Finding irregularly shaped clusters based on Entropy

Angel Kuri-Morales[1], Edwin Aldana-Bobadilla[2]
[1] Department of Computation,
Autonomous Technological Institute of Mexico,
Rio Hondo No. 1,
Mexico City, Mexico

[2] Institute of Research in Applied Mathematics and Systems,
Autonomous University of Mexico,
University City,
Mexico City, Mexico
akuri@itam.mx, ealdana@uxmcc2.iimas.unam.mx

Abstract. In data clustering the more traditional algorithms are based on similarity criteria which depend on a metric distance. This fact imposes important constraints on the shape of the clusters found. These shapes generally are hyperspherical due to the fact that each element in a cluster lies within a radial distance relative to a given center. In this paper we propose a clustering algorithm that does not depend on such distance metrics and, therefore, allows us to find clusters with arbitrary shapes in n-dimensional space. Our proposal is based on some concepts stemming from Shannon's information theory and evolutionary computation. Here each cluster consists of a subset of the data where entropy is minimized. This is a highly non-linear and usually non-convex optimization problem which disallows the use of traditional optimization techniques. To solve it we apply a rugged genetic algorithm (the so-called Vasconcelos' GA). In order to test the efficiency of our proposal we artificially created several sets of data with known properties in a tridimensional space. The result of applying our algorithm has shown that it is able to find highly irregular clusters that traditional algorithms cannot. Some previous work is based on algorithms relying on similar approaches (such as ENCLUS' and CLIQUE's). The differences between such approaches and ours are also discussed.

Keywords: clustering, data mining, information theory, genetic algorithms

1 Introduction

Clustering is an unsupervised process that allows the partition of a data set X in k groups or *clusters* in accordance with a similarity criterion. This process is unsupervised because it does not require a priori knowledge about the clusters. Generally the similarity criterion is a distance metrics based in *Minkowsky Family* [1] which is given by:

$$d_{mk}(P,Q) = \sqrt[p]{\sum_{i=1}^{n} |P_i - Q_i|^p} \quad \textbf{(1)}$$

where *P* and *Q* are two vectors in an n-dimensional space. From the geometric point of view, these metrics represent the spatial distance between two points. However, this distance is sometimes not an appropriate measure for our purpose. For this reason sometimes the clustering methods use statistical metrics such as *Mahalanobis'* [2], *Bhattacharyya's* [3] or *Hellinger's* [4], [5]. These metrics statistically determine the similarity of the probability distribution between random variables *P* and *Q*. In addition to a similarity criterion, the clustering process typically requires the specification of the number of clusters. This number frequently depends on the application domain. Hence, it is usually calculated empirically even though there are methodologies which may be applied to this effect [6].

1.1 A Hierarchy of Clustering Algorithms

A large number of clustering algorithms has been proposed which are usually classified as follows:

Partitional. Which discover clusters relocating iteratively elements of the data set between subsets. These methods tend to build clusters of proper convex shapes. The most common methods of this type are k-means [7], k-medoids or PAM (Partitioning Around Medoids) and CLARA (Clustering Large Applications) [8].

Hierarchical. In which large clusters are merged successively into smaller clusters. The result is a tree (called a *dendrogram*) whose nodes are clusters. At the highest level of the *dendrogram* all objects belong to the same cluster. At the lowest level each element of the data set is in its own unique cluster. Thus, we must select the adequate cut level such that the clustering process is satisfactory. Representative methods in this category are BIRCH [9], CURE and ROCK [10].

Density Based. In this category a cluster is a dense (in some pre-specified sense) region of elements of the data set that is separated by regions of low density. Thus, the clusters are identified as areas highly populated with elements of the data set. Here each cluster is flexible in terms of their shape. Representative algorithms of this category are DBSCAN [11] and DENCLUE [12].

Grid Based. Which use space segmentation through a finite number of cells and from these performs all operations. In this category are STING (Statistical Information Grid-based method) described by Wang et al. [13] and Wave Cluster [14].

Additionally, there are algorithms that use tools such as fuzzy logic or neural networks giving rise to methods such as Fuzzy C-Means [15] and Kohonen Maps [16], respectively. The performance of each method depends on the application domain. However, Halkidi [17] present several approaches that allow to measure the quality of the clustering methods via the so-called "quality indices".

