Multimodal Clustering with Evolutionary Algorithms

Mikhail Bogatyrev[1][0000-0001-8477-6006], Dmitry Orlov[1] and Tatyana Shestaka[1]

[1] Tula State University, 92 Lenin ave., Tula, Russia
okkambo@mail.ru

Abstract. The paper considers the application of evolutionary clustering algorithms on multidimensional formal contexts in order to solve fact extraction problem on database data. Formal contexts are built on the data that is the result of a database query. Clustering on such contexts allows one to find certain data combinations in clusters that can be interpreted as facts. Evolutionary clustering algorithm is chosen as an alternative to the FCA-based multimodal clustering algorithms. It is applied in solving the problem of phenotyping complications of myocardial infarction, formulated on the data of patient history, treatment methods and treatment results. The results of the work of evolutionary algorithms for their various parameters are presented.

Keywords: Evolutionary Computation, formal context, multimodal clustering fact extraction.

1 Introduction

A well-known problem in cluster analysis is the problem of interpreting results of clustering. Any clustering algorithm uses some proximity measure defined on the set of objects to be clustered. Therefore, the "meaning" of the resulting clusters is determined primarily by the used measure: the objects united into one cluster because we found them coinciding according to the chosen criterion. An attempt to interpret clusters from any other positions, for example, user-oriented, actually means switching to another proximity measure of the of objects, possibly more complex and not formalized.

Formal Concept Analysis (FCA) [3] offers a different approach to this problem. In FCA, clustering is used not on one, but on two, three and, in general, on an arbitrary number of sets, this is biclustering, triclustering and multimodal clustering [9]. Here, each cluster is a combination of data from clustered sets. The very fact of combining certain data in a cluster can be important from the user's point of view and carry new information. This interpretation of clusters is not directly related to the proximity measure of objects. The methods of multimodal clustering of FCA do not use a proximity measure in the traditional sense. Clustered objects are close if they are connected to each other by means of a relation that defines a formal context, and satisfy certain conditions of closure with respect to operators applied to elements of the data sets of formal context. This can be, for example, the Galois transformation which beget formal concepts or the prime and box operators which beget clusters [3, 13].

FCA-based methods of bi- and triclustering have been studied in sufficient detail [4, 7, 13]. There are also certain solutions for *n*-dimensional multimodal clustering [8, 9].

When applying FCA-based clustering methods to even not large data, a very large number of clusters is usually obtained, which makes it difficult to interpret them. In [18] and [19], a number of solutions to this problem have been proposed: algorithms for finding clusters of a given density and clusters with close values were developed, concept interestingness measures were introduced.

In this paper, it is proposed the solution of the problem of multimodal clustering of data of formal contexts created on the results of queries to databases. A query to an extensive database can return a large number of records with a large number of attributes. The representation of the query results in the form of a formal context with the subsequent finding clusters on it allows us to solve the problem of fact extraction from the database. Here, the facts are interpreted just as combinations of certain attributes in clusters. An evolutionary computation is used as a clustering method since it has a some advantages for applying in clustering, which are discussed below.

This work is a part of research aimed at modernizing the framework for evolutionary modelling [17] and applying it to new data structures.

The rest of the paper is organized as follows. We do not expound the known FCA statements, taking into account the topic of the workshop. In Section 2, there is brief description of evolutionary approach to multimodal clustering. In the Section 3, relations between evolutionary clustering and FCA are highlighted. The results of experimental study of proposed approach are presented in the Section 4. They are illustrated on the task of phenotyping of disease of myocardial infarction.

2 Evolutionary Approach to Multimodal Clustering

Evolutionary Approach to Multimodal Clustering is based on Evolutionary Computation [12]. Evolutionary computation is a term referring to several methods of global optimization, united by the fact that they all use the concept of the evolution of a set of solutions to an optimization problem, leading to solutions corresponding to the extreme value of some function that sets the optimization quality criterion. Evolutionary computation is effective when working with multimodal functions. If such a function has a global extremum, the evolutionary algorithm finds solutions corresponding to the range of values of the quality function that are sufficiently close to the that global extremum.

2.1 Principle of Evolutionary Computation

Let $\boldsymbol{X}$ is a set of solutions of a problem. Every solution $x \in \boldsymbol{X}$ can be characterized by a quality measure named as *fitness function* $f(x)$.

Let solutions of a problem depend on a set of parameters $\boldsymbol{P}$. Such a dependence may be very complex and not being expressed analytically. In this case, it is convenient to present the solution of the problem in the form of a black box. The black box inputs are the values of the parameters, and at the outputs we get the corresponding solutions, for which we calculate the values of the fitness function.

