An EDA Approach for a Density and Grid-Based Clustering Algorithm

César Siqueira de Oliveira
Universidade Federal do Pará, Departamento de Informática,
Belém, Brasil, Pará
avcesar@gmail.com

and

Paulo Igor A. Godinho
Universidade Federal do Pará, Departamento de Informática,
Belém, Brasil, Pará
piagodinho@gmail.com

and

Aruanda S. Gonçalves Meiguins
Universidade Federal do Pará, Departamento de Engenharia Elétrica,
Belém, Brasil, Pará
aruanda@redeinformatica.com.br

and

Bianchi Serique Meiguins
Universidade Federal do Pará, Departamento de Informática,
Belém, Brasil, Pará
bianchi.serique@gmail.com

and

Alex A. Freitas
University of Kent, Computing Laboratory,
Canterbury, Kent, CT2 7NF, UK
A.A.Freitas@kent.ac.uk

Abstract

This paper presents EDACluster, an Estimation of Distribution Algorithm (EDA) applied to the clustering task. EDA is a new class of Evolutionary Algorithm used here to optimize the search for adequate clusters when very little is known about the target dataset. The proposed algorithm uses a mixed approach – density and grid-based – to identify sets of dense cells in the dataset. The output is a list of items and their associated clusters. Items in low-density areas are considered noise and are not assigned to any cluster.

Keywords: Evolutionary Algorithm, Estimated Distribution Algorithm, Clustering.

Introduction

Clustering is a Data Mining task with several applications in different areas such as engineering, biology, psychology, medicine, archeology, geology and marketing [1]. Clustering is applied to pattern recognition, image processing, feature selection in data mining, non-supervised learning, speech recognition, among others [2].

Clustering identifies groups (clusters) of items in a dataset using similarity measures. A generic clustering procedure groups similar objects in the same cluster and different objects in other groups.

There are many approaches to identify what set of items are similar enough to be part of the same cluster. Clustering algorithms can be classified according to the following clustering methods: partitioning, hierarchical, density-based, grid-based and model-based methods [3].

Density-based clustering algorithms identify high-density areas separated by low-density areas in the dataset. This technique allows the discovery of arbitrary-shape clusters.

Another recent approach to clustering is grid-based clustering algorithms. The method splits the analyzed data into multidimensional cells. The processing time is therefore independent of the size of the dataset – it depends only on the number of cells for each dimension. Both density and grid-based methods are able to identify noise and are efficient when applied to spatial databases.

The use of the combined density and grid-based approach was previously proposed by the CLIQUE algorithm [4].

This work presents an evolutionary clustering algorithm that uses a mixed approach: density and grid-based. The EDACluster algorithm uses an Estimation of Distribution Algorithm (EDA) that will be presented in section 2. Section 3 presents previously proposed Evolutionary Algorithms for clustering. Section 4 details the implementation of EDACluster. The tests results are presented in section 5 and the final remarks in section 6.

1 Estimation of Distribution Algorithms

Evolutionary Computing comprehends several techniques inspired in the Evolution Theory. Evolutionary algorithms find appropriate solutions to complex optimization problems using a simplified model of reality [5].

A generic Evolutionary Algorithm (EA) performs a number of generations (iterations) each time generating a set of individuals (chromosomes) potentially better than the previous set. Methods such as crossover, mutation and selection are used for the optimization of solutions to a certain problem.

Evolutionary Algorithms depend on a series of parameters, such as the crossover and mutation rates. Those parameters have a very important role with direct influence in the results of the EA. The quality of results is therefore sensitive to the choice of parameter values. Estimation of Distribution Algorithms (EDA) are an alternative to traditional evolutionary approaches [6].

An Estimation of Distribution Algorithm does not use crossover and mutation operators. Instead EDA are based on the probability distribution of selected individuals to generate subsequent populations. A generic EDA follows the steps below [6]:

1. Produce a population of N individuals, usually assuming a uniform distribution for each variable.
2. Select m < N individuals from the population using any of the evolutionary selection method such as tournament, roulette or ranking.
3. Evaluate selected individuals and generate an updated probabilistic model for the variables.
4. Use the distribution model to generate a new population of N individuals.
5. Return to step 2 until a stop criterion is reached.

