Entropic-Genetic Clustering

Mihaela Breaban*, Henri Luchian*, Dan A. Simovici**

*Univerity of Jassy, Dept. of Computer Science, Jassy, Romania,
e-mail:pmihaela,hluchian@infoiasi.ro
**Univ. of Massachusetts Boston, Massachusetts 02125, USA,
e-mail:dsim@cs.umb.edu

Abstract. This paper addresses the clustering problem given the similarity matrix of a dataset. By representing this matrix as a weighted graph we transform this problem to a graph clustering/partitioning problem which aims at identifying groups of strongly inter-connected vertices. We define two distinct criteria with the aim of simultaneously minimizing the cut size and obtaining balanced clusters. The first criterion minimizes the similarity between objects belonging to different clusters and is an objective generally met in clustering. The second criterion is formulated with the aid of generalized entropy. The trade-off between these two objectives is explored using a multi-objective genetic algorithm with enhanced operators. As the experimental results show, the Pareto front offers a visualization of the trade-off between the two objectives.

1 Introduction

Clustering is a problem intensively studied within the data mining community because of its wide applicability in diverse fields of sciences, engineering, economy, medicine, etc. and consists of identifying natural groups in data. There exist a wide range of clustering techniques, based on various principles which consequently deliver various solutions. Because of the vague definition of the optimum, clustering is a very difficult optimization problem. From the point of view machine learning, clustering is regarded as unsupervised learning due to the lack of any external information, except the data itself.

Data items can occur as either as tuples, which are ordered sequences of categorial or numerical values, or as simple objects for which only pairwise similarities or dissimilarities are provided. The first kind of layout offers more information and is suitable to any clustering algorithm once an appropriate metric is chosen. The second kind cannot be supplied to any clustering algorithm (e.g., k-means), but eliminates one difficult step in unsupervised clustering analysis - the definition of an appropriate similarity measure. The transition from the first kind of layout to the second is trivial when a similarity function is defined; the backward transition can be performed to some extent using multi-dimensional scaling algorithms.

This paper addresses the clustering problem given the similarity matrix of a dataset. A straightforward representation of the problem instance in this case is a weighted graph, having the objects as vertices and weighted edges expressing the similarity between objects. This

leads to a graph clustering/partitioning problem which aims at identifying groups of strongly inter-connected vertices.

There exist several formal definitions of graph clustering, depending on the practical application and domain where the problem originates. These variations are reflected in the graph structure and in the objectives aimed to be optimized. A survey on various problem definitions and methods for graph clustering is presented in Schaeffer (2007).

The graph clustering problem this paper addresses has important applicability in VLSI circuit design, image processing, and distributing workloads for parallel computation. A formal definition is given next.

A *similarity space* is a pair (S, w), where $w : S \times S \longrightarrow \mathbb{R}$ is a function such that

(i) $w(s,t) \geq 0$ for every $s, t \in S$;

(ii) $w(s,t) = w(t,s)$ for every $t, s \in S$;

(iii) $w(s,s) = 1$ for every $s \in S$.

A similarity space (S, w) can be regarded as a labelled graph $G = (S, E, w)$, referred to as the *similarity graph*, where the set of edges E is defined as

$$E = \{(s_i, s_j) \mid s_i, s_j \in S \text{ and } w(s_i, s_j) > 0\}.$$

In other words, an edge exists between two vertices only if they have a positive similarity; in this case, the edge (s_i, s_j) is labelled by $w(s_i, s_j)$.

If S is a finite set $S = \{s_1, \ldots, s_n\}$, the dissimilarity w is described by a symmetric matrix $W \in \mathbb{R}^{n \times n}$, where $w_{ij} = w(s_i, s_j)$ for $1 \leq i, j \leq n$.

A k-way clustering of a finite similarity space (S, w) is a partition $\kappa = \{C_1, \ldots, C_k\}$ of S. The sets $C_1, \ldots, C_k$ are the clusters of κ. We seek a k-way partition of S, κ such that

(i) the cut size (i.e. the sum of weights of edges between clusters in the similarity graph) is minimal, and

(ii) $|C_p| \approx |C_q|$, for $1 \leq p, q \leq k$, which means that the sizes of the clusters are as equal as possible.