Most problems being solved by using Evolutionary Computation can be formulated as the following optimization problem: it is required to find optimal values of parameters p^*which deliver maximum value of fitness function, so the following is true:

$$p^* = \underset{p^* \in P}{argmax}\, f(x) \tag{1}$$

Parameter values indirectly determine the values of the fitness function calculated on the black box output, so in the expression (1) they were defined as arguments of fitness function.

Evolutionary approach to solving this problem consists in the following.

Building encoding scheme. *Encoding scheme* is the mapping φ: $P \rightarrow S$ where set S contains objects which *encode* parameters from P. Genetic algorithms, which are widely used in Evolutionary Computation often use binary encoding and every value of $p \in P$ is represented as binary string named as *chromosome*. Encoding scheme is not necessarily binary (as it is not binary in Nature): every string position contains a symbol (*gene*) from *encoding alphabet*, and there are variants of alphabets applied in encoding schemata [11]. However, necessarily there exists an inverse mapping φ^{-1}: $S \rightarrow P$, so for every $s \in S$ there exists $p \in P$.

Actually encoding is very important and represents the essence of evolutionary approach. There is *an atomic* principle of encoding which claims that encoding scheme has to be such that it generates minimal elements which influence on the values of elements of the set of solutions X. As in biology, heredity theory claims that gene (strictly gene combinations) is the minimal element which really determines individual characteristics, as here, in Evolutionary computation, atomic encoding principle plays the same role.

Evolutionary algorithm. For given encoding scheme, the following algorithm solves the problem (1).

A. Randomly generate an initial set (population) S_0 of objects from S.

B. Start *evolution* of the populations by applying a set of operators $\boldsymbol{A}$ to population $\boldsymbol{S}_0$ and further iteratively so that for every $\boldsymbol{S}_{k+1} = \boldsymbol{A}(\boldsymbol{S}_k)$ exists at least one

$$\boldsymbol{f}[\boldsymbol{\varphi}^{-1}(s_{k+1})] \geq \boldsymbol{f}[\boldsymbol{\varphi}^{-1}(s_k)], \tag{2}$$

where $s_k \in \boldsymbol{S}_k$ and $s_{k+1} \in \boldsymbol{S}_{k+1}$.

C. Finish the evolution of the population in accordance with the stopping criterion. Most often, the criterion for stopping is the immutability of the fitness function values over several steps of evolution.

If the set of operators $\boldsymbol{A}$ consists of genetic operators of *selection*, *mutation* and *recombination* (crossover) then evolutionary algorithm is named as *genetic algorithm* [11].

Selection works so that condition (2) is supported by the following "biological" principle: good parents produce good offspring (that is not true in Nature). Therefore, the higher fitness chromosomes have more opportunity to be selected than the lower ones and good solution is always alive in the next generation.

Crossover is the genetic operator that mixes two chromosomes together to form a new offspring. It does mixing by replacing fragments of chromosome's code divided in certain one or several randomly selected points.

Mutation involves modification of the gene values by randomly selecting new value from the alphabet at random point in the strings of genes.

Being realized, the algorithm (A. – C.) provides a fast and fairly accurate solution of the problem (1).

Fairly accurate means that evolutionary algorithm stops in a neighbourhood of global extreme of fitness function *f*. The size of a neighbourhood around extreme depends on the fitness function and parameters of genetic operators. When evolutionary algorithm works too fast it may stop at local extreme. This feature is traditionally considered as the lack of the algorithm but it may be useful for clustering since local extreme of quality measure may be semantically "better" than global extreme. In our experiments we have observed just that situation.

Operating speed of genetic algorithms could not be high because they have to manage not one but a whole set of possible solutions and evaluate fitness function N times on every step of evolution, where N is the size of population. Nevertheless, they are fast as compared to other algorithms for solving the problem (1) [12].

2.2 Evolutionary Multimodal Clustering

Evolutionary approach to clustering has quite a long history [10] and has contemporary applications [12]. Most applications belong to the field of gene expression analysis [2, 6, 12].

The gene expression data (GED) has been presented in two variants. These are either matrices containing expression values corresponding to various experiment conditions, or three-dimensional tensors, where a discrete time scale is used as the third dimension. Genetic algorithms with a full set of selection, mutation and crossover operators are used as evolutionary algorithms. The features of the genetic algorithms used here are determined by the choice of the chromosome encoding scheme, the fitness function and genetic operators.

