TECHNICAL TRANSACTIONS | CZASOPISMO TECHNICZNE
AUTOMATIC CONTROL | AUTOMATYKA

2-AC/2013

ŁUKASZ SOSNOWSKI[*]

APPLICATIONS OF COMPARATORS IN DATA PROCESSING SYSTEMS

ZASTOSOWANIA KOMPARATORÓW W SYSTEMACH PRZETWARZANIA DANYCH

Abstract

This paper shows practical examples of compound object comparators and the application of the theory in various fields related to data processing systems. One can also find the necessary theoretical background needed to understand the examples.

Keywords: *compound objects, compound object comparators, networks of comparators, fuzzy sets, rough mereology*

Streszczenie

Niniejszy artykuł przedstawia praktyczne zastosowania teorii komparatorów obiektów złożonych w różnych aspektach dotyczących systemów przetwarzania danych. Dodatkowo został umieszczony skrót materiału teoretycznego pozwalającego na zrozumienie przykładów oraz ogólnej tematyki komparatorów.

Słowa kluczowe: *obiekty złożone, komparatory obiektów złożonych, sieci komparatorów, zbiory rozmyte, mereologia przybliżona*

[*] M.Sc. Łukasz Sosnowski, Ph.D. Doctoral Studies, e-mail: l.sosnowski@dituel.pl, Systems Research Institute, Polish Academy of Sciences.

Symbols

$C(a, B)$	– compound objects comparator
A	– set of input objects
B	– set of reference objects
a	– input object, an element of A
b	– reference object, an element of B

1. Introduction

Data processing systems employ many techniques that allow the synthesis of analysis, and interpretation of data. One of the subsets is a class of decision support systems. They implement solutions based on neural networks [2], evolutionary algorithms [13], immune systems, energy based [7] solutions, etc. All of these techniques are suitable for solving a selected class of problems. They require a specific approach to the construction of the solution for particular problems. The theory of compound objects as well as a network of comparators is an alternative and complementary approach to the already used methods for solving decision problems. It is a kind of methodology which helps to construct one's own solution which solves these kinds of problems. It is characterized by a common approach to every kind of problem, which greatly facilitates the management of unknown cases. Optimal solutions are not guaranteed in many cases, but in a reproducible and easy implemented way, they give the opportunity to obtain satisfactory results. The theory of comparators was described in earlier papers [15, 16, 17, 21, 22, 27] and it has evolved into the form of networks [23, 26]. This network has the ability to learn from using the known techniques [9].

The article is organized as follows: the introduction is located in this section; chapter 2 consists of an abstract of the theoretical basis used in the case of the problems described; chapter 3 presents various examples of applications developed by means of the methods presented. These examples exhibit different degrees of complexity, from very simple to advanced practical problems. The final part contains a section providing a summary of issues presented and the bibliography upon which this article is based.

2. Preliminaries

2.1. Ontology

The terms are borrowed from philosophy, but it is now also frequently found in the field of artificial intelligence. The formal definition (one of many) was introduced in 2001, see [4, 8]. Its meaning is as follows: Ontology is a system marked as O which specifies the structure of concepts, relationships between them, as well as the theory defined on a model. It is defined in the following form:

$$O = \{C, R, H_c, \text{rel}, A, L\} \tag{1}$$

where:

- C – is the set of all concepts of the model and the concept is called the idea of representing a group of objects with common characteristics,
- R – set of non-taxonomic relations defined as signed connections between concepts [1],
- H_c – collection of taxonomic relationships between the concepts,
- rel – defined non-taxonomic relationships between the concepts,
- A – set of axioms,
- L – lexicon specifies how to understand concepts (including relations).

L is the following set:

$$\{L_c, L_r, F, G\} \tag{2}$$

where:

- L_c – lexicon definitions for concepts,
- L_r – lexicon definitions for a set of relationships,
- F – references to concepts,
- G – references to relationship.

In this case, ontology will be used as a set of concepts describing objects and its structure using relations. It will help to describe the features of objects as well as designate a features reduct [25]. It will be a necessary tool for the process of recognition and identification.

2.2. Simple and compound objects

It can be said that the information-based world consists of objects that can be both the source and recipient of information. Objects that exist in the surrounding area can be divided into compound objects (X_c) and simple objects (X_s). Simple objects will be called atomic, indivisible objects which are ordinary entities that have certain characteristics, but are not composed of other objects. While compound objects have their structure and may consist of other objects, either simple or compound. The simple object is any element of the real world having its representation capable of being expressed by the adopted ontology (O). In addition, the following properties arising from their ontological representation may be assumed:

1. An object always belongs to a certain class or a fixed number of classes in ontology. A single object may belong to several classes.
2. An object has a property within a class. Features may vary from class to class.
3. An object may be related to other objects in the same ontology.

The compound object is composed of other objects defined by means of ontologies (connects them) and forms a new entity. The compound object has its specification describing structure, relations and connections between sub-objects. Compound objects satisfy the following additional properties [26]:

1. A minimum of two objects can be extracted from them and can be independent entities.
2. Component objects are interrelated with ontology O.

