Scaling-up multiobjective evolutionary clustering algorithms using stratification

Alvaro Garcia-Piquer[a], Jaume Bacardit[b], Albert Fornells[c,**], Elisabet Golobardes[d]

[a] *Institute of Space Sciences (IEEC - CSIC), Campus UAB, Faculty of Science, C5 Tower - 2nd Floor, 08193 Bellaterra, Spain*
[b] *School of Computing Science, Newcastle University, Claremont Tower, Newcastle, NE1 7RU, UK*
[c] *Research Group in Hospitality, Tourism and Mobilities, HTSI, Universitat Ramon Llull. Av. Marquès de Mulhacén 40-42, 08034 Barcelona, Spain*
[d] *GR-SETAD, La Salle, Universitat Ramon Llull. Av. Quatre Camins 30, 08022 Barcelona, Spain*

ABSTRACT

Multiobjective evolutionary clustering algorithms are based on the optimization of several objective functions that guide the search following a cycle based on evolutionary algorithms. Their capabilities allow them to find better solutions than with conventional clustering algorithms when more than one criterion is necessary to obtain understandable patterns from the data. However, these kind of techniques are expensive in terms of computational time and memory usage, and specific strategies are required to ensure their successful scalability when facing large-scale data sets. This work proposes the application of a data subset approach for scaling-up multiobjective clustering algorithms and it also analyzes the impact of three stratification methods. The experiments show that the use of the proposed data subset approach improves the performance of multiobjective evolutionary clustering algorithms without considerably penalizing the accuracy of the final clustering solution.

1. Introduction

Multiobjective clustering (MC) algorithms (Handl and Knowles, 2007) tackle the challenge of optimizing several criteria that cannot be combined in a single objective function by defining an objective function for each criterion and by optimizing them trying to obtain a trade-off between all the objectives. There are different strategies for multiobjective optimization such as Simulated Annealing (Saha and Bandyopadhyay, 2010) and Ant Colony Optimization (Iredi et al., 2000), but Multiobjective Evolutionary Algorithms (MOEAs) (Coello, 1999) have become one of the most capable strategies to solve this kind of problems (Zitzler et al., 2000) since they (1) work with a collection of solutions with different trade-offs among objectives, which are improved until a Pareto set with optimal trade-offs is obtained; (2) can be easily adapted to the type of data of the studied domain, due to the flexible knowledge representation used; and (3) are able to optimize different objectives without assuming any underlying structure of the objective functions. Therefore, MOEAs offer outstanding searching capabilities but their performance can be compromised in large databases due to their high computational and memory usage requirements (Freitas, 2002). This is an important issue considering the needs analyzing large data sets in a reasonable computational time and memory usage without considerably penalizing their accuracy (Kargupta et al., 2009). Evolutionary Algorithms (EAs) are based in the principles of evolution and natural selection, applying an iterative process where a collection of initial solutions (individuals) are evolved through pseudo-random recombination until obtaining an optimal solution. In the case of clustering, an individual is a possible group of data. From the whole process, the evaluation is the most time consuming step because each individual has to be assessed with respect to all the elements according to all the objective functions defined.

This work proposes to scale-up MOEAs when they are applied to large data using an approach based on data subsets techniques. Specifically, the approach splits the data set in several strata so the EA only uses a stratum for evaluating the individuals in each generation following a Round Robin policy to avoid bias problems. Thus, computational and memory costs associated to the evaluation of the population are drastically reduced and its application does not need to modify the algorithm structure. An ideal stratification strategy is to map the initial data set into disjoin strata of equal size and with equal class distribution and where the number of strata is defined by the user. How-

**Corresponding author: Tel.: +34-932522890
e-mail: albert.fornells@htsi.url.edu (Albert Fornells)

ever, clustering problems are unsupervised and classes cannot be used to split the instances into representative strata because they are unknown (Cano et al., 2008). Therefore, the definition of the size of subsets and the selection of their elements are not trivial and they influence the performance of the algorithm if the they are not sufficiently representative. For this reason, two unsupervised and one supervised strata generation strategies are presented and analyzed: (1) random strata, (2) strata according to clusters distribution using a fast and approximate clustering algorithm and (3) strata based on classes. Moreover, strategies are integrated in a MC algorithm based on MOEAs called CAOS (Clustering Algorithm based on multiObjective Strategies) (Garcia-Piquer et al., 2012, 2013) and they are tested with several strata size using artificial and real world problems. Finally, results are compared between them and with regard to MOCK, which is one of the most well-known MC algorithms based on MOEAs (Handl and Knowles, 2007).The results show the improvement in the computational time while the accuracy is not substantially penalized when statetification approaches are applied. Furthermore, the three strategies for building the strata are equivalent so the proposed data subset approach can be used in clustering problems due to the fact that it does need a stratification method based on classes. Finally, the results also show that the proposed approach is significantly better than MOCK in terms of computational time and accuracy.

The paper is organized as follows. Section 2 summarizes the related work on data subsets applied to clustering. Sections 3 and 4 describe CAOS and the stratification strategies. Section 5 describes the experimentation and discusses the results. Finally, Section 6 ends with conclusions and further work.

2. Related Work

Two of the most used strategies for scaling-up EAs are Parallel EA (Cantu-Paz, 2000) and data subsets (Cano et al., 2008; Derrac et al., 2010). The first strategy distributes the computational cost of the evaluation step by parallelizing the evaluation of individuals so it is necessary to adapt or redefine the algorithm for being able to parallelize it in a environment with several processors. Moreover, the parallelization may imply an additional communication cost that could decrease the performance achieved with the compute distribution. On the other hand, the second strategy uses a data subset from the original data set to evaluate the individuals so less resources are required and there is no need to modify the algorithm structure. In contrast, the data sets definition is not trivial.

There are two main ways to work with data subsets: using only one of the built data subsets, or using alternatively all the data subsets. The algorithm CLARA (Clustering LARge Applications) Kaufman and Rousseeuw (1990), one of the most representative algorithms for clustering large data sets, works under the first approach idea. This algorithm is based on selecting randomly a sample from the entire data set and, subsequently, it finds k medoids of the sample using only the built sample. After this, all the instances of the entire data set are assigned to the most similar medoid. The execution of the entire process is repeated five times, and the solution with less dissimilarity is returned as solution. Following this idea, other methods consist in extracting randomly several samples from the entire data set and applying the same clustering algorithm to each one of the samples obtaining several clustering results. After this, all the obtained results are merged in a single clustering solution. Hore et al. (Hore et al., 2009) proposed to use k-means or fuzzy k-means algorithms with large data. The idea is to obtain a set of jointed or disjointed samples and apply one of the two algorithms to each sample to obtain several clustering results. The last step consists in doing a consensus between each clustering result to obtain a final clustering solution as in ensemble clustering. The drawback of using only one sample to obtain the clustering results is that it is necessary to execute the algorithm several times or apply it to different data subsets in order to avoid the bias of using only one sample. Moreover, only a part of the entire data set is used. For this, the approaches based on using all the data subsets to obtain the clustering results in a single execution can be useful.

