RB-MOPSO: A PSO algorithm based on reference points

Olacir R. Castro Jr
Computer Science's Department
Federal University of Paraná
Curitiba, Paraná
olacirjr@gmail.com

André Britto
Computer Science's Department
Federal University of Paraná
Curitiba, Paraná
andrebc@inf.ufpr.br

Aurora Pozo
Computer Science's Department
Federal University of Paraná
Curitiba, Paraná
aurora@inf.ufpr.br

***Abstract*—Many-objective problems refer to problems containing large number of objective functions to be optimized, typically more than three. As most existing algorithms based on Pareto dominance are not efficient in handling this kind of problem, researchers have been working on alternatives to overcome these limitations. Recently, Deb et al. proposed an improved NSGA-II, the principal characteristic of which is the use of reference points to obtain a larger covered portion of the Pareto front. Inspired by the results of the improved NSGA-II, in this paper we propose a new algorithm based on Particle Swarm Optimization grounded on these concepts. The algorithm, called RB-SMPSO, presents as features improved mechanisms for (i) the update of the external archive to consider the reference points, and (ii) the selection of social leaders. In order to validate the proposal, a comparative study is presented considering a state of art MOPSO-based algorithm (Speed-constrained Multi-Objective Particle Swarm Optimization, SMPSO) using two different leader selection methods. All algorithms are evaluated using well-known scalable problems with 2, 3, 5, 10, 15 and 20 objectives, respectively. The results point out that RB-SMPSO presents the better overall performance between the compared approaches.**

***Keywords*-Particle Swarm Optimization, Multi-objective, Many-objective, Optimization**

I. Introduction

Many-objective problems are a kind of multi-objective problems which have more than three objective functions to be optimized. Many real world applications need to consider and optimize more than two or three objective functions [1], but due to lack of suitable algorithms they normally are down-sized to two or three objectives and solved [2].

The interest in solving these problems has grown because most existing optimization algorithms are not efficient in handling many objectives. This mainly occurs because the proportion of non-dominated solutions in a population increases with the number of objectives, which decrease the selection pressure towards the Pareto front. Moreover, the number of points to accurately represent the front increases exponentially with the number of objectives.

In spite of the fact that the computational resources available today allows the use of large population sizes, it is certainly difficult for a decision-maker to consider a large number of trade-off solutions. These issues require diversity techniques to yield a better distribution of the solutions [2].

Recently a new many-objective optimizer that tackles these challenges was proposed, it is called M-NSGA-II [2] and has the differential of adopting a selection mechanism based on a predefined set of reference points emphasizing the members of the non-dominated population closer to each reference point.

A promising meta-heuristic to optimize many objectives is the Multi-Objective Particle Swarm Optimization (MOPSO), which is a multi-objective population based meta-heuristic inspired by the bird flocking behavior. In this algorithm each possible solution, called particle, uses simple local rules to guide its actions, and through interactions with the group, the entire population, called swarm, achieves its objectives.

This study presents a new MOPSO algorithm called RB-SMPSO which incorporates the reference points idea from M-NSGA-II including two mechanisms to improve its performance. One of them incorporates a preference relation on the archiving method based on reference points, where a solution closer to a reference point is preferred over a farther one. The other mechanism is incorporated in the leader selection method, where instead of the particle selecting its leader, the available leaders select the closest particles to lead. These features aim to improve both the convergence and the diversity of the search.

Unlike other studies using reference points to guide the search [3], [4], in this work the objective is not to reduce the portion of the Pareto front to be explored. Here the reference points are used to keep well spread leader candidates to use them as guides to make the particles cover a larger portion of the Pareto front.

The remainder of this paper is organized as follows: The general description of the M-NSGA-II algorithm is presented in section II. The RB-SMPSO algorithm is presented in section III. A brief explanation about the benchmark problems and performance metrics, as well as the empirical study performed to compare the algorithms is presented in Section IV, and section V presents the conclusions.

II. MANY-OBJECTIVE NSGA-II

A Multi-Objective Problem (MOP) involves the simultaneous optimization of two or more objective functions. These objectives are usually in conflict, which means that MOPs do not have a single optimum solution, but a set of them. Pareto optimality theory is used to find this set of solutions [5].

