A Genetic Graph-Based Clustering Algorithm

Héctor Menéndez and David Camacho

Departamento de Ingeniería Informática, Escuela Politécnica Superior,
Universidad Autónoma de Madrid,
C/Francisco Tomás y Valiente 11, 28049 Madrid, Spain
{hector.menendez,david.camacho}@uam.es
http://aida.ii.uam.es

Abstract. The interest in the analysis and study of clustering techniques have grown since the introduction of new algorithms based on the continuity of the data, where problems related to image segmentation and tracking, amongst others, makes difficult the correct classification of data into their appropriate groups, or clusters. Some new techniques, such as Spectral Clustering (SC), uses graph theory to generate the clusters through the spectrum of the graph created by a similarity function applied to the elements of the database. The approach taken by SC allows to handle the problem of data continuity though the graph representation. Based on this idea, this study uses genetic algorithms to select the groups using the same similarity graph built by the Spectral Clustering method. The main contribution is to create a new algorithm which improves the robustness of the Spectral Clustering algorithm reducing the dependency of the similarity metric parameters that currently affects to the performance of SC approaches. This algorithm, named Genetic Graph-based Clustering (GGC), has been tested with different synthetic and real-world datasets, the experimental results have been compared against classical clustering algorithms like K-Means, EM and SC.

Keywords: Machine Learning, Clustering, Spectral Clustering, Genetic Algorithms.

1 Introduction

The unsupervised learning methods are mainly based on clustering techniques [3]. These techniques were designed to find hidden information or features in a dataset grouping the data with similar properties in clusters. The different methods are divided in three main categories[7]: *partitional* (consists in a disjoint division of the data where each element belongs only to a single cluster); *overlapping* or non-exclusive (allows each element to belong to multiple clusters) and *hierarchical* (nests the clusters formed through a partitional clustering method creating bigger partitions and grouping the clusters by hierarchical levels).

This work is focused on the first category: partitional clustering which also has three main approximations[3]: *Parametric or Model-based clustering* (consists on an estimator based on a mixture of probabilities whose parameters are estimated, it fixes the model to the dataset); *Non-Parametric clustering* (there is not an initial probability model or estimator) and *Semiparametric clustering* (a combination of both methods).

This work is based on a well-known technique of Non-Parametric Partitional Clustering: Spectral Clustering (SC). It was introduced by Ng et al. in [10]. The algorithm is divided in three main steps:

1. A Similarity Function is applied to all the pairs of data elements to generate a Similarity Graph. There are three different kind of similarity graph: **the ϵ-neighbourhood graph** (all the components whose pairwise distance is smaller than ϵ are connected), **the k-nearest neighbour graph** (the vertex v_i is connected with vertex v_j if v_j is among the k-nearest neighbours of v_i) and **the fully connected graph** (all points with positive similarity are connected with each other).
2. The Laplacian Matrix (or Spectrum) of the Similarity Graph is extracted to study its eigenvectors. There are three different Laplacian matrices[12]. They define different versions of the SC algorithm: **Unnormalized SC** (the Laplacian matrix is: $L = D - W$), **Normalized SC** (the Laplacian matrix is: $L_{sym} = D^{-1/2}LD^{-1/2}$) and **Normalized SC related to Random Walks** (the Laplacian matrix is: $L_{rw} = D^{-1}L$).
3. Kmeans (or other partitional clustering technique) is applied to the matrix formed by the k-first eigenvectors to discriminate the information and assign the final clusters.

The main problem of the SC algorithm is the computation of the eigenvectors and eigenvalues of the Laplacian Matrix and the effect that they produce on the convergence of the algorithm. is how to compute the eigenvector and the eigenvalues of the Laplacian matrix of this similarity graph. The theoretical analysis of the convergence is justified using the perturbation theory [12], random walks and graph cut theory [12]. Some of the main problems of Spectral Clustering are related to the consistency of the two typical methods used in the analysis: normalized and un-normalized spectral clustering. A deep analysis about the theoretical effectiveness of normalized clustering over un-normalized was carry out by von Luxburg in [13].