In clustering, the encoding of chromosomes is the central problem on which the success of problem solution depends. Several encoding schemes have been proposed in this area [12]. Among them there are *binary encoding* and *integer encoding*. Some of them is shown on the Fig. 1 [10, 17]. In the binary encoding scheme for clustering, the length of chromosome may be very large if it corresponds to the number of clustering objects. Integer encoding is more compact but it is naturally redundant since different genotypes may correspond to the same clustering solution.

In [17] we have proposed the simple *chain-encoding scheme* also shown on the Fig. 1 which is not redundant and demonstrated its effectiveness in clustering when proximity measure is Euclidean. Another important advantage of the proposed encoding is that it provides quite fast work of the algorithm even on long chromosomes due to quasi parallelization of calculations: several genes in the chromosome may point to the same cluster simultaneously, so clusters are formed in a quasi parallel way. Below we also tested this encoding scheme in the current research for non-Euclidean proximity measure.

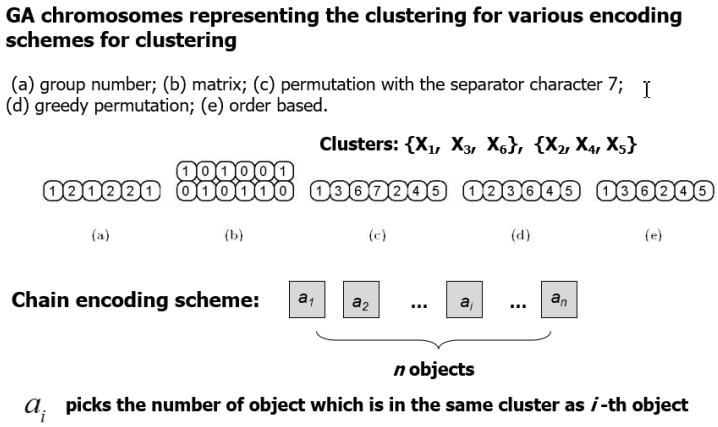


Fig. 1. Some encoding schemes for chromosomes in evolutionary algorithms for clustering.

As already mentioned, chromosomes in evolutionary clustering algorithm can have a significant length. The natural solution, which is known from practice [12, 15], is the use chromosomes with composite parts of which correspond to the data in the measurements. For GED, such chromosomes have two or three sections corresponding to genes and experiment conditions and third section corresponding to time stamps. Since number of genes in experiments may be over dozens of thousands the length of GED chromosomes may be giant. Nevertheless, the computational problem of processing very long chromosomes (usually binary) is solved now [15].

The application of genetic operators to such chromosomes has its own peculiarities. In the studying of gene expression, the genetic crossover and mutation operators are used in all sections of chromosomes. However, in other tasks, this is not justified, since the permutations of genes in certain places of the chromosomes contradict the meaning of the data and the atomic principle mentioned above.

The application crossover and mutation in each section of the chromosome entails large coverage of the search space and, possibly, fast convergence the algorithm to local extrema. However, there are other options for the implementation of operators, not necessarily covering all sections of the composite chromosomes. If, nevertheless, multisectional variants of genetic operators are used, parallelization of computations in accordance with the sectional arrangement of chromosomes is preferable.

In general, the most evolutionary clustering algorithms use fitness functions based on the distance between objects and either clusters' centroids or medoids [12].

If formal contexts are used as input data, then such proximity measures are not very effective, since instead of distances between objects and clusters (when centroids and medoids are used), the quality of clustering is characterized by such cluster' parameters as cluster density and volume of clusters.

All these and some other conditions were taken into account by when we modified our evolutionary modeling environment [17], which we used in this study.

3 Evolutionary Clustering and FCA

In the FCA, the application of Evolutionary Computation may be realized in two ways.

In the first way, Evolutionary Computation is used in FCA algorithms for constructing formal concepts, bi- and triclustering and for clustering of higher orders. Hybrid algorithms on the basis of existing FCA ones are created, and certain parameters of them are manipulated using Evolutionary Computation. The second way is creation of evolutionary algorithms for processing formal contexts as an alternative to existing FCA algorithms.

Our work relates mainly to the second way. However, we use OAC-triclustering [22] when processing formal contexts, so formally our approach partially belongs to the first way too.

A multidimensional, *n*-ary formal context is defined by a relation $R \subseteq D_1 \times D_2 \times ... \times D_n$ on data domains $D_1, D_2, ..., D_n$. The context is an $n+1$ set:

$$\mathbb{K} = <K_1, K_2, ..., K_n, R> \tag{3}$$

where $K_i \subseteq D_i$. The data from domains $D_1, D_2, ..., D_n$ is placed in a database and K_i may be treated as results of queries to database. Using queries, we can form formal contexts of certain dimension and content.

