УДК 57:015 + 573.2

New Clusterization Method Based on Graph Connectivity Search

Michael G. Sadovsky[*]
Eugene Yu. Bushmelev[†]
Institute of computational modelling SB RAS
Akademgorodok, 50/44, Krasnoyarsk, 660036
Russia
Anatoly N. Ostylovsky[‡]
Institute of Mathematics and Computer Science
Siberian Federal University
Svobodny, 79, Krasnoyarsk, 660041
Russia

Received 10.01.2017, received in revised form 30.03.2017, accepted 05.06.2017

New method is proposed to identify clusters in datasets. The method is based on a sequential elimination of the longest distances in dataset, so that the relevant graph looses some edges. The method stops when the graph becomes disconnected.

Keywords: order, complexity, clusterization, component, connectivity.
DOI: 10.17516/1997-1397-2017-10-4-443-449.

Introduction

Clusterization is a common approach to analyze the multidimensional data aiming to figure out an order and/or structuredness in the data. Tremendous growth of both bulky data, in various fields, and the computer capacities urge the development of new and advanced methods of clusterization. Clusterization as a technique has rather long story, and yields an explosive growth, currently. Number of methods to analyze data have been implemented, recently [1] (see also [2–6]); some of them are *ad hoc* techniques, some are quite general.

Clusterization plays an important role in analysis of bulky multidimensional data; another field of applications is knowledge retrieval. Informally, clusterization is a family of approaches (and relevant techniques) for presentation of multidimensional and quite complex data in compact form, through division of them in a reasonably small number of *clusters*; these latter are the groups of data objects close each other, within a cluster, and quite distinctly differing from those belonging to other clusters.

There is a number of methods to cluster data. Having no possibility to provide even a brief survey of them, we just point out some books and papers which might be used both as an introductory matter, and more prominent reading [1–6]; classic book *Algorithms for clustering data* [7] also should not be missed.

[*] msad@icm.krasn.ru
[†] eugene.bushmelev@gmail.com
[‡] hinayana@g-service.ru

Yet, there is no formal definition of clustering methodology feasible for all the data and all the cases; one has to figure out and fit a method (or set of methods) that seems to be the best for a specific goal of a research, as well as the structure of dataset. Implementation of a method may make another problem: some method might look very attractive, apparent and simple, while it request so much computational resources making senseless its implementation for any practical use; moreover, programme implementation might pose a point, itself. Meanwhile, we have to skip the discussion of that issue.

Basically, clustering methods could be divided into two major groups: the former are divisive techniques, and the latter are agglomerative ones. Here we present new method to cluster multidimensional data with quite specific structure: the objects must be the points in a metric space. The proposed method belongs to divisive methods family.

Let now introduce basic ideas and concepts. Let $\mathfrak{F}$ be the set of data points; $|\mathfrak{F}| = N$ is the capacity of the set, i. e. the total number of the points in it. Each point $\mathfrak{f}(i) \in \mathfrak{F}$, $1 \leqslant i \leqslant N$ is determined in some m-dimensional metric space $\mathbf{R}^m$. Vector $(f_1(i), f_2(i), \ldots, f_{m-1}(i), f_m(i))^{\mathbf{T}}$ is the coordinate vector of $\mathfrak{f}(i)$: $\mathfrak{f}(i) = \big(f_1(i), f_2(i), \ldots, f_{m-1}(i), f_m(i)\big)^{\mathbf{T}}$.