ILAS (Incremental Learning by Alternating Strata) (Bacardit, 2004) is a technique based on Evolutionary Algorithms for classification problems based on dividing the training set into several strata based on using a different stratum in each iteration of the evolutionary algorithm using a round-robin policy. Thus, the individuals are evaluated with all the strata, avoiding any bias of the data and increasing the generalization of the individual. The strategy followed in this paper is based on the ILAS algorithm but applied to MC problems.

3. CAOS

CAOS (Garcia-Piquer et al., 2013) is a multiobjective evolutionary algorithm system to solve clustering problems based on adapting the multiobjective optimization algorithm PESA-II (Corne et al., 2001) due to its competitiveness with respect to the state-of-the-art clustering methods and its ability to evolve accurate clusterings from domains with complex structures (Handl and Knowles, 2007). It evolves a set of mutually non-dominated clustering solutions (called Pareto set) that correspond to different tradeoffs between objectives. A solution S is non-dominated when there is not any solution better than S in all the objectives. Otherwise, the solution is dominated.

Algorithm 1 summarizes the main elements of PESA-II. It evolves an external population (EP) of individuals through a number of generation where individuals are selected, crossed and mutated following the typical evolutionary cycle. Individual are represented with real numbers that represent the coordinates (attributes) of the cluster prototype using a centroid-based representation (Hruschka et al., 2009). More specifically, each individual consists of $n \cdot t$ genes $\{x_{11}, ..., x_{1t}, ..., x_{n1}, ..., x_{nt}\}$, where n is the number of clusters of the individual, t is the number of the attributes of the data set, and x_{ij} is the value of the attribute j of the cluster centroid i. The genotypic representation is transformed into the phenotypic representation by assigning each instance to the cluster with the nearest centroid to it. In addition to EP, it also maintains an *internal population* (IP) to separate the exploration from the storage of the best solutions. That is, IP is used to explore new promising solutions and EP is employed to store a large and diverse set of non

```
1  Let $EP$ and $IP$ be an external and an internal population respectively.
   They store a maximum of $N_{EP}$ and $N_{IP}$ individuals, where ($N_{IP} < N_{EP}$)
2  Init. $IP$ with $N_{IP}$ individuals stochastically created
3  Init. the $EP$ individuals with non-dominated clustering results from $IP$
4  Evaluate all the individuals from $EP$ according to the objectives
5  foreach Generation do
6      Select $N_{IP}$ individuals from $EP$ to generate a new $IP$
7      while ($|IP|$ != $\emptyset$) do
8          Select and remove two individuals from $IP$
9          Cross and mutate them to obtain 2 new ind.: $I_{New_1}$ and $I_{New_2}$
10         foreach $I_{New_i}$ do
11             Evaluate the $I_{New_i}$ fitness according to the objectives
12             if $I_{New_i}$ dominates any individual from $EP$ then
13                 Remove the dominated individuals by $I_{New_i}$ from $EP$
14                 Add $I_{New_i}$ into $EP$;
15             else if $I_{New_i}$ is not-dominated and $I_{New_i}$ not-dominates any
               individual then
16                 if $EP$ is full then
17                     Remove an ind. from the most crowded niche
18                 Add $I_{New_i}$ into $EP$
19 Select a individual from $EP$ as a solution
```

Algorithm 1: Scheme of PESA-II algorithm.

dominated solutions found so far. Moreover, *EP* is organized in N_{niches} different niches through the placement of an hyper-grid in the objective space splitting it in hyper-rectangles, where each of them is considered as a separate niche. Therefore, solutions with similar objectives will be placed in the same niche. The replacement process uses the niching mechanism to make pressure toward balancing the allocation of solutions in different niches thus encouraging solutions to cover all the objective space. Concretely, the system creates IP with N_{IP} individuals stochastically initialized. All the non-dominated solutions of IP are used to build EP.

The final step is to select a solution from the Pareto set (composed by all non-dominated solutions) when the evolutionary process ends. This point is not trivial because there is not any single individual which is the best in all the objectives and for this reason clustering validation techniques (Halkidi et al., 2001) are required for selecting the best one. CAOS integrates some supervised and unsupervised techniques to score a solution. The supervised approach follows the idea that similar elements from the same class should be in the same cluster. In contrast, the unsupervised approaches retrieve the best solution from the Pareto set according to the quality of the clusters based on the compacting and separation between them, such as Davies-Bouldin index, Dunn index or Silhouette index, among others. Experimentation of Section 5 uses a supervised validation technique based on classes called Adjusted Rand Index (Yeung and Ruzzo, 2001) for being able to compare the performance of several strategies between them. It is important to highlight that class information is not used in the building process of the clusters, it is only used at the end of the process for comparing the obtained clustering solution between them. The Adjusted Rand Index compares a clustering solution with respect to an ideal partition of the data set by counting the number of pairwise co-assignments of instances between them and introducing a statistically induced normalization in order to yield values close to 0 for random partitions. In contrast, a value of 1 means that all the clusters correspond to the structure defined in the ideal partition. In our case, the ideal partition corresponds to the classes of the instances in the original data set.

Finally, CAOS allows the customization of the objectives, genetic operators among other features. *Deviation* and *Connectivity* are used as objective functions (Handl and Knowles, 2007) because they indicate how nearby are the elements of each cluster (intra-cluster variance) and how separated are the clusters between them (inter-cluster variance), respectively. Both objectives have to be minimized because the desired solution should contain compact clusters with examples that are close in the feature space. A one-point crossover operator (Goldberg, 2002) is used to generate two offspring from pairs of parents. One point is selected for each parent and the parts are interchanged between them, taking into account that they have to cut the individuals at the same attribute but not necessarily at the same cluster. This is an easy crossover strategy according to the size of each individual can be different. A cluster-oriented mutation operator (Hruschka et al., 2009) is used to promote the right search. It defines three different types of mutations and all of them have the same probability to be applied: to merge two clusters, to split a cluster, and to move the centroid of a cluster.

4. Data Subset Strategies

Reducing the amount of data used by an algorithm is a smart approach to reduce the computational cost of evolutionary-based machine learning techniques and it could also improve the accuracy of the system Bacardit (2004). In this sense, we want to scale-up a MC algorithm based on EAs by dividing a data set in several stratified subsets and using them alternatively during the algorithm process in order to avoid bias. Next points detail the analyzed approaches to build the strata and how to use it in MC. Finally, the impact of using these strategies in terms of computational cost and memory usage is described.

4.1. Creation of Strata

Data subset strategies map the initial data set into disjoin data subsets (strata) of equal size and with equal class distribution (Bacardit, 2004; Cano et al., 2006), where the number of strata is selected by the user (see Algorithm 2). Classes in clustering problems are usually unknown, so the stratification based on classes is not always available. To avoid this lack, two unsupervised approaches to divide the data set are proposed:

- **Random Strata.** It randomly assigns the instances to each one of the strata as Algorithm 3 shows.

