2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Unsupervised Clustering and Multi-Optima Evolutionary Search

Vassilis P. Plagianakos
Department of Computer Science and Biomedical Informatics
University of Thessaly, Lamia, GR–35100, Greece
Email: vpp@ucg.gr

***Abstract*—This paper pursues a course of investigation of an approach to combine Evolutionary Computation and Data Mining for the location and computation of multiple local and global optima of an objective function. To accomplish this task we exploit the spatial concentration of the population members around the optima of the objective function. Such concentration regions are determined by applying clustering algorithms on the actual positions of the members of the population. Subsequently, the evolutionary search is confined in the interior of the regions discovered. To enable the simultaneous discovery of more than one global and local optima, we propose the use of clustering algorithms that also provide intuitive approximations for the number of clusters. Furthermore, the proposed scheme has often the potential of accelerating the convergence speed of the Evolutionary Algorithm, without the need for extra function evaluations.**

***Index Terms*—Data Mining, Clustering, Differential Evolution, Particle Swarm Optimization, Global Optimization.**

I. Introduction

Evolutionary Algorithms (EAs) refer to problem solving optimization algorithms which employ computational models of evolutionary processes. These algorithms share the common conceptual base of simulating the evolution of the individuals (i.e. search points) that form the population using a predefined set of evolution operators. Commonly two kinds of operators are used: the *selection* and the *search* operators. The most widely used search operators are the *mutation* and the *recombination*. The selection operator mainly depends on the perceived measure of the fitness of each individual and enforces the concepts of natural selection and survival of the fittest. The search operators stochastically perturb the individuals providing efficient exploration of the search space. This perturbation is primarily controlled by the user defined recombination and mutation rates. Although simplistic from a biologist's point of view, these algorithms are sufficiently complex to provide robust and powerful search mechanisms and have shown their strength in solving hard real-world optimization problems.

On the other hand, recent advances in Data Mining research have allowed the development of tools and techniques with the unprecedented ability to extract meaningful information from diverse types of data. Utilizing these new developments to discover additional knowledge from an optimization procedure is the primary focus of this paper. The utilization of Data Clustering algorithms on the population of the evolutionary algorithm is proposed with double earnings; primarily to gain further insight and knowledge for the specific problem, and secondly to exploit that information aiming to enhance the dexterity of the evolutionary process itself.

Thus, the research goal of this paper is the combination of EAs and unsupervised clustering algorithms in an attempt to locate and compute multiple global and local optima of a multimodal objective function. More specifically, we propose the synergy of clustering algorithms that automatically provide intuitive approximations for the number of clusters, with EAs having the property to better explore the search space. This approach has the potential of locating multiple local and global minima efficiently with minimal extra computational cost and, in some cases, it can accelerate the EA's convergence speed [1].

The paper proceeds with a presentation of related work in the next Section. In the Sections III and IV, we describe the specific clustering and evolutionary algorithms employed in this study. Section V, presents the proposed scheme for the combination of Data Clustering and Evolutionary Algorithms. The experimental results in Section VI demonstrate the applicability of the proposed methodology. The paper ends with concluding remarks and a short discussion regarding future research directions.

II. Related Work

In the literature there exist numerous techniques for the discovery of multiple local and global optima that date back to the 1970s. These algorithms can be broadly divided into the following categories: i) multistart techniques, ii) objective function modification algorithms, iii) niching algorithms, iv) algorithms that learn the objective function, and v) clustering based techniques.

Multistart techniques (e.g. [2]–[4]) have their roots on pre-existing methods, such as the *Multi-Level Single Linkage* and the *Topographical Global Optimization* [5]. Objective function modification techniques transform the multimodal objective function in an attempt to remove undesired local minimizers, and are not directly connected with a particular optimization algorithm [6]–[8].

On the other hand, niching methods divide the population into semi-isolated subpopulations (*niches*), in an attempt to maintain a sufficient number of solutions at each subpopulation, preventing premature convergence. Subsequently,

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

they encourage local competition between certain members of the population, so as to allow the formation of solution clusters [9]–[11].

Algorithms that try to learn the objective function, do not try to provide multi-minima solutions, but rather use data mining techniques to better guide the evolutionary process. The *Covariance Matrix Adaptation Evolution Strategy* (CMA-ES) algorithm [12] is one of the most successful approaches. The interested reader can find similar approaches in [13], [14]. Although the majority of these algorithms were introduced as minimization methods, they can also be used to compute the maximizers of a multimodal objective function.

III. Data Clustering

Clustering techniques date back in the fourth century B.C. from the attempts by Aristotle and Theophrastos to categorize plants according to their morphological characteristics. However, the first comprehensive foundations of these methods were published in 1939 [15]. Clustering can be defined as the process of partitioning a set of patterns into disjoint and homogeneous meaningful groups, called clusters. Although there exists an abundance of clustering methods, in this study we consider two unsupervised clustering algorithms; namely the Unsupervised k–Windows (UkW) [16] and the DBSCAN [17] clustering algorithms. This selection was based on two reasons. Firstly, both algorithms have proven to be effective and efficient in many data mining tasks. Secondly and most importantly, both have the ability to automatically provide approximations for the number of clusters in a dataset. This is a central issue in modern cluster analysis, since many popular clustering methods require special handling to provide this functionality.