2.3. Compound object comparator

Compound object comparators are multi-layered structures consisting of several closely interlinked components for determining the similarity between objects. A comparator can be formally described with the following function:

$$C_B : A \rightarrow 2^{B\times[0,1]} \tag{3}$$

where:

A: is a set of input objects,
B: is a set of reference objects.

Comparator outcomes take the form of weighted subsets of reference objects:

$$C_B(a) = F\left(\{b, g(\mu(a,b)) : b \in B\}\right) \tag{4}$$

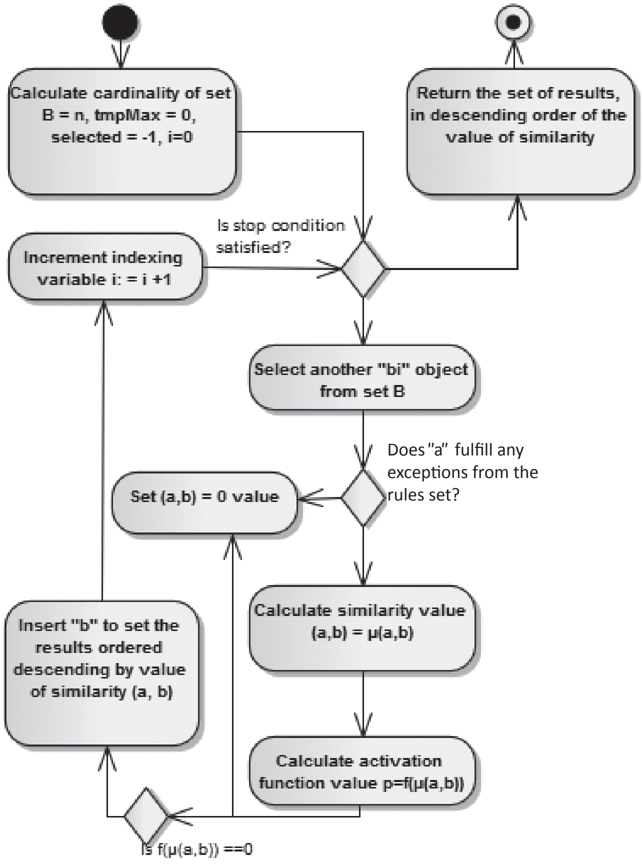
Calculate cardinality of set B = n, tmpMax = 0, selected = -1, i=0
Return the set of results, in descending order of the value of similarity
Is stop condition satisfied?
Increment indexing variable i: = i +1
Select another "bi" object from set B
Does "a" fulfill any exceptions from the rules set?
Set (a,b) = 0 value
Calculate similarity value (a,b) = μ(a,b)
Insert "b" to set the results ordered descending by value of similarity (a, b)
Calculate activation function value p=f(μ(a,b))
Is f(μ(a,b)) ==0

Fig. 1. UML activity diagram of compound object comparator

where:

F – is a function responsible for filtering partial results, e.g. min, max, top,
$\mu(a, b)$: – is a membership function of the fuzzy relation [5], which returns a similarity degree between $a \in A$ and $b \in B$,
$g(x)$: – is an activation function which filters out results that are too weak.

The following is put:

$$g(x) = \begin{cases} 0 : x < p \\ x : x \geq p \end{cases} \tag{5}$$

where:

p – denotes the lowest acceptable similarity [15].

One may also introduce some constraints which make $\mu(a, b_i) = 0$ based on the so-called exception rules [23].

2.4. Information granule

The first definition of information granules was proposed by Lotfi Zadeh [34, 35] in the 70's and read as follows: *'An information granule is a clump of objects of some sort, drawn together on the basis of indistinguishability, similarity, or functionality'* [15]. In this case, the granule is used to represent input object and its closest environment built from reference objects. We will build them through different phases of comparisons.

2.5. Rough mereology

The theory of mereology was developed by S. Leśniewski in the second and third decades of the twentieth century. The basic idea is the relation of 'being part of the whole'. Rough mereology [11] is an extension of classical mereology. It answers the question 'to what degree X is a part of Y' by using the rough inclusion defined as:

$$\mu(X, Y) : U \rightarrow [0, 1] \tag{7}$$

where:

U – is a finite set of objects called 'Universe'.

This function has been developed as a tool to approximate reasoning and has the following properties:

1. $\mu(x, y) \in [0, 1]$
2. $\mu(x, y) = 1, \forall x \in U$
3. if $\mu(x, y) = 1$ and $\mu(x, y) = 1$ then $\mu(x, z) \geq \mu(y, z)$ for any triple $x, y, z \in U$
4. there is a null object n, in the case of which $\mu(n, x) = 1, \forall x \in U$

The most common form of the rough inclusion function is based on the quantity of elements. It is defined as:

$$\mu(X, Y) = \frac{\text{card}(X \cap Y)}{\text{card}(X)} \tag{8}$$

Here, mereology is a very important part of the solution. In many cases, the entire compound object cannot be compared. The structure of objects is often very complicated. The difficulty lies in choosing appropriate measures of distance. It turns out that during decomposition of the object into sub-objects (sometimes repeatedly), a level of complexity of the structure is obtained. Therefore measuring the distance between sub-objects becomes easy. Use of mereology based on the similarity of sub-objects facilitates the determination of a degree of resemblance of the objects at higher layers (more complex).

