Evolutionary algorithms for Map of Attributes optimization

T. RZEŹNICZAK
tomek.rzezniczak@gmail.com

Military University of Technology, Faculty of Cybernetics,
Institute of Computer and Information Systems
Kaliskiego Str. 2, 00-908 Warsaw, Poland

Map of Attributes (MoA) is a visualization technique that allows to construct graphical representation of abstract entities. The technique is intended to aid recognition of the entities' representations through the effective use of human perception abilities. A certain difficulty in the application of MoA is the computational complexity of finding an optimal map. The study presents a heuristic approach, based on evolutionary algorithms (EA), to constructing MoA visualization. The method was evaluated using the repository of disease entities as an input dataset. Several different setups of EA were tested; these were configurations with well-known evolution operators, as well as setups with newly proposed operators for the matrix representation of chromosome. Detailed results and analysis of conducted experiments are presented.

Keywords: evolutionary algorithms, data visualization, evolution operators.

1. Introduction

The study presents a heuristic approach to solving the problem of finding the optimal Map of Attributes visualization for a given dataset [22], [23]. The technique was applied to the visualization of medical patterns [1], [2]: disease entities as well as the patient's health condition. For the purpose of the study an example repository of diseases was prepared. The source of the repository content was Mayo Clinic [6] (where among other information, each disease is described by the most common symptoms). The final repository is a subset of diseases for three medical specialties: pulmonary medicine, cardiology and gastroenterology. It contains 78 diseases which are defined by 143 symptoms; naturally, a single symptom can occur in multiple diseases. MoA operates by constructing a two-dimensional map of points, where each point reflects a unique symptom form the visualized repository. Then, each disease entity can be presented graphically by polygon, whose vertices are symptoms belonging to the disease, see examples in Fig. 1. A graphical representation of a patient's health condition can be similarly constructed, in the form of a polygon, out of symptoms diagnosed in the patient. The Map of Attributes technique is designed to facilitate the use of natural human perception abilities in disease recognition process by:

1. Application of a two-dimensional space for symptoms representation as points on the map – according to Mackinlay's ranking, defining a position in space is the most effective perception task [19].
2. Using polygons for building graphical representations of medical patterns – figures allows the application of shape perception in pattern recognition. Shape is a graphical characteristic of an image that carries the biggest amount of information [21].
3. Utilizing the effect of the polygon's position on a map – the position is an additional attribute that simplifies memorizing and disease entities recognition. Analogically to point 1, but on the level of diseases.
4. Adopting figures' shape perception optimization by maximizing polygons' figural goodness, which is a number of its regularities, such as symmetry and repetition [13], [21]. According to Gestalt psychology, humans process good figures better, which has a positive influence on shape perception and improves effectiveness of matching and pattern recognition [8], [9], [16].

To achieve the #4 objectives, it is required to find an arrangement of symptoms on a map in such a way that all visualized diseases form good figures. In order to solve the task, three computationally complex issues must be addressed [22]:

1. **Polygon constructing** – with $O(n!)$ computational complexity, it involves finding a polygon from a given set of n points (vertices) on a map.
2. **Figural goodness evaluation** – with $O\left(2^{nlogn}\right)$ computational complexity, it involves finding information load of a polygon with n vertices, which is preferable perceptual interpretation of the polygon [12].
3. **Map optimization** – with $O(n!)$ computational complexity, it involves finding optimal arrangement of map with n symptoms, it is such that has maximal aggregated figural goodness of all diseases (polygons) represented on the map.

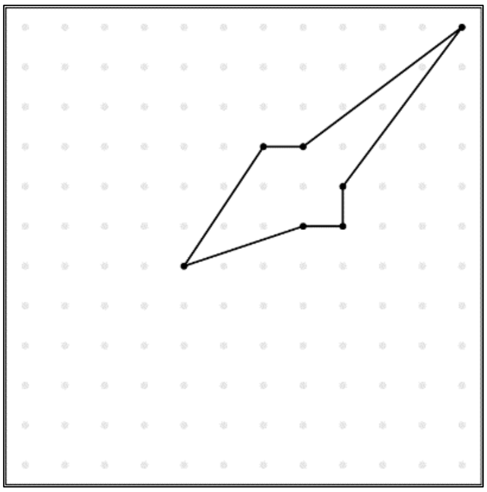


Fig. 1. Example of a disease visualization using MoA

The first two problems, despite their complexity, can be solved in finite time, due to limited size of data (the number of symptoms per disease is generally less than a few dozen [15], [18]) and efficient algorithms [23]. The biggest challenge is the $O(n!)$ complexity of *Map optimization* problem, where the number of input data (different symptoms and diseases) can be very high [14], [24]. Even in the example repository, there are 143 symptoms and 78 diseases.

Simple searching through the space of possible maps is not acceptable. Therefore, the study was focused on finding a more efficient method. Since, potentially good candidates for such an optimization problem are heuristic approaches, the application of Evolutionary Algorithms (EA) was selected [7]. Naturally, the heuristic approach is not a guarantee of finding an optimal solution, but anything close to an optimal solution can be good enough from the visualization perspective. The effectiveness of the mentioned strategy was the subject of experimental verification and detailed results are presented in next sections.

2. Evolutionary algorithms overview

Evolutionary Algorithms (EA) are a family of optimization methods based on stochastic search through the space of possible solutions [3]. EA mimics the main steps of the evolutionary process: selection of individuals for reproduction, their recombination and mutation of their genotype. Key concepts of EA are:

- **individuals** (candidates) – represent a possible solution to the problem,
- **objective function** – evaluates individuals,
- **individuals selection technique** (strategy) – decides which candidates survive,
- **evolution operators** – are techniques of reproduction and mutation that produce offspring of the candidates.

EA operation rules are as follows:

1. Generating an initial population (where the number of individuals is one of the important parameters);
2. Evaluating fitness of each of the individuals using an objective/fitness function;
3. Selecting the best individuals as parents for the next generation
4. Creating a next generation using the evolution operator (reproduction and mutation techniques);
5. Proceeding the next generation to step 2.

The algorithm ends when termination conditions are satisfied, typically these are: number of generations, stagnation or achieved required fitness level.