A. Unsupervised k–Windows Clustering Algorithm

Density based clustering algorithms operate by identifying regions of high density in dataset objects, surrounded by regions of low density. One method in this class is the "Unsupervised k-Windows" (UkW), which utilizes hyperrectangles to discover clusters. The algorithm makes use of techniques from computational geometry and encapsulates clusters using linear containers in the shape of d-dimensional hyperrectangles that are iteratively adjusted with movements and enlargements until a certain termination criterion is satisfied [16]. The movement procedure aims at positioning each window as close as possible to the center of a cluster, while the enlargement process attempts to enlarge the window so that it includes as many objects from the current cluster as possible. These two steps are illustrated in Figure 1. Notice that with proper tuning, the algorithm is able to detect clusters of arbitrary shapes.

Additionally, UkW provides an estimate for the number of clusters that exist in the dataset. The key idea is to initialize a large number of windows. When the movement and enlargement of all windows terminate, all overlapping windows are considered for merging by considering their intersection. An example of this operation is exhibited in Figure 2. For a detailed description of the UkW algorithm see [16].

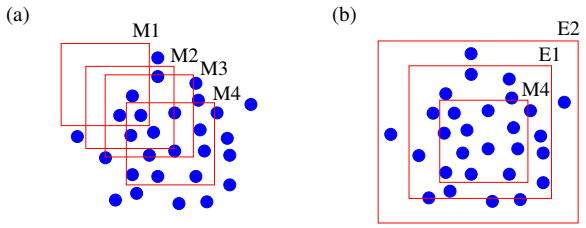

Fig. 1. (a) Sequential movements M2, M3, M4 of initial window M1. (b) Sequential enlargements E1, E2 of window M4.

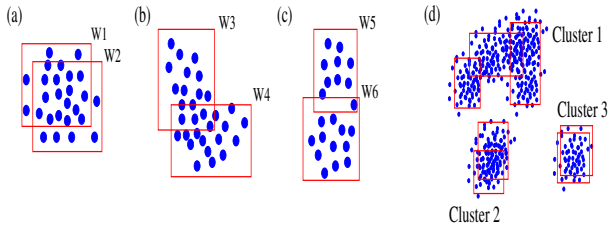

Fig. 2. (a) W_1 and W_2 satisfy the similarity condition and W_1 is deleted. (b) W_3 and W_4 satisfy the merge operation and are considered to belong to the same cluster. (c) W_5 and W_6 have a small overlap and capture two different clusters. (d) An example of how the UkW algorithm can discover non-convex shaped clusters.

B. The DBSCAN Clustering Algorithm

The DBSCAN [17] clustering algorithm is designed to discover clusters of arbitrary shape as well as to distinguish noise. More specifically, the algorithm is based on the idea that in a neighborhood of a given radius (Eps) for each point in a cluster at least a minimum number of objects ($MinPts$) should be contained. Such points are called core points and each point in their neighborhood is considered as "Directly Density-Reachable" from that. Consequently the algorithm uses the notion of density reachable chains of objects; i.e. a point q is "Density-Reachable" from a point p, if there is a chain of objects $p_1, \ldots, p_k$ such that $p_1 = q$, $p_k = p$ and p_{i+1} is "Directly Density-Reachable" from p_i for $i = 1, \ldots, k$. Finally, a point p is defined as "Density Connected" to a point q, if there is a point o that both p, q are "Density-Reachable" from that. Figure 3, illustrates an example of these definitions.

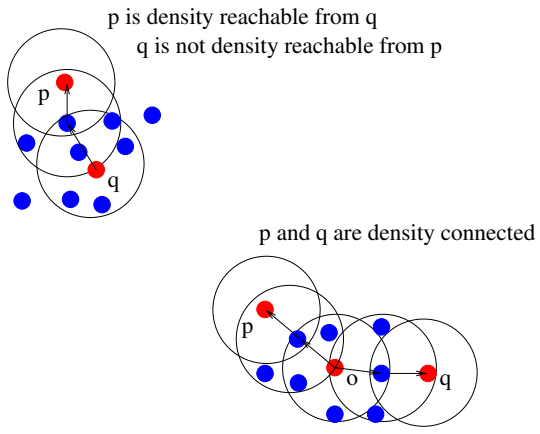

Fig. 3. An example of "Density-Reachable" and "Density Connected" points.

The algorithm considers as a cluster the subset of points from the dataset that are "Density-Reachable" from each other and additionally each pair of points inside the cluster is "Density Connected". Any point of the dataset not in a cluster is considered as noise.

To discover the clusters the algorithm retrieves density-reachable points from the data by iteratively collecting directly density-reachable objects. The algorithm scans the eps neighborhood of each point in the database. If that neighborhood has more than $MinPts$ points a new cluster C containing them is defined. Then, the neighborhood of all points q in C which have not yet been processed is checked. If the points in neighborhood of q are more than $MinPts$, then those which are not already contained in C are added to the cluster and their neighborhood will be checked in a subsequent step. This procedure is iterated until no new point can be added to the current cluster C.

An initial value of the eps parameter can be determined by using a k-distance graph of the data, where k can be derived from the number of dimensions in the data set.