MOPs presenting more than three objectives are called Many-Objective Problems (MaOPs), and the area which tackles the issues related to this kind of problem is the many-objective optimization.

M-NSGA-II [2] is a many-objective optimizer based on the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [6]. M-NSGA-II has a special selection mechanism based on reference points. To better understand the M-NSGA-II, first the NSGA-II is briefly explained.

The NSGA-II was proposed in [6] and is one evolutionary algorithm frequently used nowadays to optimize multi-objective problems [1], [2]. This popularity is mainly due to its ease of implementation and quality of the approximated frontiers generated for many problems.

Its basic operation is the following: Firstly a population P_0 is randomly created, then it is sorted in non-dominance levels $(F_1, F_2, ...)$, F_1 being composed of all the non-dominated solutions with respect to the total population, F_2 containing the solutions non-dominated after the removal of the solutions that comprise F_1 and so on (with $|F_1| < |P|$).

Binary tournament selection, recombination and mutation operators are used to create a child population Q_0 of size N. From the first generation onward, the population is generated using a combination between the parent and the child populations. The procedure for a generation $t > 0$ is shown in the Algorithm 1.

Algorithm 1: NSGA-II pseudo code

$R_t = P_t \cup Q_t$
F=$non\_dominated\_sort(R_t)$
while $|P_{t+1}| < N$ **do**
  $crowding\_distance\_assignment\ (F_i)$
  $P_{t+1} = P_{t+1} \cup F_i$
end while
$sort(P_{t+1}, \geq_n)$
$P_{t+1} = P_{t+1}[0 : N])$
Q_{t+1}= $make\_new\_population\ (P_{t+1})$
$t = t + 1$

In Algorithm 1, firstly a new combined population $R_t = P_t \cup Q_t$ is formed and classified in non-dominance levels. The new parent population P_{t+1} is formed by adding solutions from F_1 until the number of solutions in P_{t+1} exceed N. Next, the solutions from the last accepted level F_l are sorted using the niched comparison operator $(\geq_n)$, which compares the solutions according to the non-dominance level (i_n) and the crowding distance (i_d), with this operator, given two solutions $\vec{i}$ and $\vec{j}$ is defined that $\vec{i}$ is preferable to $\vec{j}$ denoted by $(\vec{i} \geq_n \vec{j})$, if $(i_n < j_n)$ or $((i_n = j_n)$ and $(i_d > j_d))$. In this case the first N solutions are selected, so $|P_{t+1}| = N$. Then this population is selected and, crossover and mutation procedures are applied to create the new population Q_{t+1}.

Besides the same basic steps of NSGA-II, M-NSGA-II utilizes a set of reference points created in an hyperplane using the procedure detailed in [7], [2] as additional criteria for the selection procedure. In the new algorithm, all members of the population from the level 1 until $(l-1)$ are included in P_{t+1} and to choose the remaining $k = N - \sum_{i=1}^{l-1} F_i$ members of the last front F_l, all members from level 1 to l are considered, constituting the set S_t.

To determine the degree of crowding of the solutions in S_t, every solution is projected in the created hyperplane and associated with its closest reference point, then each reference point will have a set of associated solutions. With H reference points, ideally a population of size N should have an average of $P_{ideal} = \frac{N}{H}$ solutions per reference point. If any set close to a reference point has less than P_{ideal} solutions, this point is said deficient, and a deficiency count $(P_{ideal} - P_{actual})$ is calculated, in which P_{actual} is the actual number of solutions associated to this reference point.

Thereafter, the most deficient reference point is found, the solution in F_l closest to this point is chosen and the deficiency count of this point is reduced by one. The next more deficient reference point is found and the solution closest to it is identified and so on. This process continues until all members from F_l are chosen to fill the N members of P_{t+1}. At the end, the cluster for each reference point is checked and if it is empty, the reference point is considered defunct and its allocated ideal cluster size P_{ideal} is distributed among its neighboring reference points. Normally this happens in disconnected fronts, in regions where is not possible to find non-dominated solutions.