A crucial aspect of EA is fitness function used to evaluate individuals. It can be defined quite freely, the only constraint is that it should reflect how close an individual is to the desired outcome. In the case of searching for optimal map arrangement, the objective function can be defined using information load – $IL(f)$ of a figure f as a bases. A proposed metric is an aggregated information load:

$$aggIL_m(O) = \sum_{x \in O} IL(p_m(x)) \tag{1}$$

where:

- x – a disease
- O − repository of diseases
- $p_m(x)$ − graphical representation of disease x on a given map m.

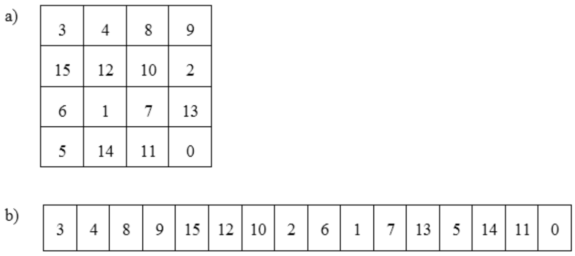
The lower *aggIL* is, the “better” map was found and more objects are represented as good figures.

3. Map optimization EA representation and operators

Several different configurations of the evolutionary algorithm for MoA optimization was evaluated. Major building blocks and at the same time parametrization areas of the algorithm are presented below. Description is limited only to those that are important in explaining issues that were encountered. The discussion is started with “standard” well-known solutions and followed by specialized methods prepared for specific characteristics of the map optimization problem.

3.1. Chromosome encoding

The initial prerequisite is the definition of individuals genotype encoding. Like in biological evolution, the encoding is called a chromosome [3]. A chromosome defines a single solution or set of parameters of a solution. There are no restrictions on the chromosome encoding method and any data structure can be applied. Typically it is a list of some sort of elements like: string of 0 and 1 or a list of cities’ names (for example in the case of the Travelling Salesman Problem) – called later list representation.



a)

3	4	8	9
15	12	10	2
6	1	7	13
5	14	11	0

b)

3	4	8	9	15	12	10	2	6	1	7	13	5	14	11	0

Fig. 2. Chromosome encoding examples:
a) matrix representation;
b) ordered list representation

In our case, a single solution is a square map of *n* equal cells, where each cell represents a single attribute from the visualized data set. Natural encoding of the chromosome is a matrix of n x n size, where the matrix’s elements correspond to the attributes – later called matrix representation. However, to be able to apply standard EA operators, the ordered list of elements representation can also be adopted. The list would reflect the order of attributes in the map reading from left-to-right and top-to-bottom (see Fig. 2). The efficiency of both types of the chromosomes was exanimated in further experiments.

3.2. Selection strategy

Selection strategy operates between each iteration of the evolutionary algorithm [3]. It defines a method of selecting candidates to become parents for the next population. Selection strategies are very important, since they decide how successive generations are growing and direct them into a promising area of a search space. There are many ready-to-use selection strategies, derived from Genetic Algorithms and Evolution Strategies, to name only a few: Truncation Selection, Roulette Wheel Selection, Rank Selection, Tournament Selection, (μ + λ) – ES and (μ, λ) – ES. Only the last one is described here, because it proved during experiments to be the most suitable for the map optimization problem.

The (μ, λ) – ES is also called comma selection [4], μ represents size of a population and $\lambda \geq 1$ is a number of descendants that is generated in each iteration. In the selection process the parent population is not included and only the fittest offspring are in the pool and can be selected to the next population – to simplify, parents are forgotten.

3.3. “Standard” evolution operators

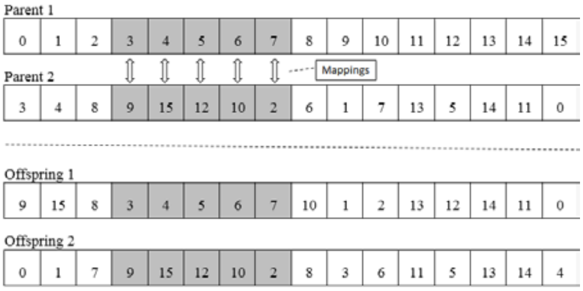
Evolution operators enable genetic diversity in a population. Applying the operators prevents getting individuals who are too similar to each other and helps to avoid local minimums [3]. The operators are applied to selected parent candidates (one or more) and result in one or more offspring. There are two types of operators: crossover and mutation. However, many different realizations of them are possible. A useful overview of the most popular ones can be found in [17] and [20]. Some of the well-known operators were used in our empirical tests and these are introduced in more detail below.

3.3.1. Crossover operators

The first type of operators is crossover [20]. Crossover operators require usually at least two parents (usually, because there are also crossover techniques for more than two parents). Since crossover involves exchanging of genotype between parents and creating one or more offspring. Crossover is typically based on swapping of genotype segments or individual genes. For the purpose of this study, two of such techniques: Partially Matched Crossover (PMX) and Position Based Crossover (POS) were used [5], [10].

Partially Matched Crossover – PMX

PMX consists of selecting a segment of chromosomes that should be preserved from parents (two parents in case of PMX), based on the segments a mapping of genes is also defined. Next, two offspring are created, each of the offspring preserves a selected segment from one of the parents and the missed genes are taken from the same positions from the second parent. If any of the genes is repeated then the mapping defined at the beginning is applied. Fig. 3 presents an example of PMX application to a chromosome from Fig. 2 and another parent (with 0 to 15 ordered elements).



Parent 1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Parent 2
3 4 8 9 15 12 10 2 6 1 7 13 5 14 11 0

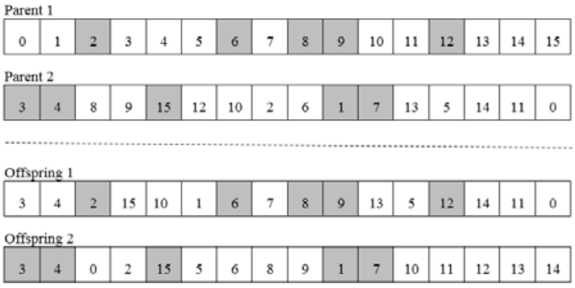
Offspring 1
9 15 8 3 4 5 6 7 10 1 2 13 12 14 11 0

Offspring 2
0 1 7 9 15 12 10 2 8 3 6 11 5 13 14 4

Fig. 3. An example of Partially Matched Crossover operator

Position Based Crossover – POS

In POS, firstly, genes that will be preserved from the first parent are randomly chosen and transferred to an offspring. Subsequently, the missing genes are copied from the second parent in unchanged order, if some of the genes are already present then it is skipped and copying moves to a next one. An example of applying POS operator can be found in Fig. 4.