According to multimodal clustering, for any dimension of formal context, the purpose of its processing is to find *n* - sets which have the closure property [9]:

$$\forall u = (x_1, x_2, ..., x_n) \in X_1, X_2, ..., X_n, u \in R, \tag{4}$$

$\forall j = 1,2,...,n, \forall x_j \in D_j \setminus X_j < X_1,...,X_j \cup \{x_j\}, ..., X_n >$ does not satisfy (4). The sets $H = < X_1, X_2, ..., X_n >$ constitute *multimodal clusters*. The multimodal *n*-adic concepts of an *n*-dimensional context (3) are exactly the maximal *n*-tuples with respect to component-wise set inclusion [8].

Two circumstances are important when applying multidimensional contexts to data analysis. The first is that not only formal concepts, but also multidimensional clusters are important for knowledge discovering from data. Multidimensional clusters are characterized by density, and formal concepts are absolutely dense clusters [7, 14].

The second feature is important for the interpretation of clusters. Each cluster is a combination of subsets of data from different domains. The very fact of combining certain data with each other can be of interest. It is this version of fact extraction that we use in this work. However, the higher the dimension of the cluster, the less certain is the information that is represented by the combination of data, and additional analysis of the clusters is required to refine it. Therefore, high-dimensional clusters are not built and mainly three-dimensional formal contexts are the subject of research here [14, 22, 24]. For simplicity, we also illustrate our approach with three-dimensional formal contexts.

The three-dimensional triadic context (tricontext) of the form $\mathbb{K} = (K_1, K_2, K_3, I)$, where $I \subseteq K_1 \times K_2 \times K_3$ is a ternary relation and in general $I \not\approx R$ after querying to

database. Traditionally, subsets K_1, K_2, K_3 are interpreted as *objects*, *attributes* and *conditions* (OAC), which have a specific meaning based on the content of the database.

The kind of triclusters as OAC-triclusters are studied in detail and demonstrated effectiveness in applications [22]. For triadic context $\mathbb{K} = (K_1, K_2, K_3, I)$ OAC-tricluster is defined in the form

$$\mathbb{C} = (X, Y, Z), X \subseteq K_1, Y \subseteq K_2, Z \subseteq K_3 \tag{5}$$

OAC-triclusters are characterized by cluster density [14], the presence of similar values in clusters [18], and interestingness measures [19].

We use cluster density, and volume of a cluster in our evolutionary clustering algorithm. The cluster density is defined as

$$d(\mathbb{C}) = \frac{|I \cap (X \times Y \times Z)|}{|X| \times |Y| \times |Z|}, \tag{6}$$

and volume of a cluster has the following form

$$\mathrm{v}(\mathbb{C}) = |X| \times |Y| \times |Z| \tag{7}$$

3.1 Genetic Clustering Algorithm

Algorithm 1 shown below is genetic algorithm, which realizes evolutionary algorithm A-C from the Section 2.1. Its chromosomes *chrom* may be encoded by two variants.

The first variant is classical one when processing GED. For three-dimensional formal context of GED, there are three sections in chromosomes, for example, “genes”, “conditions” and “time stamps”. Crossover and mutation operate in all three sections to maximize search space coverage [11]. However, among the new chromosomes generated in this way, there may be incorrect chromosomes, which do not correspond to the data in the original tensor. The *doMultipleCrossover* function accesses the original tensor in order to filter out the wrong chromosomes.

In the second variant of encoding, chromosomes have only one section containing positions of objects being clustering. Filtering out the wrong chromosomes is not needed but the algorithm may produce not proper solutions corresponded to local optima of fitness function. This variant of encoding is convenient for implementing FCA operators for clustering. Population of chromosomes represents variants of clustering and chromosomes contain only references to objects, so the corresponding them clusters can be constructed, for example, using OAC operators [22].

doSelection function realizes selection chromosomes according to selection method. There are *proportional*, *random universal*, *tournament* and *truncation* selection methods [11] realized in the algorithm.

The stopping criterion is that the fitness function of the population does not change with a certain accuracy over several steps of evolution. It may fail, and then the algorithm executes the specified maximum number of steps.