After P_{t+1} is formed, it is used to create a new offspring population Q_{t+1} by applying tournament selection, recombination and mutation operators. The tournament selection operator considers two solutions from P_{t+1} and selects the better one. In this case a hierarchy of considerations is used. First, if a solution belongs to a better non-dominated level than the other, the former is chosen. Second, if both solutions belong to the same non-domination level, but they lie on clusters of different reference points, the deficiency counts of their respective reference points are compared and the solution associated with a larger deficiency count is chosen. Third, if both solutions lie on the same non-domination level and, also on the same cluster, their distance to the reference point is computed and the closer is chosen.

Once all the reference points are well spread, the members of the population P_{t+1} are also expected to be well spread. Therefore, a recombination operator which creates offspring solutions close to the parents is used. After the offspring population Q_{t+1} is created, a new combined population is formed and the described procedure is applied again.

III. Reference Based SMPSO

This section describes the RB-SMPSO algorithm inspired by the ideas of M-NSGA-II. The algorithm is based on the Speed-constrained Multi-Objective Particle Swarm Optimization (SMPSO) [8] and presents as features improved mechanisms for (i) the update of the external archive to consider the reference points, and (ii) the selection of social leaders. First, the general structure of a MOPSO is presented.

MOPSO is a meta-heuristic based in the well-known Particle Swarm Optimization (PSO) [9] modified to deal with MOPs. Extending PSO to MOPSO consist to adapt its leader mechanism to consider a set of non-dominated solutions which are equally important, besides storing this set of solutions in a repository.

To maintain the repository (or external archive), the Pareto dominance relation is used. This archive keep the best non-dominated solutions found so far, which are used as leader candidates and returned as final approximated Pareto set at the end of the search. However, the repository size increase very quickly and must be bounded (or pruned), which makes necessary the use of an additional criterion to decide which non-dominated solutions to retain when the archive becomes full, this criterion is used in the prune procedure.

The basic steps of a MOPSO algorithm are: initialization of the swarm, initialization of the external archive, evaluation and selection of the global leaders, position and velocity update, mutation, update of the personal leader and update of the external archive [8].

Each particle $\vec{p}$, at a time step t, has a position $\vec{x}$ that represents a possible solution. The position of the particle, at time $t+1$, is obtained by adding its velocity, $\vec{v}$ to $\vec{x}$.

The velocity of a particle $\vec{p}$ is based on the best position already found by this particle ($\vec{p}_{best}$), and the best position already found by its set of neighbors ($\vec{p}_{leader}$) that is a leader from the repository. The velocity is defined as follows:

$$\vec{v}(t) = \varpi \cdot \vec{v}(t-1) + (C_1 \cdot R_1) \cdot (\vec{p}_{best}(t) - \vec{x}(t)) + (C_2 \cdot R_2) \cdot (\vec{p}_{leader}(t) - \vec{x}(t)) \quad (1)$$

R_1 and R_2, in (1), are random values in the range [0, 1]. $\vec{p}_{best}$ represent the particle best position, and $\vec{p}_{leader}$, the particles global best position. Constants C_1 and C_2 indicate how much each component influences on velocity. The coefficient ϖ is the inertia of the particle, and controls how much the previous velocity affects the current one.

The MOPSO algorithm used as base in this work is the Speed-constrained Multi-Objective Particle Swarm Optimization (SMPSO) [8], which consists of a MOPSO that uses a velocity constriction mechanism to limit the velocity of the particles when it becomes too high.

Given the differences between the NSGA-II and MOPSO, some adaptations were needed to implement the approach of reference points in this study. Here, the reference points are used in the prune procedure as additional criteria for selecting the particles.

In the new prune procedure proposed here, until the repository reach the limit, all the non-dominated solutions found are added. Once the limit is reached, if a new non-dominated solution is found and it does not dominate any solution of the repository, the following operations are executed:

1) Find the reference point $\vec{r}_c$ closer to the new solution $\vec{f}(\vec{x}_{new})$.
2) Find the set of solutions (S_c) in the repository closer to $\vec{r}_c$ than any other reference point (associated to $\vec{r}_c$).
3) Find the solution in S_c farther to $\vec{r}_c$ ($\vec{f}(\vec{x}_{ar})$).
4) If $|S_c| < \frac{A}{H}$ then include $\vec{f}(\vec{x}_{new})$ and remove the farther solution from the reference point with more solutions associated to it.
5) If $|S_c| \geq \frac{A}{H}$ then verify which solution ($\vec{f}(\vec{x}_{new})$) or ($\vec{f}(\vec{x}_{ar})$) is closer to $\vec{r}_c$ and keep/include the closer solution in the repository and discard the farther.