IV. Evolutionary Algorithms

Among the numerous methods belonging to the class of EAs, the Differential Evolution (DE) [18] and the Particle Swarm Optimization (PSO) [19] have been selected to demonstrate the proposed synergy of clustering and evolutionary algorithms. DE and PSO are optimization methods capable of locating one optimum of nondifferentiable, nonlinear and multimodal objective functions. Note that even though the benchmarks studied in the paper are mainly minimization tasks, DE and PSO are optimization methods capable of both maximizing and minimizing an objective function. Next, an brief overview of these algorithms is presented.

A. The Differential Evolution Algorithm

DE has been designed as a stochastic parallel direct search method that typically requires few, easily chosen, control parameters. Experimental results have shown that DE has good convergence properties and outperforms other well known evolutionary algorithms [20], [21]. A population of individuals is randomly initialized in the optimization domain S, by uniformly sampling *NP* n–dimensional vectors in the search space. Subsequently, the individuals are iteratively evolved in order to explore S and locate the minima of the objective function. Note that *NP* remains constant throughout the evolution.

At each iteration, called *generation*, new vectors (*offsprings*) are generated by a combination of randomly chosen vectors. This operation in our context can be referred to as *mutation*. At a next step, the *recombination* operation mixes the offspring vectors with another predetermined vector – the *target* vector. This yields the so–called *trial* vector. The trial vector replaces the target vector if and only if it yields a reduction in the value of the objective function f. This last operation can be referred to as *selection*. Performing the mutation, recombination and selection operations for all the population members constitutes a single iteration of the DE algorithm.

Below, the DE mutation operators used in this paper are outlined. Specifically, for each individual x_g^i, $i = 1, \ldots,$ *NP*, where g denotes the current generation, a new individual v_{g+1}^i (mutant vector) is generated according to one of the following equations:

$$v_{g+1}^i = x_g^{\text{best}} + \mu(x_g^{r1} - x_g^{r2}), \quad (1)$$

$$v_{g+1}^i = x_g^{r1} + \mu(x_g^{r2} - x_g^{r3}), \quad (2)$$

$$v_{g+1}^i = x_g^i + \mu(x_g^{\text{best}} - x_g^i) + \mu(x_g^{r1} - x_g^{r2}), \quad (3)$$

$$v_{g+1}^i = x_g^{\text{best}} + \mu(x_g^{r1} - x_g^{r2}) + \mu(x_g^{r3} - x_g^{r4}), \quad (4)$$

$$v_{g+1}^i = x_g^{r1} + \mu(x_g^{r2} - x_g^{r3}) + \mu(x_g^{r4} - x_g^{r5}), \quad (5)$$

where x_g^{best} is the best member of the previous generation; $\mu > 0$ is a real parameter, called *mutation constant*, which controls the amplification of the difference between two individuals so as to avoid the stagnation of the search process; and $r_1, r_2, r_3, r_4, r_5 \in \{1, 2, \ldots, i-1, i+1, \ldots, NP\}$, are random integers mutually different and not equal to the running index i.

It is evident that more such relations can be generated using the above ones as building blocks (see for example [22], where new DE operators are genetically programmed). Another such example is the trigonometric mutation operator [23]. For the rest of the paper, we call DE_1 the differential evolution algorithm that uses Equation (1) as the mutation operator, DE_2 the algorithm that uses Equation (2), and so on. The trigonometric mutation operator is also tested (DE_6).

B. The Particle Swarm Optimization

To illustrate the generality of our approach, we will also apply it to the Particle Swarm Optimization (PSO) algorithm [24]. PSO, is an optimization method inspired by the social dynamics and emergent behavior that arises in socially organized colonies [19]. For this reason, the population is usually called *swarm* and the individuals are called *particles*. The algorithm operates by iteratively moving each particle with an adaptable velocity within the search space and retains a memory of the best positions it ever encountered. Additionally, each particle is assigned to a topological neighborhood consisting of a predetermined number of particles. If all the population particles are included in this neighborhood then the algorithm is characterized as *global* PSO; otherwise as *local* PSO.

Assume a swarm consisting of *NP* particles. The i–th particle is in effect an n–dimensional vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in S$. The velocity of this particle is also an n–dimensional vector, $V_i = (v_{i1}, v_{i2}, \ldots, v_{in})^\top \in S$. The best previous position encountered by the i–th particle is a point in S, denoted by $P_i = (p_{i1}, p_{i2}, \ldots, p_{in})^\top \in S$. Assume g to be the index of the particle that attained the best previous position among all the particles in the swarm neighborhood and t to be the iteration counter. Then, the evolution of the

swarm is described by the following equations [24]:

$$V_i(t+1) = \chi\Big[wV_i(t) + c_1\, r_1\big(P_i(t) - X_i(t)\big) + \\ + c_2\, r_2\big(P_g(t) - X_i(t)\big)\Big], \quad (6)$$

$$X_i(t+1) \;\; = \;\; X_i(t) + V_i(t+1), \quad (7)$$

where $i = 1, \ldots, NP$; c_1 and c_2 are two parameters called the *cognitive* and the *social* parameter, respectively, and they are used to bias the search of a particle toward its best experience and the best experience of the whole swarm, respectively; r_1, r_2, are random numbers uniformly distributed within $[0, 1]$. The parameters χ and w are called the *constriction factor* and the *inertia weight*, respectively, and they are used alternatively as mechanisms for the control of the velocity's magnitude, giving rise to the two different PSO versions (CPSO and IPSO, respectively). The selection of the aforementioned parameters has been widely discussed and studied (see for example [25]).