2 Related Work

Evolutionary Algorithms (EA) have been successfully applied to clustering using many different approaches. EAs perform a non-deterministic, parallel and optimized search. EAs are therefore an appropriate approach to identify clusters in a dataset [7]. One possible approach combines Genetic Algorithms (GA) and Simulated Evolution [8]. Other hybrid clustering algorithms are based on partition clustering and GA [9] [10] [11]. Estimation of Distribution Algorithms was previously applied to partition clustering [6].

For instance, the following steps are an example of pseudo-code for a GA-based clustering algorithm [12]:

1. Randomly generate a population of solutions. Each individual corresponds to a valid partition of the dataset in k parts, where k is the number of clusters parameter.
2. Evaluate the fitness of each candidate solution given by the sum of the distances of the data items to the candidate cluster centers.
3. Apply selection, crossover and mutation operators to generate the next population.
4. Return to step 2 if the fitness function has varied in the last iteration.

3 The EDACluster Algorithm

The EDACluster algorithm uses a density and grid-based approach for clustering. The algorithm identifies dense areas in the database, defined by a set of adjacent high density cells. The identification of the clusters begins by a small set of cells closer to a specific cluster center. The process iteratively adds new cells while the density of the cluster is not reduced above a user-defined threshold.

3.1 Configuration Step

The first step of the algorithm organizes the data items in cells of a multidimensional grid. Each cell (unit) is delimited by equally spaced intervals corresponding to the attributes values. As all the cells have the same volume, the density measure of a cell is given by the number of its points, as in the density-based clustering algorithm CLIQUE [4].