2.6. Network of comparators

Networks of comparators are one-way networks designed for examining the similarity of compound objects in an adaptive way. This means that further layers can take into account results obtained in the previous layers. At the same time, they allow for the building of complex (composite) structures based on the captured specification of the compound object. There are two basic types of networks categorized according to their approach to the object:

Type 1 – network of monolithic objects called the homogeneous network.

Type 2 – network of structural objects called the heterogeneous network.

The network is composed of layers. They communicate through connections of their elements. Connections between neighbouring layers may be formed in several ways, depending on the type of network in question.

At the beginning of the process, the element *a* in question is put at the input of the network. It is a point to start creating an information granule around it (in accordance with the definition 2.4). The granule consists of reference objects connected through the similarity relation with input object *a.*

Elements located in a layer are not connected. Each comparator examines the selected features and returns the independent results (individual comparators perform their calculations independently). The network structure is based on groups of independent features and characteristics or analysis of the structure of object. It depends on the type of network used. The network layer can be derived either from the context associated with a group of features (treating the object as a single entity) or from the analysis of the structure of object and its relations (processing object as part of another object or considering the the object to be a set of smaller sub-objects). In the latter case, it is possible to make generalizations of object *a* as well as the decomposition of sub-objects. Depending on the position of the object in the structure (context), various object relationships are considered. Both relationships make the input granule (around *a*) grow with sub-objects or over-objects. For details about subtypes of networks see [23, 24].

2.6.1. Elements of the network

The general scheme of the network is shown in Figure 2. The network in question consists of many independent elements, i.e.:

1. *Layer* – part of a network linked to a common context of comparison. It can include common features examined by different comparators. There are three types of layers: input; intermediate; output. Input and output layers are necessary in the network, while the intermediate layer is optional. Each layer of the network has a different context in respect of the object. This context depends on the domain knowledge of the object, namely its relationship with other objects or sub-objects, e.g. both the relationship for results of decomposition of the object, or the fact that the object (in question) is a sub-object of a different object.

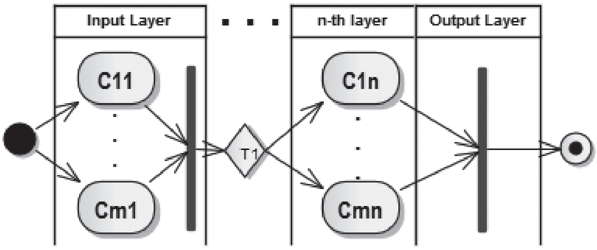

Fig. 2. General scheme of the network. C_{ji} – comparators, T_j – translators, element in output

2. *Input* – place where the input object *a* is converted into its set of representations. This is created for each comparator from the whole network. It is also a starting point of processing.
3. *Compound object comparator* – the unit enables the execution of compound object comparison, as defined in Section 2.3. There are many types of compound object comparators. Classification is listed in [18]. There may be many different types of comparators in one layer. Their selection depends on reasons, efficiency or the chosen solution to a particular problem.
4. *Translator* – component of a network associated with the adaptation of results to the context of the layer to be fed with. In other words, this element expresses the results of the previous layer (reference objects) using reference objects used in the next layer, taking into account relationships between the objects of these two layers. This element is optional due to the fact that not every network will be capable of using it or its use would be redundant in certain cases (for the same set of reference in various layers).
5. *Aggregator* – mandatory part of the network responsible for the synthesis of results obtained by comparators. This element is always present in the output layer. However, it may also be included in other layers, depending on the network structure. Aggregators are functions that operate on partial results of comparators.
6. *Output* – point where the processing results of the object in question over the network are obtained. Results are obtained in the form of a subset of reference objects which were processed by the aggregator.

2.5. Signal granule

The signal granule is based on the Information granule. In this case, the granule will be used to represent an input object and its closest surrounding area built from reference objects. It will be a representation of a signal moving through the network from layer to layer. The content will differ at every step, depending on comparisons already made and layers of the network visited. The outline of the signal granule is presented in Figure 3. The shape of the granule depends on the kind of network used and the reference set. The simplest one is built on the same reference set, so that subsequent parts of similar object are just subsets of the same reference set. In other situation they can come from totally different sets representing completely different features or groups of features.

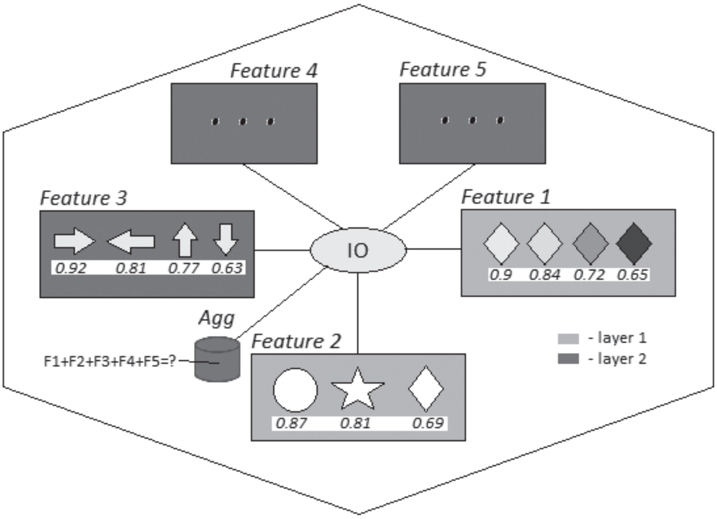

Fig. 3. General schema of signal granule used for transfer of information between comparators, aggregators, layers, in the network of comparators. Feature *n* – processed feature, IO – input object, Agg – aggregated data