- **Strata based on Clusters.** It uses a fast and approximative clustering technique to create a partition of the original data set. Next, the data set is stratified according to the obtained clusters, that is, it assigns the instances to each stratum respecting in it the same cluster distribution of the instances than in the clustered original data set. The process is described in Algorithm 4. The clusters are found with the Subtractive Clustering algorithm Chiu (1994) applied to the original data set, which is an efficient and non-iterative method for estimating cluster centers. It is usually used to determine the number of clusters and their initial values for initializing iterative optimization-based clustering algorithms. The limitation of this strategy is their computational cost $O(m^2)$, where m is the number of instances of the original data set, because with very large data sets can be expensive in computational terms. Nevertheless, in terms of spatial cost it only needs the data set information and a list with the prototypes of each cluster.

Finally, these approaches require the definition of the number of strata which will influence the performance. As the number of strata increases the computational time decreases but pattern extraction becomes more complex due to the lack of information. It is important to highlight that the idea of these strategies is to obtain data with similar distribution in each stratum, and this only can be possible if the size of each one is not very small.

4.2. Evolution Based on Strata

The idea is to use a different stratum in each iteration of the evolutionary algorithm in order to avoid the bias produced when only one stratum is used. Thus, the final individuals can generalize more than using only one of the strata and there is no need to modify the main process. More precisely, the generation of strata is done before line 1 of the Algorithm 1, and the change of stratum is done between lines 6 and 7. It is important to highlight that these strategies can be applied to CAOS due to the fact that individuals are represented by the prototypes of the clusters and then the individuals are independent of the instances. Thus, the algorithm can work with different instances of the data set in each iteration.

```
1  Let $numStrata$ be the number of strata to generate
2  Let $Strata$ be a vector of size $numStrata$ where each position is initially
   an empty list of instances
3  Let $I$ be a vector of size $numClasses$ where each position stores a list of
   the instances of the same class
4  $stratum = 0$
5  $class = 0$
6  while ($class < numClasses$) do
7      while ($|I[class]| \; != \emptyset$) do
8          Select randomly an instance $i$ from $I[class]$
9          Add $i$ to $Strata[stratum]$
10         Erase $i$ from $I[class]$
11         $stratum = (stratum + 1) \bmod numStrata$
12     $class = class + 1$
13 return $Strata$
```

Algorithm 2: Strata generation based on classes.

```
1  Let $numStrata$ be the number of strata to generate
2  Let $Strata$ be a vector of size $numStrata$ where each position is initially
   an empty list of instances
3  Let $I$ be a list of all the instances of the data set
4  $stratum = 0$
5  while ($|I| \; != \emptyset$) do
6      Select randomly an instance $i$ from $I$
7      Add $i$ to $Strata[stratum]$
8      Erase $i$ from $I$
9      $stratum = (stratum + 1) \bmod numStrata$
10 return $Strata$
```

Algorithm 3: Strata generation based on random instances selection.

4.3. Computational Performance Models

The aim of the data subsets strategies is to reduce the computational time and memory usage without considerably penalizing the accuracy. Next, this subsection analyzes the improvement in the performance of a MC algorithm based on EAs using CAOS from a theoretical perspective and section 5 will analyze the improvement using different real and artificial data sets.

The CAOS process can be divided in two main blocks: the initialization process and the clustering process. The initialization process focuses on precalculating the distances between the instances and the nearest neighbors to speed-up the clustering process avoiding the repetition of calculations. In contrast, the clustering process is referred to the evolutionary cycle that obtains the Pareto set of solutions. CAOS algorithm has the same initialization and clustering cost independently of the data subset strategy used (based on classes, random or based on clusters) but it depends on the number of strata used. However, the time of both processes is extremely lower in comparison with the time spent when the complete data set is used as Table 1 describes. It should be emphasized that both times are reduced when the size of the stratum is decreased, that is, when the number of strata increases. Nevertheless, the use of strata requires an additional cost for building them. According to this, the strategies based on random instances selection and based on classes need only one data scan to build the strata and their cost is $O(m)$, where m is the number of instances of the complete data set. In contrast, the strategy based on approximative clusters has a higher cost due to the cost related to the subtractive clustering technique ($O(m^2)$).

As it has been explained above, CAOS precalculate the distances between all the instances of the data set and the nearest neighbors of each instance to speed-up the clustering process. Thus, the memory usage would be extremely high if the complete data set is analyzed when a large data set is used. Applying any of the three strategies of data subset construction, the memory usage is considerably reduced as Table 2 shows. Even the computational time and the memory usage of the MC algorithm is considerably reduced, the accuracy of the method can be penalized due to the fact that less information is used to obtain the clustering solutions.

```
1  Obtaining the instances clustered in $numClusters$ clusters by applying the
   Subtractive Clustering algorithm to the complete data set
2  Let $numStrata$ be the number of strata to generate
3  Let $Strata$ be a vector of size $numStrata$ where each position is initially
   an empty list of instances
4  Let $I$ be a vector of size $numClusters$ where each position stores a list of
   the instances assigned to the same cluster
5  $stratum = 0$
6  $cluster = 0$
7  while ($cluster < numClusters$) do
8      while ($|I[cluster]| \; != \emptyset$) do
9          Select randomly an instance $i$ from $I[cluster]$
10         Add $i$ to $Strata[stratum]$
11         Erase $i$ from $I[cluster]$
12         $stratum = (stratum + 1) \bmod numStrata$
13     $cluster = cluster + 1$
14 return $Strata$
```

Algorithm 4: Strata generation based on approximative clusters.

Table 1. Computational cost of CAOS applied to the complete data set and to data subsets (CAOS – CD and CAOS – DS respectively) broken down in initialization cost and clustering cost. Where g is the number of generations, $|IP|$ is the internal population size, m and t are the number of instances and attributes of the data set respectively, $\bar{n}_{cd}$ is the average of the number of clusters of the individuals (the minimum number of clusters is 1 and the maximum m), $\bar{n}_{ds}$ is the average of the number of clusters of the individuals (the minimum number of clusters is 1 and the maximum $\frac{p}{numStrata}$), ℓ is the percentage of the nearest elements taken into account, and $numStrata$ is the number of strata generated.

Algorithm	Initialization cost	Clustering cost
CAOS – CD	$O(m^3 \cdot \ell)$	$O(g \cdot \lvert IP\rvert \cdot m \cdot \bar{n}_{cd} \cdot t)$
CAOS – DS	$O(numStrata \cdot (\frac{m}{numStrata})^3 \cdot \ell)$	$O(g \cdot \lvert IP\rvert \cdot \frac{m}{numStrata} \cdot \bar{n}_{ds} \cdot t)$

Table 2. Memory usage of CAOS applied to the complete data set and to data subsets (CAOS – CD and CAOS – DS respectively) to store the nearest neighbors. Where m is the number of instances of the data set, ℓ is the percentage of instances considered neighbors, $numStrata$ is the number of strata generated and $sizeof(data\ type)$ is the size in bytes of the data type.

