well-known multi-objective optimization (MOO) algorithms. Helbig and Engelbrecht [11] investigated the effect archive management has on performance in the context of dynamic environments using the Dynamic VEPSO (DVEPSO) algorithm. While studying the performance of a newly introduced knowledge transfer strategy (KTS), Harrison *et al*. [10] compared the VEPSO algorithm against the optimized multi-objective particle swarm optimization (OMOPSO) [6] and speed-constrained multi-objective particle swarm optimization (SMPSO) [7] algorithms. Similar to OMOPSO and SMPSO, VEPSO makes use of an archive to store the non-dominated solutions found during the search process. Unlike OMOPSO and SMPSO, VEPSO does not have a well-described archive management algorithm. Parsopoulos and Vrahatis did not define the number of solutions that should be kept in, or how solutions should be removed from the archive when the archive size limit is reached.

This paper presents an analysis of the effect the chosen archive deletion approach and archive size have on the distribution of solutions along the Pareto optimal front (POF). The well-known spread, solution distribution, maximum spread, and spacing metrics are analyzed with respect to various archive deletion approaches and archive sizes.

The remainder of this paper is organized as follows. Section II discusses the VEPSO algorithm along with the knowledge transfer strategies and archive management. Section III describes the POF diversity metrics used throughout this paper. Section IV describes the experimental procedure and test sets. Section V presents an analysis and discussion of the results obtained from the experimental work. Finally, section VI presents the findings and conclusions.

II. Vector Evaluated Particle Swarm Optimization

This section describes the VEPSO algorithm, knowledge transfer strategies, and archive management used throughout this paper.

A. VEPSO Algorithm

VEPSO is a PSO based cooperative MOO algorithm. Each objective of the multi-objective optimization problem (MOP) is represented and optimized by a separate swarm. Information is passed between the swarms through the use of a KTS. The KTS is implemented by replacing the neighborhood best particle in the standard PSO velocity update equation with the global guide particle selected using the KTS. At the end of each iteration, the non-dominated solutions are stored in an archive if they are not dominated by any solution already in the archive. The VEPSO velocity update equation is formally defined as follows:

$$\vec{v}_i(t+1) = w\vec{v}_i(t-1) + c_1\vec{r}_1(t)(\vec{y}_i(t) - \vec{x}_i(t)) + c_2\vec{r}_2(t)(\vec{\hat{y}}_i(t) - \vec{x}_i(t)) \quad (1)$$

where $\vec{v}_i(t)$ is the velocity of particle i at iteration t, w is the inertia weight, c_1 and c_2 are acceleration constants, $\vec{r}_1(t)$ and $\vec{r}_2(t)$ are random vectors with components sampled uniformly from $(0, 1)$ at iteration t, $\vec{x}_i(t)$ is the position of particle i at iteration t, $\vec{y}_i(t)$ is the local (also referred to as the personal best position) guide of particle i at iteration t, and $\vec{\hat{y}}_i(t)$ is the global guide of particle i at iteration t. The VEPSO algorithm selects $\vec{\hat{y}}_i(t)$ using the KTS.

B. Knowledge Transfer Strategies

Two KTSs were used throughout the work presented in this paper.

Random Personal Best KTS: The global guide, $\vec{\hat{y}}_i(t)$, is selected randomly from the personal best positions from a randomly selected sub-swarm [12]. Grobler [12] found that the random personal best KTS outperformed other KTSs for at least two performance measures.

Parent Centric Crossover Archive (PCXA) KTS: The global guide, $\vec{\hat{y}}_i(t)$, is calculated as the offspring of the parent centric crossover (PCX) operator [13] applied to three randomly selected non-dominated solutions from the archive [14]. Harrison *et al.* [14] compared six different KTSs and found that PCXA achieved the best solution diversity.

C. Archive Management

The original VEPSO algorithm does not specify how the archive should be managed. While several different archive management techniques have been developed [6], [7], [10], [15], it is well known that more research in archive management is needed to further the performance of multi-objective optimization algorithms [15], [16].

Archives store non-dominated solutions found during the search process. When a new solution is admitted into the archive, all the solutions dominated by the new solution are removed. Archives can be bounded or unbounded.

Bounded archives limit the number of solutions that can be kept in the archive. The archive limit can be preset or dynamic [17]. Once the archive size limit is reached, new solutions can only be admitted if existing solutions in the archive are removed. If no solutions in the archive are dominated by the new solution, the solution can either not be admitted [18] or a deletion approach (also known as a removal function or pruning method) can be used to remove a solution from the archive [15], [19]. The experimental work presented in this paper made use of the following four different deletion approaches:

Crowding Distance: Crowding distance was introduced by Deb *et al.* [20] to estimate the density of solutions surrounding a particular solution. The crowding distance of a point is calculated as the average distance of the two points on either side of the specified point along each of the objectives. The solution with the lowest crowding distance is removed from the archive. OMOPSO [6] and SMPSO [7] both use archives with a crowding distance based deletion approach. It should also be noted that various enhancements to crowding distance have been proposed to improve calculation performance and stability when more than two objectives are used [21]–[23].

Distance Metric: Bart-Beielstein *et al.* [15] proposed a distance metric deletion approach based on relative distances in the archive. The relative distance is calculated as follows:

$$f_{del,i} = \sum_{\forall j \neq i} \left(\sqrt{\sum_{k=1}^{K} \left(\frac{x_{i,k} - x_{j,k}}{max_k - min_k} \right)^2} \right)^{-1} \quad (2)$$

where $f_{del,i}$ is the relative distance for solution i in the archive, K is the number of objectives, $x_{i,k}$ and $x_{j,k}$ is the k'th objective function value for the i'th and j'th solutions in the archive respectively, max_k and min_k are the maximum and minimum values reached by an archive member for the k'th objective.

Nearest Neighbor: Harrison *et al.* [10] used a nearest neightbor deletion approach. The nearest neighbor is calculated similarly to the distance metric. However, instead of computing the distance between each pair of solutions in the archive, only the n-closest neighboring solutions are used in the calculation.

Random: The random deletion approach removes a randomly selected solution from the archive.

III. Pareto Optimal Front Diversity Metrics

This section presents a summary of the popular diversity metrics that are used throughout the literature.