Next, s metrics $\rho\big(\mathfrak{f}(i), \mathfrak{f}(j)\big) \geqslant 0$ is defined, for any two points $\mathfrak{f}(i)$ and $\mathfrak{f}(j)$. The choice of metrics falls beyond the scope of the paper, and hereafter we use the standard Euclidean metrics

$$\rho\left(\mathfrak{f}(i), \mathfrak{f}(j)\right) = \sqrt{\sum_{k=1}^{m} \Big(f_k(i) - f_k(j)\Big)^2}\,. \qquad (1)$$

Finally, we stipulate that there are no gaps and/or lacunae in the data set; in other words, each data point is indeed a vector $\mathfrak{f}(i) = \big(f_1(i), f_2(i), \ldots, f_{m-1}(i), f_m(i)\big)^{\mathbf{T}}$, so that all the coordinates of each vector are known. Thus, we do not consider a problem of incomplete data analysis here and the ways to cluster such data. This problems falls beyond the scope of the paper and will be presented later.

1. The method

Basically, the idea of the method is rather apparent: let gather into a cluster the points that are located closed each other, than any other ones, within the dataset $\mathfrak{F}$. An implementation of the idea starts from the development of complete weighted graph $\mathbf{G}_N$ with data points being the vertices, and the segments connecting each couple of points being the edges. The length $\rho\big(\mathfrak{f}(i), \mathfrak{f}(j)\big)$ (see (1)) is the weight of the edge connecting i-th point to j-th one.

The procedure to separate (originally united) dataset into a set of cluster consists in step-by-step elimination from the (initially complete) graph representing dataset of the edges corresponding to the longest segments. Clusterization is stipulated to be completed, as soon as the graph losses connectivity; Fig. 1 illustrates the method. The idea to figure out clusters through the loss of the relevant graph connectivity was originally proposed in [12]; yet, they have used drastically other condition for edge elimination.

Following are the advantages of the method:

i) the method stops always;

ii) the method is unambiguous: any starting point yields the same final distribution of the points into the clusters;

iii) the method is free from a development of a representative, or similar additional constructions implementation.

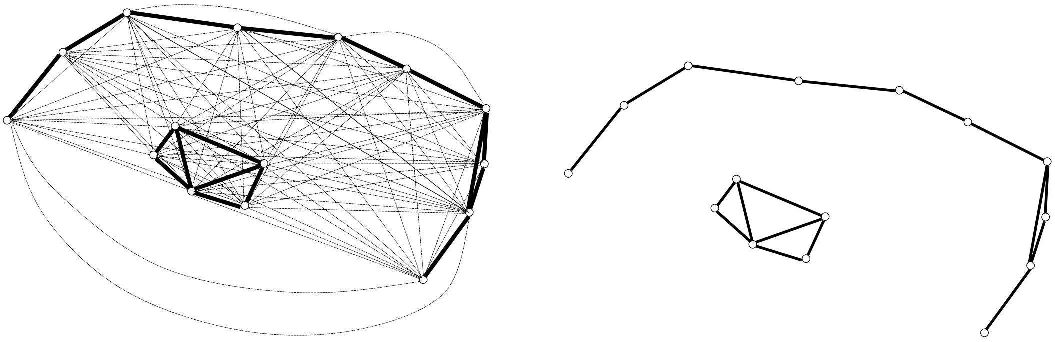

Fig. 1. An illustration of the new method to cluster the multidimensional data. Initially complete graph (shown left) transforms into the final clusterization (shown right). The edges that comprise two clusters are shown in bold solid lines, just to increase visibility

1.1. Graph connectivity examination

That is a common fact that any graph is equivalent to a $(0, 1)$ incidence matrix, and disconnected graph always could be presented with a block-diagonal matrix [8–10]. A brute force method implies that such $(0, 1)$ matrix requires a series of $N!$ permutations of rows and columns, where N is the matrix order (here $N = |\mathfrak{F}|$).

Fortunately, a connectivity (or disconnectedness) of a graph is unambiguously related to the question on the connected component search, within a graph. Indeed, any graph is unambiguously decomposed into a number of connected components; a connected graph has the unique connected component, and vice versa [8–10]. From computational point of view, connected components search is mush less costly procedure, in comparison to the block diagonalization. Hence, an examination for graph connectivity must be changed for the connected components search, in the graph.