Algorithm 1 Genetic clustering algorithm

Input: *tensor* is multidimensional context as the set of *n* samples on the axes of measurements;
Parameters:
sizePop is the size of population of chromosomes;
numpoints is the number of points of crossover;
mutationRate is the probability of mutation;
coefDensity is the cluster density-scaling factor;
coefSize is the cluster volume-scaling factor;
limitPop is the maximal number of populations;
sel is the type of selection;
countPop is the number of steps of evolution;
popFitness is the value of the fitness function for the entire population.
Output: *clusters* is the set of clusters in the form of a set of subsets.

population ⟵ *createPopulation*[*tensor, sizePop*] creating a population of chromosomes *chrom*

1: **while** *countPop* ≤ *limitPop* **do**
2: | **while** *stopping*[*population*] is false **do**
3: | | **for all** *chrom* **do**
4: | | | *clusterDensity*[*chrom, tensor*]
5: | | | *clusterVolume*[*chrom, tensor*]
6: | | | *fitnessFunction*[*chrom, tensor*, *coefDensity*, *coefSize*]
7: | | **end**
8: | | *popFitness*[population] calculating the value of the fitness function for the entire population.
9: | | *doMultipleCrossover*[{*chrom*1, *chrom*2}, *numpoints*, *tensor*]
10: | | *doMutation*[*chrom*, *mutationRate, tensor*]
11: | | *doSelection*[*chrom*, *popFitness*, *sel*]
12: | | **for all** *chrom* **do**
13: | | | {clusters} ⟵ *getSubTensorChrom*[*chrom, tensor*] forming clusters from tensor
14: | | **end**
15: | **end**
16: **end**

The purpose of other functions of the algorithm is clear from their names.

4 Experimental Study

Experimental study of the proposed approach was carried out in order to solve some tasks related to the problem of phenotyping diseases. Disease phenotyping refers to the determination of the form of the disease based on the clinical profile. A clinical profile

is a cluster that can include various data describing both the disease itself and the methods of its treatment, as well as the conditions of patients and sometimes the treatment results.

Our goal was also to study the efficiency and performance of the evolutionary clustering algorithm for its various parameters.

4.1 Data Sets

Usually the whole data about patients and disease is stored in clinical database and the data sets for the study can be obtained by performing queries to the database.
Depending on the database model and the DBMS platform, queries may be intellectual of some degree and DBMS generates corresponding complicated results in the output. However, relational databases and the SQL query language are still commonly used here. Standard results of the queries are tables, the fields of which contain data corresponding to the attributes of patients, diseases and treatment methods. Such tables constitute as binary as multi-valued formal contexts. Solving the clustering problem on such contexts, we get combinations of objects and attributes in clusters, which are further analyzed as sources of facts. A contexts of dimensions greater than two can be built on the results of queries, if we are interested in additional attributes retrieved from the database.

We use Myocardial Infarction Complications Data Set [16] for experiments. It contains information about 1700 patients having disease of myocardial infarction. All patients are anonymous and presented with identification numbers (ID). We use seven formal contexts acquired from the whole set which number of objects and attributes are shown in Table 1 where ECG is electrocardiogram. Among attributes, there are ones about patients (ID only), their anamnesis, their treatment methods, and complications after the treatment. An attribute may be binary or has a value as natural or real number.

Table 1. Number of objects and attributes of formal contexts

Context	Objects	Attributes
Anamnesis	1700	33
Therapy	1700	24
Analyzes	1700	19
Infarct	1700	6
ECG	1700	27
Therapy results	1700	14
Full data	1700	123

Some formal contexts such as Therapy, Analyzes and Therapy results have a third dimension in the form of days. The standard maximum treatment time for myocardial infarction is 21 days (in Russia), which defines the scale of the third dimension.

4.2 Evolutionary Clustering

Evolutionary clustering was performed using variants of genetic Algorithm 1 with two different encoding schemes and various types of crossover.

Chromosome encoding. After analyzing the existing variants of chromosome encoding [12], we settled on two of them. The first variant is our chained integer-encoding scheme [17] showed on Fig. 1.

The second encoding scheme is a binary scheme organized according to the principle of "one chromosome – one cluster". It has one, two or three sections in chromosomes according with the variant of encoding (see Section 3.1) and dimension of a context. Chromosomes for three-dimensional contexts have sections "patients", "attrbutes" and "days". In the sections, a number of gene is the number of patient, number of attribute from corresponding context from the Table 1 or number of a day according with objects order in the corresponding subsets in formal tricontext. Different chromosomes form different clusters. Because of the evolution of many such chromosomes, really k different chromosomes from n members of the population should remain. In this case, it turns out that some objects will be included in different clusters, i.e. there will be an intersection of clusters.

Fitness function. As in FCA, we control cluster density (6), its volume (7) and special kind of interestingness. There is the trade-off problem between the density and the volume of triclusters [7]. Depending on the data, density and volume may be contradictory characteristics of clusters. Myocardial infarction data are sparse, and if we collect enough units in a cluster, it will be simultaneously voluminous. Therefore, we do not use the volume of clusters in the fitness function, but only use their density. Nevertheless we calculate cluster volumes during evolution.