A. Maximum Spread

Zitzler [24] introduced the maximum spread metric to measure the length of the diagonal of a hyper-box formed by the extreme function values. Tan *et al.* [25] modified the maximum spread metric to normalize the objective function values. The maximum spread metric is calculated as follows:

$$\bar{D} = \sqrt{\frac{1}{K} \sum_{k=1}^{K} \left(\frac{max_k^* - min_k^*}{max_{true,k} - min_{true,k}} \right)^2} \quad (3)$$

with $max_k^* = \min\left\{max_{i=1}^{|Q|}\{x_{i,k}\}, max_{true,k}\right\}$ and $min_k^* = \max\left\{min_{i=1}^{|Q|}\{x_{i,k}\}, min_{true,k}\right\}$ where Q is the set of solutions in the archive, $|Q|$ is the cardinality of the set Q, $max_{true,k}$ and $min_{true,k}$ are the maximum and minimum values reached by the true POF for the k'th objective.

B. Distribution

Goh and Tan [26] defined the distribution metric, based on the spacing metric [27], to give an indication of the distribution of the solutions along the discovered POF. The distribution metric is calculated as follows:

$$D = \frac{1}{|Q|} \sqrt{\frac{1}{|Q|} \sum_{i=1}^{|Q|} \left(d_i - \bar{d} \right)^2} \quad (4)$$

with $\bar{d} = \frac{1}{|Q|}\sum_{j=1}^{|Q|} d_j$ where d_i is the Euclidean distance in objective space between i and its nearest neighbor in $|Q|$. Smaller values for D indicate a more uniformly distributed POF.

C. Spacing

Schott [27] introduced the spacing metric to calculate the relative distance between consecutive solutions. The spacing metric is calculated as follows:

$$S = \sqrt{\frac{1}{|Q|-1}\sum_{i=1}^{|Q|}\left(\bar{d} - d_i\right)^2} \tag{5}$$

with $d_i = min_{j \in Q \wedge j \neq i}\sum_{k=1}^{K}|x_{i,k} - x_{j,k}|$ and $\bar{d} = \sum_{i=1}^{|Q|}\frac{d_i}{|Q|}$. The spacing metric measures the standard deviations of different d_i values. When the solutions are near uniformly spread, the resulting spacing metric will be small.

D. Spread

Deb *et al.* [20] introduced the spread metric to measure the distance between solutions while also taking the extreme solutions into account. The spread metric is calculated as follows:

$$\Delta = \frac{\sum_{k=1}^{K} d_k^e + \sum_{i=1}^{|Q|}|d_i - \bar{d}|}{\sum_{k=1}^{K} d_k^e + |Q|\bar{d}} \tag{6}$$

where d_i can be any distance measure between neighboring solutions, $\bar{d}$ is the mean of the d_i values and d_k^e is the distance between the extreme solutions of $|Q|$ and the true POF corresponding to the k'th objective.

An ideal distribution will have $\Delta = 0$ if the distances between solutions are equal and $|Q|$ contains the extreme solutions of the true POF, otherwise $\Delta > 0$.

IV. Experimental Setup

This section describes the parameterization and benchmark functions used throughout this paper.

A. Parameterization

All algorithm implementations were done and executed using the CIlib framework [28]. The results presented in this study were taken over 30 independent runs of 2000 iterations of each algorithm for each problem. Each run had 50 particles per swarm, increasing the swarm size had no notable effect on the analysis presented in this study. Known well-performing values were used for the inertia weight, $w = 0.729844$, and the acceleration constants, $c_1 = c_2 = 1.49618$ [29]. n was set to 2 for the nearest neighbor deletion approach.

B. Benchmark Functions

The experimental work presented in this paper made use of the Zitzler, Deb and Thiele (ZDT) [30] test set. The test set provides a mix of challenges to test MOO algorithms against. Table I present a summary of the properties of each of the problems in the aforementioned test set.

TABLE I
Properties of the ZDT Problems

Name	Separability	Modality	Geometry
ZDT1	separable	unimodal	convex
ZDT2	separable	unimodal	concave
ZDT3	separable	unimodal/multimodal	disconnected
ZDT4	separable	unimodal/multimodal	convex
ZDT6	separable	multimodal	concave

V. Analysis

Figs. 1–8 depict the measurement values for the four metrics for ZDT1 through ZDT6 over 2000 iterations for archive sizes 50, 150, and 500. The next four subsections present an analysis for each of the metrics.

A. Spread – Δ

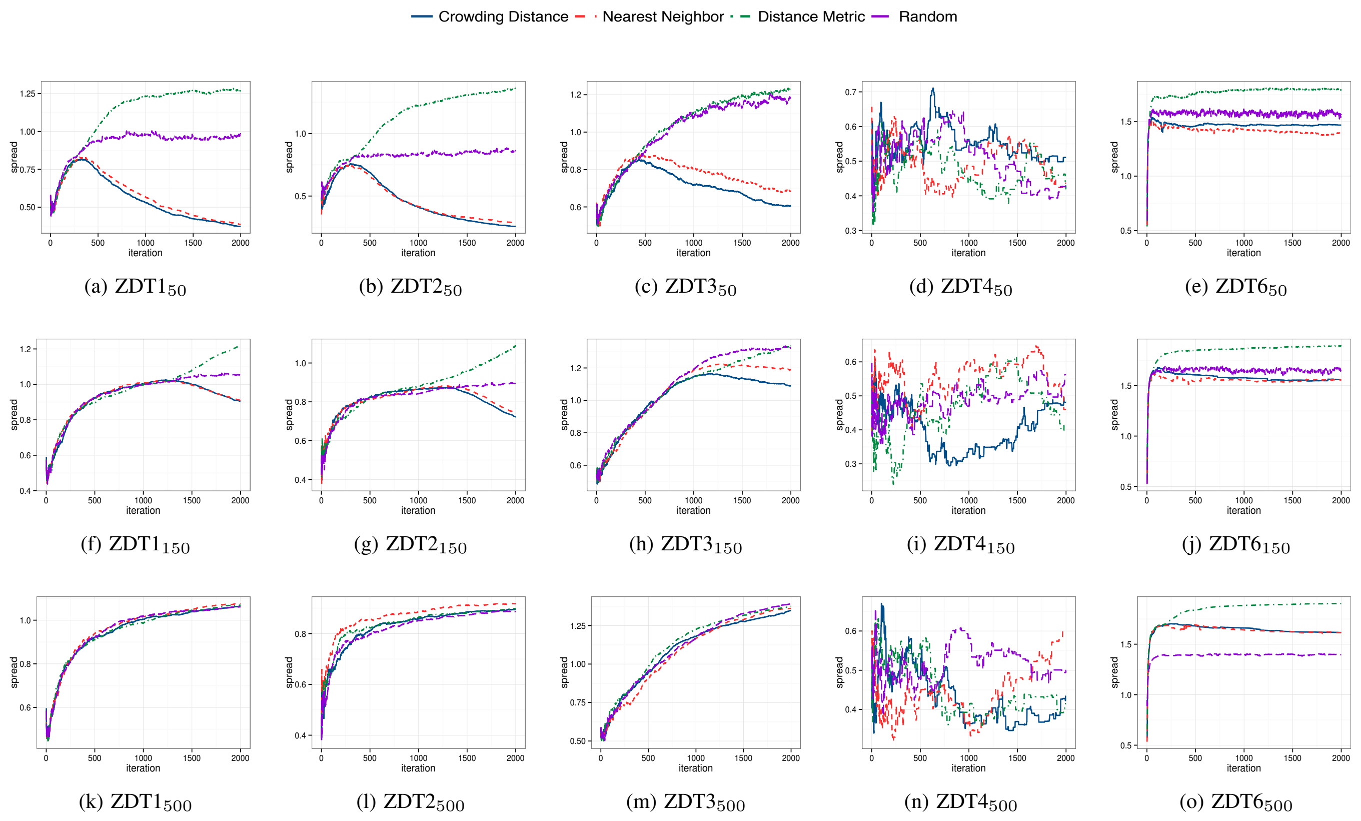
Figs. 1 and 2 depict the Δ (spread) values for both the VEPSO (Random) and VEPSO (PCXA) algorithms. Results for each of the four archive deletion approaches are shown. From Figs. 1(a) – 1(c) the effect of the archive size is clearly visible. The four archive deletion approaches for VEPSO (Random) had similar Δ values up to the point where the archive size was reached. Figs. 9(a) – 9(c) depict the number of solutions in the archive corresponding to the Δ values in Figs. 1(a), 1(f) and 1(k). Once the archive size was reached, the Δ decreased only in the cases where the crowding distance and nearest neighbor deletion approaches were used. For each of the problems the distance metric deletion approach led to worse spread than the random deletion approach once the archive size was reached.