3. Examples of applications

3.1. Standardization of pharmaceutical data – the introductory example

The task is to match the text data from different data sources (pharmaceutical warehouses). Each system might have a different ontology as well as local dictionaries of pharmaceutical products [15]. Input objects and reference objects have a common structure (according to the international standard – brand, form, dose etc.). Let us assume that the data sets $A = \{a_1\}$, $B = \{b_1, \mathrm{b}_2\}$ and $Z_1 = \{z_{11}\}$, $Z_2 = \{z_{21}\}$ are the following:

a_1 – 'Vitamin C, 100 mg, Acme, tablets',
b_1 – 'Vitamin E 100 mg, Acme tablets',
b_2 – 'Vitaminum C 100 mg, Acme tablets',
z_{11} – 'C',
z_{21} – 'E'.

and there is a factor $p = 0.6$. This simple example illustrates the usage of a single compound object comparator. Let us consider the similarity of elements a_1 and b_1, b_2 using the elements z_{11}, z_{21} of the sets of exceptions. The fuzzy relation, which is the base of the comparator in question, is the following:

$$\mu(a,b) = 1 - \frac{D_L(a,b)}{\max\{n(a), n(b)\}} \tag{9}$$

where:

$D_L(a, b)$ – Levenshtein distance between a and b,

$n(a)$ – length of string a,
$n(b)$ – length of string b,
and the characteristic function of comparator is max().

First of all, Levenshtein distance for the corresponding pairs is calculated and hence the following is obtained: $D_L(a_1, b_1) = 5$, $D_L(a_1, b_2) = 5$, then the value of the maximum function of the length of the string is calculated and gives respectively:

$$\max(n(a_1), n(b_1)) = 31 \text{ and } \max(n(a_1), n(b_2)) = 31$$

The following formula with the membership function is obtained: $\mu(a_1, b_1) = 0.8387$ and $\mu(a_1, b_2) = 0.8387$. One may note that as far as the terms of membership (similarity) are concerned, the two pairs are equivalent. The exceptions criterion is the only thing to be checked. Thus, for a pair (a_1, b_1) it has to be checked whether a_1 contains a sub-string defined by z_{11} (rule condition). In this case, it can be noted that the rule is satisfied, so item b_1 must be rejected for item a_1. The only thing that remains to be checked is whether the rule is satisfied for the pair (a_1, b_2). In this case it is clear that the string represented by a_1 does not contain a string represented by z_{21} ('E'). Only the pair (a_1, b_2) is to be considered further in accordance with the algorithm. Finally, the value of the activation function is checked. For the pair in question, the function value is 0.8387, which is within the range [0.6, 1], which sets the parameter p. Extract the maximum value of membership function. In this case, it is 0.8387. Thus, in this example, element $((a_1, b_2), 0.8387)$ is the result of standardization.

The above mentioned example shows how easy it is to use comparators for resolving data processing problems. This particular usage is very simple, based only on a single comparator. It can be compared with single neuron solutions known from neural networks [2]. Despite the fact that this is only a single comparator, it is a very powerful solution. We can choose many different membership functions and in this way, solve many other processing problems only with this change.

3.2. Contour map identifications

This example shows the practical implementation of the comparators theory in query by example solution of searching contour area maps in Poland (voivodships, counties, communes). The algorithm devised to do that, introduced in [21], consists of three phases: segmentation, granulation [10] and identification. Firstly, objects are acquired, e.g., a given image of contour map or a drawn map. Secondly, granules of objects are extracted, their characteristics are computed, and synthesized into granular descriptions of objects. Only two types of characteristics are considered, although the framework is open for adding more types. In the case of the first type, referred to as coverage, the histogram technique is used (see [3]) to compute a vector of overlaps of image's granules with area that the image represents. In the case of the second type, referred to as contour, linguistic description [28] of directions of lines connecting the extrema points of the area's contour within each granule is produced. In the third phase, similarities between the input and reference objects are computed with respect to each type of granular description, and similarity scores are synthesized. For linguistic descriptions of contours, the following membership function is defined:

$$\mu_{\mathrm{contour}}(a, b) = 1 - \frac{D_L(a, b)}{\max\{n(a), n(b)\}} \tag{10}$$

where:

$D_L(a, b)$ – Levenshtein distance between linguistic descriptions of objects a and b,
$n(a)$ – length of string a,
$n(b)$ – length of string b.

With regard to the granules' coverage, the following may be considered:

$$\mu_{\mathrm{cov}\,erage}(a, b) = 1 - \frac{\sum_{i}^{n\times m} |\,\mathrm{cov}_i^a - \mathrm{cov}_i^b\,|}{n \times m} \tag{11}$$

where:

cov_i^a – factor of number of pixels belonging to interior area of granule i of object a to all pixels of granule,
n, m – granulation parameters.