1.2 Desired Properties of Clustering Algorithms

In general a good clustering method must:

- Be able to handle multidimensional data sets.
- Be independent of the application domain.
- Have a reduced number of settings.
- Be able to display computational efficiency.
- Be able to yield irregular shaped clusters.

With respect to last point, the great majority of the clustering methods restrict the shape of the clusters to hyperspherical shapes (in the space of the metric) owing to the use of some sort of distance as a similarity criterion. Thus, necessarily the distance between each point inside a cluster and its center is smaller than the radius of an n-dimensional sphere. This is illustrated in Figure 1 for the simplest case where n=2 (and, thus, yields a circle).

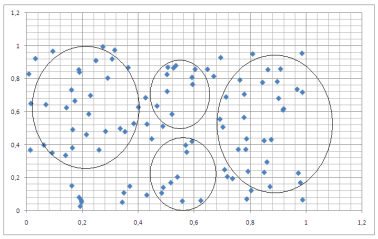

Fig. 1. Clusters with circular shapes

An ideal case would allow us to obtain arbitrary shapes for the clusters that adequately encompass the data. Figure 2 illustrates this fact.

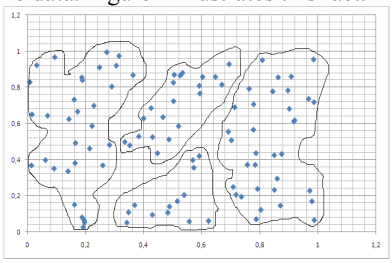

Fig. 2. Clusters with arbitrary shapes

Therefore, we propose a clustering algorithm that allows us to find clusters with irregular shapes that represent the data set better than clustering approaches that use distance metrics.

2 Related Works

Our proposal is based on maximizing density in terms of the entropy of the area of the space that represents a cluster. There exist previous works with similar approaches. Cheng et al [18], for example, present an algorithm called ENCLUS (Entropic Clustering) in which they link the entropy with two concepts that the authors call *coverage* and *density*. These are determined by the space's segmentation. Segmentation is made iteratively. Henceforth, several conditions have to be satisfied for every iteration of the algorithm. The space segmentation is a partition on non-overlapping rectangular units based on CLIQUE (Clustering in Quest) algorithm where a unit is dense if the fraction of the elements contained in the unit is greater than a certain threshold. A cluster is the maximum set of connected dense units. Another work is the so-called COOLCAT algorithm [19] which also approaches the clustering problem on entropic considerations but is mainly focused on categorical sets of data. The difference of our proposal is that the space is quantized through a hypercube that encapsulates all elements of the data set. The hypercube is composed of units of quantization that called "hypervoxels" or, simply, "voxels". The number of voxels determines the resolution of the hypercube. Contrary to ENCLUS, our algorithm does not iterate to find the optimal space quantization. Here the hypercube is unique and its resolution is given a priori as a parameter. The units of quantization become the symbols of the source's alphabet which allow an analysis through information theory. Our working hypothesis is that areas with high density have minimum entropy with respect to areas with low density.

3 Generalities

In what follows we make a very brief mention of most of the theoretical aspects having to do with the proper understanding of our algorithm. The interested reader may consult the references.

3.1 Information Theory

Information theory addresses the problem of collecting and handling data from a mathematical point of view. There are two main approaches: the statistical theory of communication (proposed by Claude Shannon [20] and the so-called algorithmic complexity (proposed by Andrei Kolmogorov [21]. In this paper we rely on the statistical approach in which information is a series of symbols that comprise a *message,* which is produced by an *information source* and is received by a *receiver* through a *channel*.

Where:
Message. It is a finite succession or sequence of symbols.

Information Source. It is a mathematical model denoted by *S* which represents an entity which produces a sequence of symbols (message) randomly. The space of all possible symbols is called source alphabet and is denoted as Σ [22].
Receiver. It is the end of the communication's channel which receives the message.
Channel. It is the medium used to convey a *message* from an *information source* to a *receiver*.

In this document we apply two key concepts which are very important for our proposal.