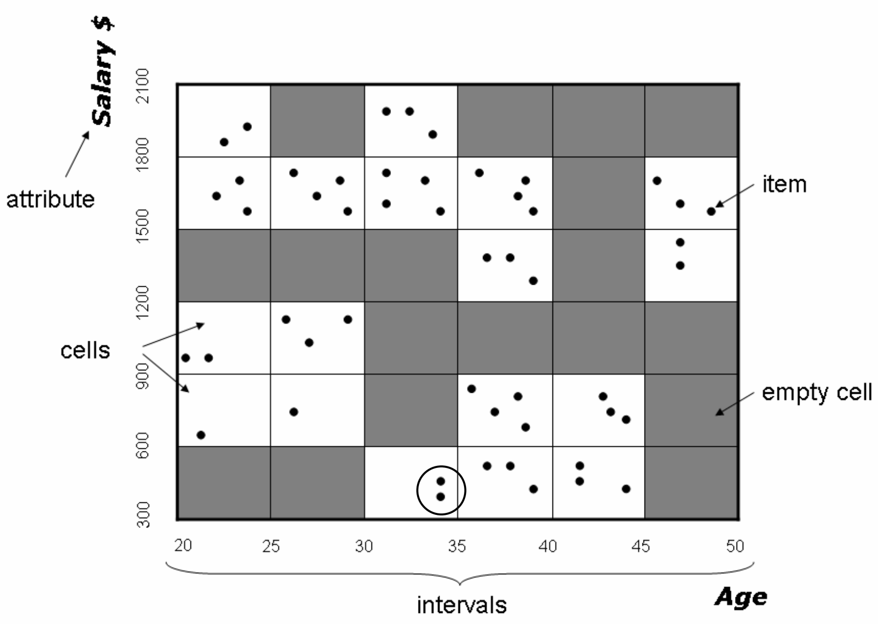


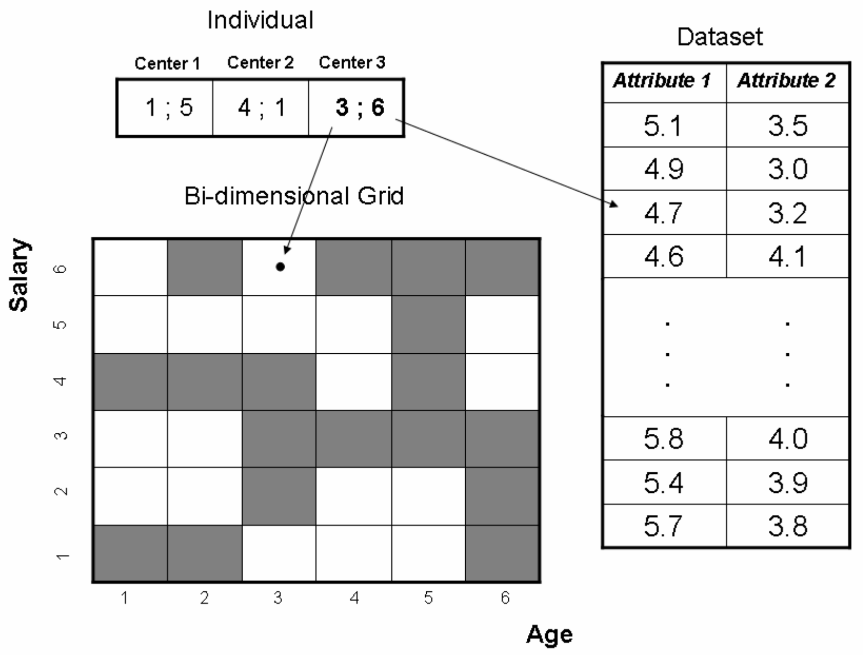
Figure 1. Configuration Step: data items in grid cells

Figure 1 presents a configuration example for a bi-dimensional data grid (age x salary) split into six intervals. After the configuration step, data items are referenced by their cells and not longer by their original attributes values. For instance, in Figure 1 two employees are between 30 and 35 years with salaries from $ 300.00 to $ 600.00. They will be referenced by their cell identifier: 3-1.

The number of intervals for each dimension is an important parameter of the algorithm. For a high number of intervals, i.e. for a fine granularity grid, the average cell density will naturally be low.

3.2 Individual Representation

The individual is formed by a set of cluster center candidates, each associated to any data item. The number of centers may be fixed or variable. In the fixed mode, all the individuals have the same number of clusters; in the variable mode, different individuals may propose a different number of clusters.
The individuals in the initial population are created by a random selection of data items. Data items are represented by their cells in the multidimensional grid. Figure 2 presents an example of the individual representation. The individual length (number of centers) is 3. There are two values separated by semicolon for each center corresponding to the attributes in the dataset.



Attribute 1	*Attribute 2*
5.1	3.5
4.9	3.0
4.7	3.2
4.6	4.1
. . .	. . .
5.8	4.0
5.4	3.9
5.7	3.8

Figure 2. Individual representation

3.3 Fitness Function

The fitness function of an individual i is associated to the density of each cluster given by (1). The fitness function is defined in (2) as the average density considering each cluster center.

$$Density(cluster) = \frac{Number\ of\ items}{Number\ of\ cells} \qquad (1)$$

$$Fitness = \frac{\sum_{i=1}^{k} Density(i)}{k} \qquad (2)$$

Therefore, the fitness function favors individuals associated to high-density clusters.

3.4 Population Generation

Each generation of the algorithm creates a new population of individuals according to the previously updated probabilistic model.
A number m < N of individuals is selected by tournament selection [13]. In the tests tournament was applied to 50% of the population. A probability model is set up for the selected individuals using the PBIL method [14]. Initially, a uniform distribution is used. The model indicates the probability of each value

interval for each attribute. The model is then use to generate more adapted individuals for the new population. Each generation updates the probability model.