Other problem of the Spectral Clustering algorithm is its sensitivity to the definition of the similarity function. It produces several problems when there is noisy information as Chang and Yeung exposed in [2]. Some solutions to this problem were based on the improvements of the parameters selection for the similarity function [2]. Other solutions are focused on the selection of the partitional clustering algorithm for the third step of SC [14]. This work develops a new algorithm based on Genetic Algorithms (GA) to improve the robustness of the clusters selection taking the similarity graph as a starting point.

Genetic Algorithms have been traditionally used in optimization problems. The complexity of the algorithm depends on the codification and the operations that are used to reproduce, cross, mutate and select the different individuals (chromosomes) of the population [4]. The algorithm applies a fitness function which guides the search to find the best individual of the population.

Different approximation of genetic codifications to the clustering problem were profound studied by Hruschka et al. in [7]. They show the different codifications, operations and fitness functions applied in several genetic algorithms to solved the clustering problem. Our previous work was also focused on resolve this problem using GA, but it was centred on overlapping clustering [1].

This work presents a Genetic Graph-based Clustering (GGC) algorithm which is inspired on the Spectral Clustering algorithm (it takes the same Similarity Graph as a starting point) and improves the robustness of the solution. The algorithm is experimentally compared with Spectral Clustering, Kmeans [9] and Expectation Maximization (EM) [9] to test its accuracy. The experimental study is also focused in a comparison between the robustness of the SC and GGC algorithms.

The rest of the work is structured as follows: Section 2 presents the Genetic Graph-based Clustering Algorithm; Section 3 shows the experimental results. Finally, Section 4 gives the conclusions and future work.

2 Genetic Graph-Based Clustering Algorithm (GGC)

This section explains the algorithm which has been implemented. It is mainly based on a simple Genetic Algorithm (GA). It is necessary to give a number of clusters initially. The algorithm begins with a Similarity Graph in the same manner that the Spectral Clustering algorithm. The population of the GGC algorithm is a set of possible solutions (partitions) which evolves until the best solution is found or the number of generations is ended. The fitness function is a quality measure for the solutions.

2.1 Codification and Genetic Operators

The codification is a simple vector-based numerical representation. Each individual is a n-dimensional vector (where n is the number of data instances) which has integer values between 1 and the number of clusters. They represent a cluster selection for the dataset. During the evolution process, the operators can create invalid individuals. These individuals represent solutions where one or more clusters have no elements. In this problem of partitional clustering these solutions are not valid because the number of clusters is initially given. To avoid the invalid individuals generation problem, they receive a 0 fitness value. The operators used in the GGC algorithm are the traditional ones extracted from the GA literature, they can be briefly summarized as follows:

- **Selection:** The selection process selects a subset of the best individuals. These chromosomes are reproduced and also pass to the next generation. It is called a $(\mu + \lambda)$ selection [4], where μ represents those chromosomes which are chosen, and λ the new chromosomes generated.
- **Reproduction:** The reproduction randomly selects two individuals (using the classical wheel algorithm [4]), and applies the crossover operation to the chosen chromosomes creating two new individuals.

Algorithm 1. Pseudo-code of the Fitness Function

```
Require: A $n$-vector of elements with values between 0 and $k$ where $k$ is the number
    of clusters and a variable $neighbours$ which represents the number of neighbours
    for the KNN measure.
Ensure: A value between 0 and 1 which corresponds with the fitness achieved.
 1: TotalKNN = 0.;
 2: TotalMC = 0.;
 3: Generate the set of $k$ Clusters: $C$.
 4: for all $C_a \in C$ do
 5:     if $C_a = \emptyset$ then
 6:         return 0
 7:     end if
 8:     SumKNN = 0; SumMC = 0.
 9:     for all $ind \in C_a$ do
10:         SumKNN += $PofKNN(neighbours, ind)$ {It calculates the percentage
            of neighbours for the individual $ind$ which are assigned to the same clus-
            ter.}
11:         SumMC += $AvEdWCut(ind)$ {It calculates the average value of the edge
            weights which have been cut from $ind$.}
12:     end for
13:     TotalKNN += SumKNN / $|C_a|$; {$|C_a|$ represents the number of elements of
        $C_a$.}
14:     TotalMC += SumMC / $|C_a|$;
15: end for
16: return $\frac{TotalKNN}{|C|} \times \left(1 - \frac{TotalMC}{|C|}\right)$
```

- **Crossover:** The main problem of the crossover operation is those individuals which have different numerical values but represents the same solution. These individuals need to be relabelled before the application of the operation. For this reason, a measure which compares the number of commons elements between the clusters is used to find the similarity degree of the chromosomes. After, one of the two chromosomes is relabelled trying to maximize the similarity. Finally, the crossover exchanges strings of numbers between the two chromosomes (both string have the same length).
- **Mutation:** The mutation randomly choose different chromosomes to change the values of some of their alleles. The new value is a random number between 1 and the number of clusters.