Self Information. It is the information contained in a symbol s_i, which is defined as[1]:

$$I(s_i) = -\log_2 p(s_i) \quad (2)$$

Where $p(s_i)$ is the probability that the symbol s_i is generated by the source *S*. We can see that the information of a symbol is greater when its probability is smaller. Thus, the self information of a sequence of statistically independent symbols is:

$$I(s_1 s_2 \dots s_n) = I(s_1) + I(s_2) + \cdots + I(s_n) \quad (3)$$

Entropy. The entropy is the expected value of the information of the symbols generated by the source *S*. This value may be expressed as:

$$H(S) = \sum_{i=1}^{n} p(s_i) I(s_i) = -\sum_{i=1}^{n} p(s_i) \log_2 p(s_i) \quad (4)$$

Where *n* is the size of the alphabet Σ. Therefore, we see that entropy is greater the more uniform the probability distribution of symbols is.

3.2 Genetic Algorithms

Genetic Algorithms (GA) (a very interesting introduction to genetic algorithms and other evolutionary algorithms may be found in [23]) are optimization algorithms which are frequently cited as "partially simulating the process of natural evolution". Although this a suggestive analogy behind which, indeed, lies the original motivation for their inception, it is better to understand them as a kind of algorithms which take advantage of the implicit (indeed, unavoidable) granularity of the search space which is induced by the use of the finite binary representation in a digital computer.

In such finite space numbers originally thought of as existing in $\Re^n$ actually map into $\boldsymbol{B}^m$ space. Thereafter it is simple to establish that a genetic algorithmic process is

[1] The base for the logarithms is arbitrary. When (as above) we choose base 2 the information is measured in "bits".

a finite Markov chain (MC) whose states are the populations arising from the so-called genetic *operators*: (typically) selection, crossover and mutation. As such they display all of the properties of a MC. From this fact one may infer the following mathematical properties of a GA: 1) The results of the evolutionary process are independent of the initial population and 2) A GA preserving the best individual arising during the process will converge to the global optimum (albeit the convergence process is not bounded in time). For a proof of these facts the interested reader may see [24]. Their most outstanding feature is that, as opposed to other more traditional optimization techniques, the GA iterates simultaneously over *several* possible solutions. Thereafter, other plausible solutions are obtained by combining (*crossing over*) the *codes* of these solutions to obtain hopefully better ones. The solution space (SS) is, therefore, traversed stochastically searching for increasingly better plausible solutions. In order to guarantee that the SS will be globally explored some bits of the encoded solution are randomly selected and changed (a process called *mutation*). The main concern of GA-practitioners (given the fact that well designed GAs, in general, will find the best solution) is to make the convergence as efficient as possible. The work of Forrest et al. has determined the characteristics of the so-called *Idealized GA* (IGA) which is impervious to GA-hard problems [25].

3.3 Vasconcelos' Genetic Algorithms

The implementation of the IGA is unattainable in practice. However, a practical approximation called the Vasconcelos' GA (VGA) has been repeatedly tested and proven to be highly efficient [26]. The VGA, therefore, turns out to be an optimization algorithm of broad scope of application and demonstrably high efficiency.

A statistical analysis was performed by minimizing a large number of functions and comparing the relative performance of six optimization methods[2] of which five are GAs. The ratio of every GA's absolute minimum (with probability $P=0.95$) relative to the best GA's absolute minimum may be found in Table 1 under the column "Relative Performance". The number of functions which were minimized to guarantee the mentioned confidence level is shown under "Number of Optimized Functions".

[2] VGA: Vasconcelos' GA; EGA: Eclectic GA; TGA: Elitist GA; SGA: Statistical GA; CGA: Canonical (or Simple) GA; RMH: Random Mutation Hill Climber.

Table 1. Relative Performance of Different Breeds of Genetic Algorithms

Algorithm	Relative Performance	Number of Optimized Functions
VGA	1.000	2,736
EGA	1.039	2,484
TGA	1.233	2,628
SGA	1.236	2,772
CGA	1.267	3,132
RHC	3.830	3,600

It may be seen that the so-called Vasconcelos' GA (VGA) in this study was the best of all the analyzed variations. Interestingly the CGA (the classical or "canonical" genetic algorithm) comes at the bottom of the list with the exception of the random mutation hill climber (RHC) which is not an evolutionary algorithm. According to these results, the minima found with the VGA are, on the average, more than 25% better than those found with the CGA. Due to its tested efficiency, we now describe in more detail the VGA.