For the binary encoding scheme, fitness function has the form:

$$f(d) = \frac{1}{N}\sum_{i=1}^{N} \alpha_i d(C_i), \tag{8}$$

where α_i is user defined coefficient, which in general depends on cluster density, N is the number of chromosomes in population which is equal to the maximal number of clusters.

For the chained integer-encoding scheme fitness function is the following:

$$f(d) = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{K_j}\sum_{i=1}^{K_j} \alpha_i d(C_i) \tag{9}$$

where K_j is the number of clusters in the j-th chromosome.

Interestingness of a cluster. It is known in clustering analysis that "*the criteria relate quite indirectly to the major goal of clustering which is improving of our understanding of the world*" [21]. According to the fitness of the chromosomes, the whole fitness of population is namely such criterion. It hides the features of individual chromosomes. But if selection leaves chromosomes with maximum fitness, then there is a chance that they will lead evolution to good solutions. Patient ID values found in clusters, other

attributes corresponding to them from the “treatment” and “treatment outcomes” domains are evaluated for the presence of information in them that can be treated as facts. The formal criteria for selecting such “interesting” clusters are:

- the presence of a single cluster at the end of evolution;
- the most dense clusters among the received;
- clusters of the maximum volume among received;
- clusters with given values of density and volume.

4.3 Fact Extraction with Clustering

The most serious complication of a myocardial infarction is a lethal outcome of the disease. In our data set, the lethal outcome is set by the attribute LET_IS, which has following 8 values: 0: unknown (alive), 1: cardiogenic shock, 2: pulmonary edema, 3: myocardial rupture, 4: progress of congestive heart failure, 5: thromboembolism, 6: asystole, 7: ventricular fibrillation. We selected attributes related to the treatment of patients to find out whether the treatment affects the lethal outcome. For this purpose, the formal context was constructed, containing 27 attributes and 110 objects as the numbers of patients who had a lethal outcome of any of the 7 variants. A similar formal context was also constructed for patients who did not have a lethal outcome. It contains 1590 objects. We have added a third dimension to these contexts, reflecting the use of drugs on certain days. For example, a point with coordinates (7, NA_R_1_n, 1) corresponds to a unit, which means that the patient number 7 got opioid drugs at the first day of hospitality.

To solve the task of the effect of patient treatment on the lethal outcome, triclustering was performed in both contexts using an evolutionary algorithm.

Facts Extracted. We were interested in special clusters. First of all, these are clusters with large groups of patients characterized by certain combinations of attributes from the domains "patient", "treatment", "treatment results". Several such groups were obtained.

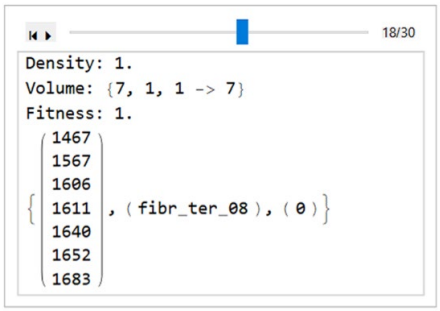
1. We have found that the lethal outcome of myocardial infarction is inherent in elderly patients over 60 years of age. This fact is consistent with the known data of cardiology.
2. In more detail, cases of heart attack in the anamnesis correlate with a fatal outcome, which also looks natural.

For both this groups of patients, we found absolutely dense clusters built on tensors with age and anamnesis attributes.

Unexpected result. We have found one unexpected result, which is as follows. On the data of myocardial infarction, there are stable (not changing according with different parameters of the genetic algorithm) and rather dense clusters in which a subgroup of patients with a lethal outcome have not got certain drugs. At the same time, patients with a non-lethal outcome had these drugs.

Comparison with Data-Peeler. We were also interested in absolutely dense clusters, the formal concepts. As expected, there were few such clusters, which follows

from the sparsity of the data. One of them is shown in Fig. 2. In it, we see that 7 patients had no fibrinolytic therapy by Streptodecase (attribute fibr_ter_08) what is confirmed by the query to the database.



Select 1 Results (MIC) - ISQL Session 1

Id	Age	Sex	Fibr Ter 08
1467	61	0	0
1567	67	0	0
1606	60	0	0
1611	55	1	0
1640	57	0	0
1652	68	1	0
1683	61	0	0

Fig. 2. The dense cluster and the query result.

To compare our results with well known another algorithm, we selected Data-Peeler [9] and modernized its code [23] by adding graphical user interface. Comparison of the results is shown in Table 2.

