Neurocomputing 180 (2016) 68–78

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

C-Multi: A competent multi-swarm approach for many-objective problems

Olacir R. Castro Jr.[a,*], Roberto Santana [b], Aurora Pozo [a]

[a] Computer Science Department, Federal University of Paraná (UFPR), PO 19081, 81531-970 Curitiba, Brazil
[b] Intelligent Systems Group, Department of Computer Science and Artificial Intelligence, University of the Basque Country (UPV/EHU), Paseo Manuel de Lardizabal 1, 20080 San Sebastián, Guipúzcoa, Spain

ARTICLE INFO

Article history:
Received 27 March 2015
Received in revised form
18 June 2015
Accepted 19 June 2015
Available online 11 November 2015

Keywords:
Particle swarm optimization
Many-objective
Estimation of distribution algorithm
Competent algorithm

ABSTRACT

One of the major research topics in the evolutionary multi-objective community is handling a large number of objectives also known as many-objective optimization problems (MaOPs). Most existing methodologies have demonstrated success for problems with two and three objectives but face significant challenges in many-objective optimization. To tackle these challenges, a hybrid multi-swarm algorithm called C-Multi was proposed in a previous work. The project of C-Multi is based on two phases; the first uses a unique particle swarm optimization (PSO) algorithm to discover different regions of the Pareto front. The second phase uses multiple swarms to specialize on a dedicate part. On each sub-swarm, an estimation of distribution algorithm (EDA) is used to focus on convergence to its allocated region. In this study, the influence of two critical components of C-Multi, the archiving method and the number of swarms, is investigated by empirical analysis. As a result of this investigation, an improved variant of C-Multi is obtained, and its performance is compared to I-Multi, a multi-swarm algorithm that has a similar approach but does not use EDAs. Empirical results fully demonstrate the superiority of our proposed method on almost all considered test instances.

1. Introduction

Recently, as many real-world applications involve four or more objectives [1], the evolutionary multi-objective optimization (EMO) community has focused its attention to handle large number of objectives (Many-objective optimization problems, MaOPs). Since, several studies pointed that Pareto based algorithms scale poorly in MaOPs [2–4] because of the increase in the number of non-dominated solutions which deteriorates the selection pressure compromising the convergence to the Pareto front and diversity of the solutions.

In a previous work [5] we presented a hybrid algorithm called C-Multi to deal with this challenge. The project of C-Multi is based on two phases: the first uses a unique particle swarm optimization algorithm (PSO) [6] to discover the different regions of the Pareto front. The second phase uses multiple swarms to specialize on a dedicate part. On each swarm, an estimation of distribution algorithm (EDA) [7] is used to focus on convergence to its allocated region. The study featured a comparative study involving the C-Multi and the I-Multi algorithms using the DTLZ [8] family of benchmark problems. I-Multi is a multi-swarm algorithm that has a similar project to C-Multi but does not incorporate probabilistic modeling to the search as C-Multi does. The result of the comparison was that C-Multi achieved good results in some problems, but performed poorly in general.

Here, in this study, our goal is to investigate the following hypothesis: H_1 the performance of C-Multi can be further improved by an appropriate adjustment of two critical components of the algorithm; the type of archiving method, and the number of swarms. We initially hypothesized that finding a robust setting for these two components, together with the use of EDAs, could enhance C-Multi and help it to overcome I-Multi in the general case. To investigate this hypothesis, firstly we conducted two studies on the impact of C-Multi components. One of them evaluated the effect of the archiver used in the multi-swarm phase of the algorithm. The other assessed the effect of the number of sub-swarms. Next, we conducted extensive experimentation to compare the performances of C-Multi, enhanced with our findings from the previous studies, and I-Multi. Moreover, we compared both algorithms to a state-of-the-art algorithm called MOEA/D-DRA [9] in order to assess the effectiveness of the new algorithm. The obtained results indicate that C-Multi can be a competitive

* Corresponding author.
E-mail addresses: olacirjr@gmail.com (O.R. Castro Jr.), roberto.santana@ehu.es (R. Santana), aurora@inf.ufpr.br (A. Pozo).

http://dx.doi.org/10.1016/j.neucom.2015.06.097
0925-2312/

algorithm and the use of EDAs is a promising area in many-objective optimization.

The remaining of this paper is organized as follows: the next section presents related works. Some background concepts are described in Section 3. Section 4 presents the C-Multi algorithm. Experimental studies to investigate the influence of components of C-Multi, as well as an empirical study comparing it to I-Multi and MOEA/D-DRA are reported in Section 5. And finally, Section 6 presents the conclusions.

2. Related works

Multi-Objective Evolutionary Algorithms (MOEAs) modify EAs by incorporating a selection mechanism that is based on Pareto optimality and by adopting a diversity preservation mechanism that avoids the convergence to a single solution [10]. Although most of the studies on MOPs have focused on problems that have a small number of objectives, practical optimization problems involve a large number of criteria [11]. Therefore, research efforts have been oriented toward investigating the scalability of these algorithms with respect to the number of objectives [12]. Several studies have shown that MOEAs scale poorly in MaOPs [12,13]. The main reason for this scaling property is that the number of non-dominated solutions increases exponentially with the number of objectives. The following consequences occur: First, the search ability deteriorates because it is not possible to impose preferences for selection purposes, since most elite preserving mechanisms of MOEAs employ Pareto dominance as a major selection criterion. Second, the number of solutions that are required for approximating the entire Pareto front increases, therefore, in a high-dimensional objective space a limited number of solutions are likely to be far away from each other.

Among the studies presented in the literature, several papers address the issues of Many-Objective Optimization and deserve to be highlighted. In [14], an extension of the NSGA-II algorithm applied to MaOPs is presented. The new algorithm, known as NSGA-III, uses a set of reference points that are aimed at guiding the search toward the Pareto front without losing diversity. These points help the convergence and the diversity of the algorithm. Basically, the proposed algorithm builds niches for each reference point. Thus, the specialization of the algorithm enables convergence, and the different niches enable diversification. With these goals, the authors proposed a new density estimator that calculates the concentration of solutions around the reference points. Solutions that are close to less crowded reference points obtain an advantage in the selection process, similar to the crowding distance in NSGA-II. The NSGA-III was compared to MOEA/D [15], a decomposition based algorithm, and the best results were presented for different many-objective scenarios. More recently, in [16] a unified framework exploits dominance and decomposition based approaches from NSGAII and MOEAD/D to tackle MaOPs. The results of this unified framework have demonstrated its capability to find a well converged and well distributed approximation of the Pareto front.

In spite of the existence of different studies that address MaOPs, until very recently, most of these research studies focused on a small group of algorithms, often the NSGA-II or MOEA/D. In our project, the behavior of the PSO in MaOPs is investigated, since this approach has shown to be very efficient both in single-objective and in multi-objective optimization problems from different domains [17,18]. Its idea of moving across the search space towards the best solutions found so far while keeping a diverse population, points to the convenience of applying this method to MaOPs, where convergence and diversity are needed for a good coverage of the Pareto front. Therefore, PSO is a suitable algorithm for continuous many-objective optimization, but it is still under-explored in the literature.

In this direction, it can be mentioned the work presented in [4] that investigates several archiving methods, among them the Ideal and Multi-level Grid Archiving (MGA) [19]. The Ideal archiving method increases the convergence of the non-dominated solutions towards the Pareto-optimal front. On the other hand, the MGA approach obtains good diversity of solutions. The conclusion of the work highlights the main challenge of MaOPs: convergence to the true Pareto front and diversity of the obtained solutions covering the entire Pareto front. Recently, to overcome this limitation, researches proposed the use of multiple swarms. I-Multi algorithm [3] combines Ideal and MGA archivers in a multi-swarm search. This algorithm uses these two archiving methods at different phases of the process: first, the MGA is used in a single swarm, and after the obtained front is split and different swarms are executed in parallel using Ideal archiver. I-Multi algorithm presents good results in terms of convergence to the Pareto front and diversity of the obtained solutions on a set of MaOP benchmark problems.

These researches motivated our previous work [5] where the main goal was to explore other strategies of combination of these good elements in the design of algorithms for MaOPs. In [5], a possible alternative for the second phase of the I-Multi was investigated. The algorithm C-Multi was proposed whose main feature was to use an EDA [7]. EDAs have the capacity of achieving good convergence by generating solutions learned from the shape of the Pareto front. This work is an extension of the investigation presented in [5]. Our goal is to investigate if the performance of C-Multi can be enhanced by a more appropriate choice of its components. To evaluate the behavior of the algorithm, we compare to I-Multi, an algorithm that is similar to C-Multi but does not incorporate EDAs as part of the optimization process and to MOEA/D-DRA, a highly efficient method recently introduced in [9] as an improvement to MOEA/D [15].