Outline of Vasconcelos' Genetic Algorithm (VGA)

1. Generate random population of *n* individuals (suitable solutions for the problem).
2. Evaluate the fitness *f(x)* of each individual *x* in the population.
3. Order the *n* individuals from best (top) to worst (bottom) for *i=1, 2,. . . , n* according to their fitness.
4. Repeat steps A-D (see below) for $i = 1,2,...,\lfloor n/2 \rfloor$.
   A. *Deterministically* select the *i-th* and the $(n - i + 1) - th$ individuals (the *parents*) from the population.
   B. With probability *Pc* cross over the selected parents to form two new individuals (the *offspring*). If no crossover is performed, offspring are an exact copy of the parents.
   C. With probability *Pm* mutate new offspring at each locus (position in individual).
   D. Add the offspring to a new population
5. Evaluate the fitness *f(x)* of each individual *x* in the new population
6. Merge the newly generated and the previous populations
7. If the end condition is satisfied, stop, and return the best solution.
8. Order the *n* individuals from best to worst (*i=1, 2, . . . , n*) according to their fitness
9. Retain the top *n* individuals; discard the bottom *n* individuals
10. Go to step 4

As opposed to the CGA, the VGA selects the candidate individuals deterministically picking the two extreme (ordered according to their respective fitness) performers of the generation for crossover. This would seem to fragrantly

violate the survival-of-the-fittest strategy behind evolutionary processes since the genes of the more apt individuals are mixed with those of the least apt ones. However, the VGA also retains the best *n* individuals out of the *2n* previous ones. The net effect of this dual strategy is to give variety to the genetic pool (the lack of which is a cause for slow convergence) while still retaining a high degree of elitism. This sort of elitism, of course, guarantees that the best solutions are not lost. On the other hand, the admixture of apparently counterpointed plausible solutions is aimed at avoiding the proliferation of similar genes in the pool. In nature as well as in GAs variety is needed in order to ensure the efficient exploration of the space of solutions[3]. As stated before, all GAs will eventually converge to a global optimum. The VGA does so in less generations. Alternatively we may say that the VGA will outperform other GAs given the same number of generations. Besides, it is easier to program because we need not to simulate a probabilistic process. Finally, the VGA is impervious to negative fitness's values.
We, thus, have a tool which allows us to identify the best values for a set of predefined metrics possibly reflecting complementary goals. This metric(s) may be arbitrary and this suits us well because one way to establish which of a set of software systems is better than the other is to: a) Define a metric or set of metrics. b) Determine the one system whose combination is best for the systems. For these reasons we use in our work the VGA as the optimization method. In what follows we explain our proposal based in the concepts mentioned above.

4 Evolutionary Entropic Clustering

Let X be a data set of elements x_i such that $x_{i=}\{x_{i1}, x_{i2}, \ldots, x_{in}\}$, let D be an n-dimensional space such that $xi \in D$ and let c_j be a subset of D called cluster. Then we must find a function that associates each element of X to the *j-th* cluster c_j as:

$$f(x_i) = c_j; \forall X \; and \; 2 \leq j \leq k \tag{5}$$

Where k is the number of clusters and $f(x_i)$ is called the membership function. Now we describe a method which attempts to identify those elements within the data set which share common properties. These properties are a consequence of (possibly) high order relationships which we hope to infer via the entropy of a quantized vector space. This space, in what follows, will be denoted as the *Hypercubic Wrapper*.

4.1 Hypercubic Wrapper

A *Hypercubic Wrapper* denoted as is an n-dimensional subspace of D such that:

[3] The Latin American philosopher José Vasconcelos proposed that the admixture of all races would eventually give rise to a better one he called the *cosmic* race; hence the algorithm's name.

$$x_i \in HW \quad \forall x_i \in X \tag{6}$$

HW is set of elements v_m called voxels, which are units in n-dimensional that can contain zero or more elements of the set *X*. The cardinality of *HW* is given by the number the voxels that we specify in each dimension of the space *D* such that:

$$|HW| = \prod_{i=1}^{n} L_i \tag{7}$$

Where L_i is the number of voxels in the *i-th* dimension and *n* is the dimension number of *D*. From equation (7) it follows that $\forall v_m \in HW$:

$$0 < m \leq \prod_{i=1}^{n} L_i \tag{8}$$

Figure 4 shows a graphical representation of *HW* in a tridimensional space, where the cardinality is equal to 27 (because there are three voxels in each dimension, i.e. $L_i = 3$).