Algorithm	Storage of distances	Storage of nearest neighbors
CAOS – CD	$m^2 \cdot sizeof(float)$	$(\ell \cdot m)^2 \cdot sizeof(integer)$
CAOS – DS	$(\frac{m}{numStrata})^2 \cdot sizeof(float)$	$numStrata \cdot (\frac{\ell \cdot m}{numStrata})^2 \cdot sizeof(integer)$

5. Experiments, Results, and Discussion

This section evaluates the performance improvement in CAOS using the proposed data subset approach with the three stratification strategies described in section 4. The performance is considered in terms of accuracy using the Adjusted Rand Index value of the solution returned by CAOS and the computational time required to find it.

The organization of this section is as follows. First, the collection of 105 artificial and real-world data sets and the experimental methodology are described. Next, CAOS results and a comparison between the proposed data subset approach in CAOS and MOCK (Handl and Knowles, 2007) is presented in order to emphasize the performance improvement of our approach. Finally, the performance of the most suitable strategy is analyzed in data sets of medium and large size.

5.1. Test Bed

The experimentation uses different typologies of artificial and real-world problems. Concretely, 75 artificial data sets were created according to different number of instances (from 800 to 24000), attributes (from 2 to 100) and classes (from 2 to 30). They were built adapting the tool used in (Handl and Knowles, 2007) where three parameters are used to create the data sets: the number of attributes, the number of classes related to the number of instances, and the separation between the classes. Each class has a data distribution for each attribute, which can only have numerical values. The distribution can be a normal or uniform distribution, and it is randomly selected to model each attribute. Also, the separation between classes were modeled, obtaining 25 data sets with well-separated classes, other 25 data sets with nearer classes, and the last 25 with overlapped classes .On the other hand, other 30 real-world problems were selected according to different number of instances (from 150 to 58000), attributes (from 2 to 60) and classes (from 2 to 26) from the UCI repository (Frank and Asuncion, 2010). The characteristics of each data set are detailed in Table 5 and Table 6 in supplementary material. It must be emphasized that the class assigned to each instance of the data sets is known in order to apply the stratification strategy based on classes and to evaluate the accuracy of the clustering results.

5.2. Experimental Methodology

The performance of the three approaches based on data subsets (CAOS$_{DS}$) were compared with respect to the approach that uses the complete data set (CAOS$_{CD}$) in terms of accuracy and computational time. The accuracy is compared using the Adjusted Rand Index, which is based on the initial classes of the data set, where 1 is the best accuracy (all the clusters correspond to the original classes) and 0 the worst. The computational time represents the sum of the precalculation time and clustering time. The first one includes the time needed to build the data subsets and to precalculate the distance and nearest neighbors structures necessaries to the clustering process. The second one is referred to the time needed to do the evolutionary process that obtains the Pareto set of solutions. Finally, each CAOS$_{DS}$ strategy is executed dividing the original data set in 2, 3, 4, 10, 15, 20 and 25 data subsets which means the 50%, 34%, 25%, 20%, 10%, 7% and 4% of the instances of the original data sets are considered in each data subset respectively.

Each CAOS configuration was run with 20 different seeds and with the following parameters (see Section 3 for notation details): ℓ is 5% of the number of instances used, the maximum size of the initial population is 100, N_{EP} is 1000, N_{IP} is 50, N_{niches} is 5, the number of generations is 400, P_c is 0.7 and P_μ is $1/m$. The minimum and maximum number of clusters for the initial individuals is 2% and 20% of m respectively.

On the other hand, the recommendations pointed out by Demšar (Demšar, 2006) were followed to perform the statistical analysis of the accuracy of the algorithms, which was based on the use of nonparametric tests. More specifically, we followed the process given in (García and Herrera, 2008) to compare them using the software freely provided by the authors. First, the Friedman's test (Friedman, 1940) with $\alpha = 0.05$ was applied to contrast the null hypothesis that all the learning algorithms obtained the same results on average. Then, if the Friedman's test rejected the null hypothesis, pair-wise comparisons were performed by means of the Holm's step-down procedure (Holm, 1979). Following this procedure, we could distinguish pairs of learners that performed significantly differently.

5.3. Discussion of CAOS Results

The accuracy of CAOS$_{CD}$ and the three CAOS$_{DS}$ strategies were empirically tested with the presented 105 data sets using the Holm's test (see Table 7 in the supplementary material). Analyzing the results it can be observed that independently of the size of the strata, all the CAOS$_{DS}$ strategies are not significantly different between them. However, all of them are significantly different to CAOS$_{CD}$ due to the fact that they are using less

Table 3. Comparison of the accuracy of the algorithms in the (a) artificial and (b) real data sets using the post-hoc Holm's procedure with $\alpha = 0.05$. The algorithms compared are CAOS using the complete data set CAOS_{CD} and the three CAOS_{DS} strategies to generate data subsets: based on classes (CAOS_{DS} Classes), random (CAOS_{DS} Random) and based on clusters (CAOS_{DS} Clusters). The results are showed for 34%, 10% and 4% of information used from the complete data set. The symbols $\oplus$ and $\ominus$ show that the method in the row obtained results that were significantly higher/lower than those obtained with the method in the column. Similarly, the symbols + and − denote a non-significant higher/lower results. The Iman and Davenport statistic is calculated according to F-distribution with 3 and 222 degrees of freedom.

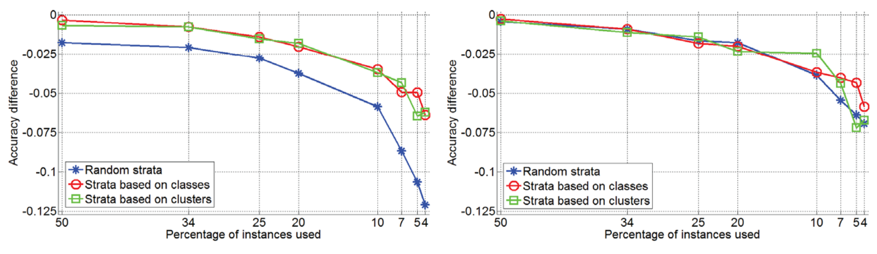
(a)

% Instances	Strategies	CAOS_{CD}	CAOS_{DS} Classes	CAOS_{DS} Random	CAOS_{DS} Cluster	Avg. Rank	p-value
34%	CAOS_{CD}		$\oplus$	$\oplus$	$\oplus$	1.871	$1.705E-5$
	CAOS_{DS} Classes	$\ominus$		+	+	2.662	
	CAOS_{DS} Random	$\ominus$	−		+	2.689	
	CAOS_{DS} Clusters	$\ominus$	−	−		2.777	
10%	CAOS_{CD}		$\oplus$	$\oplus$	$\oplus$	1.730	$2.634E-8$
	CAOS_{DS} Classes	$\ominus$		+	+	2.716	
	CAOS_{DS} Random	$\ominus$	−		−	2.817	
	CAOS_{DS} Clusters	$\ominus$	−	+		2.736	
4%	CAOS_{CD}		$\oplus$	$\oplus$	$\oplus$	1.676	$4.689E-10$
	CAOS_{DS} Classes	$\ominus$		+	+	2.662	
	CAOS_{DS} Random	$\ominus$	−		−	2.946	
	CAOS_{DS} Clusters	$\ominus$	−	+		2.716	