3. Preliminaries

In this section, we first introduce some basic knowledge about many-objective optimization. Then, we briefly introduce the general mechanism of multi-objective particle swarm optimization (MOPSO) and I-Multi that are related to our work. Finally, we review basic knowledge about EDAs.

3.1. Many-objective optimization

Multi-objective optimization problems (MOPs) require the simultaneous optimization (maximization or minimization) of two or more objective functions. These objectives are usually in conflict, so these problems do not have only one optimal solution (as in single objective optimization problems), but a set of them. This set of solutions is usually found using Pareto optimality theory.

A general unconstrained MOP can be defined as optimizing $\vec{f}(\vec{x}) = (f_1(\vec{x}), \ldots, f_m(\vec{x}))$, where $\vec{x} \in \Omega$ is an n-dimensional decision variable vector $\vec{x} = (x_1, \ldots, x_n)$ from a universe Ω, and m is the number of objective functions.

An objective vector $\vec{f}(\vec{x})$ dominates a vector $\vec{f}(\vec{y})$, denoted by $\vec{f}(\vec{x}) \preceq \vec{f}(\vec{y})$ (in case of minimization) if and only if $\vec{f}(\vec{x})$ is partially less than $\vec{f}(\vec{y})$ i.e., $\forall i \in \{1, \ldots, m\}$, $f_i(\vec{x}) \leq f_i(\vec{y}) \wedge \exists i \in \{1, \ldots, m\} : f_i(\vec{x}) < f_i(\vec{y})$.

A vector $\vec{f}(\vec{x})$ is non-dominated if there is no $\vec{f}(\vec{y})$ that dominates $\vec{f}(\vec{x})$. If $\vec{f}(\vec{x})$ is non-dominated, $\vec{x}$ is Pareto optimal. The set of Pareto optimal solutions is called Pareto optimal set, and

the image of these solutions in the objective space is called Pareto front [10].

MaOPs are a type of MOPs that present more than three objective functions to be optimized simultaneously. Several studies have indicated that Pareto based algorithms scale poorly in MaOPs [2–4]. The main reason for this is that the number of non-dominated solutions greatly increases with the number of objectives. As a consequence, the search ability of the algorithms is deteriorated because it is not possible to impose preferences for selection purposes.

3.2. MOPSO

PSO [6] is a stochastic meta-heuristic based on the movement of bird flocks looking for food, created to optimize nonlinear functions. In this method a swarm (population) of particles (solutions) moves across the search space (evolves) guided by personal and social leaders.

To expand the PSO to solve multi-objective problems, and create a multi-objective particle swarm optimization (MOPSO) [20] algorithm, some modifications are needed. The first of them is the creation of an external archive (repository) to store the better (non-dominated) solutions found so far, another modification is in the leader selection scheme, which has to choose from a set of equally good leaders according to some criterion. As the number of non-dominated solutions may become very large, an archiving method is needed to prune the repository and keep only a predefined number of solutions, discarding some non-dominated solutions according to its criterion.

A MOPSO that has shown very good results in the literature is the Speed-constrained Multi-objective PSO (SMPSO) [21]. It was noted that in some conditions the velocity of the particles in a MOPSO can become too high, generating erratic movements towards the limits of the decision space. To avoid such situations, SMPSO presents a velocity constriction mechanism based on a factor χ that varies based on the values of the influence coefficients of personal and global leaders (C_1 and C_2 respectively). In SMPSO the (global) leader selection method uses a binary tournament based on the Crowding Distance metric from [22], and the archiving strategy also uses the Crowing Distance.

As the global leaders are selected from the repository, its size and content (managed by the archiving method) have a great impact on the results, especially in MaOPs, where the number of non-dominated solutions quickly increases. In [4] several archiving methods were compared in the optimization of MaOPs. The methods that do not limit the number of solutions in the Pareto set presented better results, and among the methods that limit the size of the archive, the Ideal [4] and MGA [19] stand out.

In Multi-level Grid Archiving (MGA) [19], when the repository becomes full, the objective space is divided into boxes and every solution in the archive has a box index. In a subsequent step, the domination relationships between the boxes are determined. If the new solution to be added belongs to one of the dominated boxes, it is not included in the repository, otherwise one of the solutions that have dominated box index is removed randomly. If there is no dominance relation between the boxes, the objective space is split again into smaller boxes until at least one dominated box is found. In this way, the number of boxes in MGA can change automatically at each iteration to adapt to the characteristics of the objective space.

In the Ideal [4] archiver, when the repository becomes full a process is triggered. Firstly, the ideal point of the repository is computed. The ideal point is a vector that has the best values for all objectives. Then, the Euclidean distance from each solution in the archive to the ideal point is computed. The solution that has the highest distance is removed.

3.3. I-Multi

Recently a new MOPSO called I-Multi [3] was proposed. I-Multi is a MOPSO designed to deal with MaOPs and that uses multiple swarms to cover different areas of the objective space. Its search procedure can be divided into two phases: diversity and multi-swarm searches. In the first phase, a diversity search is performed using SMPSO [21] with the MGA archiver [19] to generate a set of well-distributed non-dominated solutions (basis front) for multi-swarm initialization. Multi-swarm search begins using the basis front to assign a seed to each sub-swarm and around each seed (within a specified search region), a population of solutions is randomly generated to form a sub-swarm.

During a predefined number of iterations each sub-swarm runs independently, using the SMPSO with the Ideal archiver [4] to enhance its convergence. After that, the repository of each sub-swarm is integrated to the basis front so, only the non-dominated solutions regarding all repositories are kept. At the end of this process, the basis front is split into sub-swarms as before. This process of joining and splitting the fronts is called split iteration, and it is repeated a predefined number of times. This process enables an indirect communication between the sub-swarms.

In [3], three methods are presented to define the seed and the archive of each sub-swarm. In this work the I-Multi centroid is used, where the basis front is split into clustered groups (using a K-Means algorithm in the decision space) and the centroid of each group is chosen as seed for each sub-swarm. The solutions of each cluster are stored in the repository of each swarm. The number of clusters is defined as the same number of swarms. In this strategy, the archive is updated with all solutions clustered in the same cluster of the seed.

3.4. Estimation of distribution algorithms

In spite of the success of the Evolutionary Algorithms (EAs), they face difficulties in some real-world problems that exhibit characteristics like non-linearity, ill-conditioning and deception. Often, EAs' poor behavior in these problems can be explained by the fact that they do not properly consider the dependencies and relationships between decision variables and are not able to thoroughly exploit the information obtained so far.

The use of probabilistic graphical models (PGMs) [23] in EAs offers a systematic way of acquiring and representing problem regularities. In this approach, and in order to generate new solutions, the traditionally applied genetic operators are replaced by learning and sampling from probabilistic models. Machine learning methods are applied to learn the PGMs from the selected solutions, and sampling strategies serve to generate the new solutions from the models. The incorporation of probabilistic modeling in EAs has led to a new paradigm known as competent EAs or EDAs [24,7].

General EDAs iterate three steps until some termination criterion is satisfied: select good solutions from a population, estimate the probability distribution from the selected individuals (learn a model) and generate new solutions (sample) from the estimated distributions (model).

The EDA used in this work is the real-coded Bayesian optimization algorithm (rBOA) [25], a continuous, real-coded version of the Bayesian optimization algorithm (BOA) [26]. BOA is one of the most efficient and extensively applied EDAs [27], able to represent higher order dependencies between discrete variables by means of Bayesian networks. rBOA inherits some of the suitable attributes of BOA. Our choice of rBOA was also motivated by its good results in previous applications and extensive documentation. Its pseudo-code is presented through Algorithm 1.

Algorithm 1. rBOA.

```
//Step 1: Initialization
P=generateRandomPopulation()
for i=1 to It do
        //      Step 2 : Selection
    S = selectPromisingCandidates(P)
        //      Step 3 : Learn
    M = learnProbabilisticModel(S)
        //      Step 4 :
    O = generateOffspring(M)
        //      Step 5 :
    P = generateNewPopulation(O, P)
end for
return bestFrom(P)
```

In this algorithm, firstly a population P is randomly generated, then for a predetermined number of iterations (It) promising candidate solutions are selected from the whole population to be used in the learning phase to create a probabilistic model M. A set of offspring solutions is sampled from this probabilistic model and the new population is updated based on the previous one and the set of new offspring solutions. At the end of the iterations, the better solution from the population is returned as a result of the algorithm.