Fig. 3. Hypercubic Wrapper in a tri-dimensional space

However, in general, $L_i \neq L_j \ \forall i, j$ and it is, therefore, possible to define different *HWs* by changing the values of the L_i's as shown in Figure 4

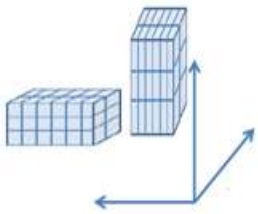

Fig. 4. Hypercubes with different lengths per dimension

Once this wrapper's characteristics are defined, we may use a clustering method based in the concept of *Self Information*, *Entropy*, and *VGA*. Now we describe our proposal which we call the Fixed Grid Evolutionary Entropic Algorithm (FGEEA).

4.2 Fixed Grid Evolutionary Entropic Algorithm

Definition 1. Every voxel that includes at least one element of the data set *X* is called a non-empty voxel; otherwise it is called an empty voxel

Definition 2. For the purpose of entropy calculation, any non-empty voxel is identified with the i-th symbol and will be denoted by s_i.

Definition 3. The space of all symbols is an alphabet denoted by Σ.

Definition 4. The data set is equivalent the source of information S. Such source only produces symbols of Σ.

Definition 5. The probability that the symbol s_i is produced by *S* is the number of its elements divided by the cardinality of the data set *X*. This probability is denoted as $p(s_i)$.

Corollary. The density of a symbol s_i is proportional to its probability $p(s_i)$.

It follows that:

$$|\Sigma| \leq |HW| \tag{9}$$

$$H(S) = \sum_{i=1}^{n} p(s_i) I(s_i) = -\sum_{i=1}^{n} p(s_i) \log_2 p(s_i) \tag{10}$$

Our idea is to use the entropy for determine the membership of every symbol in the *j-th* cluster, based on the follows assumptions:

Assumption 1. The source *S* always produces the same symbols for a given data set *X*.

Assumption 2. The spatial position of each symbol (voxel) is invariant for a given data set *X*. Therefore, H(S) is constant.

According to the working hypothesis, the areas with high density have minimum entropy with respect to areas with low density. Then the areas with minimum entropy possibly identify a cluster.

To determine the entropy of a possible cluster we introduce a term we call *intracluster entropy*, defined as:

$$H(c_i) = -\sum p(s_j) \log_2 p(s_j) \quad \forall\ s_j \in c_i \tag{11}$$

Where $H(c_i)$ is the intracluster entropy of *i-th* cluster. In order to determine that s_j belongs to c_i we use a genetic algorithm, as discussed in what follows.

Application of Vasconcelos' Genetic Algorithm

Our aim is to find the areas of the subspace *HW* where the entropy is minimal. This is to find groups of voxels such as each group have minimal entropy (intracluster entropy).

Clearly this is an optimization problem. For reasons explained above we use a VGA. The individuals of the algorithm have been encoded as follows. a) The length of the genome is equal to the cardinality of Σ. [It is composed by all symbols (or g*enes*)]. b) Each gene is assigned a label that represents the cluster to which it belongs. c) It has a sequential index. Such index will allow mapping all symbols to subspace HW. Fig.5 exemplifies a genome for *k=3*.

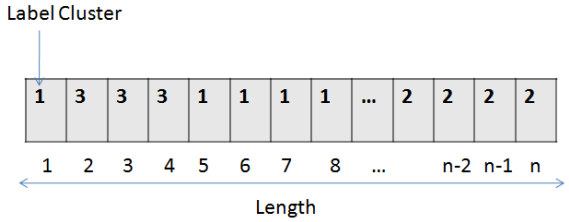

Fig. 5. Genome of the individual (*k=3*).

Now, we defined the fitness function as:

$$f(individual_j) = min \sum_{i=0}^{k} H(c_i) \quad for\ j \leq N \tag{12}$$

Subject to:

$$\sum_{i=0}^{k} H(c_i) \geq H(S) \tag{13}$$

$$\left|\sum_{i=0}^{k} H(c_i) - H(S)\right| \geq \Delta_1 \tag{14}$$

Where N is the size of the population and Δ_1 is a parameter that represents a threshold of the difference between the sum of intracluster entropies and the entropy of source *S*. Additionally we have introduced a constraint called *intracluster density* defined as:

$$dc_i \leq \varepsilon \tag{15}$$

where ε is the threshold density. One last constraint is the intracluster density (dc_i). It is the number of elements of data set *X* which belong to the symbols of *i-th* cluster:

$$dc_i = \frac{\alpha}{\beta} \tag{16}$$

where α is the number the elements that belong to the data set *X* and β is the number of symbol within cluster *i*. This constraint ensures that entropy is minimal within any given cluster. The algorithm yields a best individual which represents a set of clusters of symbols that are map into sets of voxels in the subspace *HW*, as shown in Fig. 6.