The first objective is generally met in any clustering problem: the elements belonging to different clusters should be dissimilar. In network applications this would correspond to minimizing the communication time. The second objective expresses a load balancing constraint, inherent in network applications.

This multi-objective problem was intensively addressed in literature in the last thirty years. One of the earliest approaches is the method obtained by Kernighan and Lin (1970) which refines a given (randomly generated) partition in a greedy manner by reallocating pairs of nodes between clusters; like all greedy optimizers, this is a local improvement algorithm. Several improvements with regard to time complexity were later proposed. The most used methods are recursive algorithms: in a first step a two-way partitioning is obtained after which, each of the clusters is bisected to obtain a four-way partitioning and the process continues until the desired number of clusters is reached. Recursive spectral bisection algorithms (see Simon (1991)) are known to deliver good solutions but at high computational cost because they require eigenvector computations.

To deal with large graphs, multilevel algorithms were proposed. They consist basically of three phases: coarsening, partitioning and uncoarsening. In the coarsening phase the graph

is compressed by successively collapsing nodes. A partitioning procedure (such as the one obtained by Karypis and Kumar (1998) or a spectral method presented by Barnardand and Simon (1993)) is applied on the coarsened graph. In the uncoarsening phase, a partition is built for the original graph by assigning the collapsed nodes to the same cluster; a costless refining phase may be used at this level.

An extensive state-of-the-art of the methods and comparative studies can be found in Fjällström (1998), Karypis and Kumar (1998) and Alpert (1998).

Because of the multi-objective nature of the problem, we tackle the graph partitioning problem with a multi-objective genetic algorithm with enhanced operators. The benefits of such an approach are obvious: instead of delivering a single solution, a set of several non-dominated solutions approximating the Pareto front is returned. As the experimental results show, the Pareto front offers a visualization of the trade-off between the two objectives; the shape of the Pareto front offers valuable information for the identification of the optimum solution.

The paper is structured as follows. Section 2 examines the two objectives which have to be optimized as stated in the problem definition. Section 3 provides a brief survey on the genetic algorithms for clustering with an emphasis on the multi-objective formulation; the representation and the operators we used are detailed. Section 4 presents experimental results. The paper concludes with a short discussion.

2 Clustering as multi-objective optimization

Let $\kappa = \{C_1, \ldots, C_k\}$ a clustering of the objects of the set $S = \{s_1, \ldots, s_n\}$. The matrix $X \in \mathbb{R}^{n \times k}$ defined by

$$x_{ip} = \begin{cases} 1 & \text{if } s_i \in C_p, \\ 0 & \text{otherwise,} \end{cases}$$

represents the clustering κ. Note that each row of this matrix contains a single 1 and that the total number of 1 entries equals the number n of elements of the set S.

The matrix $Y = X'X \in \mathbb{R}^{k \times k}$ is given by

$$y_{pq} = \sum_{i=1}^{n} x'_{pi} x_{iq} = \sum_{i=1}^{n} x_{ip} x_{iq} \tag{1}$$

for $1 \leq p, q \leq k$. Since any two clusters C_p, C_q are disjoint, this a diagonal matrix. Its diagonal elements are $y_{pp} = |C_p|$ for $1 \leq p \leq k$.

Let $\mathcal{G} = (S, E, w)$ be the similarity graph of S. The symmetric matrix $W \in \mathbb{R}^{n \times n}$ is defined by

$$w_{ij} = \begin{cases} w(s_i, s_j) & \text{if } i \neq j, \\ 1 & \text{if } i = j, \end{cases}$$

for $1 \leq i, j \leq n$.