Different EDAs can be characterized by the method of learning a probabilistic model (Step 3) and its performance is affected directly by the efficiency of this model learning. In general, the learning of probabilistic models consists of two tasks: model selection and model fitting. Model selection determines the structures of promising probabilistic models and model fitting estimates the conditional probability distributions with regard to the found structures [28,29].

rBOA model selection obtains a Bayesian factorization, which is represented by a directed acyclic graph called Bayesian factorization graph where the nodes and edges identify the corresponding variables and its conditional dependencies respectively. Two components determine the model selection step: a scoring metric and a search procedure. The scoring metric assesses the quality of the structure of the Bayesian factorization graph and the search procedure efficiently traverses the space of all feasible structures for finding the best one with regard to the given scoring metric. BOA and rBOA employ a Bayesian Information Criterion (BIC) [30] as the scoring metric and an incremental greedy algorithm as the search procedure [25].

In rBOA model fitting, a set consisting of a node and its parents in the Bayesian factorization graph represents a component sub-problem of decomposable problems. For efficiency, maximal connected sub-graphs of a Bayesian factorization graph are considered as sub-problems and must be independently fitted. Mixture models are employed and the aim of using them is twofold: comprehending the type of dependency between variables and traversing the search space effectively. Each mixture component can model the linearity of the variables, thus they can approximate any type of dependency by a combination of piecewise linear interaction models [25].

4. Competent Multi-swarm

This section presents a hybrid algorithm called Competent Multi-swarm (C-Multi). The feature that distinguishes C-Multi from the other previously introduced multi-swarm algorithms is the incorporation of a model-based search component implemented using rBOA [25]. C-Multi combines the strengths of two efficient algorithms: I-Multi [3] and rBOA. I-Multi presents a high diversity of solutions through the use of multiple swarms to spread its particles across the front. rBOA is able to achieve a good convergence by generating solutions learned from the shape of the Pareto front. By hybridizing these two algorithms, C-Multi is able to have a better performance, especially in hard problems.

As the project of I-Multi, the project of C-Multi is based on two phases: the first uses a unique PSO to discover the different regions of the Pareto front. The second phase uses multiple swarms to specialize on a dedicate part. On each swarm, an EDA is used to focus on convergence to its allocated region.

In the first phase, a traditional SMPSO [21] with the MGA [19] archiver is used to obtain a diverse set of non-dominated solutions (basis front). Next, the multi-swarm phase begins by splitting the basis front in NS sub-fronts (set as parameter) by using the K-Means algorithm. Each sub-swarm is initialized with a seed (centroid of the cluster) and the non-dominated solutions contained in a sub-front. This seed is used to define the bounds to truncate the solutions (if necessary) to make sure each sub-swarm will explore only a limited region avoiding overlaps.

In each loop of the multi-swarm phase, the non-dominated solutions of each sub-swarm are used by rBOA to learn a model. Next, the model is sampled to create a new population (or replace a previous one) for the sub-swarm. Then, the population of the sub-swarm is added to the front if the criteria of the archiver are satisfied, ending the loop.

Algorithm 2. C-Multi.

```
//Phase 1: Diversity search
F_b=Run-MGA-SMPSO()
//Phase 2: Multi-swarm search
for s=1 to SI do
    F⃗ = SplitFront(F_b)
    for k=1 to NS do
        for i=1 to It do
            P_k = rBOA(F_k)
            Truncate(P_k)
            Evaluate(P_k)
            F_k = Archiver(P_k)
        endfor
    endfor
    F_b = Non − dominated(F⃗)
end for
return F_b
```

During the multi-swarm phase, the loops above are interrupted a predefined number of times (SI, set as parameter) to perform a split iteration. In this procedure, the fronts of all sub-swarms are merged into a single basis front (only the non-dominated solutions concerning all fronts are kept) that is split as before. Then, the multi-swarm phase is restarted. The goal of the split iterations is to allow an indirect communication between the sub-swarms in addition to eliminating duplicated solutions across the fronts. A pseudo-code of C-Multi is presented through Algorithm 2.

In this algorithm, firstly the diversity phase is conducted by running the SMPSO algorithm with the MGA archiver for a predefined number of iterations to obtain a well diversified basis front (F_b).

Next the multi-swarm stage begins. This stage is characterized by SI split iterations, where the basis front F_b is split into NS

sub-fronts ($\vec{F} = (F_1, \ldots, F_{NS})$) by using a K-Means algorithm in the decision space. Each of these sub-fronts (F_k) is composed of a set of non-dominated solutions and a seed (the centroid of the cluster).

Then, for each front, during a predetermined number of iterations, the following procedure is executed: a model is learned for each front F_k; a new population P_k is sampled from this model and its decision vector is truncated (if necessary) to stay within the search region around the seed; the population is evaluated through the objective function; and finally the front F_k is updated with the better solutions from the population.

After all sub-swarms were executed independently for several iterations (It), a new split iteration begins. The non-dominated solutions from all the sub-swarms are joined in the basis front F_b, which is split again into NS sub-swarms and the process repeats. At the end of the algorithm, F_b containing all the non-dominated solutions regarding all fronts is returned as result from the algorithm.

The structure of C-Multi is basically the same as the I-Multi, however in the multi-swarm search, instead of using the SMPSO, the rBOA learning method is used to create a model and sample new particles. This replacement of the update method is used to increase the convergence capacity in this stage of the search, causing a moderate increase in the computational cost.

5. Empirical study

This session presents the empirical studies conducted involving the C-Multi algorithm. In the experiments, we perform an analysis of some important components of C-Multi, as the archiver used in the multi-swarm stage and the number of sub-swarms. Furthermore the performance of our algorithm is compared to its base algorithm I-Multi [3], and to the state-of-the-art algorithm, winner of the CEC 2009 MOEA contest MOEA/D-DRA [9].

5.1. Experimental setup

For the study of the impact of the components of the algorithm, i.e., the archiver (Section 5.2) and the number of swarms (Section 5.3), the problems WFG1, WFG4 and WFG6 were chosen because they are representative of the WFG [31] family of benchmark problems.

The comparison of C-Multi to I-Multi and MOEA/D-DRA used the entire WFG family of problems to analyze the performance of the algorithms in different scenarios. A summary of the characteristics of the problems used is presented in Table 1 adapted from [32]. The WFG test suite poses a significant challenge for algorithms to obtain a well converged and well distributed solution set. These benchmark problems can scale both in number of objectives and in number of decision variables, also the true Pareto optimal front is known.

Table 1
Characteristics of problems.

Problem	Characteristics
WFG1	Mixed, biased
WFG2	Convex, disconnected, multi-modal, non-separable
WFG3	Linear, degenerate, non-separable
WFG4	Concave, multi-modal
WFG5	Concave, deceptive
WFG6	Concave, non-separable
WFG7	Concave, biased
WFG8	Concave, biased, non-separable
WFG9	Concave, biased, multi-modal, deceptive, non-separable

Table 2
Parameters.

C_1, C_2	Varies randomly in [1.5,2.5]
Number of objectives (m)	3, 5, 8 and 10
Number of decision variables	$k+l$ where $k=2\times(m-1)$ and $l=20$
Initial phase duration	100 iterations
Initial phase population	100 particles
Multi-swarm phase duration	100 iterations
Multi-swarm phase population	(750/number of swarms) particles
Repository maximum size	200 solutions
Multi-swarm region size	Decrease from 0.5 to 0.1
Number of split iterations	5
Leader threshold (rBOA)	0.3
BIC complexity factor λ (rBOA)	0.5

The algorithms were tested using different numbers of objectives to verify how they perform as the number of objectives scales up. The main parameters used for the algorithms are summarized in Table 2.

The parameters C_1 and C_2, that control the effect of the personal and global best particles in the velocity respectively, are set according to the recommendation given in the original SMPSO paper [21], by changing these values is possible to control the trade-off between convergence and diversity in the algorithm. The number of decision variables was set according to the recommendation given for the problems in [31], decreasing or increasing the number of decision variables is an easy way of making the problem easier or harder respectively. The complexity factor of the BIC score controls how much the algorithm penalizes the complexity of a probabilistic model, hence impacts on the trade-off between the computational cost and efficiency of the model. The complexity factor of BIC score λ as well as the threshold used in the BEND leader algorithm (used for model fitting) is set for rBOA according to the values proposed in its original paper [25]. The leader threshold of BEND algorithm used for model fitting impacts the number of clusters used in this stage, and can impact the ability of the model in efficiently modeling the dependencies between variables.