In contrast to VEPSO (Random), the resulting Δ values for VEPSO (PCXA) are much more consistent for the different archive sizes. This can be attributed to the VEPSO (PCXA) algorithm reaching the archive size much faster than the VEPSO (Random) algorithm, as can be noted in Figs. 9(d) – 9(f). The speed at which the archive size is reached reduces the number of iterations where the Δ values are similar for the four archive deletion approaches. Once the archive size is reached, the crowding distance deletion approach performs best, followed by the nearest neighbor deletion approach. Again, the distance metric deletion approach performed worse than the random deletion approach in terms of the Δ.

For ZDT4 the Δ was erratic for both VEPSO (Random) and VEPSO (PCXA) for all the deletion approaches over all three archive sizes. The Δ did not converge to any value.

For ZDT6 the archive size was reached much faster than in the case of ZDT1, ZDT2, and ZDT3. Similar behavior for the Δ can be noted once the archive size was reached.

Overall it is shown that Δ is extremely sensitive to the choice of the archive deletion approach in cases where the archive size is reached. For smaller archives, the archive size is reached faster and the effect on the Δ is thus larger in these cases. The crowding distance deletion approach achieved the

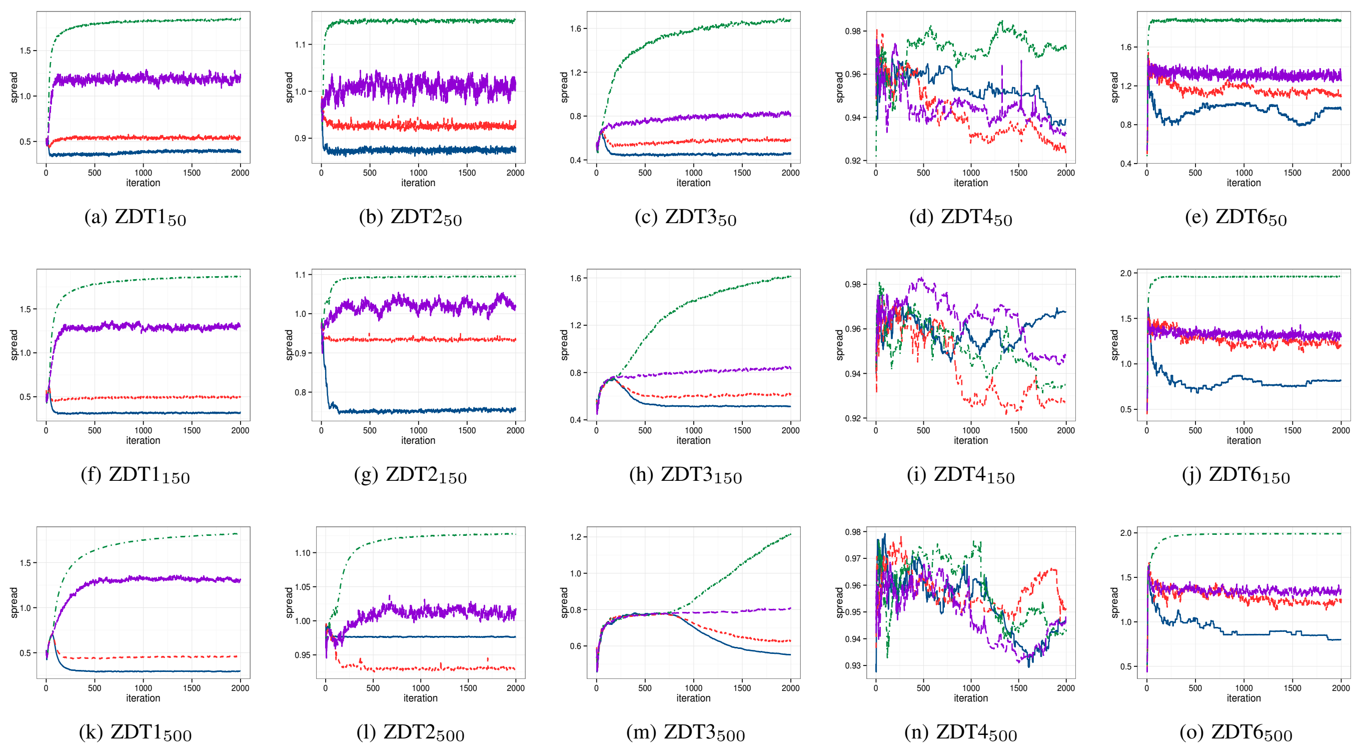


(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 1. VEPSO (Random) Spread, Δ, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