Let $Z = X'WX \in \mathbb{R}^{k \times k}$. We have

$$z_{pq} = \sum_{i=1}^{n} \sum_{j=1}^{n} x'_{pi} w_{ij} x_{jq} = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ip} w_{ij} x_{jq}$$

for $1 \leq i, j \leq n$. Therefore, for the distinct clusters C_p, C_q, z_{pq} is precisely the value of $\mathsf{cut}(C_p, C_q)$. Note also that

$$z_{pp} = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ip} w_{ij} x_{jp}$$

equals the sum of the similarities between the objects of the clusters C_p. Clearly, to achieve maximal intra-clustering cohesion and minimal inter-clustering dissimilarity it is necessary that the trace of the matrix Z (that is, the sum of the diagonal elements of Z) to be maximal and the sum of the off-diagonal elements of Z to be minimal,

Since Z is a non-negative matrix, its norm $\parallel Z \parallel_1 = \sum_{p=1}^{k} \sum_{q=1}^{k} |z_{pq}|$ coincides with the sum of its elements. Moreover, $\parallel Z \parallel_1 = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}$ and is constant for a given similarity matrix W, regardless of the clustering X. Therefore, the total weight of the inter-cluster cuts equals $\parallel Z \parallel_1 - \mathsf{trace}(Z)$ and minimizing it is equivalent to maximizing the total within clusters similarity which is given as $\mathsf{trace}(Z) = \sum_{p=1}^{k} z_{pp}$.

We use a novel approach to insure that the clusters of κ are balanced. To this end, we use the generalized entropy of partitions of finite sets (see Simovici and Djeraba (2008)) introduced by Daróczy (1970) and by Havrda and Charvat (1967) and axiomatized by Simovici and Jaroszewicz (2002). The use of entropy is suggested by the fact that it is a natural instrument for evaluating the balancing quality of a probability distribution, and, therefore, the balancing quality of a partition of a finite set.

For a partition $\kappa = \{C_1, \ldots, C_k\}$ of a set S and a number $\beta > 1$, the β-entropy is defined by

$$\mathcal{H}_\beta(\kappa) = \frac{1}{1 - 2^{1-\beta}} \left(1 - \sum_{p=1}^{k} \frac{|C_p|^\beta}{|S|} \right).$$

Note that $\lim_{\beta \to 1} \mathcal{H}_\beta(\kappa) = -\sum_{p=1}^{k} \frac{|B_p|}{|S|} \log_2 \frac{|B_p|}{|S|}$. In other words, the Shannon entropy is a limit case of the generalized entropy.

An important special case of the entropy is obtained for $\beta = 2$. We have

$$\mathcal{H}_2(\kappa) = 2 \left(1 - \sum_{p=1}^{k} \frac{|C_p|^2}{|S|} \right),$$

and this is the well-known Gini index, $\mathsf{gini}(\kappa)$ used frequently in statistics. Although we use the Gini index in this paper, our approach can be extended to use other types of entropies and it would be interesting to examine the impact of the specific type of entropy on the performance of the algorithm.

The largest value of $\mathcal{H}_\beta(\kappa)$ is obtained when κ consists of singletons, that is, when $k = n$ and $\kappa = \alpha_S = \{\{s_i\} \mid 1 \leq i \leq n\}$ and is $\mathcal{H}_\beta(\alpha_S) = \frac{1}{1-2^{1-\beta}} \left(1 - \frac{|1|}{|S|}^{\beta-1}\right)$; the least value of $\mathcal{H}_\beta(\kappa)$ is obtained for $\kappa = \omega_S$ and equals 0.

For a prescribed number k of blocks (where k is a divisor of $|S|$), the maximum value of $\mathcal{H}_\beta(\kappa)$ corresponds to a partition having blocks of equal sizes. If k is not a divisor of S, then the more uniform the sizes of the cluster, the larger is the value of $\mathcal{H}_\beta(\kappa)$. This indicates that maximization of the entropy can be used as a criterion for ensuring the uniformity of the cluster sizes. We will use the Gini index of κ because it presents certain computational advantages as shown next.

Theorem 2.1 *Let $\kappa = \{C_1, \ldots, C_k\}$ a clustering of the objects of the set $S = \{s_1, \ldots, s_n\}$ and let $X \in \mathbb{R}^{n \times k}$ be the characteristic matrix of the clustering. We have*

$$\textit{gini}(\kappa) = 2(1 - \textit{trace}(X'XX'X)).$$

Proof. The definition of the matrix Y implies $\mathsf{trace}(X'XX'X) = \mathsf{trace}(Y^2)$. Since Y is a diagonal matrix, we have

$$\mathsf{trace}(Y^2) = \sum_{p=1}^{k} |C_p|^2$$

by Equality (1). Thus,

$$\mathsf{gini}(\kappa) = 2\left(1 - \sum_{p=1}^{k} \frac{|C_p|}{|S|}^2\right) = 2(1 - \mathsf{trace}(Y^2)) = 2(1 - \mathsf{trace}(X'XX'X)),$$

which concludes the proof.