The number of split iterations as well as the multi-swarm region size was calibrated in [3] and we use the best values found, since they are representative for C-Multi as well. A split iteration can be beneficial for the search, since it removes dominated and repeated solutions as well as promoting communication among the swarms, but it can be prejudicial to the swarm, since it may reduce the number of solutions contained in each swarm, hence there are less particles to be used as guide on the search (on I-Multi) and less solutions available to be learned (on C-Multi). Regarding the multi-swarm region size, if it is increased, more overlaps are expected, hence more repeated solutions may arise among the swarms, and each swarm need to focus its search on a larger area, weakening its specialization ability, on the other hand if we decrease this value, it can create “holes” on the front, or areas that are not covered by any swarm.

The number of iterations (initial and total), the initial size of the population, the maximum size of the repositories and the total number of particles also were set as proposed in [3] since these values can be considered a good compromise between the search exploration capabilities of PSO and its computational cost. Regarding the number of particles per swarm, if the total number is not divisible by the amount of sub-swarms, the remaining particles are distributed among the sub-swarms.

To assess the quality of the Pareto fronts generated by each algorithm, two well-known quality indicators are used: The modification of the Inverted Generational Distance (IGD) known as IGD_p [33], and the hypervolume [34]. Up to eight objectives the exact hypervolume calculation was used, and for ten objectives an

approximated version based on Monte Carlo sampling [35] was used. Before calculating both metrics we normalized the fronts obtained in the interval [0, 1], based on the maximum and minimum values found for each objective, due to this, the reference point used for the hypervolume calculation was 1.01 in all dimensions.

The results measured using these two indicators in 30 independent runs of the algorithms for every number of objectives are compared using the Kruskal–Wallis statistical test [36] at 5% significance level, the post-test indicates if there are statistical differences between the results and the ranks of the indicator values are used to determine the best performing algorithms.

5.2. Archiver of sub-swarms

In multi-objective continuous optimization it is possible to have infinite non-dominated solutions that need to be stored in an external archive or repository. Usually the repository is limited to contain a maximum number of solutions in order to avoid its size of growing excessively along with the computational cost to maintain it. Hence, when the repository is full and a new non-dominated solution is found an archiver is used to decide whether this new solution enters in the archive and if so, which non-dominated solution is removed.

In this section we compare three archivers from the literature that can be used in the multi-swarm phase of C-Multi. Crowding Distance (CD) [22], Ideal [4] and Multi-level Grid Archiving (MGA) [19] in order to identify which is the best in most cases.

The results of this comparison are shown in Tables 3 and 4. Regarding the IGD_p results, in problem WFG1 there were no significant differences among the archivers. In problem WFG4, significant differences were only found for eight objectives, where CD and Ideal performed better than MGA. In problem WFG6, CD had better results for three and eight objectives, while no significant differences were found for five and ten objectives.

Considering the hypervolume, no significant differences were found in WFG1. In WFG4, CD had the best results for all numbers of objectives. In WFG6, for three and eight objectives CD also had the best results, while for five and ten objectives all archivers tied.

As an overall result, considering both metrics, we can state that the archiver of sub-swarms had small influence on the search, however, in general the archiver CD had better results in more combinations of problems and objective numbers.

These good results obtained by the CD archiver can be attributed to its characteristic of always keeping the extreme solutions of the sub-swarm. By keeping these solutions, the EDA models the region comprising these points, and has a higher probability of generating more solutions between these points. Other archivers that usually remove the extreme solutions, like Ideal, can create "holes" in the front, because the model does not learn the extremes of the front, and can be concentrated only near the center of its search region.

Table 3
Kruskal–Wallis ranks of the IGD_p for different archivers.

Obj.	Archiver	WFG1	WFG4	WFG6
3	CD	**48.43 (2.00)**	**43.53 (2.00)**	**22.90 (1.00)**
	Ideal	**45.23 (2.00)**	**44.00 (2.00)**	62.50 (2.50)
	MGA	**42.83 (2.00)**	**48.97 (2.00)**	51.10 (2.50)
5	CD	**44.00 (2.00)**	**44.20 (2.00)**	**37.80 (2.00)**
	Ideal	**45.17 (2.00)**	**44.87 (2.00)**	**49.23 (2.00)**
	MGA	**47.33 (2.00)**	**47.43 (2.00)**	**49.47 (2.00)**
8	CD	**44.53 (2.00)**	**34.47 (1.50)**	**26.07 (1.50)**
	Ideal	**45.57 (2.00)**	**29.07 (1.50)**	**36.23 (1.50)**
	MGA	**46.40 (2.00)**	72.97 (3.00)	74.20 (3.00)
10	CD	**49.77 (2.00)**	**43.83 (2.00)**	**36.37 (1.50)**
	Ideal	**44.57 (2.00)**	**39.67 (2.00)**	47.07 (2.00)
	MGA	**42.17 (2.00)**	**53.00 (2.00)**	53.07 (2.50)

Table 4
Kruskal–Wallis ranks of the hypervolume for different archivers.

Obj.	Archiver	WFG1	WFG4	WFG6
3	CD	**49.40 (2.00)**	**35.10 (1.50)**	**24.50 (1.00)**
	Ideal	**47.83 (2.00)**	45.00 (2.00)	50.80 (2.50)
	MGA	**39.27 (2.00)**	56.40 (2.50)	61.20 (2.50)
5	CD	**45.27 (2.00)**	**25.40 (1.00)**	**44.53 (2.00)**
	Ideal	**47.37 (2.00)**	54.80 (2.50)	**43.17 (2.00)**
	MGA	**43.87 (2.00)**	56.30 (2.50)	**48.80 (2.00)**
8	CD	**49.80 (2.00)**	**16.07 (1.00)**	**23.53 (1.00)**
	Ideal	**43.03 (2.00)**	51.57 (2.00)	46.90 (2.00)
	MGA	**43.67 (2.00)**	68.87 (3.00)	66.07 (3.00)
10	CD	**43.73 (2.00)**	**31.37 (1.00)**	**37.07 (2.00)**
	Ideal	**45.30 (2.00)**	49.80 (2.50)	**47.20 (2.00)**
	MGA	**47.47 (2.00)**	55.33 (2.50)	**52.23 (2.00)**

This behavior can be seen in Fig. 1, where we plotted the best front (according to hypervolume) generated by C-Multi using each of the three archivers. These fronts were generated when optimizing the three objective instance of problem WFG4. Additionally, Fig. 1(d) shows a discretization of the Pareto optimal front for this problem. Fig. 1(a) shows a front obtained using the CD archiver, where the solutions are well-spread along the objective space. A front obtained using the Ideal archiver is presented in Fig. 1(b) where we can see several areas of the objective space that are not covered by solutions due to the decrease on diversity. Fig. 1 (c) shows a front generated using the MGA archiver, this figure shows that this archiver is able to maintain a set of solutions well spread despite the MGA method not being able to guarantee the maintenance of the extreme solutions.

5.3. Number of sub-swarms

In C-Multi, the number of swarms is an important parameter since it can impact in the content of the repository. If there are less swarms, each swarm will have more particles to cover a larger area of the front, hence we expect a higher diversity within the swarm, but since the repository size is limited, more non-dominated solutions will be discarded. On the other hand, if we increase the number of swarms, each swarm will have less particles to cover a smaller area of the front, being able to specialize in a small region, thus we expect a smaller diversity within the swarm and less solutions available for learning.

Since the model is learned from the solutions contained in the repository, its diversity and size impacts the algorithm. This section investigates this impact in the results obtained with the C-Multi algorithm and identifies the best number of sub-swarms to be used. To this end, we conducted a comparative study varying the number of sub-swarms between 10 and 100. In order to keep the same computational cost, the total number of particles on the multi-swarm phase was set to 750, then the number of particles per swarm is (750/number of swarms). If this division does not have an integer result, the remaining particles are distributed among the sub-swarms.

Since in the previous section the CD archiver presented general better results, in this study we use this archiver in the multi-swarm phase of the algorithm. The results obtained from these experiments are presented through Tables 5 and 6 where the Kruskal–Wallis ranks of the IGD_p and hypervolume are displayed.