For the purposes of this example, the aggregated similarity will be used:

$$\mu_{agg}(a, b) = \frac{1}{2}\left(\mu_{\mathrm{contour}}(a, b) + \mu_{\mathrm{cov}\,erage}(a, b)\right) \tag{12}$$

Finally, an additional procedure is implemented for cases that meet the following conditions:

1. If a given reference object was not chosen for any investigated object, then it should be used for the most similar unidentified object, even if its degree of similarity is not greater than the activation threshold.
2. If a given reference object was chosen for many investigated objects, then it should be used for the most similar of them and the remaining ones should be re-identified, excluding the already used reference objects.

The method presented above introduces the use of comparators in networks. There are 2 comparators in one layer and an aggregator. There are a lot of possible variations of implementing aggregators, e.g. using an elections algorithms [20]. The advantages of this method is the ease of construction solutions and that it is a human readable solution. We can easily modify the solution by adding more comparators that are specifically defined tasks. The weak point is the manual selection of a feature to examine. It requires a domain knowledge in form of an expert or well-built ontology and instances described by concepts. In practice, mixed solutions are preferred, that there is an ontology already built and also access to expert providing guidance.

3.3. Identification of authors of scientific publications

This example is based on the data processed in the SYNAT project (abbreviation of Polish 'SYstem NAuki i Techniki'), which was exported and prepared in the form of a flat file for the identification process. There is a set of authors, but saved in various publication standards with possible minor errors (mistypings). The task is to identify the same objects inside the collection and produce a unique set of references. The input data are in the form of a table with three columns (id_instance, name, surname). The target set of input objects is very large (millions of records). To solve this problem, the previously described homogeneous network [23] can be used. This network has 3 layers: input; output; intermediate. The input layer consists of three comparators running concurrently: C_{sfl} – surname first letter comparator; C_{nfl} – name first letter comparator; C_{slen} – surname length comparator. The first two examine the first letter of the name and surname. The third one examines the length of surname with certain tolerances in both directions (e.g. one character). All of these comparators examine computationally simple features (so they can be efficiently computed), but they will be able to limit the search space for the next layer.

Table 1

Distribution of data in the input layer of the network of authors' identification

Initial of surname	Quantity	Initial of name	Quantity	Length of surname	Quantity
s	987	m	1027	6	1820
m	843	j	1014	7	1567
b	812	a	875	5	1515
c	645	s	814	8	1337
k	607	c	625	4	1016
l	588	r	560	9	886
h	577	d	518	10	494
g	507	p	463	3	439
p	489	k	435	11	285
w	479	g	432	12	170
d	449	t	428	2	153
r	438	l	407	13	119
t	388	e	369	14	83
a	355	h	355	16	56
f	315	b	304	15	43
v	293	f	252	17	36
n	262	n	243	19	21
j	213	y	219	18	20
e	178	i	167	20	13
z	170	w	167	21	9
o	154	v	141	23	7
y	145	o	92	22	5
i	94	z	57	24	2
u	55	x	57	1	1
x	34	u	57	26	1
q	21	q	20	–	–

The intermediate layer uses the results of the input layer by feeding the reference set with the results. In this layer, there are five comparators investigating the similarity of attributes: name, surname, sorted acronym, *nlm* category publications and co-authors. Results of these calculations are synthesised by the global aggregator. At the end, the result is obtained and is interpreted as follows: if the network indicates similarity to a reference object, and it meets the minimum requirements for quality (activation parameter *p*), then the input object is matched with an already existing element of the reference set. If the answer of the network is empty, then the input object is treated as a unique object (new), and it is added to the set of reference objects. Such proceedings shall be set for each element of the input.

The scheme used in the network is shown in Figure 4. The sample of experiments consisted of 10098 records. Distribution of data for the comparator input layer is presented in Table 1. It can be noted that the largest class of objects amounts respectively to: ‘s’ – 987, ‘m’ – 1027, ‘6’ – 1820. The aggregator of the input layer selects data to be passed to the second layer. It calculates the arithmetic mean of the values of partial similarities (coming from a single comparator) for each pair (a, b_i), where b_i is a reference object. Subsequently, only those with the highest value are selected. The last two comparators from the intermediate layer will be mentioned. In this case, some additional domain knowledge of an input object is needed. Information about the source publication and its assigned category is needed, too. The comparator examines whether the defined publication falls into the categories of publications assigned to authors who were objects in the reference set. In the future, one may consider relevant categories and explore partial membership in a particular category (as a membership in the related class). The last comparator examines the co-authors, e.g., whether a given author has publications with co-authors from the set of reference.