Parent 1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Parent 2
3 4 8 9 15 12 10 2 6 1 7 13 5 14 11 0

Offspring 1
3 4 2 15 10 1 6 7 8 9 13 5 12 14 11 0

Offspring 2
3 4 0 2 15 5 6 8 9 1 7 10 11 12 13 14

Fig. 4. An example of Position Based Crossover operator

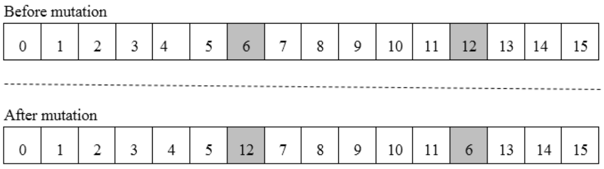
3.3.2. Mutation operators

Mutation operators apply to a single individual, therefore, they are usually simpler. The idea is simply to alter a single or multiple genes in a chromosome. To generalize, for each gene of a chromosome a decision is made whether to change it or leave it unchanged. Alteration is done with a defined probability, if the probability is too high, evolution can be reduced to a random search. The mutation operator used in the experiments is Repeated Exchange Mutation Operator (REM) [4].

Repeated Exchange Mutation – REM

REM re-orders a random element of a parent genotype with an element occurring x positions before or after it (where x < number of elements). An example of REM operation is presented in Fig. 5. The operator has two parameters:

- **mutation count** – defines how many times the mutation is applied, it can be a fixed number or determined according to a defined probability schema;
- **mutation amount** – defines the number of positions by which the element is shifted, it can be also fixed or randomized.



Before mutation
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

After mutation
0 1 2 3 4 5 12 7 8 9 10 11 6 13 14 15

Fig. 5. An example of Repeated Exchange Mutation

3.3.3. Operators characteristics

Let's consider now some characteristics of the above operators. PMX preserves the selected segment and absolute positions of elements from the second parent. Therefore, it may be especially suitable for problems where final solution can be built of such optimized subparts and absolute positions of genes are important. On the other hand, POS partially preserves absolute positions of genes (genes of the first parent) and partially emphasizes the genes order (genes of the second parent).

The map optimization problem has its specific characteristic. Although a solution can be represented as an ordered list of chromosomes, the position of a single gene influences not only its neighbors, but many other related genes. This is because of co-occurrence of many attributes in a single object and between objects. Furthermore, because of the 2D nature of the map optimization problem, the position of the gene in 2D space is an important aspect, not the position on the 1D list. Therefore, intuitively a better approach should be one that preserves positions of genes, which indicates PMX. Analyzing POS, its drawback is focusing on order of elements when copying from the second parent, which is not important from our perspective. However this operator has also a potential advantage, POS preserves absolute positions of selected elements of the first parent.

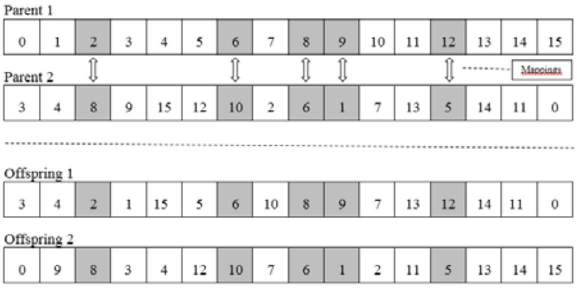
Regarding REM operator, its typical implementation is dependent on the list representation, because elements are selected to be swapped using a random distance in a given sequence of elements. Therefore, its effectiveness may be weaker in 2D space problems, which is our case.

3.4. Specialized evolution operators

“Standard” operators were built with list representation of chromosomes in mind. Additionally, some of them are focused more on the ordering of genes than their absolute or relative positions. Therefore, an attempt was made to prepare specialized operators, which can cope with specific issues of 2D problems and the related 2D matrix representation of chromosome, like in the map optimization problem. All operators presented in this section are original propositions constructed during research on the map optimization problem.

Position Based Mapped Crossover – PBMX

POS operator has its advantage in preserving the absolute positions of genes from the first parent, at the same time genes of the second parent are copied keeping their order but not positions. While the position preservation of the second parent genes is a feature of PMX operator. The Position Based Mapped Crossover (PBMX) operator is a mixture of POS and PMX approaches. It can be applied to both: list and matrix chromosome representations.



Parent 1

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Parent 2

3 4 8 9 15 12 10 2 6 1 7 13 5 14 11 0

Offspring 1

3 4 2 1 15 5 6 10 8 9 7 13 12 14 11 0

Offspring 2

0 9 8 3 4 12 10 7 6 1 2 11 5 13 14 15

Fig. 6. An example of Position Based Mapped Crossover operator

In the first phase PBMX works like POS: a set of genes that will be transferred to offspring is randomly selected for both parents. In the next step, similarly as in PMX, a mapping between the selected genes of the parents is produced. Then, the algorithm continues with PMX approach, which means that two offspring are created, each of the offspring receives the selected genes from one of the parents and missed genes are taken from the same positions from the second parent. In the case when some of the genes are repeated, defined mapping is applied to fix this issue, see Fig. 6 for an example.