Table 1. Example of probability model

Interval	Attribute1	Attribute2	P1(At1)	P1(At2)
1	4	3	4/20	3/20
2	6	7	6/20	7/20
3	2	5	2/20	5/20
4	8	5	8/20	5/20
Total	20	20	1	1

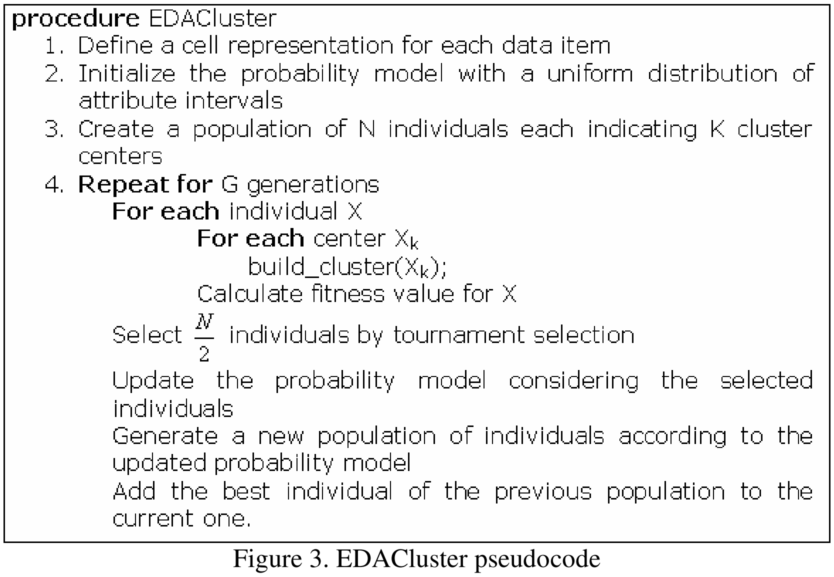
Table 1, for instance, presents the probabilities of a 10 individual population with two cluster centers. It presents the number of centers assigned to each attribute interval and the probability given by the number of centers in each interval divided by the total number of centers. For "Attribute1", the major probability distribution P1(At1) is associated to "Interval 4". For "Attribute2" the major probability is associated to "Interval 2." Those intervals have therefore greater probability to be selected for individuals of the next generation.

The probability model is updated according to the PBIL method. The new probability pi(x) for an attribute interval is given by (3) in terms of its previous probability pi(x)', its current probability pi(x)" and the α parameter. This parameter varies from 0 to 1 and indicates how much the new probability value is influenced by the previous one. The performed tests used α = 0.5.

$$p_i(x) = (1-\alpha)p_i(x)' + \alpha * p_i(x)'' \quad (3)$$

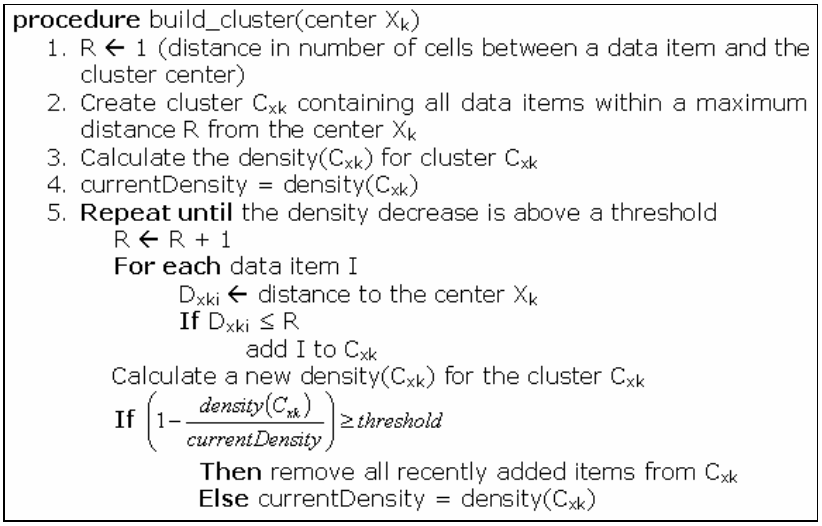
3.5 EDACluster Pseudocode

Figures 3 and 4 present the EDACluster main pseudocode and the procedure that creates density-based clusters, respectively. Two parameters are required: the number of intervals used in the configuration step (section 4.1) and the density reduction threshold, used to validate each new set of adjacent cells assigned to a cluster.



```
procedure EDACluster
   1. Define a cell representation for each data item
   2. Initialize the probability model with a uniform distribution of
      attribute intervals
   3. Create a population of N individuals each indicating K cluster
      centers
   4. Repeat for G generations
        For each individual X
              For each center $X_k$
                   build_cluster($X_k$);
              Calculate fitness value for X
        Select $\frac{N}{2}$ individuals by tournament selection
        Update the probability model considering the selected
        individuals
        Generate a new population of individuals according to the
        updated probability model
        Add the best individual of the previous population to the
        current one.
```