V. The Proposed Scheme

To effectively exploit the above described notions in practice, we present an algorithmic scheme combining the DE and an unsupervised clustering algorithm algorithm. Note that this approach is not DE exclusive, but can be applied to any EA variant. We have also use it with the PSO algorithm and the results were similar. The proposed approach applies the unsupervised clustering method only once, after a user–defined number of generations. For the case of DE, only a small number of generations is needed to sufficiently explore the search space. Afterwards, the clusters of individuals around the optima are determined and subpopulations are confined within each region. Each subpopulation has NP/β individuals, where β is the number of detected clusters. Notice that β is automatically computed by the unsupervised clustering algorithm; no user intervention is required. If a region contains more individuals, the best NP/β of them are selected. On the other hand, if a cluster contains less individuals, new ones are randomly initialized in the region. The results obtained in this manner consist of the minimizers computed in a single run. This algorithmic scheme (Clustering assisted DE) is outlined in Algorithm 1.

Although not necessary, to better utilize the proposed approach it is possible to start the DE algorithm using a mutation operator that permits adequate exploration of the search space (for example DE_2 or DE_6). Once the clusters around the minimizers have been determined by the unsupervised clustering algorithm, one can switch to a DE mutation operator that has faster convergence speed (for example DE_1) [1]. Note that the DE algorithm in Steps 16 and 17 of the above scheme can be replaced by any optimization method (even a non evolutionary one).

For very hard optimization problems, when the objective function is defined in many dimensions and possesses multitudes of local and global minima, the clustering algorithm could be called more than once. The same might be true for real–life optimization tasks, where the function value of the global minimum is unknown. Each consecutive call of the clustering algorithm will result in more promising subregions of the original search space and will save unneeded objective function evaluations, since the subpopulations will stay focused on regions containing desirable minima. For all the experiments conducted in this paper, one call of the clustering algorithm is sufficient for the location of many local and (possibly) global minima.

Algorithm 1 Clustering assisted DE

1: Initialize the population of *NP* individuals
2: Evaluate the fitness of each individual
3: **repeat**
4: **for** $i = 1$ **to** *NP* **do**
5: Mutation(x_g^i) $\rightarrow$ Mutant$_g^i$
6: Recombination(Mutant$_g^i$) $\rightarrow$ Trial$_g^i$
7: **if** f(Trial$_g^i$) $\leqslant f(x_g^i)$ **then**
8: accept Trial$_g^i$ for the next generation
9: **end if**
10: **end for**
11: **until** the search space is sufficiently explored
12: Call an unsupervised clustering algorithm once
13: Automatically estimate the number of clusters (β)
14: Assign each DE individual to its corresponding cluster
15: **for** $j = 1$ **to** β **do**
16: Confine NP/β individuals within each cluster
17: Use the DE algorithm to compute each minimum
18: **end for**
19: **return** all the computed minima

To determine the applicability and the efficiency of the DE and clustering synergy, we applied it to the following simple multimodal 2–dimensional function:

$$f(x_1, x_2) = \sin(x_1)^2 + \sin(x_2)^2, \quad (8)$$

where $(x_1, x_2) \in \mathbb{R}^2$. This function has an infinite number of global minima in $\mathbb{R}^2$, with function values equal to zero, at the points $(\kappa\pi, \lambda\pi)$, where $\kappa, \lambda \in \mathbb{Z}$. Restricted in the hypercube $[-5, 5]^2$ the function f has 9 global minima. An illustrative clustering result of the individuals using the UkW algorithm is exhibited in Figure 4. The final windows of the UkW algorithm are also depicted.

VI. Presentation of Experiments

To evaluate the performance of the proposed scheme we have conducted two independent sets of experiments. First, in Section VI-A, we have employed two popular multimodal test functions to evaluate the applicability and the properties of the two unsupervised clustering algorithms considered here. Next, Section VI-B is devoted to the comparison of our approach against similar techniques found in the literature.

In the tables below, we report the average number of Function Evaluations (FE) needed before an algorithm was stopped by any of the following three termination criteria: i) the global minimum was located with accuracy 10^{-6}, ii) at any time the largest Euclidean distance among the population

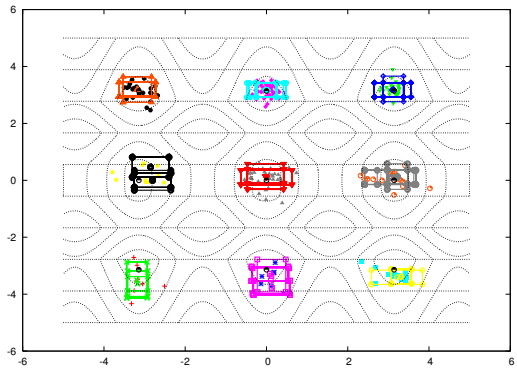

Fig. 4. Clusters and global minimizers discovered in f

members is less than 10^{-6} (i.e. the population is considered converged), and iii) a predefined number of generations is reached.

In the first set of experiments, three different approaches are evaluated: the original DE algorithm (six different mutation operators), the UkW assisted DE, and the DBSCAN assisted DE. For the latter two algorithms, we also report the average number of detected minima. Note that in the first set of experiments, the UkW assisted DE and the DBSCAN assisted DE algorithms always located the global minimum of the objective functions. For the second set of experiments, to make the comparisons straightforward, we have tried to use algorithm settings that are as close as possible to the settings in the original publications. All the simulations mentioned above were performed 100 times with different random seeds to ensure statistical accuracy. The mutation and recombination constants of the DE algorithms had fixed values $\mu = 0.6$ and $\rho = 0.8$, respectively. These values are commonly encountered in DE publications and no attempt was made to fine-tune the algorithm.