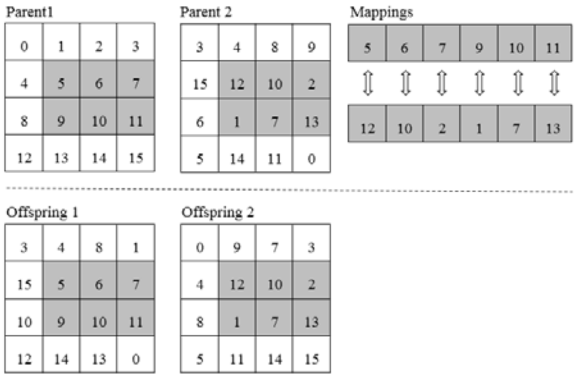
Region Mapping Crossover – RMX

A possible disadvantage of PMX operator is that it was built on segments of genes, which are subsequences of a 1D list. This may not be sufficient in the case of its application to 2D related problems. On the other hand, as it was noticed before, the preservation of gene positions while copying from the second parent is a very welcomed behavior. To overcome the 1D sequences issue, a specialized crossover operator was proposed – Region Mapping Crossover (RMX). The operator is region based, so instead of selecting a sequence of genes, a region in the matrix representation of chromosome is selected, where the region is understood as a submatrix. Further behavior of RMX is similar to PMX. Elements included in the region from the first parent are mapped to elements at the same positions from the second parent. Next, the selected region of the first parent is copied to the first offspring, and the missed genes are taken from the second parent left-to-right and top-to-bottom. Their original positions are kept, but if some of the genes are already present in the offspring, the mapping prepared earlier is applied and its replacement is used. The same procedure is applied to the second pair of parent and offspring, see Fig. 7 for an example.

An important comment is that the method of selecting the region is not explicitly defined; therefore, different strategies are possible. Assuming that common first step is determining randomly the size of the submatrix, further possible strategies are:

- **wrapping strategy** – randomly selecting a cell of the matrix, which will be the top-left corner of the region. If the region size does not fit on the matrix, the region is wrapped to the opposite side of the matrix.
- **non-wrapping strategy** – as in the wrapping strategy, with the difference that if the region size does not fit on the matrix, the surplus portion of the region is not taken into account.
- **fitting strategy** – randomly selecting a cell of the matrix which will be the top-left corner of the region, but available cells are limited to those that guarantee the whole region fits on the matrix;

It should be noticed that only the wrapping strategy is fair and selects each of the cells with the same probability. All the other strategies prefer some cells, at the same time this does not mean that they cannot give good results for certain types of problems.



Parent1

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Parent 2

3	4	8	9
15	12	10	2
6	1	7	13
5	14	11	0

Mappings

5	6	7	9	10	11
⇕	⇕	⇕	⇕	⇕	⇕
12	10	2	1	7	13

Offspring 1

3	4	8	1
15	5	6	7
10	9	10	11
12	14	13	0

Offspring 2

0	9	7	3
4	12	10	2
8	1	7	13
5	11	14	15

Fig. 7. An example of Region Mapping Crossover operator

Figure Preservation Heuristic Crossover – FPHX

In analyzing the already discussed crossovers, these are general operators, in the sense that they can be used for any kind of a problem as long as it is represented as list or matrix chromosome. Their behavior differs and they emphasize diverse aspects, like ordering of genes, position or segment/region preservation. However, these are still solutions that can be applied without any deeper knowledge on the problem. Another type of crossovers is when specific knowledge of a problem is incorporated into the operator to facilitate the algorithm convergence and effectiveness. The idea was introduced by Grefenstette, when he proposed a class of heuristic crossover operators for TSP [11].

The objective function of the map optimization problem is to minimize aggregated information load of the map or, in other words, maximize the figural goodness of the objects' graphical representations. Risk associated with the previous crossovers is that they can break the building blocks of the solution which are good figures. In the course of the algorithm, some figures which present high figural goodness may be found and can be treated as already optimized (the assessment can be done using figural goodness coefficient which will be discussed later). Such figures should be preserved during the crossover operation. This is a basis of the heuristic used to develop a new operator – Figure Preservation Heuristic Crossover (FPHX).

The general flow of FPHX operator is as follows (the whole procedure is carried out separately for both parents). First, genes belonging to figures that have figural goodness coefficient on some required level are selected from the first parent. Then, a mapping that maps those selected to genes holding the same absolute positions in the second parent chromosome is built. Next, a new offspring is created, starting with copying the genes selected from the first parent, followed by transferring missing genes from the second parent. The transfer is similar to PMX operator, genes' original positions are kept and it is done from left-to-right and top-to bottom, if a transferred gen is already present in the offspring the mapping is applied. The same procedure is performed to create the second offspring, but the roles of parents are swapped. It should be noted that another set of figures/genes is selected and another mapping is prepared after the swap. This is due to the fact that the operator is not symmetrical and the previous mapping would not be valid for the second parent. An example of FPHX is presented in Fig. 8.

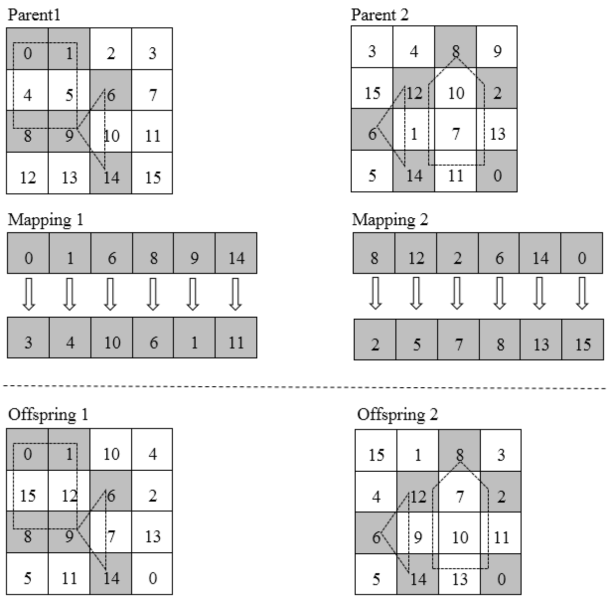


Fig. 8. An example of Figure Preservation Heuristic Crossover operator

For the purpose of the FPHX parametrization a figural goodness coefficient (*fgc*) was defined:

$$fgc(f) = \frac{minIL(f)}{maxIL(f)} \qquad (2)$$

where:

- *maxIL(f)* – information load of a maximal code of a figure *f*. The maximal code of *f* is a pure contour code without looking for regularities (for details see [23]).
- *minIL(f)* – the information load of the minimal code of a figure *f*, where the minimal code of *f* represents preferred perceptual interpretation of the figure (for details see [23]).

In the basic version of FPHX operator, the figural goodness coefficient parameter is fixed through the whole execution of an evolutionary algorithm. This version was tested during the experiments and provided the best results. Although, other extended versions are possible, for example FPHX could use relative values of figures' coefficients and select only top N figures, where N is another parameter that needs to be set before the algorithm execution. FPHX (top N) version can also be combined with forcing minimal level of the figural goodness coefficient among the top N figures. Unfortunately, the initial results of the extended versions were not promising and, therefore, detailed experiments of them were not conducted.