Thus, two objectives can be used to find a balanced k-clustering κ:

(i) minimization of the total cut of the clustering partition, which amounts to minimization of

$$f_1(X) = \| Z \|_1 - \mathsf{trace}(Z) = \| X'WX \|_1 - \mathsf{trace}(X'WX) \tag{2}$$

(ii) maximization of cluster uniformity, which is equivalent to the maximization of the Gini index of κ, or to the minimization of

$$f_2(X) = \mathsf{trace}(X'XX'X) \tag{3}$$

We seek X subjected to the conditions $x_{ip} \in \{0, 1\}$ for $1 \leq i \leq n$ and $1 \leq p \leq k$. Depending on the aspects we need to emphazise in the clustering we can use a convex combination of these criteria:

$$\begin{aligned} \Phi_a(X) &= a\, f_1(X) + (1-a)\, f_2(X) \\ &= a\, (\| X'WX \|_1 - \mathsf{trace}(X'WX)) + (1-a)\, \mathsf{trace}(X'XX'X), \end{aligned}$$

where $a \in [0, 1]$.

To simultaneously minimize criteria f_1 and f_2, also a non-linear combination can be used:

$$\Psi(X) = \frac{f_1(X)}{n^2 - f_2(X)} = \frac{\| X'WX \|_1 - \mathsf{trace}(X'WX)}{n^2 - \mathsf{trace}(X'XX'X)}. \tag{4}$$

The criterion $\Psi(X)$ measures the average link between clusters, because the denominator is proportional with the number of pairs of items that occur in distinct clusters.

3 The clustering algorithm

We use a genetic algorithm (GA) to deal with the graph clustering problem, because such algorithms provide a good exploration of the search space and are able to deliver high-quality

solutions. GAs are soft-computing paradigms inspired from natural evolution. In contrast with the rigid/static models of hard computing, these nature-inspired models provide self-adaptation mechanisms which aim at identifying and exploiting the properties of the instance of the problem being solved.

GAs are iterative algorithms. They work with a population of candidate solutions (also called individuals or chromosomes) which evolve in order to adapt to the "environment" defined by a "fitness function". They involve a degree of randomness which classify them as probabilistic methods. Several approximate good solutions are returned.

An important advantage over classical computational methods is their extended usability. GAs are general-purpose heuristics that can be used to solve diverse optimization problems, extract patterns from data in the machine learning field (eg. classifier systems) or can be useful tools in the design of complex systems.

In a standard GA an individual/chromosome encodes a candidate solution as a bitstring. The initial population is generated randomly or with the aid of greedy heuristics. An iterative process begins, each iteration consisting of three main steps: evaluation, selection and recombination.

The evaluation phase assigns a fitness value to each individual in the population. The fitness function is constructed based on the objectives to be optimized in the problem being solved: the higher the fitness assigned to an individual is, the better the solution encoded by the individual is.

The selection phase simulates natural selection, the competition for survival. At this stage individuals are chosen to breed and create offsprings, mainly based on the individual merit measured by the fitness function. Usually, selection is not deterministic, and it involves a degree of randomness which is essential for maintaining diversity in the population. The lack of diversity is the main culprit for premature convergence and sub-optimal solutions.

The recombination phase aims at creating new individuals by applying genetic operators which are inspired from natural reproduction: crossover and mutation. Crossover is a binary operator which combines the information encoded in two chromosomes and usually produces two offsprings. Mutation is an unary operator: it alters the information encoded in one chromosome and produces one offspring. Not all chromosomes returned in the selection phase are subjected to these operators. Two parameters acting as probability rates decide the frequency of these operations. Along with the population size, these are numeric parameters of the algorithm and are usually empirically determined, in the so-called optimization phase of the algorithm.