Where A represent the number of solutions in the archive, and H is the number of reference points.

In the steps 1 to 3 the distance is calculated as follows: firstly both solutions are translated to a normalized hyperplane using the Equation 2.

$$f'_i(\vec{x}) = \frac{f_i(\vec{x})}{\sum_{j=1}^{m} f_j(\vec{x})} \quad (2)$$

Where $\vec{f}'(\vec{x})$ is an objective vector translated to the normalized hyperplane, $f_i(\vec{x})$ is the i-th objective of the vector $\vec{f}(\vec{x})$, $\vec{x}$ represent one solution and m is the number of objectives. In this equation the main idea is to draw a line segment from the origin to the particle to be translated and identify the point in which this line intercepts the hyperplane in $\sum_{i=1}^{m} \vec{f}_i(\vec{x}) = 1$.

After the calculation of the translated objective vector of both solutions though Equation 2, the Euclidean distance between them is used in the steps 1 to 3. This procedure is done to better identify the region where the solution is located, avoiding the distance between the particle and the front to interfere in the calculation. In the step 4, the Euclidean distance is used directly, since in this case it is important to take the closeness to the front in consideration.

These operations are performed to increase the portion of the front covered guaranteeing the association of $\frac{A}{H}$ non-dominated solutions to each reference point. This technique requires that the number of particles in the swarm always be greater than or equal to the number of reference points and the repository size.

The design of the algorithm aims that all the solutions in the repository converge to the true Pareto front, as well as be widely distributed over it. Thus, it is important to have a leader selection mechanism that emphasizes that by creating pressure over the entire front, and without causing

the particles to move unnecessarily, since this behavior may result in loss of convergence [10].

In this new leader selection method, instead of each particle selecting its own leader, the opposite happens. Each leader candidate selects the particles closest to it to lead, avoiding the unnecessary movements of the particles.

Firstly the number of particles per leader is calculated as $n = \frac{P}{A}$, where n represent the number of particles per leader, P is the population size, and A is the number of candidates within the repository. Usually, it is chosen $P = A$, resulting in $n = 1$ when the repository is full, however population and repository sizes can be different, being mandatory $P \geq A$.

Next, each leader calculates its distance to every particle using the Equation 2 and selects the n closest particles that do not have leader yet. With this procedure is expected that the leaders choose particles in its respective regions and that the particles cover the entire front.

As the RB-SMPSO algorithm implement its own leader selection method as presented, two other methods are used in this work with the SMPSO algorithm in the comparative study: The Sigma method [11], which presents good results at the literature [10], and the binary tournament using Crowding Distance [6], which is the original leader selection method in SMPSO. Explanations about these methods were not included in this work for brevity.

IV. Empirical Study

This section presents the empirical study done in this paper to assess the performance of the RB-SMPSO algorithm. The comparison use the original SMPSO, which have its leader selection based on binary tournament considering the Crowding Distance (CD) [6], and a different version of this algorithm which incorporates one of the best leader selection methods found in [10], named Sigma method [11].

The benchmark problems used in these experiments are part of the well-known DTLZ class of multi-objective optimization problems [12]. In this paper two problems of this family are used, namely DTLZ2 and DTLZ4.

To properly assess the performance of the compared algorithms, two quality indicators are used. The first of them is the well-known Generational Distance (GD). This indicator measures how far, on average, the Pareto front approximation achieved using an algorithm (PF_{known}) is from the true Pareto front (PF_{true}), measuring the *convergence* towards the true Pareto front. A value of zero indicates PF_{known} = PF_{true} [5].

The second metric used here, is the Inverted Generational Distance (IGD). This metric measure if PF_{known} is well distributed over PF_{true}, representing mainly the *diversity* of PF_{known}. It is calculated basically the same way as the original GD, but the roles of PF_{true} and PF_{known} are inverted in the GD definition [5]. Here the PF_{true} used is represented by the reference points targeted in the Pareto surface, as done in [2].