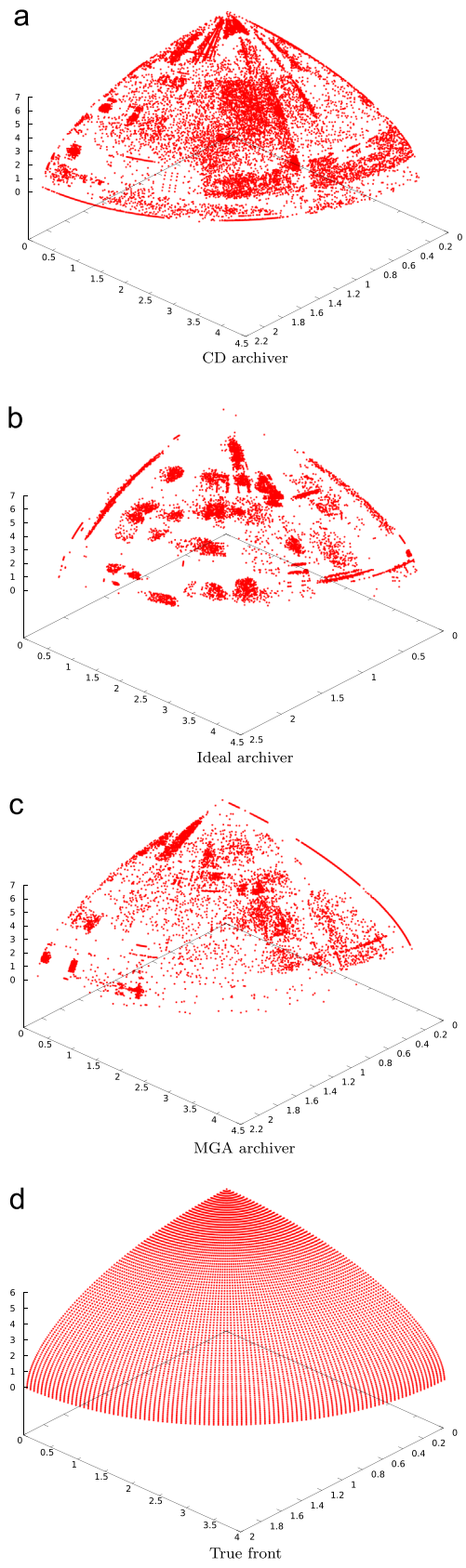

Fig. 1. Better approximation fronts obtained using different archivers and the true front.

Since we obtained different results according to problem and objective numbers, is hard to take overall conclusions, hence we included in Tables 7 and 8 the Friedman ranks obtained for the overall analysis of the algorithms. In these tests, the average of the 30 independent runs of each subproblem (problem/objective number) is considered. A significance level of 5% is considered for this test as well.

Table 5
Kruskal–Wallis ranks of the IGD_p for different numbers of sub-swarms.

Obj.	Swarm number	WFG1	WFG4	WFG6
3	10	225.27 (8.50)	281.87 (9.50)	271.20 (9.50)
	20	**122.63 (5.00)**	251.53 (9.00)	226.63 (9.00)
	30	**127.27 (5.00)**	204.70 (8.00)	160.27 (5.00)
	40	**116.13 (5.00)**	175.93 (6.00)	**116.27 (4.50)**
	50	**114.00 (5.00)**	121.80 (4.00)	**120.00 (4.50)**
	60	**148.80 (5.00)**	113.40 (4.00)	**104.30 (4.50)**
	70	164.23 (5.50)	111.47 (4.00)	**117.90 (4.50)**
	80	**149.07 (5.00)**	**70.50 (3.50)**	**116.73 (4.50)**
	90	173.43 (5.50)	**72.03 (3.50)**	**153.10 (4.50)**
	100	164.17 (5.50)	**101.77 (3.50)**	**118.60 (4.50)**
5	10	251.77 (9.50)	281.37 (9.00)	262.63 (9.50)
	20	187.50 (5.50)	249.77 (9.00)	228.57 (9.00)
	30	**140.43 (5.00)**	214.03 (8.00)	164.93 (5.00)
	40	**115.60 (5.00)**	164.23 (6.00)	**141.27 (4.50)**
	50	**131.17 (5.00)**	161.63 (6.00)	**141.83 (4.50)**
	60	**124.57 (5.00)**	114.33 (4.00)	**123.33 (4.50)**
	70	**145.50 (5.00)**	104.87 (4.00)	**108.93 (4.50)**
	80	**135.47 (5.00)**	**80.57 (3.00)**	**113.33 (4.50)**
	90	**139.63 (5.00)**	**73.40 (3.00)**	**115.23 (4.50)**
	100	**133.37 (5.00)**	**60.80 (3.00)**	**104.93 (4.50)**
8	10	230.53 (8.50)	277.77 (9.00)	275.53 (9.50)
	20	190.00 (7.00)	252.80 (9.00)	248.40 (9.00)
	30	155.30 (5.00)	220.93 (8.00)	181.97 (7.50)
	40	**110.00 (4.50)**	178.93 (6.50)	158.67 (5.00)
	50	**103.40 (4.50)**	162.50 (6.50)	151.50 (4.50)
	60	**110.40 (4.50)**	108.43 (4.00)	108.03 (4.00)
	70	134.77 (5.00)	**85.57 (3.00)**	102.50 (4.00)
	80	130.70 (5.00)	**85.27 (3.00)**	95.17 (4.00)
	90	169.33 (5.50)	**62.83 (3.00)**	98.43 (4.00)
	100	170.57 (5.50)	**69.97 (3.00)**	**84.80 (3.50)**
10	10	224.73 (8.50)	278.87 (9.00)	277.97 (9.50)
	20	177.80 (5.50)	245.93 (8.50)	250.17 (9.00)
	30	153.47 (5.50)	219.13 (8.00)	180.90 (6.50)
	40	**131.93 (5.00)**	180.00 (6.50)	151.17 (4.50)
	50	**150.87 (5.00)**	148.23 (5.50)	145.97 (4.50)
	60	153.40 (5.50)	112.70 (4.00)	116.00 (4.50)
	70	**148.40 (5.00)**	111.07 (4.00)	110.17 (4.50)
	80	**128.43 (5.00)**	**69.00 (3.00)**	**98.77 (4.00)**
	90	**116.80 (5.00)**	79.80 (3.50)	**82.77 (4.00)**
	100	**119.17 (5.00)**	**60.27 (3.00)**	**91.13 (4.00)**

These tables indicate that in general when considering the IGD_p indicator, the more swarms the better, since except from two cases (60–70 and 80–90), the Friedman ranks decrease as the number of swarms increase, also the better final ranking is obtained with 100 swarms.

Regarding the hypervolume indicator, we see a similar effect, but towards 40 swarms, where the ranks increase as the number of sub-swarms distances from 40, except between 50 and 70. Also, the best final ranking is obtained with 40 swarms.

It is known that IGD_p and hypervolume have different characteristics, as demonstrated in this section. Since each one points us to a different setting, in this work we followed the results obtained using the hypervolume due to its stronger mathematical properties.

5.4. C-Multi vs. I-Multi vs. MOEA/D-DRA

In this section we compare C-Multi to I-Multi [3] and MOEA/D-DRA [9], a state-of-the-art algorithm winner of the CEC 2009 MOEA contest. In this comparison, the entire WFG family of benchmark problems was used to provide an overview of the behavior of the algorithms as the number of objectives scales up.

Table 6
Kruskal–Wallis ranks of the hypervolume for different numbers of sub-swarms.