2.2 The GGC Fitness Function

The fitness function is a combination of the classical K-Nearest Neighbourhood (KNN) [9] algorithm and the Minimal Cut measure [11]. KNN assigns an element to a cluster if its neighbours are in the same cluster. It is useful to ensure the continuity condition that is common in the Spectral Clustering solutions. To control the separation between the elements of the clusters, the Minimal Cut measure is used. It guarantees that those elements which clearly belongs to different clusters are not assigned to the same cluster. The K value for KNN is initially given.

Algortihm 1 shows the pseudo-code of the fitness: KNN covers all the nodes and check if the K-closest elements are in the same cluster (lines 9 to 12). The fitness value of this metric is the mean of the percentage of well-classified neighbours of all the individuals in a cluster (lines 10 and 13). The Minimal Cut measure calculates the average value edge weights which have been removed (lines 11 and 14). The final value of the fitness if the product of the KNN metric and the subtraction between one and the Minimal Cut metric (line 16), both metrics have the same range: [0,1]. Therefore, the algorithm maximizes the value of $\frac{TotalKNN}{|C|} \times \left(1 - \frac{TotalMC}{|C|}\right)$ (line 16) where:

$$TotalMC = \sum_{x \in C} \frac{\sum_{y \notin C_x} w_{xy}}{|\{y | y \notin C_x\}|}$$

$$TotalKNN = \sum_{x \in C} \frac{|\{y | y \in \Gamma(x) \wedge y \in C_x\}|}{|\Gamma(x)|}$$

In these formulas, C represents the set of clusters and $\Gamma(x)$ represents the neighbourhood of the element x. It reduces the weight values of the edges which are cut and improve the proximity of the neighbours.

3 Experimental Results

This section shows the different experiments carried out to evaluate the behaviour of our approach. These experiments are both synthetic and real-world experiments. First, the experiments analyse the distance dependency problem of the Spectral Clustering algorithm compared with the GGC algorithm. This initial analysis shows the robustness of the algorithm to the metric parameters. Second, the GGC algorithm results are compared with other algorithms (Kmeans, EM and Spectral Clustering) using synthetic datasets. Finally, these algorithms are applied to real-world datasets, which are classified, to test their results.

The parameters of the GGC algorithm have been experimentally fixed. To find these values, 100 experiments have been executed using the synthetic data (see Section 3.2).the selected parameters are:

- Population: 200
- Generations: 2000
- Crossover probability: 0.3
- Mutation probability: 0.5
- Selection $(\mu + \lambda)$: The 50^{th} best individual are selected from the previous generation.
- K value of the KNN metric: 2

3.1 The Robustness of the GGC Algorithm

An important problem of the Spectral Clustering algorithm is its dependency to the parameters of the similarity function. The GGC algorithm has been designed to avoid this problem. The KNN metric which is applied in the fitness calculation provides a higher robustness to the algorithm compared to the Spectral Clustering algorithm, it does not depend of the order of magnitude of the distances calculated by the metric. Figure 1 shows a clear example. In this case, the Spectral Clustering algorithm (implemented in the "kernlab" package of CRAN [8]) is compared with the GGC algorithm. In the "kernlab" package, Karatzoglou et al. implements the Random Walks Normalized Spectral Clustering algorithm. They use the Gaussian RBF Kernel to set the similarity graph. It is defined by: $K_{ij} = e^{-\sigma ||x_i - x_j||^2}$, where K is the similarity matrix, x_i, x_j are data instances, and σ is the parameter which changes the order of magnitude. The experimental results show that the clustering technique clearly depends of the σ parameter. Figure 1 shows the different clustering results of the Spectral Clustering and the GGC algorithm modifying the σ parameter.