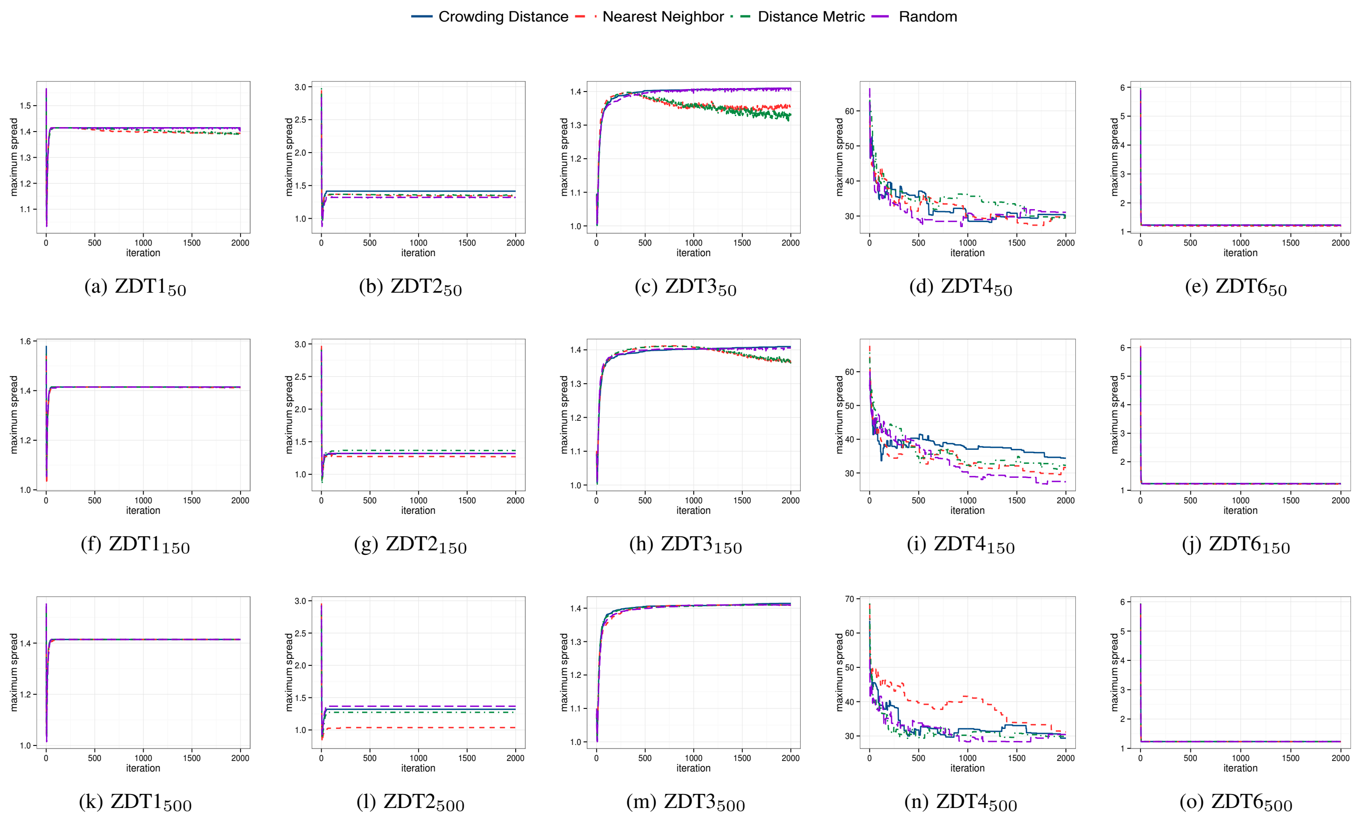


(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 2. VEPSO (PCXA) Spread, Δ, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

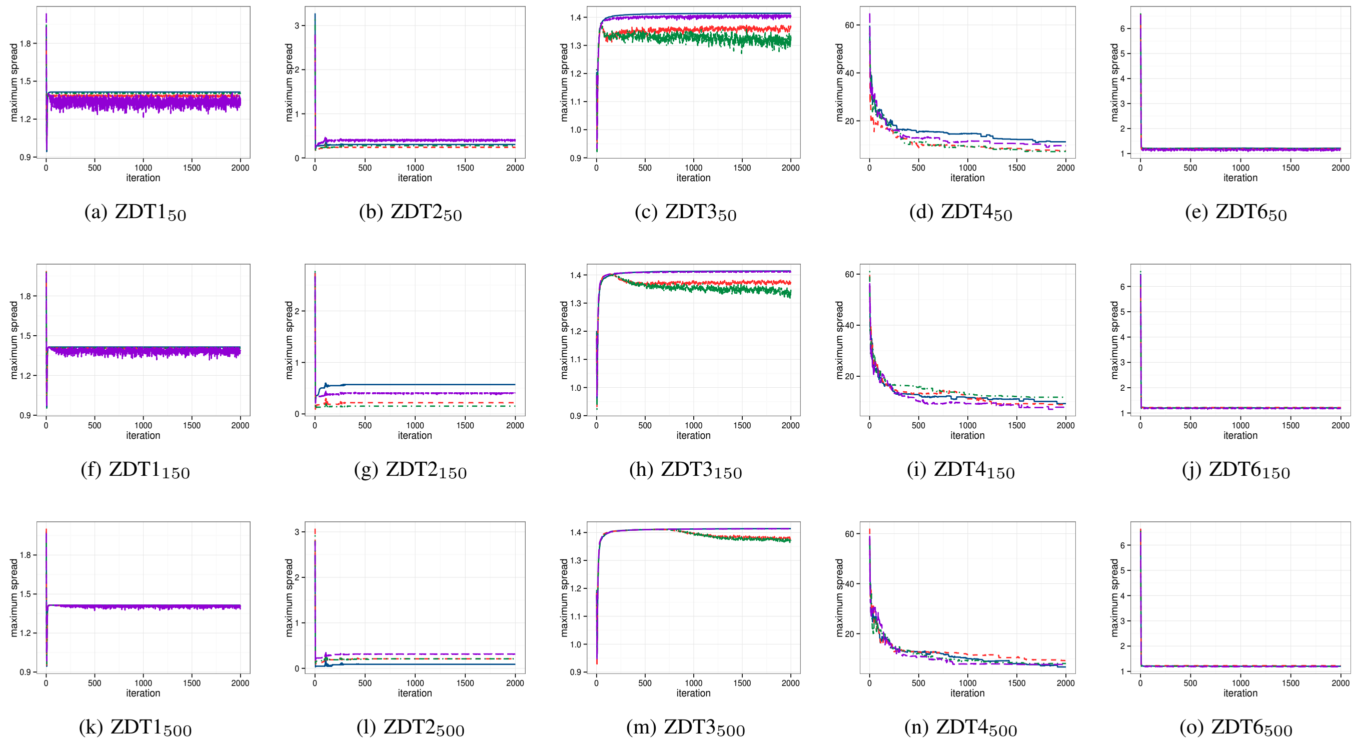


(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 3. VEPSO (Random) Maximum Spread, $\bar{D}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 4. VEPSO (PCXA) Maximum Spread, $\bar{D}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