Obj.	Swarm number	WFG1	WFG4	WFG6
3	10	209.27 (7.00)	203.17 (7.00)	258.20 (9.50)
	20	138.77 (5.50)	171.10 (5.50)	205.20 (8.50)
	30	**119.97 (5.00)**	165.90 (5.50)	155.77 (5.00)
	40	**127.83 (5.00)**	**127.20 (5.00)**	**100.27 (4.50)**
	50	**123.73 (5.00)**	**127.47 (5.00)**	**127.67 (4.50)**
	60	149.30 (5.50)	148.83 (5.50)	**115.90 (4.50)**
	70	144.27 (5.50)	**122.87 (5.00)**	**129.77 (4.50)**
	80	141.73 (5.50)	150.60 (5.50)	**128.50 (4.50)**
	90	178.37 (5.50)	132.83 (5.50)	158.43 (5.00)
	100	171.77 (5.50)	155.03 (5.50)	**125.30 (4.50)**
5	10	211.97 (7.50)	198.83 (7.50)	211.37 (7.50)
	20	156.90 (5.50)	**103.30 (3.50)**	161.17 (5.50)
	30	161.03 (5.50)	**99.83 (3.50)**	**108.57 (5.00)**
	40	**132.33 (5.00)**	**101.33 (3.50)**	148.20 (5.50)
	50	**122.30 (5.00)**	116.43 (4.00)	143.37 (5.50)
	60	**114.87 (5.00)**	156.93 (5.50)	**136.57 (5.00)**
	70	142.23 (5.50)	142.80 (5.50)	**125.83 (5.00)**
	80	162.17 (5.50)	176.63 (7.00)	**128.10 (5.00)**
	90	173.17 (5.50)	204.27 (7.50)	167.07 (5.50)
	100	**128.03 (5.00)**	204.63 (7.50)	174.77 (5.50)
8	10	**179.10 (5.50)**	144.47 (4.50)	**150.77 (5.50)**
	20	**151.43 (5.50)**	102.03 (4.00)	**181.20 (5.50)**
	30	**136.77 (5.50)**	111.20 (4.00)	**142.97 (5.50)**
	40	**153.50 (5.50)**	**78.10 (3.50)**	**152.23 (5.50)**
	50	**156.07 (5.50)**	114.90 (4.00)	**159.97 (5.50)**
	60	**135.33 (5.50)**	158.20 (6.00)	**125.03 (5.50)**
	70	**130.97 (5.50)**	149.20 (4.50)	**140.20 (5.50)**
	80	**121.67 (5.50)**	189.07 (7.50)	**159.33 (5.50)**
	90	**151.77 (5.50)**	227.83 (8.50)	**138.90 (5.50)**
	100	**188.40 (5.50)**	230.00 (8.50)	**154.40 (5.50)**
10	10	**163.42 (5.50)**	119.70 (4.00)	**173.00 (5.50)**
	20	**147.87 (5.50)**	**81.43 (3.00)**	**125.55 (5.50)**
	30	**163.90 (5.50)**	91.40 (3.50)	**140.97 (5.50)**
	40	**139.87 (5.50)**	121.07 (4.00)	**153.43 (5.50)**
	50	**145.90 (5.50)**	109.15 (4.00)	**153.08 (5.50)**
	60	**149.95 (5.50)**	158.80 (6.00)	**153.07 (5.50)**
	70	**135.77 (5.50)**	171.13 (6.50)	**142.30 (5.50)**
	80	**171.50 (5.50)**	194.97 (8.00)	**143.17 (5.50)**
	90	**123.02 (5.50)**	231.65 (8.00)	**157.30 (5.50)**
	100	**163.82 (5.50)**	225.70 (8.00)	**163.13 (5.50)**

Table 7
Overall Friedman ranks of the IGD_p for different numbers of sub-swarms.

10	20	30	40	50	60	70	80	90	100
120.0 (9.0)	101.0 (7.5)	87.0 (6.0)	63.0 (5.0)	58.0 (5.0)	52.0 (4.5)	56.0 (5.0)	42.0 (4.5)	44.0 (4.5)	**37.0 (4.0)**

Table 8
Overall Friedman ranks of the hypervolume for different numbers of sub-swarms.

10	20	30	40	50	60	70	80	90	100
103.0 (7.5)	66.0 (5.5)	46.0 (5.0)	**41.0 (4.5)**	55.0 (5.5)	52.0 (5.0)	50.0 (5.0)	74.0 (5.5)	82.0 (5.5)	91.0 (6.0)

Since the CD archiver had the best overall results for C-Multi, we use it to prune the solutions in the multi-swarm phase of C-Multi. I-Multi used the Ideal archiver as it was designed. In order to keep a fair comparison, we used the same number of sub-swarms (40), for C-Multi and I-Multi. Also we used as stop criterion for all three algorithms the number of fitness evaluations, fixed in 85 100. The results from this experimental study are presented through Tables 9 and 10 that present the Kruskal–Wallis ranks of the IGD_p and hypervolume respectively.

Table 9
Kruskal–Wallis ranks of the IGD_p for C-Multi, I-Multi and MOEA/D-DRA.

Obj.	Algorithms	WFG1	WFG2	WFG3	WFG4	WFG5
3	C-Multi	73.33 (3.00)	39.83 (2.00)	49.17 (2.00)	**19.23 (1.00)**	**39.43 (1.50)**
	I-Multi	46.53 (2.00)	**23.03 (1.00)**	71.83 (3.00)	41.77 (2.00)	**24.33 (1.50)**
	MOEA/D-DRA	**16.63 (1.00)**	73.63 (3.00)	**15.50 (1.00)**	75.50 (3.00)	72.73 (3.00)
5	C-Multi	**40.03 (1.50)**	**31.80 (1.50)**	58.57 (2.50)	**15.57 (1.00)**	**15.73 (1.00)**
	I-Multi	**27.93 (1.50)**	**29.27 (1.50)**	62.17 (2.50)	45.43 (2.00)	45.27 (2.00)
	MOEA/D-DRA	68.53 (3.00)	75.43 (3.00)	**15.77 (1.00)**	75.50 (3.00)	75.50 (3.00)
8	C-Multi	44.83 (2.00)	**39.70 (1.50)**	48.10 (2.50)	**18.70 (1.00)**	**17.00 (1.00)**
	I-Multi	**18.20 (1.00)**	**28.97 (1.50)**	**27.00 (1.00)**	42.30 (2.00)	44.00 (2.00)
	MOEA/D-DRA	73.47 (3.00)	67.83 (3.00)	61.40 (2.50)	75.50 (3.00)	75.50 (3.00)
10	C-Multi	46.80 (2.00)	**36.43 (1.50)**	**36.03 (1.50)**	**19.77 (1.00)**	**18.70 (1.00)**
	I-Multi	**16.67 (1.00)**	42.97 (2.00)	**24.97 (1.50)**	41.23 (2.00)	42.30 (2.00)
	MOEA/D-DRA	73.03 (3.00)	57.10 (2.50)	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)

Obj.	Algorithms	WFG6	WFG7	WFG8	WFG9
3	C-Multi	**16.37 (1.00)**	64.90 (2.50)	65.77 (2.50)	**19.17 (1.00)**
	I-Multi	63.13 (2.50)	56.10 (2.50)	55.23 (2.50)	49.87 (2.00)
	MOEA/D-DRA	57.00 (2.50)	**15.50 (1.00)**	**15.50 (1.00)**	67.47 (3.00)
5	C-Multi	42.43 (2.00)	**15.50 (1.00)**	**15.50 (1.00)**	**28.70 (1.50)**
	I-Multi	**18.57 (1.00)**	45.50 (2.00)	45.50 (2.00)	**32.30 (1.50)**
	MOEA/D-DRA	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)
8	C-Multi	**26.80 (1.50)**	**15.57 (1.00)**	**15.50 (1.00)**	41.90 (2.00)
	I-Multi	**34.20 (1.50)**	45.43 (2.00)	45.50 (2.00)	**19.10 (1.00)**
	MOEA/D-DRA	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)
10	C-Multi	**23.30 (1.50)**	**15.87 (1.00)**	**15.50 (1.00)**	44.77 (2.00)
	I-Multi	**37.70 (1.50)**	45.13 (2.00)	45.50 (2.00)	**16.23 (1.00)**
	MOEA/D-DRA	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)	75.50 (3.00)

Regarding the IGD_p results, for three objectives C-Multi had competitive performance, achieving the best rankings in problems WFG4, WFG6 and WFG9 and tying with I-Multi on WFG5. Besides tying with C-Multi on WFG5, I-Multi only achieved superior performance on WFG2. MOEA/D-DRA outperformed both algorithms on problems WFG1, WFG3, WFG7 and WFG8. For five objectives the results of C-Multi were better, outperforming the other algorithms in problems WFG4, WFG5, WFG7 and WFG8 and tying with I-Multi on WFG1, WFG2 and WFG9. I-Multi, besides tying to C-Multi, outperformed the other algorithms on WFG6. MOEA/D-DRA only had better rankings on WFG3.

For eight objectives, C-Multi had better rankings than the other algorithms on WFG4, WFG5, WFG7 and WFG8, and tied with I-Multi on WFG2 and WFG6. I-Multi, besides tying with C-Multi, achieved better performance on WFG1, WFG3 and WFG9. MOEA/D-DRA did not outperformed the others in any problem. For ten objectives, C-Multi only lost to I-Multi on problems WFG1 and WFG9, and tied to it on WFG3 and WFG6. MOEA/D-DRA did not have the best rankings in any problem.

Fig. 2 shows examples of boxplots for the IGD_p results obtained by the algorithms. Due to lack of space we only included boxplots for the problem WFG4. From this figure, we can see that for all the numbers of objectives, in general, C-Multi obtained better (smaller) IGD_p results, followed closely by C-Multi. MOEA/D-DRA did not achieve good results regarding IGD_p on this problem.

Considering the hypervolume, for three objectives, C-Multi outperformed the other algorithms on WFG6 and WFG9. I-Multi had better rankings on WFG4 and tied to MOEA/D-DRA on WFG5. In the remaining problems, MOEA/D-DRA outperformed both algorithms. For five objectives, the three algorithms tied on WFG9, I-Multi achieved better results on WFG6, and MOEA/D-DRA outperformed I-Multi and C-Multi in the remaining problems. For eight and ten objectives, MOEA/D-DRA presents very good results, outperforming both algorithms in all problems.