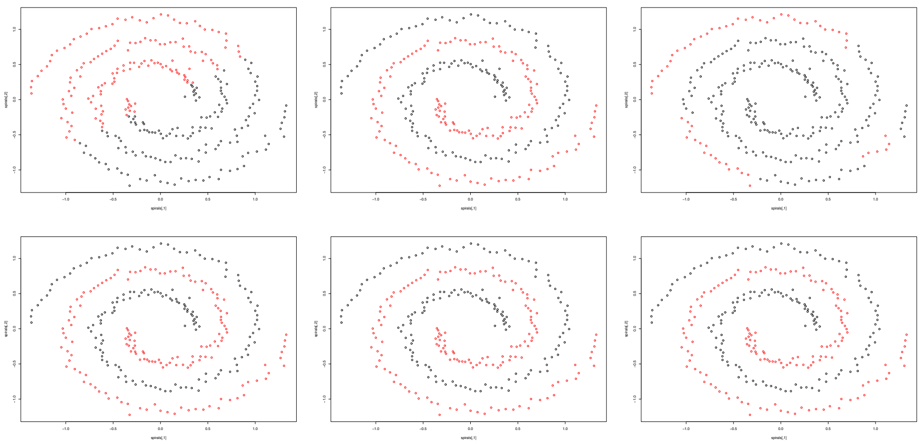


Fig. 1. Spectral Clustering and Genetic Algorithm results for the spirals[8] dataset with $\sigma = 2$, $\sigma = 500$, $\sigma = 2000$, respectively

These results show that the parameters used in the definition of the kernel are very important because these parameters defined the degree of the similarity. Ng et al. introduced a method to calculate the optimal σ in [10], however, as Figure 2 shows, this technique is not always enough. GGC obtains always the same results because it consider metrics which do not depend of the value of the distances, they depend of the order relation between the distances.

3.2 Experiments on Synthetic Data

In this section the different datasets which are used for the experimentation are explained and analysed. These datasets have been extracted from different clustering works which study the behaviour of the algorithms in difficult conditions.

Data Description. The GGC algorithm has been tested with different datasets. These datasets are 2-Dimensional data which can be separated by human intuition but are problematic for the classical clustering algorithms. The following datasets have been analysed:

– **Aggregation**[6]: This dataset is composed by 7 clusters, some of them can be separated by parametric clustering.
– **Compound**[15]: There are 6 clusters with are only separable by non-parametric methods (or special kernels if parametric clustering is applied).
– **Spiral**[2]: In this case, there are 3 spirals close to each other.

Results of Data Test. Figure 2 shows the classification results of the different datasets. Table 1 shows the best fitness values achieved by the GGC algorithm. In these cases the σ parameter to generate the similarity matrix of the Spectral Clustering and the GGC algorithms is 100 (it has been approximated using the method described by Ng et al. [10]). All the algorithms have been run 50 times and their best results have been selected. GGC and SC use the RBF kernel. EM and Kmeans use the Euclidean distance metric.

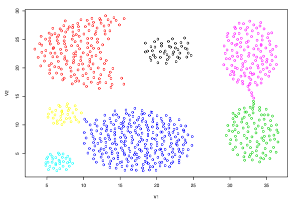
GGC and SC correctly classify Aggregation (GGC achieves a fitness value of 0.9928 which is the maximum value of fitness achieved by the algorithm; it is a consequence of those elements which could belong to two clusters) while EM and Kmeans have problems related to the form of the data. These problems could be a consequence of local minimum convergence for the centroids. Compound is impossible to classify with these parametric algorithms and the Euclidean distance. Also SC has problems related to the different distributions of the data. The GGC algorithm correctly classifies the Compound problem with a fitness value of 0.9552 (this value is also the maximum fitness achieved by the algorithm; in this case, there are elements assigned to different clusters which are closed to other clusters). Finally, Spirals classification is also impossible for the parametric methods while GGC and SC classify it correctly (in this case GGC achieves the maximum fitness).