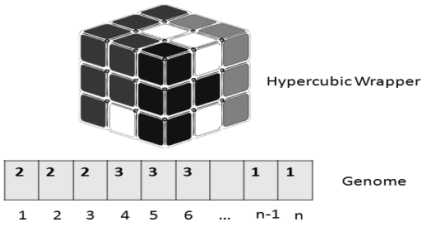

Fig. 6. Possible clustering delivered by the VGA. Different intensities in the cube represent a different cluster. White voxels are empty.

In what follows we show some experiments that allow us to test the effectiveness of the algorithm presented previously

5 Experimental Results

Our algorithm was tested with a synthetic data set which consists of a set of points contained by three disjoint spheres. The features and parameters of the first test are given in Table 2. The values of the parameters were determined experimentally.

Table 2. Features and parameters first test.

Feature	Value	Parameter	Value
Sample size	192	N (Number of individuals)	500
Elements per cluster	64	G (Generations)	1000
Dimensions	3	Pm (Mutation Probability)	0.001
Data Distribution	Disjoint sphere	Pc (Crossover Probability)	0.99
		Δ_i	$3.5<\Delta_i<3.6$
Cardinality of $\sum$	26	ε	5

The VGA was run 20 times (with different seeds of the pseudo random number generator) yielding an average effectiveness of 98%. Notice that no information other than the number of clusters is fed to FGEEA.

The same data set was tested with other algorithms such as *Kohonen Maps* and *Fuzzy C-Means*. The results obtained are shown in Table 3.

Table 3. Results obtained with Kohonen Maps and Fuzzy C-Means.

Algorithm	Average Effectiveness
Kohonen Maps	0.99
Fuzzy C- Means	0.98

This us allow see that the result of our proposal is similar to result given by some traditional algorithms. The high effectiveness in all cases is probably due to the spatial distribution of data set.

Next, we test with other data set whose spatial distribution yields presents overlapping clusters as is shown in Fig. 11. For clarity we show a bi-dimensional example. The actual runs consisted of three dimensional data.

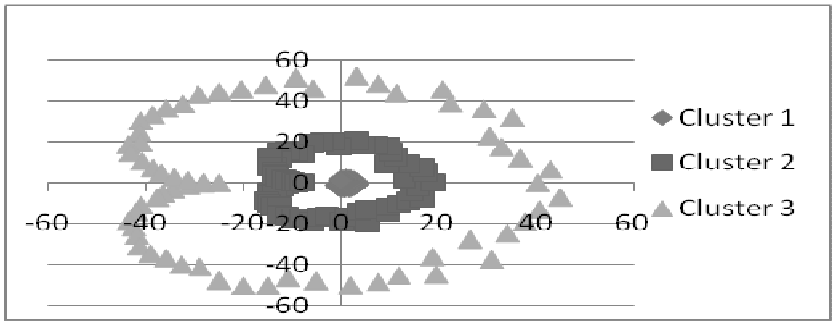

Fig. 7. Overlapping clusters.

In this case also the size sample is 192 elements distributed in three clusters a priori whose cardinality is 64 elements. The results obtained are shown in Table 4.

Table 4. Results of FGEEA with overlapping data set

Algorithm	Average Effectiveness
Kohonen Maps	0.62
Fuzzy C- Means	0.10
FGEEA	0.73

Here the effectiveness decreases significantly in general. But FGEEA showed the better results.

Finally we tested our algorithm with a data set in tridimensional space with an unknown spatial distribution. For $k = 3$ (number of clusters) the algorithm found a solution that is shown in Fig. 8. Here the clusters are irregularly shaped.

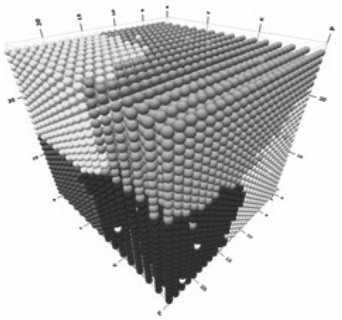

Fig. 8. Irregular clusters. The number of voxels is 15625 (25 voxels per dimension). The white voxels are empty.