There is a number of algorithms to seek for connected components, in a graph. Some of them look rather slow (e. g., a "lazy" algorithm makes about N^3 operations, where N is the number of vertices), others are faster. For example, the Tarjan procedure (see, e. g., [11]) has the complexity of $O(|E| + |V|)$, where $|E|$ is the number of edges, and $|V|$ is the number of vertices; thus, for the worst case (of a complete graph, to start with), one has $O(N^2)$ complexity, and that latter rapidly goes down, as the newly described clusterization algorithm proceeds. Hence, the heaviest problem is over.

Clusterization is equivalent to the connectivity loss of the (originally) complete graph. A step-by-step elimination of the longest edge requires an examination of the connectivity of the graph. Fortunately, there is no need for such examination followed by an edge elimination. Indeed, an elimination of an edge from a complete graph never results in a connectivity loss, unless $N > 2$.

Hence, for the worst case, one gets an isolated vertex (as a null-graph) and the subgraph with $(N - 1)$ vertices removing $N - 1$ edges to get the decomposition, and this is the necessary minimum of edges to be eliminated. At the next stage one must eliminate not less than $N - 2$ edges, and so no. So, at the k-th stage one must eliminate not less than $N - k$ edges, and so on. This observation yields a good estimation for the number k of examinations of connected components occurrence, during the clusterization development by the proposed method:

$$\frac{N(N-1)}{2} - (N-1) - (N-2) - \cdots - (N-k) = N - 1\,, \qquad (2)$$

where N is the vertices number in originally complete graph. Formula (2) could easily be transformed into

$$\frac{N(N-1)}{2} - kN + \frac{k(k+1)}{2} = N - 1\,. \tag{3}$$

For both formulae (2, 3) we assume that examination of connected components stops as soon, as a path is found. For the worst case ($k = N$) the number of examinations is N, not N^2 as it is necessary, if each eliminated edge is followed with the examination for connectivity.

So, a comprehensive description of the method converted into a chart looks like following:

Step 1. Calculate all pairwise distances between the points from the original dataset $\mathfrak{F}$.

Step 2. Develop the complete weighted graph corresponding to the dataset $\mathfrak{F}$ so that the points be vertices, and the distances be the edges.

Step 3. Remove $N-1$ edges corresponding to $N-1$ longest distances.

Step 4. Check the connectivity of the graph. Stop, if the graph is disconnected, otherwise.

Step 5. Remove $N-k$ edges corresponding to next $N-k$ longest distance, and go to Step 4.

Here k is the number of the cycle in this scheme, and the estimation (3) may not be improved.

Few words should be said towards the weighted graph used to implement the method. It is rather specific weighted graph, since the weights assigned to the edges yield a number of constraints: for any three vertices v_i, v_j and v_k, three edges incident to them $e_{i,j}$, $e_{i,k}$ and $e_{j,k}$ must meet the triangle inequality:

$$w(e_{i,j}) \leqslant w(e_{i,k}) + w(e_{k,j})\,, \tag{4}$$

where $w(e_{i,j})$ is the weight assigned to the edge $e_{i,j}$. This constraint significantly decreases a freedom in the choice of vertices and edges when developing a connected component.

2. Straps in the datasets

The problems arises from the method itself: there might be two (or more) clusters that are apparently identified by a researcher, while the connectivity remains intact, so that the method fails to discrete the dataset into clusters. An example of such data configuration is shown in Fig. 2., right. Indeed, the dataset shown in the left of Fig. 2. seems to consist of two cluster; a strap consisting of the points labeled in red in the right of the Fig. 2. joins two clusters in a single one.

There are few ways to address the point shown above. The first answer is that the method does not identify two clusters, and the dataset has no cluster structure. Indeed, the answer seems quite natural, since the key factor determining the clusterization is graph connectivity. In other words, if dataset looks like a field of points located in $\mathbf{R}^M$ in the manner allowing the attainability of any point via a set of edges connecting the points, under the constraint of the longest edge, then an absence of clusterization looks natural.