The solution returned by a genetic algorithm is the best individual in the last iteration of the algorithm. Several solutions can be reported if necessary.

Like all meta-heuristics, GAs are weak, general optimizers. To increase their performance, the representation of the candidate solutions and the operators must be adapted to incorporate specific knowledge. First attempts to use evolutionary techniques in clustering date back to 1991 (see Krovi (1991)) when genetic algorithms were proposed to search for the optimal partition with an apriori-fixed number of clusters. The encoding used is a straightforward one: solutions are strings of integers, of length equal with the size of the data set, the i^{th} integer signifying the cluster number of data item i. When used in conjunction with the standard genetic operators, this encoding suffers from several drawbacks like redundancy and invalidity and determines a slow convergence of the algorithm.

In Luchian et al. (1994) a new encoding is proposed which considers only cluster representatives, allowing for simultaneous search of the optimum number of clusters and the optimum partition. The partition is constructed in a manner similar to k-means: the data items are assigned to clusters based on the proximity to the cluster representatives. Other more complex encodings were proposed over time (e.g. rules which build a grid in the feature space), but this encoding became the most used when approaching clustering with evolutionary techniques. The criterion optimized in approaches of this kind is the compactness computed based on the distances between the data items and the cluster centers.

Real-world problems necessitate most of the times the optimization of several conflicting objectives. Usually, this is achieved by combining the objectives into a single function. However, some objectives may be more important than others for a given problem and their relative importance cannot be established beforehand. To deal with this kind of problems, multi-objective GAs were proposed. These algorithms optimize simultaneously several objectives and return a set of non-dominated solutions which approximate the Pareto front. For a problem involving m objectives denoted with $f_i, 1 \leq i \leq m$ which have to be minimized, a solution x is dominated by a solution x^* if

$$f_i(x^*) \leq f_i(x),\ \forall i = 1..m,\ and\ \exists 1 \leq j \leq m\ s.t.\ f_j(x^*) < f_j(x).$$

The Pareto optimal set of solutions X^* consists of all those solutions for which no improvement in an objective can be made without a simultaneous worsening in some other objective. In other words, the Pareto front consists of all solutions that are not dominated by any other solution.

The multi-objective scheme we use to tackle the graph clustering problem is PESA-II obtained by Corne et al. (2001). The algorithm maintains two populations of solutions. An external population stores mutually non-dominated clustering solutions, which correspond to different trade-offs between the two objectives. At each iteration an internal population is created by selecting chromosomes from the external population. This selection phase takes into account the distribution of solutions across the two objectives by maintaining a hypergrid of equally sized cells in the objective space. The solutions are selected uniformly from among the populated niches such that highly populated niches do not contribute more solutions than less populated ones. After selection, the crossover and mutation operators are applied within the internal population. The external population is updated by joining the two populations and eliminating the dominated solutions. The general scheme of the algorithm is presented in Algorithm 1.

This multi-objective algorithm was previously used by Handl and Knowles (2005) in unsupervised clustering (when the number of clusters is not fixed and evolve during search) with very good results. The objectives they optimized were connectivity and the within-cluster deviation and because the number of clusters was allowed to vary, a different representation and consequently different operators were used.

The straightforward representation of a solution for the partitioning problem is a string, encoding the cluster membership of each data item. This is the representation used in our GA: an individual is a string of length n (the number of vertices in the graph), taking values in the set $\{1, \ldots, k\}$, where k is the number of clusters.

If the standard operators would be used, this encoding would suffer from several drawbacks like redundancy and invalidity and would determine a slow convergence of the algo-

FIG. 1 – *PESA-II*

Initialize IP (internal population)
Evaluate IP
Initialize EP (external population) to include all mutually
    non-dominated solutions from IP
while $halting\ condition\ not\ met$ **do**
  delete the current content of IP
  fill IP with individuals selected uniformly among the niches from EP
  apply crossover and mutation in IP
  Evaluate IP
  Update EP
end while

rithm. However, due to the multi-objective scheme used and the new operators we propose, this drawbacks are eliminated.