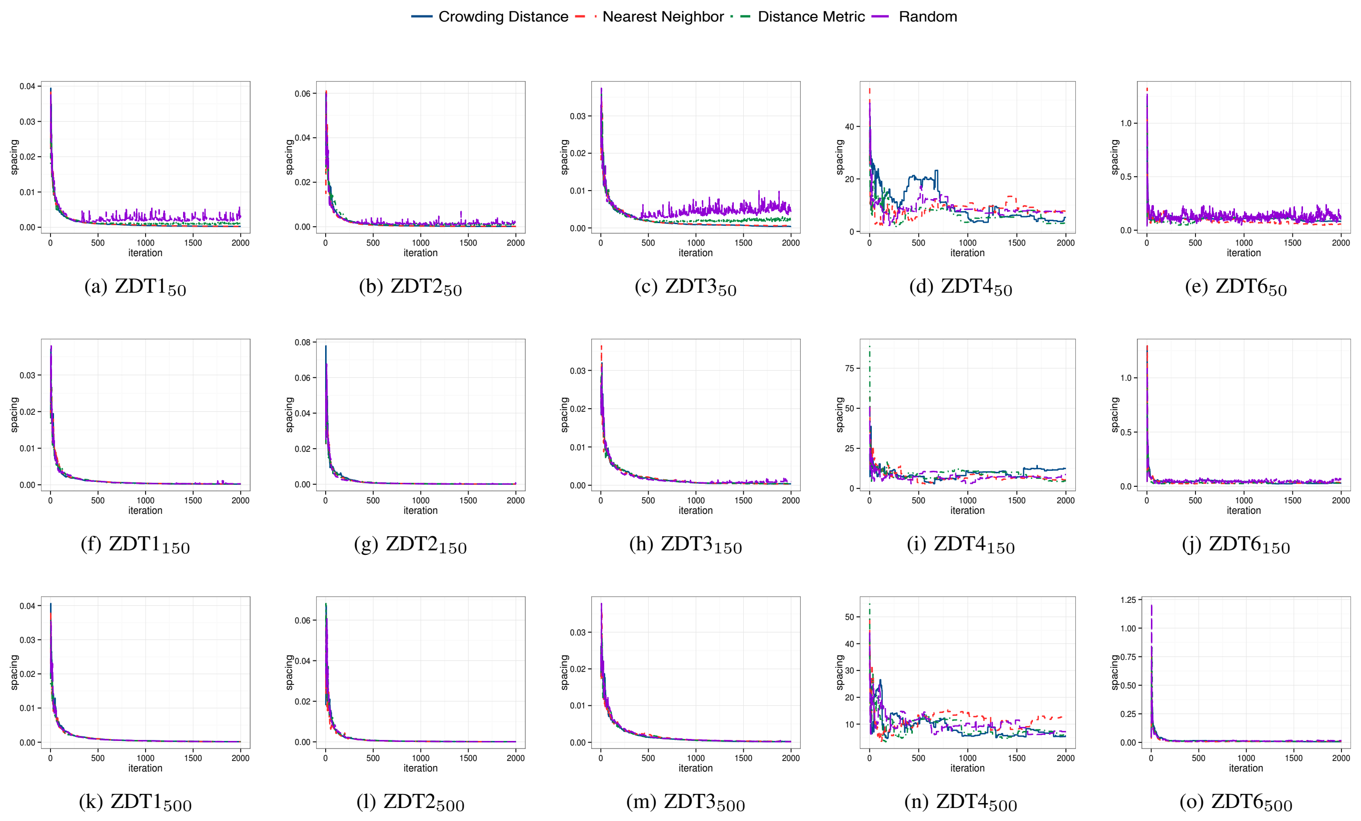


Fig. 5. VEPSO (Random) Spacing, S, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