Table 2. Clustering results compared with Data-Peeler

Formal context	Number of clusters	Number of dense clusters	Number of Data-Peeler concepts
Anamnesis	30	14	449639
Therapy	30	19	28599
Analyzes	30	17	162
Infarct	30	20	65
ECG	30	10	689011
Therapy results	30	12	7798
Full Data	30	4	12564890

The results in the last row of Table 2 can be explained by the high sparsity of data in this formal context. Accordingly, the Data-Peeler algorithm has built a lot of small concepts.

4.4 Algorithm Performance.

The results of the algorithm performance study are as follows.

1. The algorithm processes very long three-section chromosomes of about 2000 genes fairly quickly. This allowed us to perform experiments in a wide range of changes in the parameters of the algorithm. Fig. 3 shows clustering execution time for each of

the seven contexts. On the Fig. 3-a it is shown for two-dimensional formal contexts and on the Fig. 3-b it is shown for three-dimensional formal contexts. At the same time, in some contexts, the third dimension was introduced artificially.

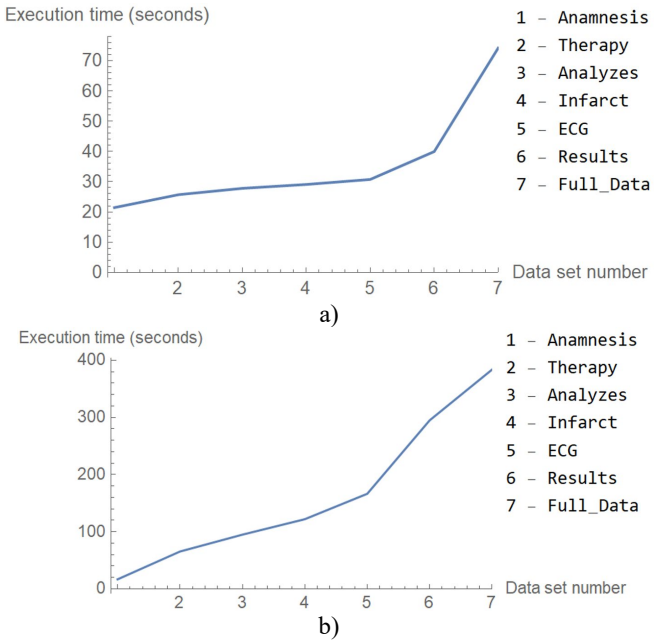


Fig. 3. Clustering execution time for several formal contexts.

The executions were performed on a standard PC 3.59 GHz with 4 Core-Processors and 8 GB RAM.

2. Encoding "one chromosome – one cluster" was more effective than chain encoding on a non-Euclidean fitness functions (6), (7) combining the density and volume of clusters. Since the chain encoding is more complex and multi-linked, the execution of crossover operators on chromosomes led to the "mixing"of genes, the appearance of many "incorrect" chromosomes, and as a result, a decrease in performance.

3. A multipoint crossover is more efficient than a single-point crossover. The use of multipoint crossover in all three sections of chromosomes accelerated the convergence of the algorithm and was effective namely on the encoding scheme "one chromosome – one cluster".

5 Conclusion and Future Work

This paper proposes an approach to multimodal clustering on multidimensional formal contexts using evolutionary computation. This approach is effective in experiments on clustering three-dimensional formal contexts based on data of patients with myocardial

infarction. The genetic algorithm builds dense clusters in any case, even for a local extrema of the fitness function.

The presented experimental results reflect the initial stage of research in this area. In the future, it is planned to do the following.

1. Evaluate the informativeness of the obtained clusters not manually, but using a user interface focused on doctors.

2. Experiments have confirmed that the criteria of cluster density and volume contradict each other. Therefore, it is necessary to apply multi-objective evolutionary clustering with appropriate algorithms.

3. Transition to the dimension of formal contexts greater than three. Separate groups of parameters can be represented as dimensions. Then their combinations obtained in clusters will reflect in more detail the relationships in heterogeneous data.

Acknowledgments. We thank anonymous reviewers for their remarkы and advices. The reported study was funded by Russian Foundation of Basic Research, the research project № 19-07-01178 and RFBR and Tula Region according to research project № 19-47-710007.