In the initialization phase a minimum spanning tree (MST) is constructed using Prim's algorithm. Half of the population is initialized with candidate solutions created by repeating the following procedure: $k-1$ edges are randomly removed from MST and the connected components are marked as individual clusters. The rest of the population is filled with chromosomes generated randomly.

The crossover operator takes as input two partitions (individuals in the population) and computes their intersection. Since the new partition has more than k clusters, clusters are merged until the required number is reached. The decisions are made with regard to the two objectives to be optimized and therefore two distinct crossover operators are use. One operator aims at decreasing the cut size and therefore performs some iterations of the hierarchical agglomerative clustering algorithm using average linkage metric. The second operator merges iteratively the two smallest clusters aiming at balancing the clusters, until a number of k clusters is reached.

The mutation operator takes as input a single partition and reallocates a randomly chosen vertex and its most similar adjacent vertices to a randomly chosen cluster. The number of adjacent vertices to be reallocated decreases during the run so that in final iterations only small perturbations are allowed. This strategy allows for a quick exploration/diversification phase of the search space in first iterations of the algorithm which degenerates into an exploitation/intensification phase in last iterations.

The fitness functions used in our multi-objective genetic approach are based on the two objectives presented in Section 2 and are formulated for minimization. We maximize the entropy by minimizing the Gini index criterion 3 and minimize the average cut size as expressed by Equation (4).

4 Experiments

Experiments were conducted on synthetic datasets containing well-defined clusters of various sizes. The synthetic generator designed by Handl and Knowles [1] was used to create five datasets, each one consisting of 1500 data items grouped into 3 clusters. The clusters in a data set are built iteratively based on covariance matrices which need to be symmetric and positive definite. Overlapping clusters are rejected and regenerated, until a valid set of clusters is found. The datasets are named as $n_1 - n_2 - n_3$ with n_p denoting the size of cluster p.

The size of the internal population was set to 10. The maximum size for the external population containing non-dominated solutions was set to 500 but in our experiments it did not exceed 250 elements. The number of iterations was set to 10000.

Figure 4 presents the set of non-dominated solutions returned in the last iteration of the genetic algorithm. The fitness values corresponding to the two criteria to be optimized are normalized in range $[0, 1]$ and are plotted as follows: the horizontal axis corresponds to Criterion (3) expressing how unbalanced the clusters are and the vertical axis corresponds to Criterion (4) expressing the average cut size. The solution closest to the real partition of the dataset is marked as a square; in this regard, the Adjusted Rand Index (see Hubert (1985)) is used to evaluate the quality of the partitions. The partition corresponding to the best/minimum score computed as sum between the two objectives is marked as a triangle.

The shape of the Pareto front plotted for datasets of various degrees of uniformity is an indicator of the interaction between the two objectives. Because both objectives are formulated for minimization, the desirable position of a clustering is towards the southwestern corner of the diagram. Experimental results show that the average cut size cannot be lowered indefinitely without severely affecting the balancing of the clusters. A gap is recorded for the criterion measuring the uniformity once the optimum solution (with regard to the true partition) is met. This gap is due to the dependency between the two objectives: the second criterion measuring the average cut size is built using both the cut size and the entropy (the first objective).

The best solution with regard to the real partitioning of the dataset is very close to the solution retrieved as a convex combination between the two objectives. In all cases the Adjusted Rand Index takes values higher than 0.95, which indicates a very close match to the real partition. Experimental results show that a convex combination between the two criteria is able to identify a near-optimum solution if the final set of non-dominated solutions is normalized within the same range for both objectives.

To highlight the advantages of our multi-objective approach over other graph clustering methods, the well-known recursive partitioning algorithm METIS [2] is used, which delivers only perfectly-balanced clusters, even though in practice this may not be the best solution from the point of view of the cut size.

Table 1 presents comparative results. The Adjusted Rand Index (ARI) is reported for the solutions corresponding to: 1) the partitioning with the highest ARI value, 2) the best partitioning under the convex combination (average) over the two criteria normalized in range [0,1] and 3) the best balanced partitioning from the non-dominated set of solutions delivered by the genetic algorithm, which corresponds to clusters of equal size. Also, the ARI is reported for the partition computed with METIS.