(b)

% Instances	Strategies	CAOS_{CD}	CAOS_{DS} Classes	CAOS_{DS} Random	CAOS_{DS} Cluster	Avg. Rank	p-value
34%	CAOS_{CD}		$\oplus$	+	+	1.816	0.005
	CAOS_{DS} Classes	$\ominus$		−	−	2.917	
	CAOS_{DS} Random	−	+		+	2.650	
	CAOS_{DS} Clusters	−	−	+		2.617	
10%	CAOS_{CD}		$\oplus$	$\oplus$	$\oplus$	1.533	$4.853E-6$
	CAOS_{DS} Classes	$\ominus$		−	−	2.966	
	CAOS_{DS} Random	$\ominus$	+		−	2.933	
	CAOS_{DS} Clusters	$\ominus$	+	+		2.567	
4%	CAOS_{CD}		$\oplus$	$\oplus$	$\oplus$	1.400	$8.677E-8$
	CAOS_{DS} Classes	$\ominus$		+	+	2.667	
	CAOS_{DS} Random	$\ominus$	−		−	3.000	
	CAOS_{DS} Clusters	$\ominus$	−	+		2.933	

information in the clustering process. For this reason, we repeat the same Holm's test but separating data sets in artificial and real-world datasets in order to analyze in more detail the performance of the different strategies as Table 3(a) and 3(b) illustrates using 34%, 10% and 4% of instances for the artificial and real data sets respectively (see Tables 8(a) and 8(b) from the supplementary material for the complete experimentation). The results with respect to the artificial data sets shows that the three CAOS_{DS} strategies are significantly worse in terms of accuracy regarding CAOS_{CD} independently of the number of instances considered. Nevertheless, the three CAOS_{DS} are not significantly different in terms of accuracy between them.
From a quantitative point of view, Figure 1(a) shows the average of accuracy difference among the three CAOS_{DS} strategies and CAOS_{CD}. Globally, the strategies based on classes and on clusters follow a similar pattern, and the accuracy is not considerably decreased until less than the 20% of the instances are used. In contrast, the random selection approach seems to underperform the other two due to the fact that the classes structure is complex in some data sets and the random strategy is not able to build representative strata. Figure 2(c) shows the average speedup for each data set of the three CAOS_{DS} strategies. The speedup of each CAOS_{DS} strategy applied to a specific data set corresponds to the time needed to apply CAOS by CAOS_{CD} divided by the time needed by the corresponding CAOS_{DS} strategy. Thus, a high speedup is desired (e.g., a CAOS_{DS} strategy with speedup of 3 indicates that it is three times faster than the CAOS_{CD} approach). This speedup is divided in two partial



(a) (b)

Fig. 1. Accuracy difference of the three CAOS_{DS} strategies regarding CAOS_{CD}. (a) Artificial data sets and (b) real-world data sets.

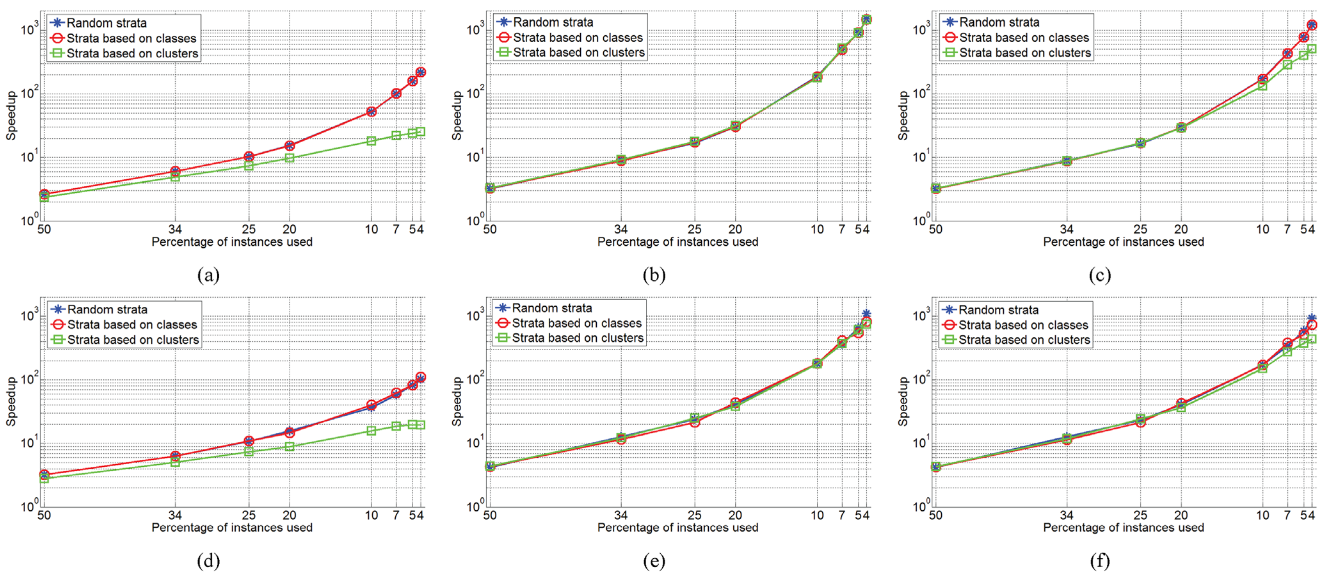


Fig. 2. Speedup of the three CAOS$_{DS}$ strategies regarding CAOS$_{CD}$ in artificial (a,b,c) and real-world data sets (d,e,f). Figures (a,d) are referred to the speedup of the precalculation time. Figures (b,e) show the speedup of the clustering time to do the evolutionary process that obtains the Pareto set of solutions. Figures (c,f) are related to the speedup of the overall time taking into account both times.

speedups (1) regarding the precalculation time needed to build the data subsets and to precalculate the distance and nearest neighbors structures (see Figure 2(a)), and (2) regarding to the clustering time (see Figure 2(b)). The precalculation speedup is the same in the random strata than in the strata based on classes strategies. However, the strategy based on clusters has a lower precalculation speedup because it needs to roughly cluster the instances before building the data subsets. On the other hand, the speedup of the clustering time is the same in the three strategies since the strata method does not affect to the clustering process. Analyzing the overall speedup according to the percentage of instances used from the complete data set, it can be observed that using a 50% of the instances, that is the lowest improvement, the speedup is 3, so CAOS$_{DS}$ strategies are three times faster than CAOS$_{CD}$. Moreover, using a 4% of the instances of the complete data set the speedup is 500 for the strategy based on clusters and 1200 for the other two strategies. Also it can be observed that there are not speedup differences between the strategies until more than a 10% of the instances are used. These analysis showed that CAOS$_{DS}$ strategies applied to the proposed artificial data sets are faster than CAOS$_{CD}$ and do not considerably penalize the accuracy in quantitative terms. Nevertheless, CAOS$_{DS}$ is worst and significantly different, in statistical terms, regarding CAOS$_{CD}$.