Table 10
Kruskal–Wallis ranks of the hypervolume for C-Multi, I-Multi and MOEA/D-DRA.

Obj.	Algorithms	WFG1	WFG2	WFG3	WFG4	WFG5
3	C-Multi	73.20 (3.00)	68.60 (3.00)	47.57 (2.00)	75.37 (3.00)	64.50 (3.00)
	I-Multi	47.03 (2.00)	52.40 (2.00)	73.43 (3.00)	**15.50 (1.00)**	**35.20 (1.50)**
	MOEA/D-DRA	**16.27 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	45.63 (2.00)	**36.80 (1.50)**
5	C-Multi	69.03 (3.00)	74.37 (3.00)	47.27 (2.00)	75.50 (3.00)	50.17 (2.00)
	I-Multi	51.93 (2.00)	46.63 (2.00)	73.73 (3.00)	44.57 (2.00)	70.83 (3.00)
	MOEA/D-DRA	**15.53 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**16.43 (1.00)**	**15.50 (1.00)**
8	C-Multi	72.87 (3.00)	71.33 (3.00)	45.50 (2.00)	46.23 (2.00)	47.07 (2.00)
	I-Multi	47.90 (2.00)	49.67 (2.00)	75.50 (3.00)	74.77 (3.00)	73.93 (3.00)
	MOEA/D-DRA	**15.73 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**
10	C-Multi	74.10 (3.00)	69.33 (3.00)	45.50 (2.00)	45.50 (2.00)	46.53 (2.00)
	I-Multi	41.60 (2.00)	51.67 (2.00)	75.50 (3.00)	75.50 (3.00)	74.47 (3.00)
	MOEA/D-DRA	**20.80 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**

Obj.	Algorithms	WFG6	WFG7	WFG8	WFG9
3	C-Multi	**23.40 (1.00)**	75.27 (3.00)	75.27 (3.00)	**22.73 (1.00)**
	I-Multi	49.33 (2.50)	45.73 (2.00)	45.73 (2.00)	51.53 (2.50)
	MOEA/D-DRA	63.77 (2.50)	**15.50 (1.00)**	**15.50 (1.00)**	62.23 (2.50)
5	C-Multi	75.50 (3.00)	73.03 (3.00)	73.87 (3.00)	**42.90 (2.00)**
	I-Multi	**15.50 (1.00)**	47.97 (2.00)	47.13 (2.00)	**39.33 (2.00)**
	MOEA/D-DRA	45.50 (2.00)	**15.50 (1.00)**	**15.50 (1.00)**	**54.27 (2.00)**
8	C-Multi	75.50 (3.00)	46.70 (2.00)	49.27 (2.00)	74.33 (3.00)
	I-Multi	43.17 (2.00)	74.30 (3.00)	71.73 (3.00)	46.63 (2.00)
	MOEA/D-DRA	**17.83 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**15.53 (1.00)**
10	C-Multi	75.50 (3.00)	46.67 (2.00)	47.10 (2.00)	75.43 (3.00)
	I-Multi	45.50 (2.00)	74.33 (3.00)	73.90 (3.00)	40.03 (2.00)
	MOEA/D-DRA	**15.50 (1.00)**	**15.50 (1.00)**	**15.50 (1.00)**	**21.03 (1.00)**

Fig. 3 shows examples of boxplots for the hypervolume results obtained by the algorithms. Due to lack of space we only included boxplots for the problem WFG4. From this figure we can see that in general MOEA/D-DRA had better (higher) hypervolume results, except for three objectives, where I-Multi performed better. For three and five objectives, I-Multi outperforms C-Multi, however for eight and ten objectives C-Multi presents better results than I-Multi.

Considering all the results presented, we can conclude that despite the excellent performance of the state-of-the-art MOEA/D-DRA regarding the hypervolume, C-Multi can be a very competitive algorithm if we take into account a different quality indicator like IGD_p. Moreover, in general C-Multi achieved competitive results compared to I-Multi on hypervolume, and outperformed it on IGD_p.

A possible explanation for this poor performance on hypervolume is that even using an archiver that benefits the diversity, parts of the fronts are not being properly covered by C-Multi, specially the extremes. In general, the results obtained can be considered promising, pointing to the usefulness of keep investigating the hybridization of MOPSO with EDAs for many-objective optimization.

6. Conclusion

In this paper, we have built on a previous study where a hybrid algorithm called C-Multi was introduced. C-Multi was designed to meet two conflicting goals in MaOPs: convergence to the Pareto front by using an EDA, and diversity of the solutions by using multi-swarms. In the previous paper, despite presenting good results in some difficult problems, C-Multi had overall poor performance.

To overcome the limitation observed on C-Multi, we have investigated in this paper which is the role of the archiving method and the number of swarms, and how an appropriate selection of these components can enhance the searching capabilities of the algorithm. We have conducted a detailed empirical analysis of the effect of these components.

For the archiving method, we have analyzed how three different archivers (CD, Ideal and MGA) influence the multi-swarm phase of the algorithm. This study indicated that using the Ideal archiver generates the unwanted effect of creating “holes” in the resulting front. Therefore, it was concluded that it is better to use an archiver like CD that guarantees the maintenance of generated extreme solutions of the swarms favoring the diversity of the solutions kept in the archiver.

The second experimental study involved investigating the effect of the number of sub-swarms used in the multi-swarm phase of the search. We used ten different values for the number of sub-swarms: 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. This study obtained conflicting results regarding the indicators used, hence

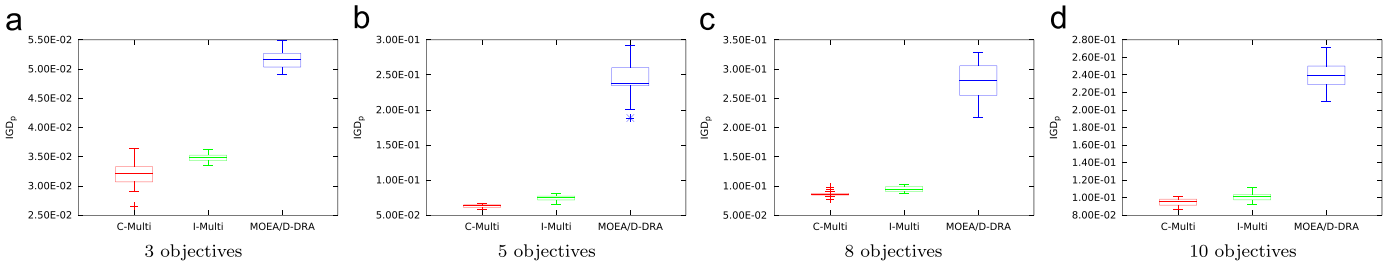

Fig. 2. IGD_p boxplots for WFG4.

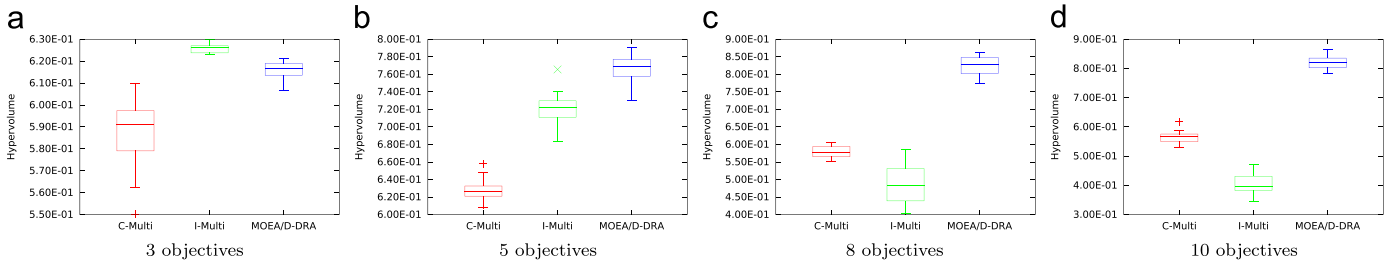

Fig. 3. Hypervolume boxplots for WFG4.

we configured the algorithm according to the most widely accepted indicator, in this case hypervolume.

Finally, experiments were conducted to compare C-Multi to I-Multi and the state-of-the-art MOEA/D-DRA. To consider a variety of difficult optimization scenarios, the entire WFG family of problems for 3, 5, 8 and 10 objectives, was used. The results obtained indicate that despite not being able to outperform MOEA/D-DRA in most cases regarding hypervolume, C-Multi had very good results when considering the IGD_p indicator. Moreover, C-Multi was able to outperform I-Multi in most cases regarding IGD_p, and presented competitive results according to hypervolume. A possible explanation for this poor performance on hypervolume is that part of the fronts are not being properly covered by C-Multi, specially the extremes. However, considering the results obtained, we can confirm our original hypothesis concerning the improvement of C-Multi's performance by an appropriate choice of the archiving strategy and the number of sub-swarms.