To assess if there are significant differences in the results found for each utilized measure, the results are submitted to the Friedman statistical test, which is a non-parametric statistical test used to detect differences between data sets [13]. In this study the significance level used was 0.05, and the three algorithms were compared per objective number on each test problem.

The basic parameters of both algorithms were defined according to [8]. The number of iterations used for the problems DTLZ2 and DTLZ4 were 200 and 500 respectively, the population always kept in 300 and the size of the repository also fixed in 300. In both problems where used objectives varying between two and twenty. The number of divisions per axis (p variable) is defined according to [2].

The CD and Sigma leader selection methods were used with the standard archiving method of SMPSO (Crowding Distance), while the new reference based approach implements its own leader and archiving methods and do not need additional parameters. Each configuration was executed 30 times with each number of objectives.

The results of the experiments are presented through charts where each curve represents one different algorithm for the problems DTLZ2 and DTLZ4 respectively. Each point is the mean of the quality indicator values for each number of objectives. The algorithms showing the best results and no statistical difference according to Friedman test for each number of objectives are presented in Table I.

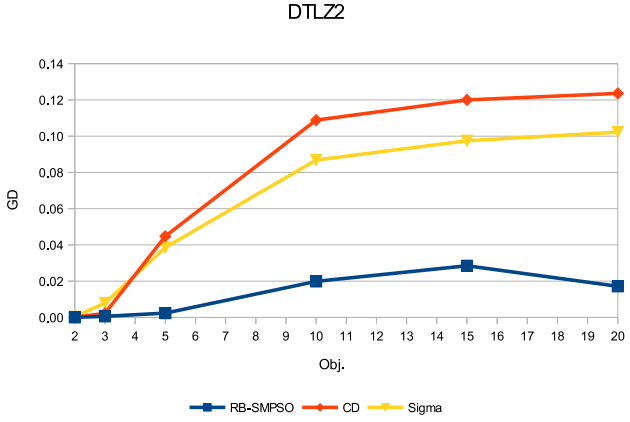

Figure 1. GD values for the DTLZ2 problem.

Fig. 1 shows the GD values for the DTLZ2 problem. For two objectives is not possible to see differences of performance between the algorithms, while for three objectives the SMPSO with Sigma show a slightly worse performance. From five to twenty objectives the differences become more visible, with the RB-SMPSO presenting the better convergence followed by the SMPSO with Sigma, while the original SMPSO presents the worst results.

The Friedman test for GD values from the DTLZ2 problem is presented at Table I. RB-SMPSO is best for all numbers of objectives with exception of two objectives,

Table I
COMPARISON OF MOPSO ALGORITHMS FOR DTLZ2 AND DTLZ4 ACCORDING TO FRIEDMAN TEST.

Prob	Obj	Best Algorithms	
		GD	IGD
DTLZ2	2	CD	CD
	3	RB-SMSPO	RB-SMSPO
	5	RB-SMSPO	RB-SMSPO
	10	RB-SMSPO	RB-SMSPO
	15	RB-SMSPO	RB-SMSPO
	20	RB-SMSPO	RB-SMSPO and Sigma
DTLZ4	2	RB-SMSPO	CD
	3	RB-SMSPO and CD	RB-SMSPO and CD
	5	RB-SMSPO	CD
	10	RB-SMSPO	RB-SMSPO, CD and Sigma
	15	RB-SMSPO	RB-SMSPO
	20	RB-SMSPO	Sigma

where the original SMPSO presents the better performance.

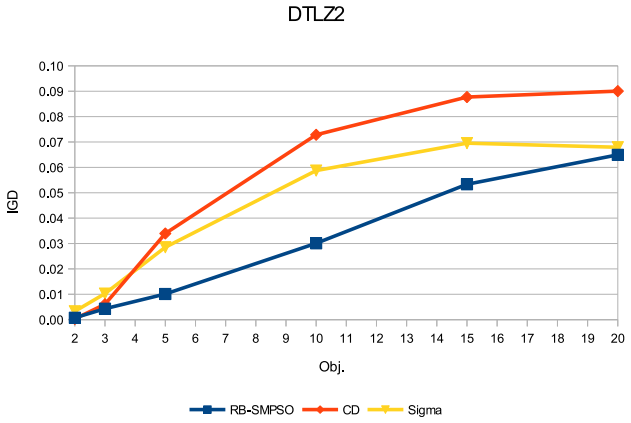

Figure 2. IGD values for the DTLZ2 problem.