With respect to the results of the Holm's test applied to the real-world data sets, the Friedman's test cannot reject the null hypothesis that all the strategies obtain the same results on average when in each data subset is considered a 50% of the instances (2 data subsets). Thus, the three CAOS$_{DS}$ strategies cannot be considered different than CAOS$_{CD}$ in terms of accuracy. When the 34% and the 25% of the instances are considered, the CAOS$_{DS}$ strategy based on random strata and on clusters is not significantly different in terms of accuracy regarding CAOS$_{CD}$. Nevertheless, the accuracy of the other strategy is worst and significantly different than CAOS$_{CD}$. When it is used less than the 25% of the instances of the complete data set in each data subset, the three CAOS$_{DS}$ strategies are significantly different in terms of accuracy regarding CAOS$_{CD}$, but they are not significantly different between them. Figure 1(b) shows that there is virtually no accuracy difference between CAOS$_{DS}$ strategies regarding CAOS$_{CD}$, because the data sets used do not have shapes as complex than the artificial ones used. Figure 2(f) shows that, in terms of speedup, the behavior in the used real-world data sets is similar than in artificial data sets. The maximum speedup is lower because some of the real-world data sets are small and the speedup of using less than a 10% of the instances is not as high than in larger data sets. Also, it can be observed than in the used real-world data sets if it is used a 50% of the instances, it is four times faster than CAOS$_{CD}$ and it obtains the same clustering results (see Table 11 and Table 12 in the sup. material with the results of the execution time). It must be emphasized that the speedup obtained applying this kind of techniques is very high and, consequently, the computational performance of the system is considerably improved. Moreover, assuming that the best strata generation is based on the original classes, the results show that the other two strategies to build the strata are not significantly different in terms of accuracy independently of the kind of data sets tested. In terms of accuracy, the random and the cluster based strategies are as useful as the strategy based on classes but without the requirement of having the original class of each instance. In terms of computational time, the random and classes based strategies have similar speedup regarding CAOS$_{CD}$. Nevertheless, the cluster based strategy has a lower speedup. According to these observations, it seems that the most suitable strategy to build the data subsets in CAOS$_{DS}$ is the random one, because it does not require the original classes of the instances, it is not significantly different in terms of accuracy than the other two strategies and it has a high speedup.

5.4. Discussion of CAOS and MOCK Results

The proposed approach has demonstrated that it is able to scale-up CAOS without considerably penalizing the accuracy and it has been compared with respect to MOCK (Handl and Knowles, 2007) in order to emphasize the results. MOCK is also based on the PESA-II algorithm but it uses the locus-based adjacency representation proposed in (Park and Song, 1998) for representing the individuals. That is, each individual encodes a reflexive directed unlabeled graph that connects pairs of examples using an integer encoding scheme. More specifically, each individual consists of m genes $\{x_1, x_2, ..., x_m\}$, where m is the number of examples of the training data set and x_i ranges in [1, m]. Thence, each gene x_i indicates that there exists an arrow connecting instance i with instance x_i. It can be observed that this representation cannot properly scale-up the memory usage when the algorithm is applied to large data because the individual size depends on the number of instances. Thus, this representation does not allow the use of data subsets due to the fact that all the instances are needed for building the individuals.

The accuracy of the best results of MOCK, CAOS$_{CD}$ and the three CAOS$_{DS}$ strategies of the real-world data sets is analyzed using the Holm's test and illustrated in Table 4(a) For carrying out this analysis, four data sets (*letter-recognition*, *magic*, *pendigits* and *shuttle*) where MOCK was not able to obtain the results with a reasonable memory usage are not considered. Analyzing the results it can be observed that MOCK is significantly worse than CAOS$_{CD}$ and than all the CAOS$_{DS}$ strategies when a 20% of information or more is used. When less of a 20% of instances are used, MOCK is worse but not significantly different than CAOS$_{DS}$ strategies. Finally, MOCK has a lower computational time than CAOS$_{CD}$ for the different data sets as Table 11 and Table 12 show in the suplementary material. However, CAOS$_{DS}$ strategies are significantly better than MOCK in terms of computational time independently of the size of the strata as Table 4(b) shows. Thus, the proposed data subset approach can obtain better results than MOCK in a lower computational time, and it can be affirmed that CAOS$_{DS}$ is a promising scalable MC algorithm based on MOEAs.

Table 4. Comparison of (a) the accuracy and (b) the execution time of the algorithms in the real-world data sets (the four data sets where MOCK cannot obtain results are excluded from the analysis, so 26 data sets are used) using the post-hoc Holm's procedure with $\alpha = 0.05$. The algorithms compared are MOCK and CAOS using the CAOS$_{CD}$ and the three CAOS$_{DS}$ strategies to generate data subsets: based on classes (CAOS$_{DS}$) Classes, random (CAOS$_{DS}$ Random) and based on clusters (CAOS$_{DS}$ Clusters). The results are showed for 34%, 10% and 4% of information used from the complete data set. The symbols $\oplus$ and $\ominus$ show that the method in the row obtained results that were significantly higher/lower than those obtained with the method in the column. Similarly, the symbols $+$ and $-$ denote a non-significant higher/lower results. Symbol $=$ indicates that the algorithms have the same rank. The Iman and Davenport statistic is calculated according to F-distribution with 4 and 100 degrees of freedom. The results with the rest of % of instances are detailed in Table 9 and 10 in supplementary material.

(a)

Instances	Strategies	MOCK	CAOS$_{CD}$	CAOS$_{DS}$ Classes	CAOS$_{DS}$ Random	CAOS$_{DS}$ Clusters	Average Rank	p-value
	MOCK		$\ominus$	$\ominus$	$\ominus$	$\ominus$	4.346	
	CAOS$_{CD}$	$\oplus$		$+$	$+$	$+$	1.962	
34%	CAOS$_{DS}$ Classes	$\oplus$	$-$		$-$	$-$	3.077	$3.449E-7$
	CAOS$_{DS}$ Random	$\oplus$	$-$	$+$		$+$	2.769	
	CAOS$_{DS}$ Clusters	$\oplus$	$-$	$+$	$-$		2.846	
	MOCK		$\ominus$	$-$	$-$	$\ominus$	4.269	
	CAOS$_{CD}$	$\oplus$		$\oplus$	$\oplus$	$\oplus$	1.577	
10%	CAOS$_{DS}$ Classes	$+$	$\ominus$		$=$	$-$	3.192	$1.108E-9$
	CAOS$_{DS}$ Random	$+$	$\ominus$	$=$		$-$	3.192	
	CAOS$_{DS}$ Clusters	$\oplus$	$\ominus$	$+$	$+$		2.769	
	MOCK		$\ominus$	$-$	$-$	$-$	4.000	
	CAOS$_{CD}$	$\oplus$		$\oplus$	$\oplus$	$\oplus$	1.538	
4%	CAOS$_{DS}$ Classes	$+$	$\ominus$		$+$	$+$	3.038	$4.580E-8$
	CAOS$_{DS}$ Random	$+$	$\ominus$	$-$		$-$	3.346	
	CAOS$_{DS}$ Clusters	$+$	$\ominus$	$-$	$+$		3.077	