References

1. Hartigan J A. Direct clustering of a data matrix. Journal of the American statistical association, 67(337): 123—129 (1972)
2. Madeira SC, Oliveira AL. Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Trans. Comput. Biol. Bioinform. Jan-Mar;1(1):24-45. (2004) DOI: 10.1109/TCBB.2004.2.
3. Ganter, Bernhard; Stumme, Gerd; Wille, Rudolf, eds., Formal Concept Analysis: Foundations and Applications, Lecture Notes in Artificial Intelligence, No. 3626, Springer-Verlag. Berlin (2005) DOI:10.1007/978-3-540-31881-1
4. Kaytoue, Mehdi, Kuznetsov, Sergei, Napoli, Amedeo. Biclustering Numerical Data in Formal Concept Analysis. 135-150. (2011). DOI: 10.1007/978-3-642-20514-9_12.
5. Ignatov D. I., Kuznetsov S. O., Zhukov L. E., Poelmans J., Can triconcepts become triclusters? // International Journal of General Systems, Vol. 42. No. 6 (2013)
6. Henriques R., Madeira S. Triclustering Algorithms for Three-Dimensional Data Analysis: A Comprehensive Survey. ACM Comput. Surv. V. 51. № 5. P. 1–43. (2019) DOI: 10.1145/3195833.
7. Dmitry V. Gnatyshak, Dmitry I. Ignatov, Sergei O. Kuznetsov, From Triadic FCA to Triclustering: Experimental Comparison of Some Triclustering Algorithms. In: Proceedings of the Tenth International Conference on Concept Lattices and Their Applications (CLA'2013), La Rochelle: Laboratory L3i, University of La Rochelle, pp. 249-260, (2013)
8. Voutsadakis, G. Polyadic concept analysis. – Order. Vol. 19 (3). Pp. 295–304 (2002)
9. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed Patterns Meet N-ary Relations. In: ACM Trans. Knowl. Discov. Data. 3, 1, Article 3, 36 p. (2009)
10. R. M. Cole, Clustering with Genetic Algorithms, MSc Thesis, University of Western Australia, Australia (1998)
11. Goldberg D.E. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, Reading, MA, USA (1989)
12. Hruschka E., Campello R., Freitas A., de Carballo A. A Survey of Evolutionary Algorithms for Clustering. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE

Transactions on Evolutionary Computation. V. 39. P. 133–155. (2009) DOI: 10.1109/TSMCC.2008.2007252.
13. S.O. Kuznetsov and S.A. Obiedkov, Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 14, no. 2-3, pp. 189-216, 2002.
14. Ignatov D. I., Gnatyshak D. V., Sergei O. Kuznetsov, Boris G. Mirkin, Triadic Formal Concept Analysis and triclustering: searching for optimal patterns. In: Machine Learning, April, 2015, pp. 1-32.
15. Ma P., Chan K., Yao X., Chiu D. An evolutionary clustering algorithm for gene expression microarray data analysis. IEEE Transactions on Evolutionary Computation. V. 10. P. 296–314 (2006) doi: 10.1109/TEVC.2005.859371.
16. Myocardial infarction complications Data Set. http://archive.ics.uci.edu/ml/machine-learning-databases/00579/
17. M. Y. Bogatyrev, A. P. Terekhov. Framework for Evolutionary Modelling in Text Mining. - Proceedings of the SENSE'09 - Conceptual Structures for Extracting Natural Language Semantics. Workshop at 17th International Conference on Conceptual Structures (ICCS'09), p.p. 26-37 (2009)
18. Mehdi Kaytoue, Sergei O. Kuznetsov, Juraj Macko, Wagner Meira Jr., Amedeo Napoli, Mining Biclusters of Similar Values with Triadic Concept Analysis. In: Proc. 8th International Conference on Concept Lattices and Their Applications (CLA 2011), INRIA Nancy - Grand Est and LORIA, pp. 175 - 190, 2011.
19. Kuznetsov S., Makhalova T. Concept interestingness measures: a comparative study, in: Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications Clermont-Ferrand, France, October 13-16, 2015 Vol. 1466. Clermont-Ferrand : CEUR Workshop Proceedings. P. 59-72 (2015)
20. Mirkin, B. G., & Kramarenko, A. V. Approximate bicluster and tricluster boxes in the analysis of binary data. In Rough sets, fuzzy sets, data mining and granular computing, LNCS, Vol. 6743, pp. 248–256. (2011)
21. Mirkin, Boris, Muchnik, Ilya. Combinatoral Optimization in Clustering. Handbook of Combinatorial Optimization. D.-Z. Du and P.M. Pardalos (Eds.) pp. 261-329 (2000)
22. Dmitry I. Ignatov, Alexander Semenov, Daria Komissarova, Dmitry V. Gnatyshak: Multimodal Clustering for Community Detection. Formal Concept Analysis of Social Networks 2017: 59-96
23. https://github.com/ibrahim85/d-peeler