A. Applying and Comparing the Clustering Algorithms

In the first set of experiments the Shekel's Foxholes and the 10–dimensional Griewangk test functions were considered. Note that in the experiments below the clustering step was applied after 20 generations. The Shekel's Foxholes is a relatively easy test function and was included in the experiments to investigate the performance of the unsupervised clustering algorithms when applied to problems in which the DE algorithm requires a relatively small number of generations to locate the global minimum.

In Table I the average performance of the algorithms on the Shekel's foxholes test function is exhibited. It is clear that the use of the unsupervised clustering algorithms results in the computation of many minima at once, but the problem is so easy that there is always an increase in the number of function evaluations needed to locate the global minimum. This fact indicates that the proposed approach is better suited to difficult optimization tasks, when more than one minimum exist.

On the contrary, the experimental results from the Griewangk test function, illustrated in Table II, show that the clustering greatly accelerates the DE algorithms (ranging from 80% to 260%). DE_1 is the only exception; although on average 14 minima are located the proposed algorithm requires 27% additional generations.

	original DE	UkW assisted DE		DBSCAN assisted DE	
	FE	Minima	FE	Minima	FE
DE_1	1540	7.81	2516	5.58	3182
DE_2	2590	12.58	2738	10.22	4448
DE_3	408	9.34	2486	4.40	3572
DE_4	2150	8.13	2448	4.63	3844
DE_5	2372	9.87	3370	11.02	4558
DE_6	552	14.33	2716	8.75	4328

TABLE I
RESULTS FOR THE SHEKEL FUNCTION: AVERAGE NUMBER OF FUNCTION EVALUATIONS (FE) NEEDED AND AVERAGE NUMBER OF COMPUTED MINIMA, WITH AND WITHOUT THE AID OF A CLUSTERING ALGORITHM

	original DE	UkW assisted DE		DBSCAN assisted DE	
	FE	Minima	FE	Minima	FE
DE_1	60468	14.19	83110	1.05	85600
DE_2	174786	8.58	69262	1.88	191734
DE_3	159812	23.36	87964	3.93	242180
DE_4	256084	4.40	71304	0.21	255800
DE_5	363218	1.65	114260	0.08	224100
DE_6	123886	18.09	65662	3.08	177816

TABLE II
RESULTS FOR THE GRIEWANGK FUNCTION: AVERAGE NUMBER OF FUNCTION EVALUATIONS (FE) NEEDED AND AVERAGE NUMBER OF COMPUTED MINIMA, WITH AND WITHOUT THE AID OF A CLUSTERING ALGORITHM

The experimental results indicate that the utilization of an unsupervised clustering algorithm aids the location of multiple minima. In relatively easy problems, there may be an increase in the number of function evaluations needed to locate the global minimum. However, in difficult optimization tasks the clustering step usually accelerates the convergence of the DE algorithms. Additionally, the UkW assisted DE algorithm exhibited increased convergence speed and located more minima when compared against the DBSCAN assisted DE. Thus, the use of the UkW clustering algorithm is proposed and for the next experiment only UkW experimental results are reported.

B. Comparison with other approaches

Here we conduct an experimental analysis that allows to compare the performance of the proposed methodology against other methods [9], [26]–[30].

Firstly, the proposed approach will be compared against the method proposed in [27]. To this end, we utilize the Levy No. 5 test function, which has has about 760 local minima and one global minimum with function value $f_{\text{levy}}(-1.3068, -1.4248) = -176.1375$. The large number of local optimizers makes it difficult for any method to locate the global minimizer.

The experimental results are exhibited in Table III. As shown the use of the unsupervised clustering algorithm enhances the performance of the DE algorithms. In detail, there is an average acceleration of the algorithm's convergence speed ranging from 30% to 80%. Additionally, as many as 20 minima (including the global one) were simultaneously computed. The only exception is DE_1 where a slight increase in the function evaluations is observed (3%), but the modified algorithm locates, in average, the global as well as 6 local minima.

	original DE	UkW assisted DE	
	FE	Minima	FE
DE_1	6642	5.97	6872
DE_2	14132	20.52	10852
DE_3	12818	11.96	7954
DE_4	13022	20.22	10050
DE_5	26602	22.70	14170
DE_6	12978	19.20	10048

TABLE III
RESULTS FOR THE LEVY FUNCTION: NUMBER OF FUNCTION EVALUATIONS (FE) NEEDED AND AVERAGE NUMBER OF COMPUTED MINIMA, WITH AND WITHOUT THE AID OF A CLUSTERING ALGORITHM

Using the deflection technique proposed in [27] in combination with PSO, the authors show that is possible to detect 2 minima of the function, over as many as $3,026.5$ iterations of a swarm size of 20 particles. This corresponds to a mean total of $3,026.5 \times 20 = 60,530$ function evaluations. Compared to the worst results of $14,170$ function evaluations required by DE_5 to discover on average more than 20 minima is a significant improvement.