(b)

Instances	Strategies	MOCK	CAOS$_{CD}$	CAOS$_{DS}$ Classes	CAOS$_{DS}$ Random	CAOS$_{DS}$ Clusters	Average Rank	p-value
	MOCK		$+$	$\ominus$	$\ominus$	$\ominus$	1.731	
	CAOS$_{CD}$	$-$		$\ominus$	$\ominus$	$\ominus$	1.308	
34%	CAOS$_{DS}$ Classes	$\oplus$	$\oplus$		$+$	$+$	4.231	0
	CAOS$_{DS}$ Random	$\oplus$	$\oplus$	$-$		$+$	3.961	
	CAOS$_{DS}$ Clusters	$\oplus$	$\oplus$	$-$	$-$		3.769	
	MOCK		$+$	$\ominus$	$\ominus$	$\ominus$	1.692	
	CAOS$_{CD}$	$-$		$\ominus$	$\ominus$	$\ominus$	1.308	
10%	CAOS$_{DS}$ Classes	$\oplus$	$\oplus$		$+$	$+$	4.154	0
	CAOS$_{DS}$ Random	$\oplus$	$\oplus$	$-$		$+$	4.135	
	CAOS$_{DS}$ Clusters	$\oplus$	$\oplus$	$-$	$-$		3.712	
	MOCK		$+$	$\ominus$	$\ominus$	$\ominus$	1.692	
	CAOS$_{CD}$	$-$		$\ominus$	$\ominus$	$\ominus$	1.308	
4%	CAOS$_{DS}$ Classes	$\oplus$	$\oplus$		$+$	$+$	4.442	0
	CAOS$_{DS}$ Random	$\oplus$	$\oplus$	$-$		$+$	4.192	
	CAOS$_{DS}$ Clusters	$\oplus$	$\oplus$	$-$	$-$		3.365	

5.5. CAOS Results in Large and Medium Data Sets

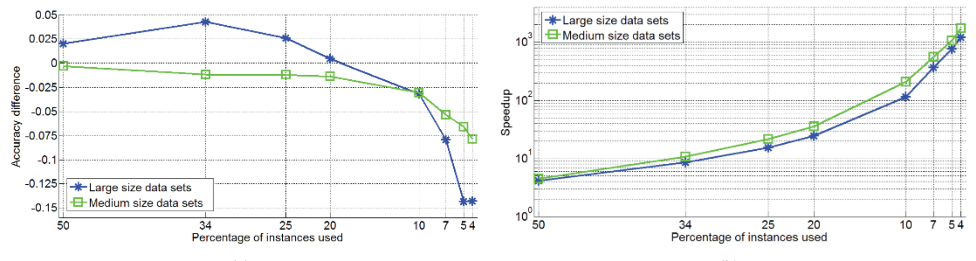
The analysis of the performance of the three CAOS$_{DS}$ strategies in artificial and real-world data sets concluded that the random strata strategy is the most suitable one. In order to analyze the potential of CAOS$_{DS}$ with this strategy applied to data sets with a considerable number of instances, it was tested with 33 data sets considered of medium size and with 33 data sets considered of large size. The medium size data sets have a number of instances from 2000 to 8000, and the large size data sets have a number of instances from 13000 to 50000.

The experiments show that the speedup of both type of data sets is very high (see Figure 3(b)), and this represents an important improvement (e.g., in the shuttle data set, which has 50000 instances, CAOS$_{CD}$ needs near 40 hours to obtain a clustering result, and CAOS$_{DS}$ with a random strata strategy needs 10 hours using the 50% of instances and 3 hours using the 25% of them, as Table 11 in supplementary material shows). It is obvious that if less data is considered, the accuracy results will be worse, as statistical tests show in Subsection 5.3. However, the loss of accuracy is not very considerable when more than the 20% of instances are used (see Figure 3(a)). Thus, the results obtained when the 25% of the instances are used in each cycle of the EA are analyzed to better illustrate this issue because it is considered (according to Figure 3) that it is a configuration with a good trade-off between accuracy and execution time.

Figure 4 and Figure 5 show the accuracy result and the overall execution time of CAOS$_{CD}$ and the random strata strategy of CAOS$_{DS}$ for each one of the large and medium-sized data sets. Both figures show the average results of the 20 runs done. It can be observed that the accuracy results are preserved or slightly penalized but, on the other hand, the execution time is strongly improved. Moreover, it is important to emphasize that the main idea of CAOS$_{DS}$ is to considerably improve the performance of the algorithm with small impact on the accuracy, and this can be clearly observed in data sets of medium and large size.

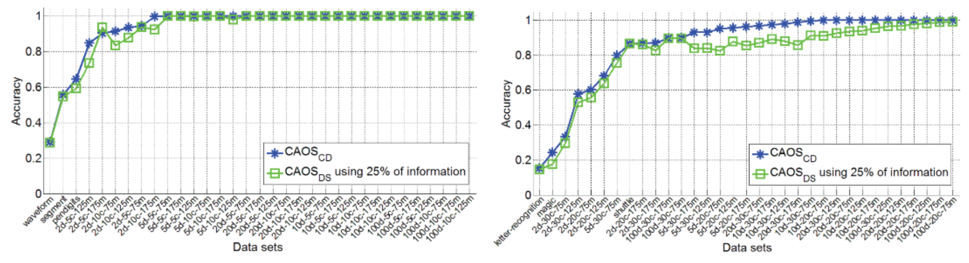
6. Conclusions and Further Work

MC based on EA is a data mining technique focused on identifying data relationships according to multiple criteria to properly understand huge and complex databases. Although its searching capabilities outline from the rest of similar techniques, its main lack is the high cost in terms of computational time and memory usage when it is applied to a large data sets.



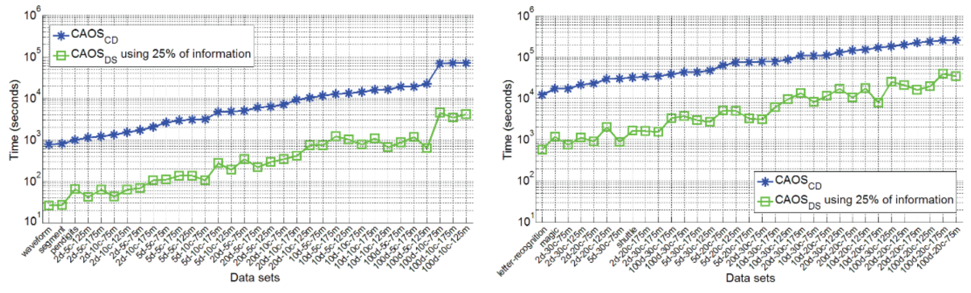
(a) (b)

Fig. 3. Accuracy difference (a) and overall speedup (b) of the CAOS$_{DS}$ random strata strategy in large and medium size data sets.