Figure 3. EDACluster pseudocode



```
procedure build_cluster(center $X_k$)
  1. R ← 1 (distance in number of cells between a data item and the
     cluster center)
  2. Create cluster $C_{xk}$ containing all data items within a maximum
     distance R from the center $X_k$
  3. Calculate the density($C_{xk}$) for cluster $C_{xk}$
  4. currentDensity = density($C_{xk}$)
  5. Repeat until the density decrease is above a threshold
       R ← R + 1
       For each data item I
           $D_{xki}$ ← distance to the center $X_k$
           If $D_{xki} \le R$
               add I to $C_{xk}$
       Calculate a new density($C_{xk}$) for the cluster $C_{xk}$
       If $\left(1 - \frac{density(C_{xk})}{currentDensity}\right) \ge threshold$
           Then remove all recently added items from $C_{xk}$
           Else currentDensity = density($C_{xk}$)
```

Figure 4. Procedure build_cluster

Each iteration of the build_cluster(Xk) procedure evaluates the addition of a set of adjacent cells to the cluster of center Xk. The procedure iteratively increases the maximum distance R between each data item and the cluster center. The Spearman footrule distance is used to estimate the absolute distance between two data items [15].

4 Experimental Results

Tests used four public databases: Hungary Heart Disease, Bupa, Auto-mpg and Iris datasets [16] [17]. Comparative tests were applied to EDACluster and the density-based clustering algorithm DBSCAN [18]. Table 2 presents the distribution specified by the algorithms for items in each cluster and each dataset. Both algorithms identified part of the items as noise.

Table 2. Distribution of data items by cluster

Algorithm	Dataset	C1	C2	C3
EDACluster	Iris	58%	24%	18%
DBSCAN	Iris	33%	66%	1%
EDACluster	Hungary	93%	5%	2%
DBSCAN	Hungary	97%	2%	1%
EDACluster	Auto-mpg	55%	35%	10%
DBSCAN	Auto-mpg	60%	38%	2%
EDACluster	Bupa	42%	19%	39%
DBSCAN	Bupa	97%	2%	1%

The evaluation of the identified clusters was based on an adaptation of the Clest method [19] originally proposed to estimate the number of clusters of a database. The method splits the dataset into a training dataset and a test dataset. It applies the clustering algorithm to the training dataset. Each data item is assigned to a cluster or identified as noise. Next, the classification algorithm J48 [17] is applied to the same base using the label of the cluster as the data item class. The accuracy of this classifier in the test dataset is used as a quality measure for the proposed clustering.

As the accuracy of the classifier may be influenced by the fraction of items with the same class label, the metric accuracy lift [20] is used to evaluate the accuracy gain in relation to the default accuracy. In (4), Acc is the algorithm accuracy and DefAcc is the default accuracy, defined as the percentage of items in the majority class. The function produces a value between -1 (when Acc = 0) and 1 (when Acc=1). For instance, Acc = 94% may seem a high accuracy value, but if DefAcc is 95% that Acc value is actually a low one. The accuracy lift measure recognizes that, since in that case accuracy lift = -0.01.

$$AccuracyLift = \begin{cases} \dfrac{Acc - DefAcc}{1 - DefAcc} & se \quad Acc > DefAcc \\ \dfrac{Acc - DefAcc}{DefAcc} & se \;\; Acc \leq DefAcc \end{cases} \tag{4}$$

Table 3 presents the evaluation results for EDACluster e DBSCAN. When clustering the Auto-mpg dataset, DBSCAN achieved better results. The difference in terms of accuracy-lift, however was very low. In all other cases EDACluster performed better. DBSCAN bad performance was often due to the large number of items assigned to the same cluster. EDACluster cluster assignments were more uniformly distributed.