1. http://dbkgroup.org/handl/generators/generators.pdf
2. http://glaros.dtc.umn.edu/gkhome/

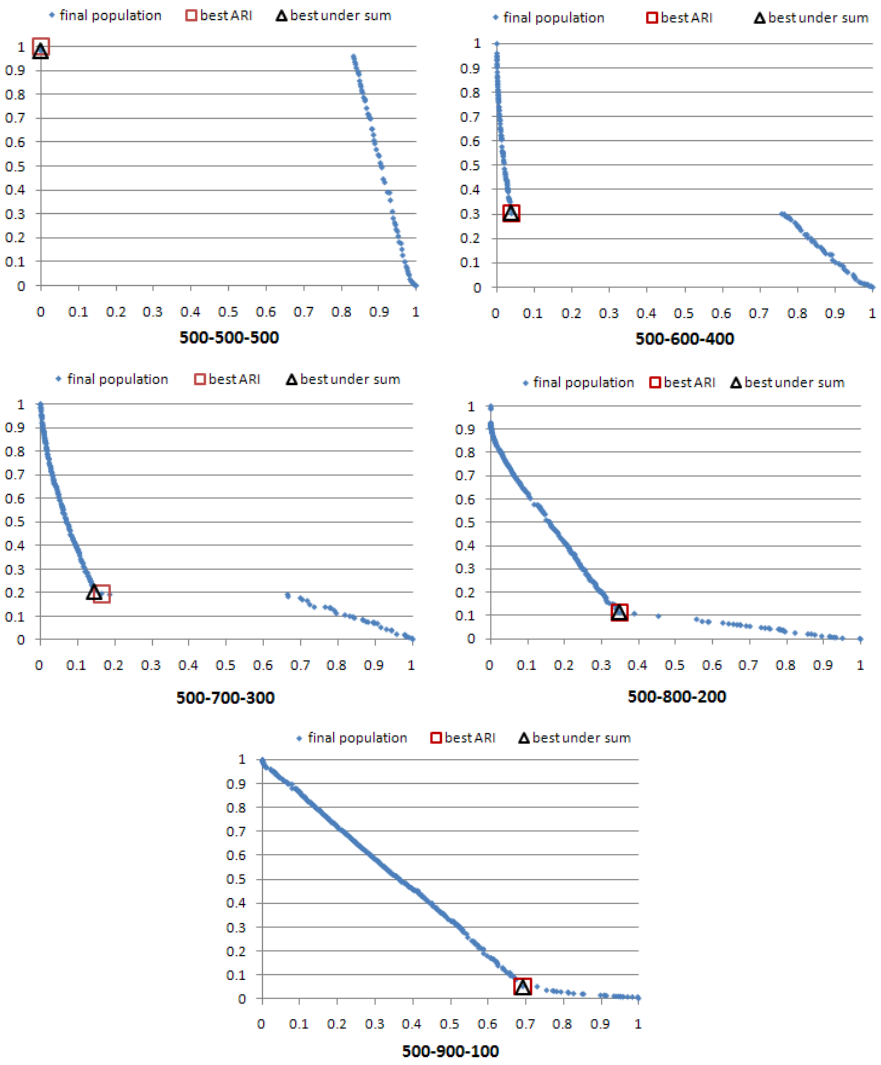

FIG. 2 – *The set of non-dominated solutions for various datasets. The horizontal axis corresponds to criterion 3 expressing how unbalanced the clusters are and the vertical axis corresponds to criterion 4 expressing the average cut size. The best match to the real partition is marked as a square. The partition corresponding to the minimum score computed as sum between the two objectives is shown as a triangle.*

Instance	best under ARI	best convex combination	best balanced	METIS
500-500-500	0.9999	0.9880	0.9999	0.9999
500-600-400	0.9909	0.9909	0.8111	0.8118
500-700-300	0.9625	0.9535	0.6588	0.6817
500-800-200	0.9839	0.9839	0.5764	0.5954
500-900-100	0.9950	0.9950	0.5615	0.5493

TAB. 1 – *Comparative Results*

Experimental results show that our algorithm is comparable with METIS with regard to the quality of the balanced partitioning. However, a near-optimal match with the true partitioning of the dataset can be extracted from the final the set of non-dominated solutions in a unsupervised manner, using a convex combination of the two criteria we use. Furthermore, this set can be explored to extract the most convenient solution for the problem being solved.