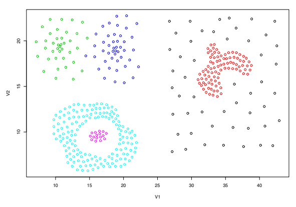
Table 1. Fitness values achieved by GGC (see Figure 2)

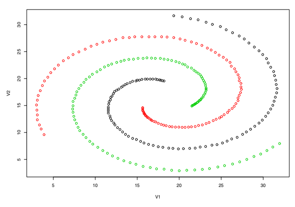
Dataset	Fitness achieved
Aggregation	0.9928
Compound	0.9552
Spiral	1.0

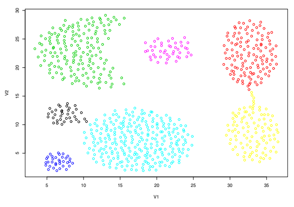
3.3 Experiments on UCI Datasets

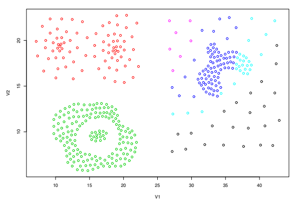
In this section the experiments are focused on real-world datasets which have been previously classified. Here, the accuracy of the algorithm is tested. In these databases the correct number of clusters is known. The measures used are the Euclidean Distance and the RBF kernel because they are the best known of the dissimilarity measures for these databases and they have been employed in previous works for all the methods used here. The GGC algorithm has been

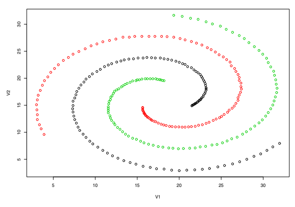

(a) Ideal Aggregation

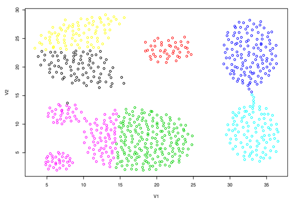

(b) Ideal Compound

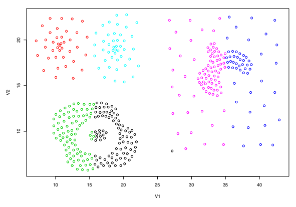

(c) Ideal Spirals

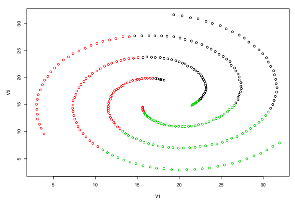

(d) SC Aggregation

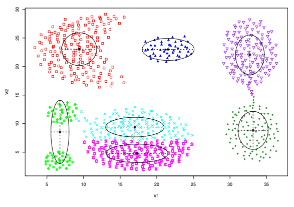

(e) SC Compound

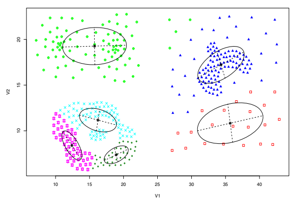

(f) SC Spirals

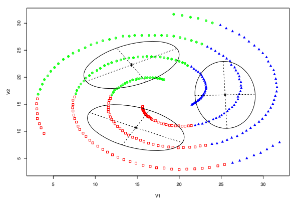

(g) Kmeans Aggregation

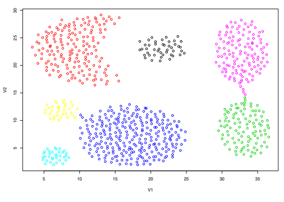

(h) Kmeans Compound

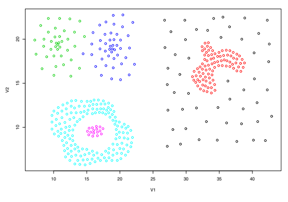

(i) Kmeans Spirals


(j) EM Aggregation


(k) EM Compound


(l) EM Spirals


(m) GGC Aggregation


(n) GGC Compound

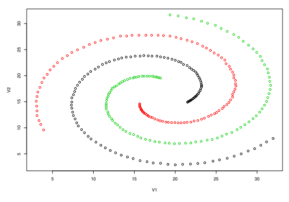

(o) GGC Spirals

Fig. 2. Each row represents the cluster selection of each algorithm for the synthetic datasets from left to right: "aggregation", "compound", "spiral". The rows represents from top to bottom: the ideal results, the Spectral Clustering results, the Kmeans results, the EM results and the GA results.

applied on 2 real and classified datasets (without missing values) extracted from the UCI Machine Learning Repository [5]:

- **Iris**: This dataset is a well-know dataset. It has 150 instance of 3 different classes (50 in each class). Each class refers to a type of iris plant. Each instance has 4 attributes.
- **Wine**: This dataset has 178 instance of 3 different classes (not balanced). Each class refers to a type of wine. Each instance has 13 attributes.