Zonal Repeated Exchange Mutation – ZREM

Zonal Repeated Exchange Mutation is another new operator devolved for the purpose of this study. As the name may suggest, it is an equivalent to Repeated Exchange Mutation that operates in the defined zone/region of the matrix representation. Application of REM operator to the map optimization problem has the same disadvantages as any other operator originated from the list representation, it does not take into account the 2D characteristic of the problem. Therefore, polygons influenced by the REM operator can be destroyed by transfer of one of their vertices to a completely random position. To prevent this, a set of transfer-eligible positions should be limited to a certain distance from the vertex in 2D space. The idea of ZREM is that better mutation effects can be achieved if random transfer is limited to a parametrized zone around a vertex.

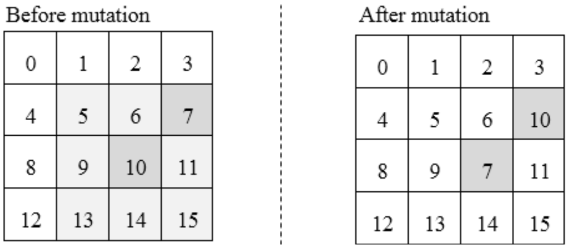


Fig. 9. An example of Zonal Repeated Exchange Mutation operator

The zone can be defined freely, in the implementation used in this study the zone is a square submatrix around a random gene defined by a single parameter. The parameter is a maximal distance from the gene in four directions (top, down, left, right). Fig. 9 shows an example of ZREM operator application. Summarizing, ZREM operator can be parametrized by:

- **mutation count** – defines how many times the mutation is applied, it can be a fixed number or determined, according to some probability schema;
- **mutation zone** – with four sub-parameters (top (T), down (D), left (L), right (R)) it defines a zone in which an element is shifted, the shift itself is to a random position inside the zone. If the zone size does not fit on the matrix fully, the zone is partially wrapped to the opposite side of the matrix.

4. Results

The EA approach for the map optimization problem was tested in experiments conducted using the example repository of diseases (see section 1) with 143 symptoms and 78 entities. As it was discussed, the complexity of the problem was divided into three tasks: Figure constructing $O(n!)$, Minimal code $O(2^{nlogn})$, Map optimization $O(n!)$. The first two tasks can be solved quite efficiently because of the size of n and powerful algorithms available [23]. However, in case of the last task n can be significantly larger if one would like to build a map of broad diseases' space. Even 143! gives a number of the order of 10247. Therefore, approximately 32 x 106 map verification per second speed is needed to finish the work in one year using brute force search. This is not achievable without enormous computing power, therefore, an unacceptable solution as well.

To set a background for results interpretation, let's estimate theoretical values of aggregated information load of an optimal and worst map – for the example repository. Starting from the worst map case, it would be a map where each of polygons would have no regularities. Therefore, the minimal code of each figure would be equal to its maximal code [22], which is twice the number of a figure sides (sum of the number of figure sides and the number of interior angles – one symbol per element). In our repository, there are 78 diseases and aggregated number of their symptoms is 554, which means that the worst map would have *aggIL* of 1108. Whereas, during the experiments, it was observed that average *aggIL* of a random map is 992.

In the case of the optimal map case each polygon would have maximal number of regularities, which means each polygon would be a regular polygon. A minimal code of any regular polygon is 2, since it can be described by two symbols – one representing sides length and one for angles size. The theoretical optimal map would have *aggIL* equal to 156 (78 diseases multiplied by 2). However, the figural goodness coefficient (*fgc*) (see definition 2) observed during the experiments in the best maps was on average 0.642, while the minimal level was 0.357. Assuming that all the figures would achieve *fgc* equal to 0.3, such a map would have *aggIL* around 395 (~ 0.3 x 1108).

Obviously, both of the above cases are purely theoretical and there is no proof that such maps can be constructed, considering their dimensionality, shared symptoms and dependencies between polygons. The discussion was intended only to approximate the boundaries of *aggIL* space, this will help to place the results of the experiments in some context.

4.1. Experiments description

Experiments were conducted using several different setups of evolutionary algorithms' components: crossover operators, mutation operators, selection strategies and their parameters described in section 3.

Since the space of potential combination is very large, before the final evaluation was done, a preliminary assessment of various approaches was conducted. The assessment covered especially parameters of algorithms execution like: population size, mutation amount and count, selection strategies' specific parameters and operators' specific parameters (like figural goodness coefficient in the case of the FPHX crossover operator). Only those parameters ranges and algorithm setups that showed promising results were tested further. A very important observation was that for the given problem the (μ, λ)-ES selection strategy was optimal. In the case of others which were also initially verified such as: Truncation Selection, Roulette Wheel Selection, Rank Selection, Tournament Selection, the issue was premature convergence mostly. Finally, (μ, λ)-ES was applied in all the setups.

The evaluated setups can be found in Tab. 1. Each of the setup was executed 5 times and the stop condition was stagnation of the mean fitness of the population for more than 50 generations. Evaluation of the results was done using several types of metrics. These are metrics based on objective function results, metrics based on figural goodness coefficient and metrics related to EA qualities itself. All the metrics' definitions can be found in Tab. 2.