The IGD values for the DTLZ2 problem are presented in Fig. 2 representing the diversity obtained with the algorithms. The results have similar behavior to the GD values. The Sigma presents a performance slightly worse for two and three objectives, while the RB-SMPSO and the CD presents performance very close. For five to twenty objectives the RB-SMPSO present the better performance, followed by the SMPSO with Sigma and the original SMPSO with CD presenting the worse results.

The IGD statistical tests for the DTLZ2 problem displayed at Table I confirmed the data of the chart, with the RB-SMPSO presenting the better performance for all of number of objectives with exception of two objectives, and presenting no statistically significant differences from the Sigma for twenty objectives.

Fig. 3 shows the GD values obtained for the problem DTLZ4, in which for two and three objectives the RB-SMPSO and the original SMPSO presented the better performances with the Sigma slightly worse. For five to twenty objectives the RB-SMPSO presents the better results, while the Sigma performs worse than CD for five objectives. CD and Sigma perform very close for ten objectives, and for fifteen and twenty objectives the Sigma stands out.

The GD statistical tests for the DTLZ4 problem shown at the Table I points the RB-SMPSO as better algorithm for all

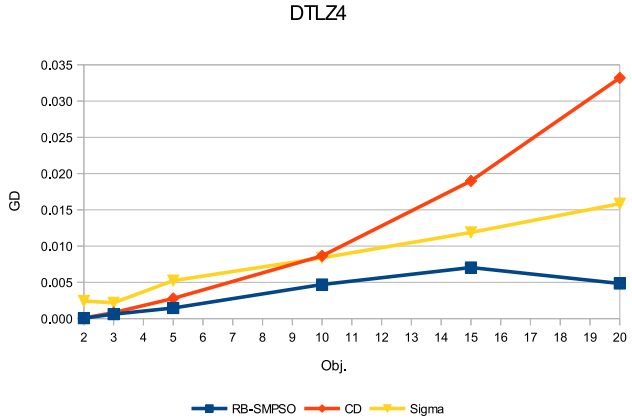

Figure 3. GD values for the DTLZ4 problem.

of number of objectives. For three objectives, the SMPSO with CD and RB-SMPSO are equivalent.

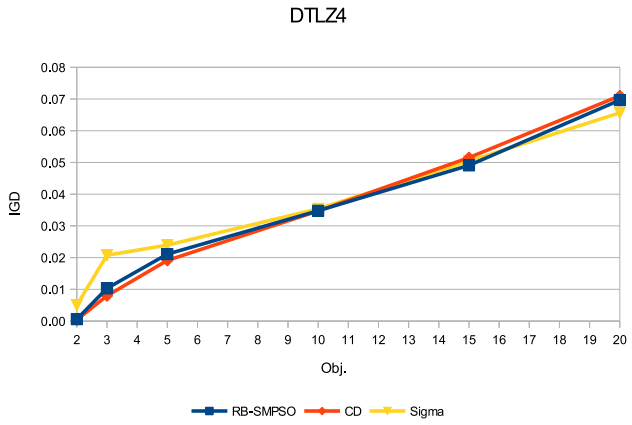

Figure 4. IGD values for the DTLZ4 problem.

The IGD values for the DTLZ4 problem are presented at Fig. 4. It is possible to observe very small differences between the values obtained with the algorithms. For two objectives the original SMPSO with CD and the RB-SMPSO presents results very close, while the Sigma performs worse. For three and five objectives the SMPSO with CD present a slightly better performance, followed by the SMPSO, with the Sigma presenting the worse results. For ten objectives, the three algorithms presents about the same results, while for fifteen the RB-SMPSO presents results slightly superior, followed by Sigma and at last by SMPSO with CD. For twenty objectives the Sigma stands out, followed by the RB-SMPSO and the original SMPSO with the worst results. The DTLZ4 IGD statistical tests are presented at Table I.