Results of the Data Test. The experiments have followed the same procedure that they followed in the synthetic datasets experiments. Also the value of σ has been approximated to 100. The results for the Iris show that EM is the best classifier (with an accuracy of the 96,67 %) and the GGC algorithm is the second (92%). The results for the Wine datasets shows that all the algorithm obtain high accuracy values (higher than the 95 %), and the GGC algorithm obtains a perfect classification with the maximum fitness value. These results are a consequence of the data distribution. Iris dataset has instances of different classes which are closed to each other, the GGC algorithm has problems to discriminate the boundary of the clusters specially when there are intersections between the clusters. The fitness value of the Iris is the higher that the algorithm has achieved, it shows that there are instance which belongs to different cluster but are closed to each other. In the case of the Wine dataset, the classes are clearer separated (as the different clustering techniques show). It improves the results of the GGC algorithm, because the boundary is clearer.

Table 2. Experimental results obtained using the UCI datasets

	Iris dataset	**Wine dataset**
Kmeans best classification	89.33%	95.50 %
EM best classification	96.67%	97.19%
Spectral Clustering best classification	89.33%	95.50%
GGC best classification	92% (Fitness=0.9946)	100% (Fitness=1)

4 Conclusions and Future Work

This work presents a new clustering method inspired by the Spectral Clustering algorithm and based on Genetic Algorithms. The GGC algorithm is defined using simple codification and operations. The main contribution of the algorithm is the fitness selection. GGC uses KNN and Minimum Cut measures. It is applied to the similarity graph which is generated in the first step of the Spectral Clustering algorithm. The combination of these measures improves the robustness of the algorithm giving a higher independence of the parameters of the similarity function. The results of Section 3 show that the new algorithm obtains good results for both synthetic and real-world datasets.

The future work will be focused on several improvements that could be made to the GGC algorithm. The effects of noisy information could be deeply analysed. The number of clusters could be automatically selected using strategies such as cross-validation. Finally, other fitness functions which could improve the convergence, and the clusters quality, of the GGC algorithm will be studied.

Acknowledgements. This work has been partly supported by: Spanish Ministry of Science and Education under project TIN2010-19872.

References

1. Bello, G., Menéndez, H., Camacho, D.: Using the Clustering Coefficient to Guide a Genetic-Based Communities Finding Algorithm. In: Yin, H., Wang, W., Rayward-Smith, V. (eds.) IDEAL 2011. LNCS, vol. 6936, pp. 160–169. Springer, Heidelberg (2011)
2. Chang, H., Yeung, D.-Y.: Robust path-based spectral clustering. Pattern Recogn. 41(1), 191–203 (2008)
3. Cios, K.J., Swiniarski, R.W., Pedrycz, W., Kurgan, L.A.: Unsupervised learning: Clustering. In: Data Mining, pp. 257–288. Springer, US (2007)
4. Coley. An Introduction to Genetic Algorithms for scientists and engineers. World Scientific Publishing (1999)
5. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
6. Gionis, A., Mannila, H., Tsaparas, P.: Clustering aggregation. ACM Trans. Knowl. Discov. Data 1(1) (March 2007)
7. Hruschka, E.R., Campello, R.J.G.B., Freitas, A.A., de Carvalho, A.C.P.L.F.: A survey of evolutionary algorithms for clustering. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 39(2), 133–155 (2009)
8. Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A.: kernlab – an S4 package for kernel methods in R. Journal of Statistical Software 11(9), 1–20 (2004)
9. Larose, D.T.: Discovering Knowledge in Data. John Wiley & Sons (2005)
10. Ng, A., Jordan, M., Weiss, Y.: On Spectral Clustering: Analysis and an algorithm. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, pp. 849–856. MIT Press (2001)
11. Schaeffer, S.E.: Graph clustering. Computer Science Review 1(1), 27–64 (2007)
12. von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)
13. von Luxburg, U., Belkin, M., Bousquet, O.: Consistency of spectral clustering. The Annals of Statistics 36(2), 555–586 (2008)
14. Wang, H., Chen, J., Guo, K.: A genetic spectral clustering algorithm. Journal of Computational Information Systems 7(9), 3245–3252 (2011)
15. Zahn, C.T.: Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Transactions on Computers C-20(1), 68–86 (1971)