Next, we compare the proposed technique against the popular NichePSO type of methods proposed in [28]–[30]. To allow direct comparisons, we will employ the same test functions and we will follow as closely as possible the experimental design described in the corresponding publications. The five test functions are defined as follows:

$$
\begin{aligned}
f_{\mathrm{niche}_1}(x) &= \sin^6(5\pi x),\\
f_{\mathrm{niche}_2}(x) &= e^{-2\log(2)(\frac{x-0.1}{0.8})^2}\sin^6(5\pi x),\\
f_{\mathrm{niche}_3}(x) &= \sin^6\left(5\pi(x^{3/4}-0.05)\right),\\
f_{\mathrm{niche}_4}(x) &= e^{-2\log(2)(\frac{x-0.1}{0.8})^2}\sin^6\left(5\pi(x^{3/4}-0.05)\right),\\
f_{\mathrm{niche}_5}(x) &= 200-(x^2+y-11)^2-(x+y^2-7)^2.
\end{aligned}
$$

Notice that these are maximization problems. The test functions f_{niche_1} and f_{niche_3} both have 5 maxima with a function value of 1.0. In f_{niche_1}, maxima are evenly spaced, while in f_{niche_3} maxima are unevenly spaced. Functions f_{niche_2} and f_{niche_4} are oscillating functions, and the local and global peaks exist at the same x-positions as in functions f_{niche_1} and f_{niche_3}. The modified Himmelblau function $f_{\mathrm{niche}_5}(x)$ has 4 equal maxima with function values equal to 200. The test functions f_{niche_1} to f_{niche_4} are investigated in the range $[0,1]$, while the optima of $f_{\mathrm{niche}_5}(x)$ are being sought in the range $[-6,6]$.

Table IV illustrates the results of the proposed scheme in terms of Mean Function Value (FV) among the computed minima and Accuracy as defined in [30] (the proportion of the total minima discovered). The results in this case are averaged over 100 independent experiments. Two different approaches are examined; the UkW assisted CPSO and the UkW assisted IPSO, corresponding to the *constriction factor* and the *inertia weight* versions of the PSO algorithm, respectively. The population size was set to 500 individuals in all cases.

	UkW assisted CPSO		UkW assisted IPSO	
	Mean FV	Accuracy	Mean FV	Accuracy
f_{niche_1}	0.99	50%	0.96	50%
f_{niche_2}	0.9	50%	0.96	48%
f_{niche_3}	0.99	44%	0.99	48%
f_{niche_4}	0.99	26%	0.99	28%
f_{niche_5}	199.37	90%	199.36	100%

TABLE IV
RESULTS OF THE CLUSTERING ASSISTED PSO METHODS

The parameters of the PSO algorithms had values commonly found in many PSO publications, i.e. $c_1 = c_2 = 2.05$ and $\chi = 0.729$. All the different neighborhood sizes from 2 to 250 were examined. Using a small neighborhood size (less than 10) produced on average the best results in terms of accuracy. On the other hand, as expected, large neighborhood sizes (larger than 100) produced the best results in terms of mean function values. The larger the neighborhood size the better the search space exploitation. Table IV reports the results for a neighborhood of size 5 particles. Notice that the reported results are worse in terms of accuracy for functions f_{niche_1} to f_{niche_4} compared to those reported in [28]–[30]. Visually inspecting the population of the particles we determined that the minima in these test functions are located very close to one another. Thus, the evolution of the swarm is mostly influenced by one or two of them and the algorithms loses the ability to cluster around all different minima. However, the proposed scheme does not aim to locate all the minima, but rather to exploit whatever clustering structure already exists in the population. In the case of f_{niche_5} however, PSO shows a clear clustering structure that is exploited by UkW and in effect the accuracy of results is increased up to 100%.

To compare against the work reported in [26], we utilize the Rastrigin test function. This function is a typical example of a non-linear multimodal function and is a fairly difficult problem due to its large number of local minima. Table V summarizes the average results of 100 experiments of the 2-dimensional Rastrigin function. The results are similar to the results for the Levy function. Faster convergence is exhibited for all the DE algorithms (except for DE_1). The average increase in convergence speed was from 10% to 130% and up to 20 minima were simultaneously located. In [26], the authors report that the 2-dimensional version of this function, requires a mean number of $2,396$ function evaluations to discover approximately 30 minima. They also show the superior results against SPSO, SCGA and SNGA algorithms [31]–[33]. The

technique proposed in this paper applied on the DE_5 requires a mean number of approximately 11000 function evaluations to discover approximately 19 minima. While these results are worse than those reported in [26], we should note that they utilize iterative execution of the clustering algorithm that strengthens the explicative capabilities of the evolution, but also increases the computational cost.

	original DE	UkW assisted DE	
	FE	Minima	FE
DE_1	4186	1.00	5268
DE_2	12334	17.19	9144
DE_3	16022	11.23	6976
DE_4	9434	13.89	8528
DE_5	19738	19.43	10948
DE_6	11052	15.72	8856

TABLE V
RESULTS FOR THE 2-DIMENSIONAL RASTRIGIN FUNCTION: AVERAGE NUMBER OF FUNCTION EVALUATIONS (FE) NEEDED AND AVERAGE NUMBER OF COMPUTED MINIMA, WITH AND WITHOUT THE AID OF A CLUSTERING ALGORITHM

On the contrary, when the 6-dimensional form of the Rastrigin function is optimized, the results are different. Table VI illustrates the results obtained using UkW assisted DE. These results are fairly comparable with those reported in [26], since in this case the method proposed here manages to discover an equivalent numbers of minima using less function evaluations. In [26] it is reported that the proposed method requires more than 40000 function evaluations to discover 21 minima. Note also that their method is unable to discover the global optimum in all cases. The latter is also happening in this case for DE_2,DE_4 and DE_5, which is the reason that their results are not reported in Table VI. The enhanced performance exhibited by the remaining DE variants can be attributed to the unsupervised UkW clustering algorithm used here, compared to the supervised fuzzy k-means variant used in [26].