Also Figure 4 shows that the non-linear criterion $\Psi(X)$ given by Equality (4) biases the search towards highly balanced clusters and can be successfully used when a perfectly balanced partition is desired. Its convex combination with the criterion measuring the balancing degree of the partitioning is necessary to retrieve the true partitioning.

5 Concluding Remarks and Future Work

The current paper presents a multi-objective approach to the graph clustering problem. A novel criterion is proposed to measure cluster uniformity, based on the generalized entropy. A multi-objective genetic algorithm that returns a set of non-dominated solutions is used to study the interaction between the two criteria and to extract the optimum solution.

Future work will be conducted towards integrating a multi-level strategy within our approach in order to make it feasible for very large problems from VLSI design.

References

Alpert, C. J. (1998). The ispd98 circuit benchmark suite. In *Proc. ACM/IEEE International Symposium on Physical Design, April 98*, pp. 80–85.

Barnardand, S. T. and H. D. Simon (1993). A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. In *Proc. 6ts SIAM Conf. Parallel Processing for Scientific Computing*, pp. 711–718.

Corne, D. W., N. R. Jerram, J. D. Knowles, and M. J. Oates (2001). Apesa-ii: regionbased selection in evolutionary multiobjective optimization. In *Proc. Genetic and Evolutionary Computation Conference*, pp. 283–290.

Daróczy, Z. (1970). Generalized information functions. *Information and Control 16*, 36–51.

Fjällström, P.-O. (1998). Algorithms for graph partitioning: A survey. *Linköping Electronic Articles in Computer and Information Science 3*.

Handl, J. and J. Knowles (2005). Improving the scalability of multiobjective clustering. In *Proceedings of the Congress on Evolutionary Computation*, Volume 3, pp. 2372–2379.

Havrda, J. H. and F. Charvat (1967). Quantification methods of classification processes: Concepts of structural α-entropy. *Kybernetica 3*, 30–35.

Hubert, A. (1985). Comparing partitions. *Journal of Classification 2*, 193–198.

Karypis, G. and V. Kumar (1998). A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing 20*, 359–392.

Kernighan, B. W. and S. Lin (1970). An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal 49*(1), 291–307.

Krovi, R. (1991). Genetic algorithms for. clustering: A preliminary investigation. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, San Francisco, pp. 540–544. IEEE Computer Society Press.

Luchian, S., H. Luchian, and M. Petriuc (1994). Evolutionary automated classification. In *Proceedings of the First Congress on Evolutionary Computation*, pp. 585–588.

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review, Elsevier I*, 27–64.

Simon, H. D. (1991). Partitioning of unstructured problems for parallel processing. *Computing Systems in Engineering 2*, 135–148.

Simovici, D. A. and C. Djeraba (2008). *Mathematical Tools for Data Mining – Set Theory, Partial Orders, Combinatorics*. London: Springer-Verlag.

Simovici, D. A. and S. Jaroszewicz (2002). An axiomatization of partition entropy. *IEEE Transactions on Information Theory 48*, 2138–2142.

Résumé

Cet article traite le problème de classification à partir d'une matrice de similarité sur un ensemble de données. En représentant cette matrice comme un graphe pondéré nous transformons ce problème en un problème de classification/séparation qui vise à identifier des groupes de sommets fortement inter-connectés. Nous définissons deux critères distincts pour obtenir des clusters équilibrés et bien separés. Le premier critère minimise la similarité entre les objets appartenant à différents groupes et constitue un objectif généralement atteint en matière de regroupement. Le deuxième critère est formulé avec l'aide de l'entropie généralisée. Le compromis entre ces deux objectifs est exploré en utilisant un algorithme génétique multi-objectifs avec opérateurs renforcés. Comme les résultats expérimentaux le montrent, le front de Pareto offre une visualisation des compromis entre les deux objectifs.