Tab. 1. The evaluated setups of evolutionary algorithms (DUD – discrete uniform distribution)

Setups/ operator	**PMX-setup**	**POS-setup**	**PBMX-setup**	**RMX-setup**	**FPHX -setup**	**ZREM-setup**	**REM-setup**
crossover operator	PMX	POS	PBMX	RMX	FPHX - wrap.strategy	N/A	N/A
mutation operator	REM	REM	ZREM	ZREM	ZREM	ZREM	REM
mutation count	Poisson dist. (λ=2)	Poisson dist. (λ=2)	Poisson dist. (λ=2)	Poisson dist. (λ=1)	Poisson dist. (λ=1)	Poisson dist. (λ=1)	Poisson dist. (λ=1)
mutation amount	DUD in: [1,8]	DUD in: [1,8]	N/A	N/A	N/A	N/A	DUD in: [1,8]
mutation zone	N/A	N/A	DUD in zone: T= 1 D= 1 L = 1 R = 1	DUD in zone: T= 1 D= 1 L = 1 R = 1	DUD in zone: T= 1 D= 1 L = 1 R = 1	DUD in zone: T= 4 D= 4 L = 4 R = 4	N/A
selection strategy	(μ, λ) μ=200 λ=400	(μ, λ) μ=100 λ=400	(μ, λ) μ=100 λ=400	(μ, λ) μ=200 λ=400	(μ, λ) μ=200 λ=400	(μ, λ) μ=20 λ=400	(μ, λ) μ=200 λ=400
crossover factor	N/A	N/A	N/A	N/A	0.5	N/A	N/A
fgc	N/A	N/A	N/A	N/A	0.65	N/A	N/A

Tab. 2. Results evaluation metrics

Category	**Metric Name**	**Definition**
Objective function related	*aggIL-top*	Lowest aggregated IL of best candidates from all executions
	aggIL-avg	Average aggregated IL of best candidates from all executions
	aggIL-worst	Worst of aggregated IL of best candidates from all executions
Figural goodness coefficient related	*fgc-avg*	Average figural goodness coefficient of the best candidate map found in all executions
	fgc-max	Maximal figural goodness coefficient observed in the best candidate map found in all executions
	fgc-min	Minimal figural goodness coefficient observed in the best candidate map found in all executions
EA qualities related	*gen-num*	Number of finished generation until the best candidate was found
	search-space	Number of evaluated candidates until the best candidate was found

4.2. Analysis of results

The overall results of the experiments can be found in Tab. 3. Starting with a brief summary, the best results in terms of objective functions and figural goodness coefficient metrics were achieved using PMX-setup. The aggregated IL of the best map that was found is 679 with the average *fgc* equal to 0.642. Comparing this to an average *aggIL* of a random map equal to 992, it is approximately a 32% improvement. The search space until the best map was found was ~340 K candidates in 851 generations. Other setups achieved worse results, the second was RMX-setup with ~26% improvement level, the third was FPHX-setup with ~22% improvement. The FPHX-setup stands out by the low number of generation and, therefore, the size of the search space until the best candidate was found, which was approximately ~64 K. It is also worth emphasizing ZREM-setup results, the setup includes only the Zonal Repeated Exchange Mutation operator without application of any crossover operator. The achieved *aggIL-top* gives ~20% improvement compared to the random map. The remaining setups, which are POS-setup and PBMX-setup, had worse performance – below 16% of improvement.

Tab. 3. Metrics for the evaluated setups of evolutionary algorithms

Metrics	**PMX-setup**	**POS-setup**	**PBMX-setup**	**RMX-setup**	**FPHX-setup**	**ZREM**	**REM**
aggIL-top	**679**	826	864	737	774	789	830
aggIL-avg	689.4	828.6	872.2	740.6	783	793,6	835.8
aggIL-worst	698	833	878	747	790	797	842
fgc-min	0.357	0.375	0.611	0.375	0.375	0.5	0.5
fgc-avg	**0.642**	0779	0.815	0.701	0.718	0.748	0.782
fgc-max	1	1	1	1	1	1	1
gen-num	851	476	319	974	**162**	672	479
search-space	~340K	~190K	~127K	~389K	**~~64K**	~268K	~155K

Let's verify how accurate the theoretical considerations on the effectiveness of operators from section 3 were. Recognized PMX operator advantages were the preservation of segments from the first parent and preservation of absolute positions of elements from the second parent. While, the potential issue that was noticed was its focus on 1D – an ordered list chromosome representation. The new specialized operator that was proposed to overcome the 1D issue was Region Mapping Crossover (RMX), which works on matrix crossover representation and instead of 1D sequences is based on 2D matrix regions. In contradiction to the theoretical assumptions, the results show that setups with

PMX operator have better performance than those with RMX operator. A possible explanation of good achievements of PMX is that even though it preserves 1D segments, the size of the segments is not limited. This means that large segments are in fact regions formed of several rows of the matrix. Harder to explain is worse performance of RMX. The reason here can be related to specific conditions of the particular problem or region selection methods. A detailed analysis of these particular observations could be very complex and since it is not a direct goal of the study it will not be elaborated further here.

The POS operator preserves position of selected genes from the first parent and ordering of genes from the second parent. The first part (positions preservations) is representation agnostic and there is no difference in applying it to 1D and 2D representations. On the other hand, the second part was criticized in the context of the objective function, since the ordering is not important in map optimization. To improve the POS operator, the PBMX operator was proposed, which was designed to replace the ordering part with preservation of the genes from the second parent. It was achieved by using the mapping technique known from the PMX operator. Finally, the setups with the PBMX operator had the worst results and the setup with the POS operator the second worst result. Analyzing the data, it turns out that the major issue with both of the operators is the first phase, where a random selection of genes for preservation is performed. The randomness causes that figures do not survive in their entirety. What is worse, even coherent fragments of figures do not survive, only random vertices. Therefore, the algorithm quickly stagnates. As the experiments showed, enforcing the second parent genes position preservation by defining a mapping causes an even quicker stagnation. The explanation here is lower diversity of candidates that the additional preservation introduces.

A certain solution to the issues reported in case of the POS and PBMX operators is the FPHX operator. The technique is very similar to PBMX, with the difference that genes from the first parent are not randomly selected but belong to figures with the highest figural goodness coefficient. The approach proved to be significantly better than POS and PBMX. It did not achieve the level of PMX, but its advantage is a significantly smaller search space to find candidates with over 20% improvement level. Still, the operator issue is low diversity and, therefore, quick stagnation.

ZREM setup execution was initially conducted to validate the new proposed mutation operator for 2D genotype representations. The effects were good and demonstrated that the ZREM operator works well for the matrix representation – better than the list representation dependent REM operator.

4.3. Discussion of the top map

The previous chapter already introduced that PMX-setup achieved the best performance in terms of the objective function. To better illustrate the result, let's analyze the disease's representation produced by the best map in a context of other close to random maps. Tab. 4 presents key metrics of two maps:

- **TopMap** – the top map (the best found map for the example repository of diseases, taking all experiments into account)
- **1stGenMap** – the best candidate map after first generation of the same run of the algorithm during which the TopMap was found.