Table 3. Comparative Results

Algorithm	Dataset	*Accuracy*	*Acc Lift*
EDACluster	Iris	0.95	0.88
DBSCAN	Iris	0.82	0.47
EDACluster	Hungary	0.94	0.45
DBSCAN	Hungary	0.78	-0.19
EDACluster	Auto-mpg	0.98	0.96
DBSCAN	Auto-mpg	0.99	0.99
EDACluster	Bupa	0.95	0.92
DBSCAN	Bupa	0.80	-0.17

5 Final Remarks

This paper presented an evolutionary density-based clustering algorithm. EDACluster presented competitive results when compared with DBSCAN, a non-evolutionary density-based clustering algorithm. A direction for future work is the adaptation of the algorithm to support arbitrary shape clusters. This will be accomplished by increasing the maximum distance iteratively in each dimension. Clusters will therefore involve different interval lengths defined for each attribute. Another research direction is the development of a parameter optimization algorithm for customization to any specific dataset.

References

[2] Tan, P. et al., Introduction to Data Mining, Addison Wesley, Boston, 2005.

[3] Mirkin, B., Clustering for Data Mining: A Data Recovery Approach., Chapman & Hall/CRC, London, 2005.

[4] Han, J., Kamber, M., Data Mining: Concepts and Techniques, 2nd edition, Morgan Kaufmann, San Francisco, 2006.

[5] R. Agrawal et al., “Automatic subspace clustering of high dimensional data for data mining applications” In: ACM-SIGMOD Int. Conf. Management of Data. Washington, 1998, pp. 1 - 12.

[6] Fogel, D. B., Evolutionary Computation: toward a new philosophy of machine intelligence. Second Edition, IEEE Press, 2000.

[7] Larrañaga, P., Lozano, J., Estimation of Distribution Algorithms: A new tool for Evolutionary Computation, Kluwer Academic Publishers, Boston, 2002.

[8] Freitas, A., Data Mining and Knowledge Discovery with Evolutionary Algorithms, Berlin: Springer, 2002.

[9] J. Bhuyan, “A Combination of Genetic Algorithm and Simulated Evolution Techniques for Clustering”, In: ACM 0-89791-737-5., 1995.

[10] K. Krishna, N. Murty, “Genetic K-Means Algorithm”, Indian Institute of Science, 1998.

[11] C-F. Tsai et al., “MSGKA: An Efficient Clustering Algorithm for Large Databases,” Taiwan, 2002.

[12] H. Zhang et al., “An Evolutionary K-Means Algorithm for Clustering Time Series Data”, In: Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 2004

[13] A. K. Jain et al., “Data Clustering: A Review”, In: ACM Computing Surveys, Vol. 31, No 3, 1999.

[14] D. E. Goldberg, K. Deb, “A Comparative Analysis of Selection Schemes Used in Genetic Algorithms”, 1991, pp. 69-93.

[15] S. Baluja, R. Caruana, “Removing the genetics from standard genetic algorithm”, In: Proceedings of the International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 38-46.

[16] P. Diaconis and R. Graham. “Spearman's footrule as a measure of disarray.” J. of the Royal Statistical Society, Series B, 39(2), pp. 262-268, 1977.

[17] D.J. Newman, S. Hettich, C.L. Blake, C.J. Merz, “UCI Repository of machine learning databases” [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[18] Ian, H. W., Eibe, F., Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[19] M. Ester et al., “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.” In: KDD-96 2 Proc. of the 2nd Intl Conf on Knowledge Discovery and Data Mining, AAAI Press, 1996, pp. 226 - 231.

[20] J. N. Breckenridge, “Replicating cluster analysis: Method, consistency and validity.” In: MULTIVARIATE BEHAVIORAL RESEARCH, 24, 1989, pp. 147 - 161.

[21] G. Pappa, A. Freitas, “Automatically Evolving Rule Induction Algorithms”, University of Kent Canterbury, UK, 2006.