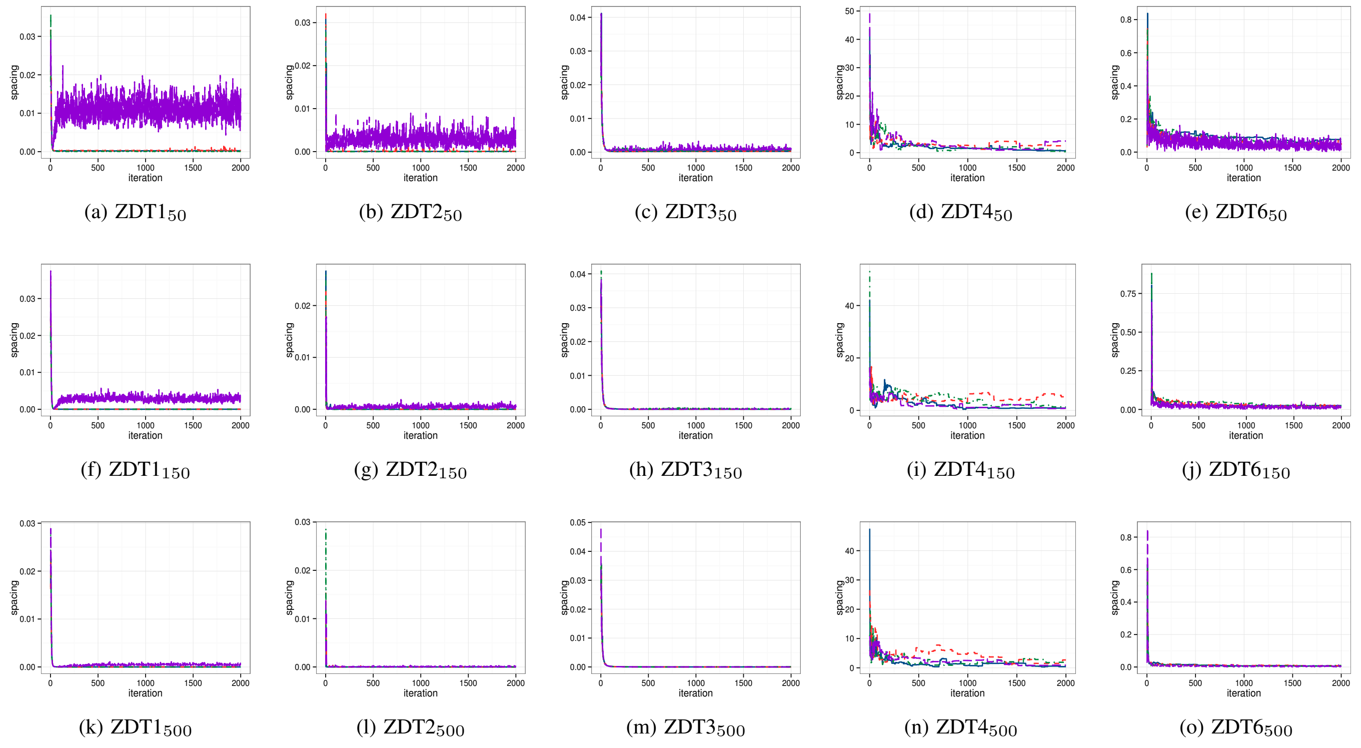
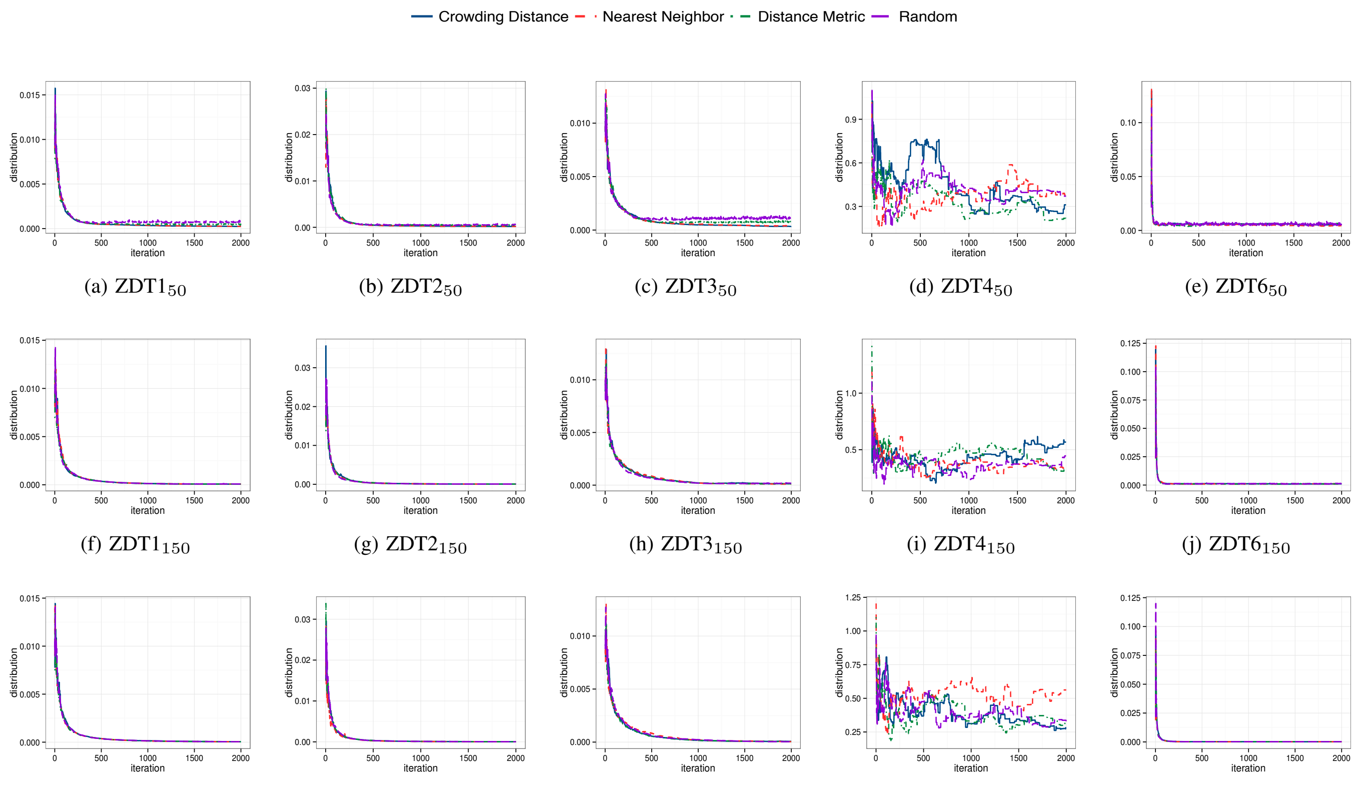


Fig. 6. VEPSO (PCXA) Spacing, S, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

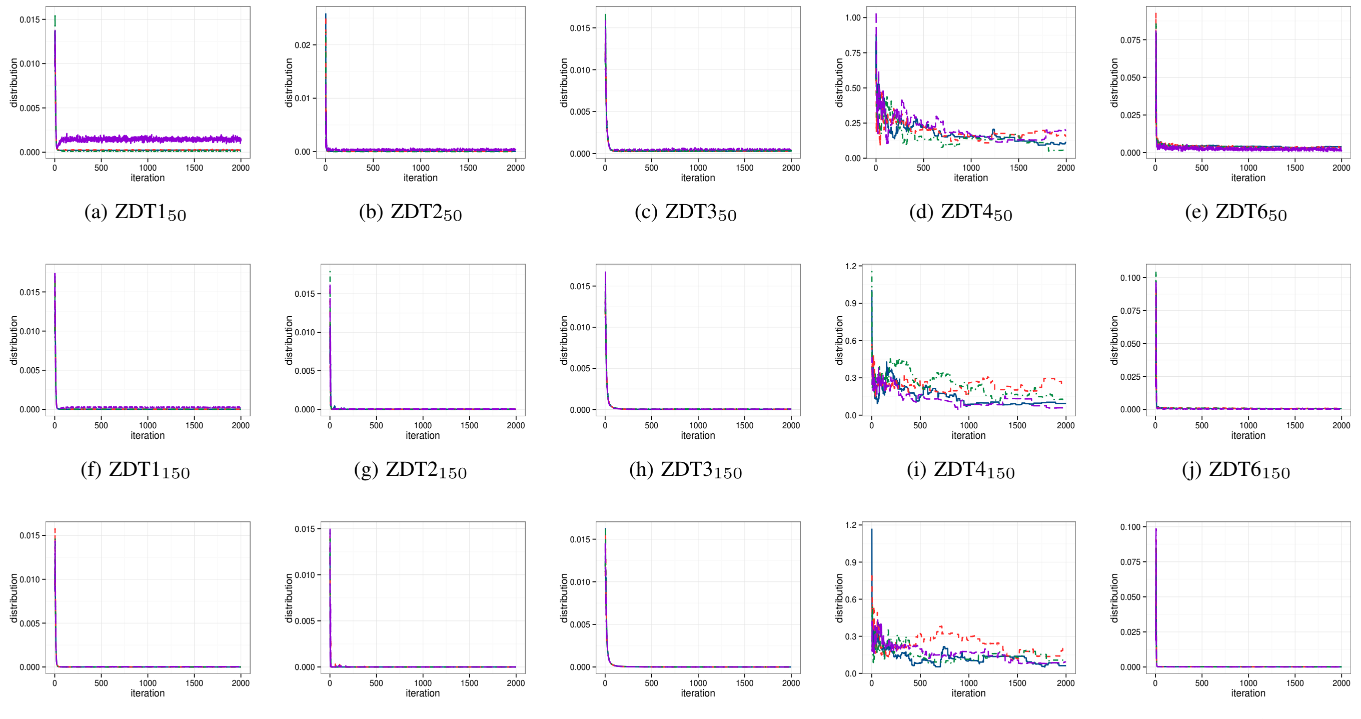


(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 7. VEPSO (Random) Distribution, D, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6