Comparing these two maps, the average figural goodness coefficient of all the represented figures has dropped from 0.914 (1stGenMap) to 0.642 (TopMap) and minimal *fgc* levels changed from 0.727 to 0.357 respectively. The practical impact on the regularities in the disease representations can be seen in Figures 10 to 16, where examples of polygons with different *fgc* are presented using both maps. As it was expected, differences are easily noticeable. The polygons show significant regularities even at the level of 0.7 *fgc* (see Fig. 15 and 16).

Tab. 4. Metrics of TopMap and 1stGenMap

Metrics	**TopMap**	**1stGenMap**
aggIL	679	970
fgc-min	0.357	0.727
fgc-avg	0.642	0.914
fgc-max	1.0	1.0

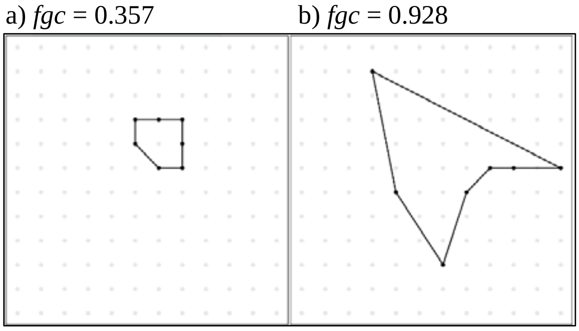

a) *fgc* = 0.357 b) *fgc* = 0.928

Fig. 10. Hypertrophic Cardiomyopathy represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

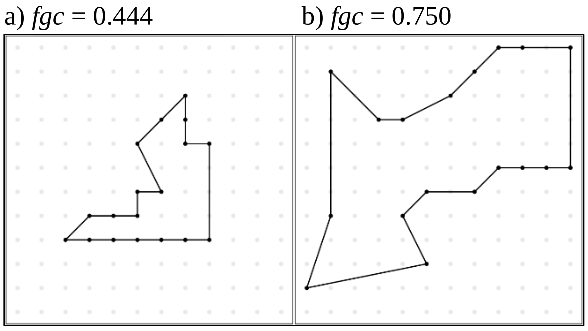

a) *fgc* = 0.444 b) *fgc* = 0.750

Fig. 11. Pulmonary Edema represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

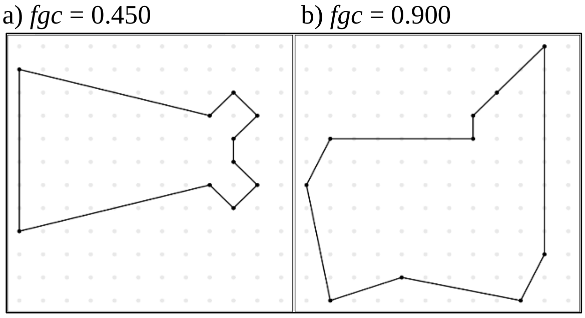

a) *fgc* = 0.450 b) *fgc* = 0.900

Fig. 12. Diverticulitis represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

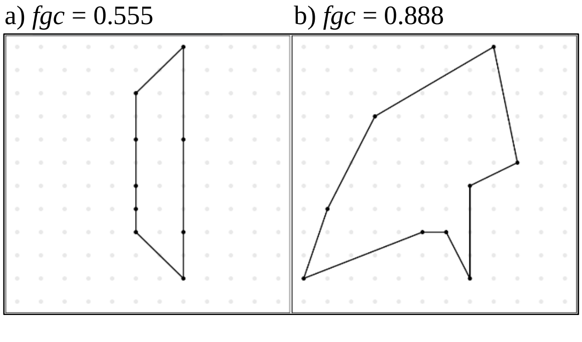

a) *fgc* = 0.555 b) *fgc* = 0.888

Fig. 13. Atrioventricular Canal Defect represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

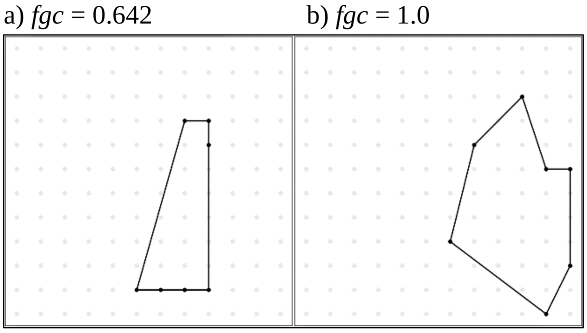

a) *fgc* = 0.642 b) *fgc* = 1.0

Fig. 14. Peptic Ulcer represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

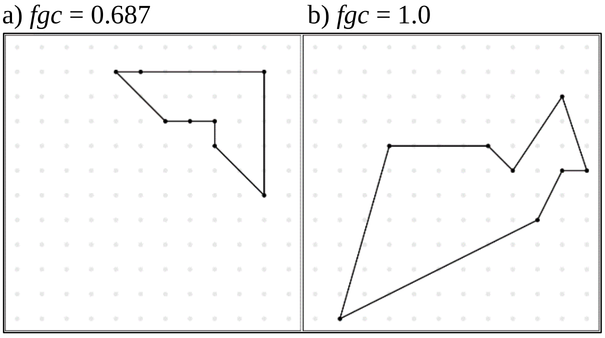

a) *fgc* = 0.687 b) *fgc* = 1.0

Fig. 15. Stomach Cancer represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

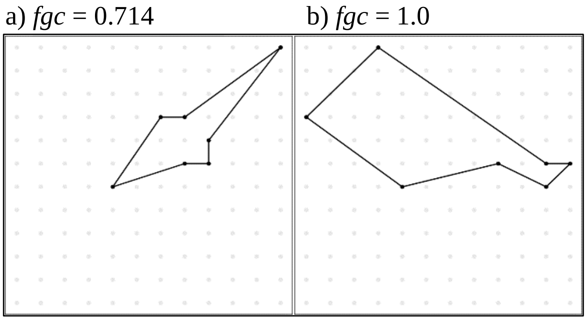

a) *fgc* = 0.714 b) *fgc* = 1.0

Fig. 16. Hemochromatosis represented on maps with different levels of *aggIL*: a) TopMap; b) 1stGenMap

5. Summary

The study presents a heuristic approach to the preparation of a map of attributes for a given data set. Evolutionary algorithms are applied to solve the map optimization problem. Multiple experiments assessing EA effectiveness were conducted using an example repository of disease entities. In addition to well-known evolutionary algorithms' operators that were tested, several new ones, specialized for matrix representation of a chromosome, were proposed. Their evaluation was done on the basis of the MoA optimization problem. Newly elaborated operators include: Region Mapping Crossover (RMX), Position Preservation Crossover (PPX), Figure Preservation Heuristic Crossover (FPHX), Zonal Repeated Exchange Mutation (ZREM).