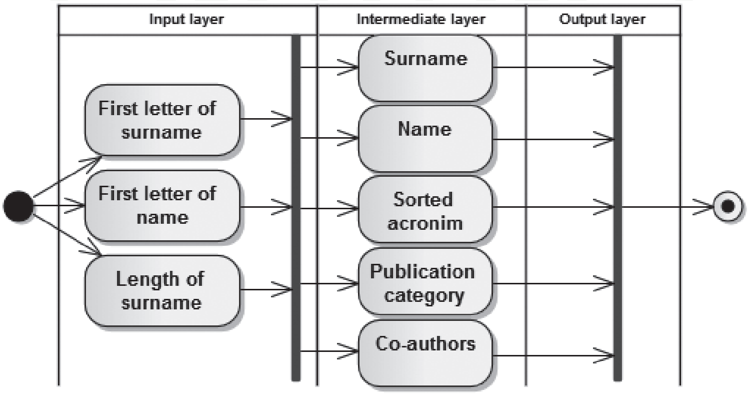

Fig. 4. Network of compound object comparators for identification of authors of scientific publications

This example is quite similar to hash algorithms known in literature to handle problems with huge data cluster processing. It consists of the pre-selection of objects, among which the final result is searched. The preselection is performed by means of simple operations that can be done very efficiently, while the exact match requires more processing power. The strength of this solution is the ability to build complex processing solutions composed of simple parts, fully comprehensible and manageable by humans.

3.4. SYNAT – identification of ‘semantic blobs’

This section refers to an academic project in the area of semantic search over a repository of scientific publications. One of the tasks was parsing and interpreting publication references.

Interpreting means understanding which part of reference signifies an author, title, year of publication etc. Such items should be interpreted, parsed and loaded as publications themselves, but the online text analysis algorithms may not handle them credibly enough. In the SYNAT project, there is the strategy of loading such descriptions into the repository as unparsed strings with no properties and related instances assigned. Then, there is a place for using a specific module based on the network of compound objects comparators for extracting important information from such strings. The idea is to discover internal structures of input bibliography descriptions by comparing their dynamically identified components with various types of objects already stored in the repository. Such components can then be treated as new entries in the system, ready for further analysis. The heterogeneous network is considered in Figure 5.

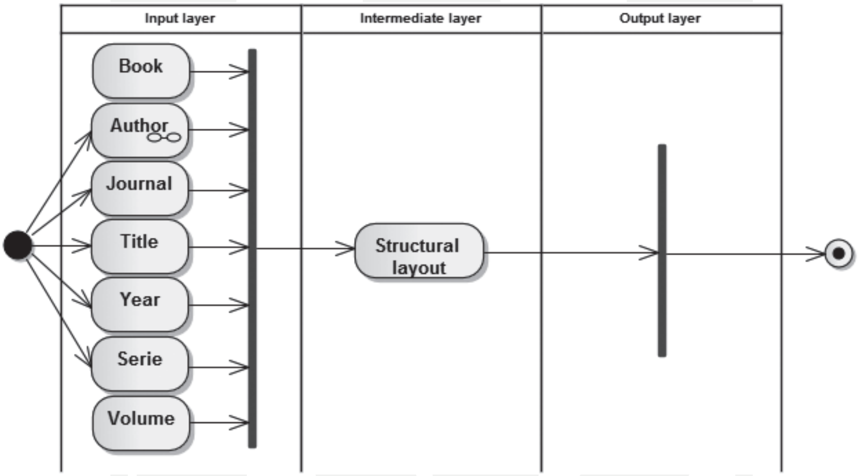

Fig. 5. Network for the identification of publication authors

The process begins with cutting the input text into pieces, e.g. by means of characters such as dots, commas, etc. For it is not known which pieces correspond to particular types of bibliography item components (more formally, properties and relations of the publication in question with some other objects), the first network's layer focuses on the computation of resemblance degrees of all pieces to reference sets maintained in the repository for scientists, authors, publication titles, years, journals and others [24].

Table 2 shows a simplified example of the outcome of stages of text decomposition and resemblance calculation. As in certain cases, the assignments of sub-strings to particular component types are problematic, the identification process is strengthened by the analysis of a bibliography item's structure at the next network layer. Table 3 presents several examples of reference structural objects, i.e. structures of bibliography items already stored in the repository. The structural layout comparator in Figure 5 determines the most reasonable hypothesis about the input's structure basing on such reference objects, in combination with information obtained in the previous layer. The final assignment is retrieved by following the most probable structure. This last step can be implemented in various ways. One of the possible solution is using a rough mereology approach on the final aggregation stage.

Table 2

Input and its decomposition with degrees of resemblance of its components to the corresponding reference sets

Example of the input object	Comp.	Substring	*A*	*T*	*Y*	*P*	*J*
M. Szczuka, Ł. Sosnowski, A. Krasuski, K. Kreński, ‘Using Domain Knowledge in Initial Stages of KDD: Optimization of Compound Object Processing’, Fundam. Inform., 2014, to appear.	1	M. Szczuka	0.83	0	0	0	0
	2	Ł. Sosnowski	0.83	0	0	0	0
	3	A. Krasuski	0.85	0	0	0	0
	4	K. Kreński	0.84	0	0	0	0
	5	Using (…)	0	0.75	0	0	0
	6	Fundam. (…)	0	0.42	0	0	0.55
	7	2014	0	0	1	0	0
	8	to appear	0	0	0	0	0

Table 3

Sample of reference structures

Abbreviation dictionary		Reference structural objects
A – authors	*T* – title	*A T B S V Y P*
J – journal	*B* – book	*E T B S Pb Y V*
Y – year	*Pb* – publisher	*A T B Y*
P – pages	*S* – series	*A T J Y*
V – volume	*E* – editors	*A T S Pb Y*
N – note ...		...