These results encourage further research on the hybridization between EDAs and MOPSOs to take advantage of their different capabilities to design more adaptive algorithms, able to explore the diverse scenarios that can be found in real-world many-objective optimization. Future works include using other EDAs to further improve convergence and the combination of decomposition and reference points to promote diversity, specially in the extremes of the search space.

References

[1] E. Hughes, Radar waveform optimisation as a many-objective application benchmark, in: Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 4403, Springer, Berlin, Heidelberg, 2007, pp. 700–714.

[2] H. Ishibuchi, N. Akedo, H. Ohyanagi, Y. Nojima, Behavior of EMO algorithms on many-objective optimization problems with correlated objectives, in: IEEE Congress on Evolutionary Computation, 2011, pp. 1465–1472.

[3] A. Britto, S. Mostaghim, A. Pozo, Iterated multi-swarm: a multi-swarm algorithm based on archiving methods, in: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, GECCO '13, ACM, New York, NY, USA, 2013, pp. 583–590.

[4] A. Britto, A. Pozo, Using archiving methods to control convergence and diversity for many-objective problems in particle swarm optimization, in: IEEE Congress on Evolutionary Computation, 2012, pp. 1–8. http://dx.doi.org/10.1109/CEC.2012.6256149.

[5] O.R. Castro Jr., A. Pozo, A hybrid competent multi-swarm approach for many-objective problems, in: Brazilian Conference on Intelligent Systems, 2014, pp. 426–431. http://dx.doi.org/10.1109/BRACIS.2014.82.

[6] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, vol.4, 1995, pp. 1942 –1948.

[7] P. Larrañaga, H. Karshenas, C. Bielza, R. Santana, A review on probabilistic graphical models in evolutionary computation, J. Heuristics 18 (5) (2012) 795–819, http://dx.doi.org/10.1007/s10732-012-9208-4.

[8] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable multi-objective optimization test problems, in: IEEE Congress on Evolutionary Computation, vol. 1, 2002, pp. 825 –830.

[9] Q. Zhang, W. Liu, H. Li, The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances, in: IEEE Congress on Evolutionary Computation, 2009, pp. 203–208. http://dx.doi.org/10.1109/CEC.2009.4982949.

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

[11] R.J. Lygoe, M. Cary, P.J. Fleming, A real-world application of a many-objective optimisation complexity reduction process, in: Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 7811, Springer, Berlin, Heidelberg, 2013, pp. 641–655. http://dx.doi.org/10.1007/978-3-642-37140-0_48.

[12] H. Ishibuchi, N. Tsukamoto, Y. Nojima, Evolutionary many-objective optimization: a short review, in: IEEE Congress on Evolutionary Computation, 2008, pp. 2419–2426.

[13] A.L. Jaimes, C.A.C. Coello, Some techniques to deal with many-objective problems, in: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, GECCO '09, ACM, New York, NY, USA, 2009, pp. 2693–2696.

[14] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach. Part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.

[15] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.

[16] K. Li, K. Deb, Q. Zhang, S. Kwong, Combining dominance and decomposition in evolutionary many-objective optimization, IEEE Trans. Evol. Comput. (99) (2014) 1. http://dx.doi.org/10.1109/TEVC.2014.2373386.

[17] S. Lalwani, S. Singhal, R. Kumar, N. Gupta, A comprehensive survey: applications of multi-objective particle swarm optimization (mopso) algorithm, Trans. Combin. 2 (1) (2013) 39–101.

[18] Yudong Zhang, Shuihua Wang, Genlin Ji, A comprehensive survey on particle swarm optimization algorithm and Its Applications, Mathematical Problems in Engineering 2015 (2015) 931256 http://dx.doi.org/10.1155/2015/931256 38 pp.

[19] M. Laumanns, R. Zenklusen, Stochastic convergence of random search methods to fixed size Pareto front approximations, Eur. J. Oper. Res. 213 (2) (2011) 414–421 http://dx.doi.org/10.1016/j.ejor.2011.03.039.

[20] C.A.C. Coello, M.S. Lechuga, MOPSO: a proposal for multiple objective particle swarm optimization, in: Proceedings of the 2002 Congress on Evolutionary Computation, CEC '02, vol. 2, IEEE Computer Society, Washington, DC, USA, 2002, pp. 1051–1056.

[21] A.J. Nebro, J.J. Durillo, J. Garcia-Nieto, C.A.C. Coello, F. Luna, E. Alba, SMPSO: a new PSO-based metaheuristic for multi-objective optimization, in: Computational Intelligence in Multi-criteria Decision-making, IEEE, Nashville, TN, 2009, pp. 66–73.

[22] K. Deb, S. Agrawal, A. Pratap, 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, PPSN VI, Springer-Verlag, London, UK, 2000, pp. 849–858.

[23] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, The MIT Press, Cambridge, MA, 2009.

[24] P. Larrañaga, J.A. Lozano (Eds.), Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Boston, Dordrecht, London, 2002.

[25] C. Ahn, R.S. Ramakrishna, D.E. Goldberg, Real-coded Bayesian optimization algorithm, in: Towards a New Evolutionary Computation, Studies in Fuzziness and Soft Computing, vol. 192, Springer, Berlin, Heidelberg, 2006, pp. 51–73. http://dx.doi.org/10.1007/3-540-32494-1_3.

[26] M. Pelikan, D.E. Goldberg, E. Cantu-Paz, BOA: The Bayesian Optimization Algorithm, Morgan Kaufmann, Orlando, FL, 1999, pp. 525–532.

[27] M. Hauschild, M. Pelikan, An introduction and survey of estimation of distribution algorithms, Swarm Evol. Comput. 1 (3) (2011) 111–128.

[28] C.W. Ahn, R. Ramakrishna, On the scalability of real-coded Bayesian optimization algorithm, IEEE Trans. Evol. Comput. 12 (3) (2008) 307–322.

[29] C.W. Ahn, Advances in Evolutionary Algorithms: Theory, Design and Practice, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[30] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 6 (2) (1978) 461–464.

[31] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Trans. Evol. Comput. 10 (5) (2006) 477–506, http://dx.doi.org/10.1109/TEVC.2005.861417.

[32] K. Li, K. Deb, Q. Zhang, S. Kwong, Combining dominance and decomposition in evolutionary many-objective optimization, IEEE Trans. Evol. Comput. (2014) 1. http://dx.doi.org/10.1109/tevc.2014.2373386.

[33] O. Schutze, X. Esquivel, A. Lara, C.A.C. Coello, Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization, IEEE Trans. Evol. Comput. 16 (4) (2012) 504–522, http://dx.doi.org/10.1109/TEVC.2011.2161872.

[34] L. While, L. Bradstreet, L. Barone, A fast way of calculating exact hypervolumes, IEEE Trans. Evol. Comput. 16 (1) (2012) 86–95.
[35] J. Bader, K. Deb, E. Zitzler, Faster hypervolume-based search using Monte Carlo sampling, in: Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems, Lecture Notes in Economics and Mathematical Systems, vol. 634, Springer, Berlin, Heidelberg, 2010, pp. 313–326.
[36] W.H. Kruskal, W.A. Wallis, Use of ranks in one-criterion variance analysis, J. Am. Stat. Assoc. 47 (260) (1952) 583–621.

Olacir R. Castro Jr. is a Ph.D. student at the Computer Science Department in Federal University of Paraná. He received the Master's degree in Computer Science at Federal University of Parana in 2013 and the B.S. degree in Information Systems from the Federal University of Acre in 2009. His main interests are evolutionary algorithms metaheuristic and multiobjective optimization.

Aurora Pozo is an associate professor of Computer Science Department and Numerical Methods for Engineering at Federal University of Paraná, Brazil, since 1997. She received an M.S. in electrical engineering from Federal University of Santa Catarina, Brazil, in 1991. She received a Ph.D. in electrical engineering from the Federal University of Santa Catarina, Brazil. Aurora's research interests are in evolutionary computation, data mining and complex problems.

Roberto Santana received the B.S. degree in computer science and the Ph.D. degree in mathematics from the University of Havana, Havana, Cuba, in 1996 and 2005, respectively, and the Ph.D. degree in computer science from the University of the Basque Country, San Sebastin–Donostia, Spain, in 2006. He is a Researcher with the University of the Basque Country. His research interests include evolutionary computation, probabilistic graphical models, neuroscience, and bioinformatics.