Finally, to compare our approach against [9], we try to locate the minimum of the Weierstrass function, which is a multimodal and non-separable minimization problem. Following the experimental approach reported in [9], we use the same parameter values. In this case there exist 16 minima, when $x_i \in \{-1.5, -0.5, 0.5, 1.5\}$. The algorithm proposed in [9] is able to discover the 16 minima (requiring a change of the parameters) and it achieves that using a population size of 1000 individuals evolved over 100 generations, resulting in 100000 function evaluations. Our approach employed a population of just 100 individuals that was initially evolved for 10 generations. Subsequently, as the proposed scheme suggests, the unsupervised clustering algorithm was applied and the population was dynamically divided in clusters. The experimental results are reported in Table VII. Although the proposed scheme may fail to locate every minimum, it needs far less function evaluations and exhibit increased convergence speed. So not only faster optimization procedure is facilitated, but also additional information (the location of other minimisers) is extracted.

	Mean Minima Detected	Function Evaluations	Variance of Function Evaluations
DE_1	35.05	32939.5	32.55
DE_3	37.44	32682.8	117.63
DE_6	16.05	36412.1	131.51

TABLE VI
RESULTS FOR THE 6-DIMENSIONAL RASTRIGIN FUNCTION: AVERAGE NUMBER OF COMPUTED MINIMA, AVERAGE NUMBER AND VARIANCE OF FUNCTION EVALUATIONS NEEDED.

	Mean Minima Detected	Function Evaluations	Variance of Function Evaluations
DE_1	6.4	4689.75	119.51
DE_2	14.25	8615.41	250.91
DE_3	13.4	7225.55	217.84
DE_4	14.05	8594.65	239.84
DE_5	14.05	9438.75	228.77
DE_6	14	8690.11	237.94

TABLE VII
RESULTS FOR THE WEIERSTRASS FUNCTION: AVERAGE NUMBER OF COMPUTED MINIMA, AVERAGE NUMBER AND VARIANCE OF FUNCTION EVALUATIONS NEEDED.

The experimental results indicate the the proposed approach aid to the location of the many optima of high–dimensional objective functions. Furthermore, in general, fewer function evaluations are required for the EA to converge Finally, comparisons against other well known similar techniques have shown the applicability of the combination of Unsupervised Clustering and Evolutionary Algorithms.

VII. CONCLUDING REMARKS

Evolutionary Algorithms have the tendency to concentrate large portions of their population in the vicinity of various local or global optima. This can be exploited to further improve the convergence properties of the EAs. In this work we propose to utilize unsupervised clustering algorithms to identify those concentration regions and subsequently confine the evolutionary search inside them. Especially in the case of multimodal objective functions, the determination of more than one regions can lead to the computation of many global and local minima simultaneously.

The UkW and the DBSCAN algorithms studied in this paper are density based clustering methods, having few and easily tuned control parameters. They have the potential to discover clusters of arbitrary shapes (e.g. non-convex shaped clusters) [16]. In brief, the proposed scheme has the following properties:

1) is able to locate the many local optima and (possibly) the global one,
2) there is no need for additional function evaluations,
3) fewer function evaluations are generally required for the EA to converge,
4) is better suited to high–dimensional objective functions having many optima.

Thus, even in the case that there is a single global optimum, the usage of the proposed approach is beneficial in terms of function evaluations. Note that there is a computational cost associated the clustering steps, that needs to be taken under consideration. Compared against other similar approaches the proposed methodology proved effective, although does not try to explicitly enforce a clustering structure on the population.

Promising future research directions include the utilization of niching techniques, like those described in [9]. This would probably enhance the explorative capabilities of the EA algorithm and thus further improve the multi-optima discovery potential of our method.

ACKNOLEDGEMENT

The author would like to thank the European Union (European Social Fund ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: Thalis: "Interdisciplinary Research in Affective Computing for Biological Activity Recognition in Assistive Environments", for financially supporting this work.

REFERENCES

[1] V. Plagianakos, "Multi-optima search using differential evolution and unsupervised clustering," in *Congress on Evolutionary Computation (CEC 2013)*, 2013, pp. 2178–2185.

[2] A. Törn, "Cluster analysis as a tool in a global optimization model," in *Third International Congress of Cybernetics and Systems, Bucharest*. Springer Verlag, 1977, pp. 249–260.

[3] G. Timmer, "Global optimization - a stochastic approach," Ph.D. dissertation, Erasmus University Rotterdam, 1994.

[4] A. Torn and A. Zilinskas, *Global Optimization*. Springer-Verlag, Berlin, 1989.

[5] C. Floudas, *State of the art in global optimization : computational methods and applications*. Kluwer Academic Publishers, 1996.