The best of evolutionary algorithm setups reaches an approximate of 32% enhancement in comparison to a random map. This result gives a significant improvement in terms of figure regularities represented on the map, still it is far from potential optimum (see section 4). Therefore, possible direction of further research can be constructing an alternative algorithm for solving the MoA optimization problem.

6. Bibliography

[1] Ameljańczyk A., "Wielokryterialne mechanizmy wspomagania podejmowania decyzji medycznych w modelu repozytorium w oparciu o wzorce", *Biuletyn Instytutu Systemów Informatycznych*, Nr 5, 1–6 (2010).

[2] Ameljańczyk A., "Multicriteria similarity models for medical diagnostic support algorithms", *Bio-Algorithms and Med--Systems*, Nr 9(1), 1–7 (2013).

[3] Back T., *Evolutionary algorithms in theory and practice*, Oxford Univ. Press., 2010.

[4] Beyer H.G., Schwefel H.P., "Evolution strategies – A comprehensive introduction", *Natural computing*, 1(1), 3–52 (2010).

[5] Davis L., *Handbook of genetic algorithms* Vol. 115, New York: Van Nostrand Reinhold, 1991.

[6] Disease and Conditions, Mayo Clinic, http://www.mayoclinic.org/diseases-conditions, accessed November 2015.

[7] Dianati M., Song I., Treiber M., *An introduction to genetic algorithms and evolution strategies. Technical report*, University of Waterloo, Ontario, N2L 3G1, Canada, 2002.

[8] Donderi D.C., "Visual complexity: a review", *Psychological Bulletin*, No. 132(1), 73–97 (2006).

[9] Garner W.R., *The processing of information and structure*, Psychology Press, NY 2014.

[10] Goldberg, D.E., Lingle R., "Alleles, loci, and the traveling salesman problem", *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, Vol. 154, Lawrence Erlbaum, Hillsdale, NJ, 1985.

[11] Grefenstette J.J., "Incorporating problem specific knowledge into genetic algorithms", *Genetic algorithms and simulated annealing*, No. 4, 42–60 (1987).

[12] van der Helm P., Leeuwenberg E., "Accessibility: A Criterion for Regularity and Hierarchy in Visual Pattern Codes", *Journal of Mathematical Psychology*, No. 35, 151–213 (1991).

[13] Hochberg J., McAlister E., „A quantitative approach to figural goodness", *Journal of Experimental Psychology*, No. 46, 361–364, (1953).

[14] International Classification of Disease, World Health Organization, http://www.who.int/classifications/icd/en/, accessed November 2015.

[15] Karges W., Al Dahouk S., *Innere Medizin. in 5 Tagen*, Springer, 2011.

[16] Kayaert G., Wagemans J., "Delayed shape matching benefits from simplicity and symmetry", *Vision research*, No. 49(7), 708–717 (2009).

[17] Larranaga P., Kuijpers C.M., Murga R.H., Yurramendi Y., "Learning Bayesian network structures by searching for the best ordering with genetic algorithms", *Systems, Man and Cybernetics, Part A: Systems and Humans*, IEEE Transactions on, No. 26(4), 487–493 (1996).

[18] Latkowski B., Lukas W., *Medycyna rodzinna – repetytorium*, Wydawnictwo Lekarskie PZWL, Warszawa, 2008.

[19] Mackinlay J., "Automating the Design of Graphical Presentations of Relational Information", *ACM Transactions on Graphics*, Vol. 5, Issue 2, New York, USA, April, 1986.

[20] Magalhaes-Mendes J., "A Comparative Study of Crossover Operators for Genetic Algorithms to Solve the Job Shop Scheduling Problem", *WSEAS Transactions on Computers*, Vol. 12(4), 164–173 (2013).

[21] Palmer S., *Vision Science*: *Photons to Phenomenology*, MIT Press, 1999.

[22] Rzeźniczak T., "Implementation aspects of data visualization based on map of attributes", *Journal of Theoretical and Applied Computer Science*, Vol. 6(4), 24–36 (2012).

[23] Rzeźniczak T., "Applying figural goodness to data visualization", *Biuletyn Instytutu Systemów Informatycznych*, Nr 11, 23–31 (2013).

[24] SNOMED CT, International Health Terminology Standards Development Organization, http://www.ihtsdo.org/snomed-ct/, accessed November 2015.

Algorytmy ewolucyjne w optymalizacji Mapy Atrybutów

T. RZEŹNICZAK

Mapa atrybutów (MoA, z ang. *Map of Attributes*) to technika wizualizacji, która pozwala konstruować graficzną reprezentację abstrakcyjnych obiektów. Celem działania techniki jest wsparcie rozpoznawania graficznej reprezentacji obiektów przez efektywne wykorzystanie percepcyjnych zdolności człowieka. Pewną trudnością stosowania MoA jest złożoność obliczeniowa znajdywania optymalnej mapy. W artykule przedstawiono heurystyczne podejście bazujące na algorytmach ewolucyjnych (EA, z ang. *evolutionary algorithms*) do konstruowania wizualizacji MoA. Metoda została zbadana z wykorzystaniem repozytorium jednostek chorobowych jako zbioru danych wejściowych. Kilka różnych konfiguracji EA zostało zweryfikowanych, były to konfiguracje z zastosowaniem dobrze znanych operatorów ewolucyjnych, jak również konfiguracje z nowo zaproponowanymi operatorami dla macierzowej reprezentacji chromosomu. Artykuł prezentuje szczegółowe wyniki oraz analizę przeprowadzonych eksperymentów.

Słowa kluczowe: algorytmy ewolucyjne, wizualizacja danych, operatory ewolucji.