These last results were not compared with other clustering algorithms. However, we can see that in principle our approach is feasible.

6 Conclusions and Future Work

These results allow us to test the feasibility of our algorithm. This is not enough, however, to assume its effectiveness in general. To achieve this proof we require testing with several data sets and applying more solid clustering validation techniques. Computationally, the analysis of the geometric and spatial membership relation between elements of a multidimensional data set is hard. Our approach showed that in principle, membership relations in a data set can be found through of its entropy without an excessive demand on computational resources. Even though the results obtained are limited (since they correspond to particular cases and in tri-dimensional data) they are promissory. Therefore, future work requires to generalize our method for a data set in n-dimensional space (with $n>3$), to analyze its computational complexity and to test its detailed mathematical formulation. We will report on these issues shortly.

References

1. Cha, Sung Hyuk: Taxonomy of Nominal Type Histogram Distance Measures, Massachusetts (2008).
2. Mahalanobis, Prasanta Chandra: On the genaralized distance in statistics. (1936).
3. Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by probability distributions, Calcutta (1943).
4. Pollard, David E.: A user's guide to measure theoretic probability. Cambridge University Press, Cambridge (2002).
5. Yang, Grace Lo y Le Cam, Lucien M.: Asymptotics in Statistics: Some Basic Concepts. Springer, Berlin(2000).
6. Li, Xin, Wai, Man y Kwong Li, Chi: Determining the Optimal Number of Clusters by an Extended RPCL Algorithm. Hong Kong Polytechnic University, Hong Kong (1999).

7. MacQueen, J.B.: Some Methods for Classification and Analysis of Multivariate Observations. In :Proceedings of 5th Berkley Sysmposium on Mathematical Statiscs and Probability, pp. 281-297, Berkley (1967).
8. Ng, R y Han, J.: Effecient and Effective Clustering Methods for Spatial Data Mining, Santiago de Chile (1994).
9. Zhang, T, Ramakrishnman, R y Linvy, M.: BIRCH: An Efficient Method for Very Large Databases, Montreal, Canada (1996).
10. Guha, S, Rastogi, R y Shim, K: An efificient Clustering Algorithm for Large Databases, (1998).
11. Ester, M, Kriegel, H., Sander, J., Xu X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, pp. 226-223, Portland (1996).
12. Hinneburg, A y Keim, D.: An Efficient Approach to Clustering in Large Multimedia Databases with noise, (2000).
13. Wang, Wei, Yang, Jiong y Muntz, Richard. STING : A Statistical Information Grid Approach to Spatial Data. In: Proceedings of the 23rd VLDB Conference, Athens (1997).
14. Sheikholeslami, Gh., Chatterjee, S. y Zhang, A.: Wavecluster: A multi-resolution clustering. In : Proceedings of the 24th VLDB conference, (1998).
15. Dunn, J. C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters, pp. 32-57, (1973)
16. Kohonen, T.: Self-Organizing Maps. In: Series in Information Sciences, (1995).
17. Halkidi, M, Batistakis, Y. y Vzirgiannis, M.: On Clustering Validation Techniques, pp. 107-145, (2001)
18. Cheng, C., Fu, Ada W. y Zhang, Y.: Entropy- based Subspace Clustering for Mining Numerical Data, (1998).
19. Barbará, D., Julia, C. y Li, Y.: COOLCAT: An entropy-based algorithm for categorical clustering, George Mason University (2001).
20. Shannon, Claude E. A mathematical theory of communication, pp. 379-423, (1948)
21. Kolmogorov, A. N.: Three approaches to the quantitative definition of information, pp. 1-7, (1948).
22. Gray, Robert M.: Entropy and Information Theory. In : Springer Verlag, (2008).
23. Bäck, T.: Evolutionary Algorithms in Theory and Practice. In: Oxford University Press, (1996).
24. Rudolph, G. Convergence Analysis of Canonical Genetic Algorithms. In. : IEEE Transactions on Neural Networks, (1994).
25. Forrest, S. and Mitchell, M. What makes a problem hard for a genetic algorithm? In : Machine Learning, (1993).
26. Kuri, Angel. A Methodology for the Statistical Characterization of Genetic Algorithms. In : Springer-Verlag, págs. 79-88, (2002).