The solution described above is a popular scientific publication processing problem. There have been many approaches to solve this problem. One of them is the network of comparators. This example shows that comparators provide a universal approach to many processing problems using the same techniques and tools. In this solution, it should be noted that there is an incremental algorithm used which adds unknown cases to the reference set. It is one of the methods of learning classified within the class of lazy learning.

3.5. Optical character recognition of digits

In order to achieve automatic character recognition, a network of compound object comparators shown in Figure 6 has been developed. Before the proper processing is initiated [6], using the designed network, the pre-processing stage for every object has to be ensured. Consequently, a segmented string as a set of individual images is obtained. Each image represents one digit (as a single object). Next, each of these objects must be converted to binary scale (only black & white). After that each image is cut in such a way that each edge of the newly created image is one pixel further from the black edge of the font.

The network constructed is a type I [23] (for monolithic objects), made up of three layers representing particular contexts. The first layer covers a very general nature (rough) features. They can significantly reduce the cardinality of the reference set. The second layer relates to an in-depth analysis of the image [12], thanks to which the final answer is obtained (decision).

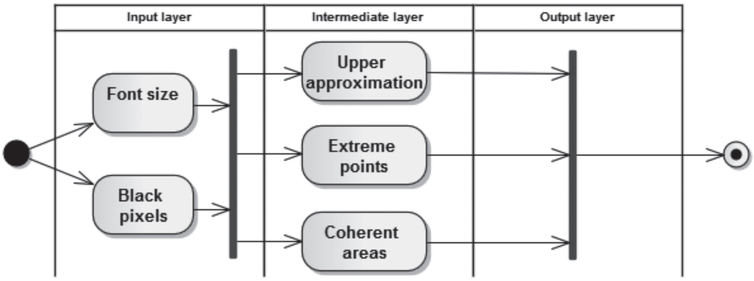

Fig. 6. Network of comparators for the OCR task

The last layer in the classical way only contains an aggregator that performs the synthesis of previous results. The reference set consists of objects which are images of searched characters (digits) grouped in terms of different sizes and font types. In this example, the following structure of the set is used: the size of the font (10, 14, 18, 24, 36, 48, 60); types of fonts (Times New Roman, Arial, Verdana, Courier). The cardinality of the entire reference set is 280 elements. The purpose of the first layer is to limit the cardinality of the reference set. The method applied is to analyse the size of the font and distribution of pixels in the image.

A set of fonts with the size closest to the input object is selected. Then synthesis of the results returned by both comparators is performed. After that, reference objects which meet the imposed specific qualifying criteria are selected. The result of the aggregator feeds the next layer, as a set of reference. In the intermediate layer, there are three comparators: Upper approximation; Extreme points; Coherent area. The first comparator compares images arising from the granules as a result of the granulation process, but only those in which there are black pixels (activation of the granule). This means that the comparison of images is converted to a very low resolution (m×n – granulation parameters).

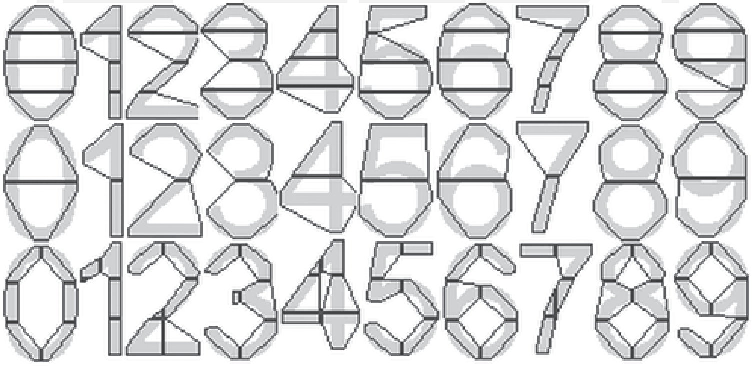

Fig. 7. Reference digits prepared for comparisons with respect to their shapes (with granulation resolution parameters 1 × 3, 2 × 2, 2 × 3 displayed in consecutive rows)

Another comparator deals with the comparison of geometric obtained by means of the connection of extreme points of subsequent granules (see Figure 7). Further action is to take contours of subsequent shapes and make comparisons with methods developed and published earlier [21]. The third comparator is prepared to compare the coherent areas within objects. In this way, easier recognition of following digits: 0,4,6,8,9 is supported. These figures represent 50% of the possible characters defined for the task. This comparator not only detects their presence, but also indicates the most resembled areas in objects of reference. Consequently, the calculated results of the subsequent comparator feed output layer as input data of the aggregator. The aggregator calculates an arithmetic mean of resemblance for specific pairs (the input object and the reference object) and the subsequent comparators in question. In consequence, results are obtained in the form of a reference object or objects. In this way, identification of individual objects is performed. It is worth noting that individual comparators operate at different representations of the input object and reference objects, although each time they refer to the same signal granule (information granule [14]).

The method presented above is one of the many known and available methods in the range of digits recognition problems. This conclusion confirms the possibility of using it in many different fields of applications. The method in this regard does not discover new things, but allows using well-known methods from other applications. This gives the possibility to use the experience gained previously to instantly develop new solutions.