(a) (b)

Fig. 4. Average accuracy of 20 runs of CAOS$_{CD}$ and CAOS$_{DS}$ using 25% of information in each generation for (a) medium and (b) large data sets.



(a) (b)

Fig. 5. Average of the overall clustering time of 20 runs of CAOS$_{CD}$ and CAOS$_{DS}$ using the 25% of information in each generation for (a) medium and (b) large size data sets.

This work has presented an approach focused on reducing the impact of the volume of data in the EA by means of the stratification of the complete data set into disjoint strata and alternate them following a Round Robin policy in each cycle of the genetic algorithm. More specifically, a supervised and two unsupervised stratification strategies are proposed and their performance is analyzed using 106 real and artificial data sets in the CAOS algorithm: (1) according to the original classes of the data set, (2) selecting random instances from the data set, and (3) according to the clusters found applying a fast method called subtractive clustering.

The experimentation showed that the speedup of the three strategies is very high, and this considerably improves the computational performance of the system. Moreover, it can be observed that the two unsupervised strategies to build the strata are not significantly different from the strategy based on classes in terms of accuracy so they can be considered equivalent to the strategy based on classes in terms of accuracy. Furthermore, the strategy based on random strata has a higher speedup than the cluster based strategy, due to the fact that the last one needs to build approximative clusters at the begin of the algorithm and this has a high cost with very large data sets. Thus, the random stratum strategy is the most suitable to scaling-up CAOS. On the other hand, there are statistically significant differences among these three strategies and the approach that use the complete data set but the accuracy differences among them are relatively small and they considerably reduce the computational time of the algorithm scaling-up it properly. It is important to highlight that the size of each stratum affect the performance of the CAOS algorithm. The computational time of CAOS will be decreased as much smaller is the size of the strata, but as much smaller is the size it will be difficult to obtain consistent stratum according to the original data set, affecting to the accuracy of the system. The last part of the experimentation has also compared the results of the proposed strategies with respect to MOCK and they have obtained results significantly better in terms of computational time and accuracy. Thus, it can be concluded that the proposed approach is a promising scalable MC algorithm based on MOEAs and it can be properly applied to large data. Moreover, this approach can also be applied in situations where losing some accuracy can be accepted if it is possible to obtain results in a reasonable time.

Finally, this analysis has set the basis for further conducting research on multiobjective evolutionary clustering applied to large data sets in two research topics. First, the consequences of applying stratification methods for scaling-up CAOS with other individual representations. Second, the application of other data mining techniques for large data sets (Bacardit and Llorà, 2009) such as Parallel EAs.

References

Bacardit, J., 2004. Pittsburgh Genetic-Based Machine Learning in the Data Mining era: Representations, generalization, and run-time. Ph.D. thesis. Enginyeria i Arquitectura La Salle, Universitat Ramon Llull.

Bacardit, J., Llorà, X., 2009. Large scale data mining using genetics-based machine learning, in: Proceedings of the 11th Annual Conference Companion on GECCO: Late Breaking Papers, ACM. pp. 3381–3412.

Cano, J.R., García, S., Herrera, F., 2008. Subgroup discovery in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes. Pattern Recognition Letters 29, 2156–2164.

Cano, J.R., Herrera, F., Lozano, M., 2006. On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining. Applied Soft Computing 2006, 323–332.

Cantu-Paz, E., 2000. Efficient and Accurate Parallel Genetic Algorithms. Kluwer Academic Publishers, Norwell, MA, USA.

Chiu, S.L., 1994. Fuzzy model identification based on cluster estimation. Journal of Intelligent and Fuzzy Systems .

Coello, C.A., 1999. A comprehensive survey of evolutionary multiobjective optimization techniques. Knowledge and Information Systems 1, 269–308.

Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J., 2001. PESA-II: Region-based selection in evolutionary multiobjective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers. pp. 283–290.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30.

Derrac, J., García, S., Herrera, F., 2010. Stratified prototype selection based on a steady-state memetic algorithm: A study of scalability. Memetic Computing 2, 183–199.

Frank, A., Asuncion, A., 2010. UCI machine learning repository. URL: `http://archive.ics.uci.edu/ml`.

Freitas, A.A., 2002. Data Mining and Knowledge Discovery with Evolutionary Algorithms. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics 11, 86–92.

García, S., Herrera, F., 2008. An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. Journal of Machine Learning Research 9, 2677–2694.

Garcia-Piquer, A., Fornells, A., Bacardit, J., Orriols-Puig, A., Golobardes, E., 2013. Large-scale experimental evaluation of cluster representations for multiobjective evolutionary clustering. IEEE Transactions on Evolutionary Computation , 36–53.

Garcia-Piquer, A., Fornells, A., Orriols-Puig, A., Corral, G., Golobardes, E., 2012. Data Classification through an Evolutionary Approach Based on Multiple Criteria. Knowledge and Information Systems 33, 35–56.

Goldberg, D.E., 2002. The Design of Innovation. Kluwer Academic Publishers, Massachusetts, US.

Halkidi, M., Batistakis, Y., Vazirgiannis, M., 2001. On clustering validation techniques. Journal of Intelligent Information Systems 17, 107–145.

Handl, J., Knowles, J., 2007. An evolutionary approach to multiobjective clustering. IEEE Transactions on Evolutionary Computation 1, 56–76.

Holm, S., 1979. A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics 6, 65–70.

Hore, P., Hall, L.O., Goldgof, D.B., 2009. A scalable framework for cluster ensembles. Pattern Recognition 42, 676–688.

Hruschka, E.R., Campello, R.J.G.B., Freitas, A.A., de Carvalho, A.C.P.L.F., 2009. A survey of evolutionary algorithms for clustering. IEEE Transactions on Systems, Man and Cybernetics, Applications and Reviews 39, 133–155.

Iredi, S., Merkle, D., Middendorf, M., 2000. Bi-criterion optimization with multi colony ant algorithms, in: Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization, Springer. pp. 359–372.

Kargupta, H., Han, J., Yu, P.S., Motwani, R., Kumar, V., 2009. Next Generation of Data Mining. Chapman & Hall/CRC data mining and knowledge discovery series, CRC Press, USA.

Kaufman, L., Rousseeuw, P.J., 1990. Finding groups in data: An introduction to cluster analysis. John Wiley & Sons .

Park, Y., Song, M., 1998. A genetic algorithm for clustering problems, in: Proceedings of the 3rd Annual Conference on Genetic Programming, Morgan Kaufmann. pp. 568–575.

Saha, S., Bandyopadhyay, S., 2010. A new multiobjective clustering technique based on the concepts of stability and symmetry. Knowledge and Information Systems 23, 1–27.

Yeung, K., Ruzzo, W., 2001. Details of the adjusted rand index and clustering algorithms. supplement to the paper "an empirical study on principal component analysis for clustering gene expression data". Science 17, 763–774.

Zitzler, E., Deb, K., Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms. Evolutionary computation 8, 173–195.