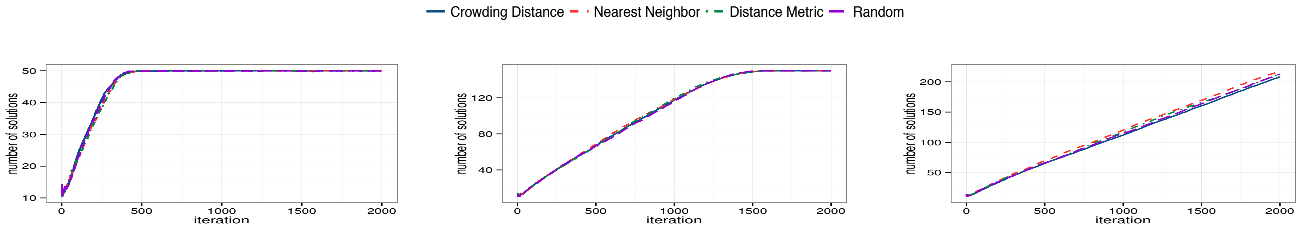


(a) ZDT1$_{50}$ (b) ZDT2$_{50}$ (c) ZDT3$_{50}$ (d) ZDT4$_{50}$ (e) ZDT6$_{50}$

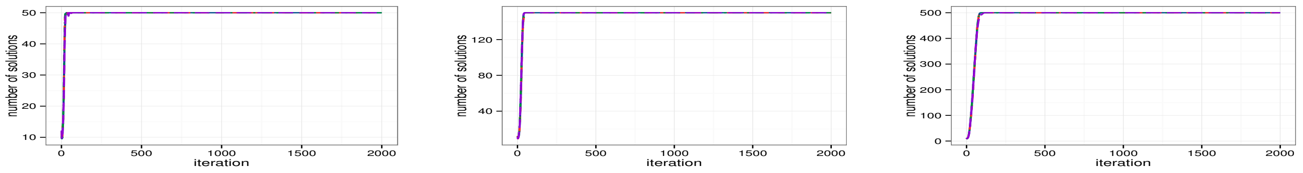
(f) ZDT1$_{150}$ (g) ZDT2$_{150}$ (h) ZDT3$_{150}$ (i) ZDT4$_{150}$ (j) ZDT6$_{150}$

(k) ZDT1$_{500}$ (l) ZDT2$_{500}$ (m) ZDT3$_{500}$ (n) ZDT4$_{500}$ (o) ZDT6$_{500}$

Fig. 8. VEPSO (PCXA) Distribution, D, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) VEPSO (Random) number of solutions$_{50}$ (b) VEPSO (Random) number of solutions$_{150}$ (c) VEPSO (Random) number of solutions$_{500}$



(d) VEPSO (PCXA) number of solutions$_{50}$ (e) VEPSO (PCXA) number of solutions$_{150}$ (f) VEPSO (PCXA) number of solutions$_{500}$

Fig. 9. VEPSO (Random) and VEPSO (PCXA) ZDT1 number of solutions over archive sizes 50, 150 and 500

best Δ when compared to the nearest neighbor, distance metric and random deletion approaches.

B. *Maximum Spread – $\bar{D}$*

$\bar{D}$ proved to be a much more stable measurement than Δ when comparing different archive sizes and deletion approaches. For VEPSO (Random) ZDT1 there was no notable difference in the $\bar{D}$ value after around 100 iterations, with only the nearest neighbor and distance metric deletion approaches showing a slightly improved $\bar{D}$ with an archive size of 50. The distance metric and nearest neighbor deletion approaches showed the most notable variance in the $\bar{D}$ value for ZDT3 over the archive sizes. Figs. 3(c), 3(h) and 3(m) depict the $\bar{D}$ for ZDT3 with archive sizes 50, 150 and 500. For smaller archive sizes the distance metric and nearest neighbor deletion approaches achieved slightly lower $\bar{D}$ values. For ZDT4, the $\bar{D}$ value is somewhat erratic, a general decrease up to around iteration 1000 can be noted.

For VEPSO (PCXA) the $\bar{D}$ value behavior was similar to VEPSO (Random) with the random deletion approach $\bar{D}$ values being slightly erratic. The performance in the case of ZDT4 was notably less erratic. ZDT3 showed a similar pattern with the distance metric and nearest neighbor deletion approaches showing a slight improvement in $\bar{D}$ for smaller archive sizes.

Overall, the $\bar{D}$ metric provided more stable results over the four deletion approaches and varying archive sizes. The $\bar{D}$ metric showed sensitivity on ZDT3 to the archive size when using the distance metric or nearest neighbor deletion approaches. No discernible pattern could be seen between the $\bar{D}$ value and when the archive size was reached.

C. *Spacing – S*

For VEPSO (Random) on ZDT1, ZDT2, and ZDT3, the S metric showed little to no sensitivity towards the choice of crowding distance, distance metric, and nearest neighbor deletion approaches over the different archive sizes. Only a slight sensitivity for the distance metric deletion approach can be noted on ZDT3 where a higher S value is obtained. Again, somewhat unstable S values can be noted for ZDT4.

For VEPSO (PCXA) on ZDT1 and ZDT2 the S metric showed extreme sensitivity when using the random deletion approach with an archive size of 50. Fig. 6(a) depicts the S value for ZDT1. Note the high variation in S values. In contrast to the Δ metric shown earlier, the S metric achieved lower values for the larger archive sizes. Similarly, for both VEPSO (Random) and VEPSO (PCXA), smaller S values were achieved for larger archive sizes on ZDT6. For VEPSO (PCXA) on ZDT1 through ZDT3, the S metric did not show sensitivity towards the choice of crowding distance, distance metric and nearest neighbor deletion approaches for a fixed archive size.

Overall it can be noted that S showed sensitivity towards the random deletion approach over all archive sizes but more so for smaller archive sizes.