[6] A. A. Goldstein and J. F. Price, "On descent from local minima," *Math. Comput.*, vol. 25, no. 3, pp. 569–574, 1971.

[7] S. Kirkpatrick, C. Gelatt, Jr, and M. Vecchi, "Science," *Optimization by simulated annealing*, vol. 220, pp. 671–680, 1983.

[8] K. Parsopoulos, V. Plagianakos, G. Magoulas, and M. Vrahatis, "Objective function "stretching" to alleviate convergence to local minima," *Nonlinear Analysis, Theory, Methods and Applications*, vol. 47, pp. 3419–3424, 2001.

[9] K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1062–1087, 2008.

[10] G. Harik, "Finding multiple solutions in problems of bounded difficulty," Urbana, IL: University of Illinois at Urbana-Champaign, Tech. Rep. IlliGAL Report. No. 94002, 1994.

[11] D. Goldberg and L. Wang, "Adaptive niching via coevolutionary sharing," *Genetic Algorithms and Evolution Strategies in Engineering and Computer Science*, pp. 21–38, 1997.

[12] N. Hansen, "The cma evolution strategy: A comparing review," in *Towards a new evolutionary computation*, ser. Advances in estimation of distribution algorithms, J. Lozano, P. Larraga, I. Inza, and E. Bengoetxea, Eds. Springer, 2006, pp. 75–102.

[13] M. Milano, P. Koumoutsakos, and J. Schmidhuber, "Self-organizing nets for optimization," *Neural Networks, IEEE Transactions on*, vol. 15, no. 3, pp. 758–765, 2004.

[14] M. Preuss, "Improved topological niching for real-valued global optimization," in *Applications of Evolutionary Computation*, ser. Lecture Notes in Computer Science, C. Chio, A. Agapitos, S. Cagnoni, C. Cotta, F. Vega, G. Caro, R. Drechsler, A. Ekrt, A. Esparcia-Alczar, M. Farooq, W. Langdon, J. Merelo-Guervs, M. Preuss, H. Richter, S. Silva, A. Simes, G. Squillero, E. Tarantino, A. Tettamanzi, J. Togelius, N. Urquhart, A. Uyar, and G. Yannakakis, Eds. Springer Berlin Heidelberg, 2012, vol. 7248, pp. 386–395.

[15] C. Tryon, *Cluster Analysis*. Ann Arbor, MI: Edward Brothers, 1939.

[16] D. Tasoulis and M. Vrahatis, "Novel approaches to unsupervised clustering through the k-windows algorithm," in *Knowledge Mining*, ser. Studies in Fuzziness and Soft Computing, S. Sirmakessis, Ed. Springer-Verlag, 2005, vol. 185, pp. 51–78.

[17] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169–194, 1998.

[18] R. Storn and K. Price, "Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[19] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings IEEE International Conference on Neural Networks*, vol. IV. Piscataway, NJ: IEEE Service Center, 1995, pp. 1942–1948.

[20] R. Storn, "System design by constraint adaptation and differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 22–34, 1999.

[21] V. Plagianakos, G. Magoulas, and M. Vrahatis, "Evolutionary training of hardware realizable multilayer perceptrons," *Neural Computing and Application*, vol. 15, pp. 33–40, 2005.

[22] N. Pavlidis, D. Tasoulis, V. Plagianakos, and M. Vrahatis, "Human designed vs. genetically programmed differential evolution operators," in *IEEE Congress on Evolutionary Computation (CEC 2006)*, Vancouver, Canada, 2006, pp. 1880–1886.

[23] H. Fan and J. Lampinen, "A trigonometric mutation operation to differential evolution," *Journal of Global Optimization*, vol. 27, pp. 105–129, 2003.

[24] R. C. Eberhart and Y. Shi, "Comparison between genetic algorithms and particle swarm optimization," in *Evolutionary Programming*, V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds. Springer, 1998, vol. VII, pp. 611–616.

[25] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, pp. 317–325, 2003.

[26] J. Alami, A. El Imrani, and A. Bouroumi, "A multipopulation cultural algorithm using fuzzy clustering," *Applied Soft Computing*, vol. 7, no. 2, pp. 506–519, 2007.

[27] K. Parsopoulos and M. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 211–224, 2004.

[28] R. Brits, A. Engelbrecht, and F. van den Bergh, "Scalability of niche pso," in *IEEE Swarm Intelligence Symposium*, 2003, pp. 228–234.

[29] A. Engelbrecht and L. van Loggerenberg, "Enhancing the nichepso," in *IEEE Congress on Evolutionary Computation*, 2007, pp. 2297–2302.

[30] J. Zhang, J.-R. Zhang, , and K. Li, "A sequential niching technique for particle swarm optimization," in *Advances in Intelligent Computing*, ser. Lecture Notes in Computer Science, 2005, vol. 3644/2005, pp. 390–399.

[31] D. Beasley, D. Bull, and R. Martin, "A Sequential Niche Technique for Multimodal Function Optimization," *Evolutionary Computation*, vol. 1, no. 2, pp. 101–125, 1993.

[32] X. Li, "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization," *Proceedings of Genetic and Evolutionary Computation Conference*, pp. 105–116, 2004.

[33] J. Li, M. Balazs, G. Parks, and P. Clarkson, "A Species Conserving Genetic Algorithm for Multimodal Function Optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 207–234, 2002.