As the data presented shows, neither of the algorithms is better for all numbers of objectives. These results indicates that the new RB-SMPSO have a good overall performance being able to maintain a good convergence of solutions in both problems and a better diversity in the problem DTLZ2

with a diversity comparable to the other algorithms in the DTLZ4 problem.

In the DTLZ2 problem, which is easier to optimize, the solutions obtained with the RB-SMPSO were more influenced by the reference points, as it is designed to be, achieving a good convergence especially for many objectives where the particles are more distant to each other favoring the convergence. The diversity was worsening as the number of objectives increases mostly because of the bigger spacing between the reference points as the objective space grows, concentrating the solutions closer to the reference points.

In the DTLZ4 problem, the RB-SMPSO achieved a better convergence, especially for many objectives, presenting a behavior similar to the observed for DTLZ2, however the diversity hardness imposed by this problem causes difficulties to obtain a good diversity for all of the algorithms.

V. Conclusion

This work presented a new MOPSO approach based on reference points aiming to better cover the entire Pareto front with a limited number of solutions. Its performance was compared with the SMPSO algorithm using two different leader selection methods.

The empirical analysis was focused in comparing the performance of the different approaches as the number of objectives scales up. The experiments were conducted with two well-known many-objective problems, and the objectives were varied between two and twenty. The convergence and the diversity of the obtained approximated Pareto fronts were analyzed through two quality indicators: Generational Distance, and Inverted Generational Distance. The Friedman statistical test was used to detect differences between the algorithms results.

From the data collected in the experiments, is possible to deduce that the proposed algorithm here presented has better overall performance than the classic approaches, mostly because of its idea of spread the particles across the entire objective space. These good results incentives further improvements on its main idea, especially to achieve a superior diversity performance on the DTLZ4 problem, which is characterized by presenting challenges in this sense.

Future works include the implementation of methods to improve the diversity in scenarios with increased pressure to small regions, like the DTLZ4. Additionally, more benchmark problems and metrics could be used to better validate the results obtained.

References

[1] E. Hughes, "Radar waveform optimisation as a many-objective application benchmark," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. Springer Berlin / Heidelberg, 2007, vol. 4403, pp. 700–714.

[2] K. Deb and H. Jain, "An Improved NSGA-II Procedure for Many-Objective Optimization Part I: Solving Problems with Box Constraints," Indian Institute of Technology, Kanpur, Tech. Rep., 2012.

[3] D. W. Corne and J. D. Knowles, "Techniques for highly multiobjective optimisation: some nondominated points are better than others," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, ser. GECCO '07. New York, NY, USA: ACM, 2007, pp. 773–780. [Online]. Available: http://doi.acm.org/10.1145/1276958.1277115

[4] U. K. Wickramasinghe and X. Li, "Integrating user preferences with particle swarms for multi-objective optimization," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, ser. GECCO '08. New York, NY, USA: ACM, 2008, pp. 745–752. [Online]. Available: http://doi.acm.org/10.1145/1389095.1389237

[5] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: Nsga-ii," in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, ser. PPSN VI. London, UK: Springer-Verlag, 2000, pp. 849–858.

[7] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, Mar. 1998.

[8] A. Nebro, J. Durillo, J. Garcia-Nieto, C. Coello Coello, F. Luna, and E. Alba, "Smpso: A new pso-based metaheuristic for multi-objective optimization," in *Computational intelligence in multi-criteria decision-making, 2009.*, Apr. 2009, pp. 66 –73.

[9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, nov/dec 1995, pp. 1942 –1948 vol.4.

[10] O. R. Castro Jr, A. Britto, and A. Pozo, "A comparison of methods for leader selection in many-objective problems," in *IEEE Congress on Evolutionary Computation*, Jun. 2012, pp. 1 –8.

[11] S. Mostaghim and J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (mopso)," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003. SIS '03*, Apr. 2003, pp. 26 – 33.

[12] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, ser. Advanced Information and Knowledge Processing, A. Abraham, L. Jain, and R. Goldberg, Eds. Springer London, 2005, pp. 105–145.

[13] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.