D. *Distribution – D*

The D values for both VEPSO (Random) and VEPSO (PCXA) exhibited similar behavior. No notable sensitivity towards the deletion approach can be noted, except in the case of VEPSO (PCXA) on ZDT1 where sensitivity towards the random deletion approach can be noted when using an archive size of 50. For ZDT3 and ZDT6 smaller archive sizes led to higher D values for all the deletion approaches. D values for ZDT4 were, similar to the other metrics, erratic.

E. *Summary*

Overall it can be noted that D had the least sensitivity to the archive size and choice of deletion approach between all the tested metrics.

VI. Findings and Conclusions

This paper presented an analysis of the effect the choice of VEPSO archive size and deletion approach has on popular solution diversity metrics. VEPSO was used to evaluate four archive deletion approaches, namely crowding distance, distance metric, nearest neighbor, and random with three different archive sizes. Each of the archive sizes and deletion approach combinations was tested with both the random and PCXA knowledge transfer strategies. The algorithms' POF diversity was measured using four popular metrics, namely maximum spread $\bar{D}$, distribution D, spacing S and spread Δ.

The experimental results showed that all the tested metrics exhibited sensitivity towards the choice of deletion approach and archive size in at least some cases. The results indicate that Δ had the highest resolution of all the metrics but also had the highest sensitivity towards the choice of deletion approach and archive size. D was the least sensitive and can also be reasoned to have had the lowest resolution of all the metrics. S had the highest sensitivity towards the random deletion approach. $\bar{D}$ proved to be the most stable measurement as it only measures the extent of the POF and not the diversity of the solutions that make up the POF. Due to the usage of solutions from the archive, VEPSO (PCXA) was influenced more than VEPSO (Random) by the choice of archive deletion approach. Overall, the crowding distance and nearest neighbor deletion approaches generally yielded good results for the POF diversity metrics.

From the experimental results, it can be concluded that care must be taken when choosing the archive size and deletion approach as the impact of the POF diversity can be notable. Comparing diversity metrics between algorithms should not be done without taking the archive size into account.

A more in-depth analysis of the spacing, S, and solution distribution, D, metrics are currently underway. Future work will include an investigation into why the S and D metrics displayed less sensitivity to changes in the archive size and archive deletion approach.

References

[1] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

[2] J. Kennedy and R. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.

[3] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through Particle Swarm Optimization," *Natural Computing*, vol. 1, no. 2, pp. 235–306, 2002.

[4] ——, "Particle swarm optimization method in multiobjective problems," in *Proceedings of the ACM Symposium on Applied Computing*, 2002, pp. 603–607.

[5] K. E. Parsopoulos, D. Tasoulis, and M. N. Vrahatis, "Multiobjective Optimization using Parallel Vector Evaluated Particle Swarm Optimization," in *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications*, vol. 2, 2004, pp. 823–828.

[6] M. R. Sierra and C. A. Coello Coello, "Improving PSO-Based Multiobjective Optimization Using Crowding, Mutation and -Dominance," in *Evolutionary Multi-Criterion Optimization: Third International Conference, 2005. Proceedings*, C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, vol. 3410, pp. 505–519.

[7] A. Nebro, J. Durillo, J. García-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, "SMPSO: A New PSO-based Metaheuristic for Multi-objective Optimization," in *Proceedings of the IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making*, no. 2, 2009, pp. 66–73.

[8] J. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, 1985, pp. 93–100.

[9] W. Matthysen, A. P. Engelbrecht, and K. M. Malan, "Analysis of Stagnation Behavior of Vector Evaluated Particle Swarm Optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium*, 2013, pp. 155–163.

[10] K. R. Harrison, A. P. Engelbrecht, and B. M. Ombuki-Berman, "A Scalability Study of Multi-Objective Particle Swarm Optimizers," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2013, pp. 189–197.

[11] M. Helbig and A. P. Engelbrecht, "Archive management for dynamic multi-objective optimisation problems using vector evaluated particle swarm optimisation," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2011, pp. 2047–2054.

[12] J. Grobler, "Particle swarm optimization and differential evolution for multi objective multiple machine scheduling," Master's thesis, University of Pretoria, 2009.

[13] K. Deb, A. Anand, and D. Joshi, "A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization," *Evolutionary Computation*, vol. 10, no. 4, pp. 371–395, 2002.

[14] K. R. Harrison, B. M. Ombuki-Berman, and A. P. Engelbrecht, "Knowledge transfer strategies for vector evaluated particle swarm optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*, 2013, pp. 171–184.

[15] T. Bartz-Beielstein, P. Limbourg, J. Mehnen, K. Schmitt, K. E. Parsopoulos, and M. N. Vrahatis, "Particle Swarm Optimization for Pareto Optmization with Enhanced Archiving Techniques," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2003, pp. 1780–1787.

[16] D. A. Van Veldhuizen, J. B. Zydallis, and G. B. Lamont, "Considerations in engineering parallel multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 144–173, 2003.

[17] M. Laumanns, L. Thiele, E. Zitzler, and K. Deb, "Archiving With Guaranteed Convergence And Diversity In Multi-objective Optimization." in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 439–447.

[18] Y. Jin, M. Olhofer, and B. Sendhoff, "Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How?" in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001, pp. 1042–1049.

[19] J. D. Knowles and D. W. Corne, "Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 100–116, 2003.

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[21] S. Kukkonen and K. Deb, "A Fast and Effective Method for Pruning of Non-Dominated Solutions in Many-Objective Problems," *Parallel Problem Solving from Nature PPSN IX*, pp. 554–562, 2006.

[22] S. Tiwari, G. M. Fadel, P. Koch, and K. Deb, "AMGA : An Archive-based Micro Genetic Algorithm for," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2008, pp. 729–736.

[23] F.-A. Fortin and M. Parizeau, "Revisiting the NSGA-II Crowding-Distance Computation," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2013, pp. 623–630.

[24] E. Zitzler, "Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications," Doctorate Thesis, Swiss Federal Institute of Technology Zurich, 1999.

[25] K. C. Tan, E. F. Khor, and T. H. Lee, *Multiobjective Evolutionary Algorithms and Applications*. Springer-Verlag London, 2005.

[26] C. K. Goh and K. C. Tan, "An Investigation on Noisy Environments in Evolutionary Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 3, pp. 354–381, 2007.

[27] J. R. Schott, "Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization," MSc Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1995.

[28] G. Pampara, A. Engelbrecht, and T. Cloete, "CIlib: A collaborative framework for Computational Intelligence algorithms-Part I," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, 2008, pp. 1750–1757.

[29] R. Eberhart and Y. Shi, "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, no. 2, 2000, pp. 84–88.

[30] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.