On the other hand, a glance at the pattern shown in Fig. 2. makes an existence of two clusters (these are a “ring” and a “ball”) rather obvious. An elimination of few points (shown in red, in the right part of Fig. 2.) changes the connected graph for a disconnected one, and two clusters become apparent. So, the point is how to find out such a strap in a dataset. Again, an interplay between metric and topological properties of the graph addresses this problem.

To do that, we shall develop all diameters of the graph. Here we leave the metric properties of the dataset, and change for topological ones. *Diameter* of a graph is the longest simple path found between a couple of two vertices [8–10]. Here the length is defined as a number of edges

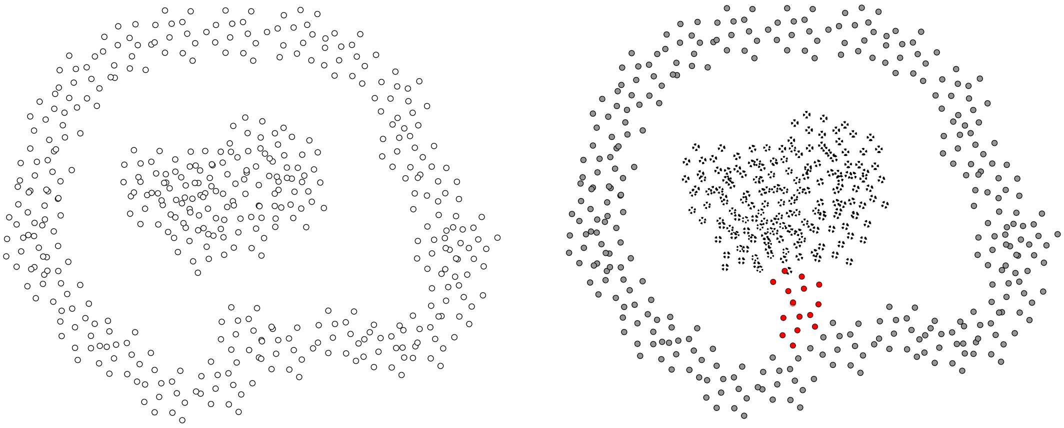

Fig. 2. An example of dataset configuration consisting of two clusters, while the connectivity remains intact, for given ρ^*

conjoining the given couple of vertices. Obviously, for the pattern shown in Fig. 2. the diameter goes through the vertices belonging to strap; indeed, any longest path found within the "ring" (for definiteness) is shorter than a path going through the strap to the "ball". Such algorithms have complexity of $O(N^2)$, thus making no problem for implementation.

Of course, the diameter itself yet does not answer the question on the identification of the vertices comprising the strap. Actually, to find out the strap, one has to identify the vertices that are incident to a huge number of (sufficiently long) paths. To do that, one can follow the way like that. First of all, randomly identify a subset $\widetilde{\mathfrak{F}}$ of vertices, in the graph suspicious for having a strap, so that $|\widetilde{\mathfrak{F}}| = \widetilde{N}$. The number $\widetilde{N}$ of the vertices should be $\sim \sqrt{N}$, where N is the capacity of $\mathfrak{F}$. Then all the shortest sequences must be developed, for each couple $(\widetilde{\mathfrak{f}}_i, \widetilde{\mathfrak{f}}_j)$ of the points from $\widetilde{\mathfrak{F}}$; $1 \leqslant i < j \leqslant \widetilde{N}$. Thus, there would be the ensemble of the sequences consisting of N elements. Finally, the number of occurrence of each vertex (to be found at the ensemble) must be counted. Obviously, the vertices comprising the strap would be counted with very high excess, in comparison to the subset of other ones. This increased occurrence number identifies the strap. Finally, one should eliminate these vertices (and incidental edges, as well) from the graph, and check the connectivity. If the graph becomes disconnected, then the cluster implementation is done.

Obviously, a dataset might be of severely complex structure; that one shown in Fig. 2. could be approximated with a manifold of rather simple topological character, i. e. a union of two manifolds of genus 0 and genus 1, respectively, or a manifold of the genus 1 (at the right of the figure). The point is that there could be more than one strap in the dataset. Growth of the genus character is not a sole problem in multidimensional data analysis; there might take place various loopings, etc. Nonetheless, the proposed method is able to treat such complicatedly organized datasets.

Conclusion

A new method to cluster multidimensional data is proposed. The method is based on the development of special complete weighter graph representing the dataset, with vertices being the points, and edges being the distances between all possible couples; the edge weight is the distance between two relevant points. Further, a sequential elimination of the heaviest (i. e. those corresponding to the longest distances) edges is carried out, till the connectivity isn't lost. Computationally hard problem of a graph connectivity search is by-passed with the connected components search that is equivalent, in this case.

The proposed method is divisive one: it always stops, and the final configuration of the connected components is always the same, with neither respect to the choice of initial point to do it. Basically, the method follows the strategy of the development of tools for approximation of multidimensional data by manifolds of low dimensionality [13].

References

[1] J.Leskovec, A.Rajaraman, J.DUllman, Mining of massive datasets, Cambridge Univ. Press, 2014.

[2] A.Fahad, N.Alshatri, Z.Tari, A.Alamri, I.Khalil, A.Y.Zomaya, S.Foufou, A.Bouras, A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis, *IEEE Trans. on emerging topics in computing*, **2**(2014), no. 3, 267–279.

[3] Dongkuan Xu, Yingjie Tian, A Comprehensive Survey of Clustering Algorithms. *Ann. Data. Sci.*, **2**(2015), no. 2, 165–193.

[4] M.Gavhale, P.D.Saraf, Survey on Algorithms for Efficient Cluster Formation and Cluster Head Selection in MANET, *Procedia Computer Science*, **78**(2016), 477–482.

[5] Ka-Chun Wong, A Short Survey on Data Clustering Algorithms, arXiv:1511.09123v1 [cs.DS], 2015.

[6] R.Xu, D.Wunsch II, Survey of Clustering Algorithms. *IEEE Trans. on neural networks*, **16**(2005), no. 3, 645–678.

[7] A.Jain, R.C.Dubes, Algorithms vor clustering data, 1988, Prentice-Hall, Inc. xiv.

[8] J.A.Bondy, U.S.R.Murty, Graph theory with applications, Elsevier, 2007.

[9] R.Diestel, Graph theory, Springer, 2000.

[10] O.Ore, Theory of graphs, AMS (3^{rd} ed.), 1962.

[11] G.W.Zobrist, J.V.Leonard, Progress in simulation, vol. **2**, Ablex Publ. Corp., New Jersey, 1994.

[12] M.Girvan, M.E.J.Newman, Community structure in social and biological networks. *PNAS*, **99**(2002), no. 12, 7821–7826.

[13] A.A.Akinduko, E.M.Mirkes, A.N.Gorban, SOM: Stochastic initialization versus principal components, Information Sciences, **364–365**(2016), 213–221.

Новый метод кластеризации на основе поиска связности графа

Михаил Г. Садовский
Евгений Ю. Бушмелёв
Институт вычислительного моделирования СО РАН
Академгородок, 50/44, Красноярск, 660036
Россия
Анатолий Н. Остыловский
Институт математики и фундаментальной информатики
Сибирский федеральный университет
Свободный, 79, Красноярск, 660041
Россия

Представлен новый метод кластеризации, основанный на последовательном исключении наиболее длинных ребер взвешенного графа, соответствующего распределению точек в пространстве. Кластеризация считается построенной, когда исходно полносвязный граф становится несвязным.

Ключевые слова: порядок, сложность, кластеризация, компонента, связность.