3. Summary

This article presents compiled examples of different applications of comparators in the field of data processing systems, putting special emphasis on decision support systems. The examples collected show how easy the implementation is for various problems (e.g. identification, matching, searching, etc.). In each of the reported cases, the procedure is analogous, which is undoubtedly a very big strength of this solution. The examples presented are achievements of earlier researches. This made it possible to collect them in this work in ahigh concentration. This allows for agood understanding of these applications and the theory in question.

Future work focuses on other applications of the framework. One of them is the identification of risks in the fire rescue actions used in the ICRA [19] project (www.icra-project.org). This is a wide field of possible applications of the said method.

The next step will also be to develop the learning methods with regard to the solution in question. One of them will be an evolutionary algorithm for searching optimal weights of comparator importance in the layer. Another will be based on the back propagation algorithm for improving the basic parameters of single comparator connections between layers.

Another possible step is the publication of the framework's implementation written in Java as a freeware licence for greater and easier access for researchers. This is important in the case of gaining popularity for this method of AI.

Acknowledgements:

The research was supported by the Polish National Centre for Research and Development (NCBiR) – Grant No. O ROB/0010/03/001 in the frame of Defence and Security Programmes and Projects: 'Modern engineering tools for decision support for commanders of the State Fire Service of Poland during Fire&Rescue operations in the buildings'.

References

[1] Dean Allemang and James Hendler. *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.

[2] Bishop C.M., *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[3] Deb S. (ed.), *Multimedia Systems and Content-Based Image Retrieval*, IGI Global, 2004.

[4] Gliński W., *Ontologies – an attempt to put terminological chaos in order*, [in:] *From scientific information to the technology of information society: collective work*, Miscellanea Informatologica Varsoviensia, SBP, 2005.

[5] Kacprzyk J., *Multistage Fuzzy Control: A Model-Based Approach to Fuzzy Control and Decision Making*, Wiley 1997.

[6] Kuznetsov S.O., *Pattern Recognition and Machine Intelligence*, 4th International Conference, PReMI, 2011.

[7] LeCun Y. et al., *A Tutorial on Energy-Based Learning*, [in:] Bakir G. et al., Predicting Structured Data, MIT Press, 2006.

[8] Maedche A., Staab S., *Comparing Ontologies – Similarity Measures and a Comparision Study*, Internal report No. 408, University of Karlsruhe, 2001.

[9] Mitchell T.M., *Machine Learning*, McGraw-Hill, 1997.

[10] Pedrycz W., Kreinovich V., Skowron A. (Eds.), *Handbook of Granular Computing*, Wiley 2008.

[11] Polkowski L., *Approximate Reasoning by Parts an Introduction to Rough Mereology*, Springer 2011.

[12] Reed T., *Digital Image Sequence Processing, Compression and Analysis*, CRC Press, 2005.

[13] Rutkowski L., *Computational Intelligence*, Methods and Techniques, Springer 2008.

[14] Skowron A., Stepaniuk J., Peters J.F., *Towards Discovery of Relevant Patterns from Parametrized Schemas to Information Granule Construction*, [in:) (Eds.) I. Masahiro, H. Shoji, T. Shusaku, *Rough Set Theory and Granular Computing*, Springer Berlin Heidelberg 2003, 97-108.

[15] Sosnowski Ł., *Inteligentne dopasowanie danych przy użyciu teorii zbiorów rozmytych w systemach przetwarzania danych*, Analiza systemowa w finansach i zarządzaniu, T. 11 pod redakcją prof. J. Hołubca, 2009.

[16] Sosnowski Ł., *Budowa systemu porównywania obiektów złożonych*, Analiza systemowa w finansach i zarządzaniu, T. 12 pod redakcją prof. J. Hołubca, 2010.

[17] Sosnowski Ł., *Identification with compound object comparators – technical aspects*, Techniki informacyjne teoria i zastosowania, T. 1 pod redakcją prof. J. Hołubca, 2011.

[18] Sosnowski Ł., *Identyfikacja obiektów złożonych przy użyciu komparatorów*, Techniki informacyjne teoria i zastosowania, T. 2 pod redakcją prof. Myślińskiego, 2012.

[19] Sosnowski Ł., Pietruszka A., Krasuski A., Janusz A., *A resemblance based approach for recognition of risks at the fire ground*, [in:] Active Media Technology Proceedings, 2014, to appear.

[20] Sosnowski Ł., Pietruszka A., Łazowy S., *Election Algorithms Applied to the Global Aggregation in Networks of Comparators*, [in:] FedCSIS 2014 Aaia Workshop, to appear.

[21] Sosnowski Ł., Ślęzak D., *Comparators for Compound Object Identification*, [in:] Proc. of RSFDGrC, LNAI 6743, Springer 2011, 342-349.

[22] Sosnowski Ł., Ślęzak D., *RDBMS Framework for Contour Identification*, [in:] Proc. of the international workshop CS&P 2011, Białystok University of Technology, 2011, 487-498.

[23] Sosnowski Ł., Ślęzak D., *Networks of Compound Object Comparators*, [in:] FUZZ-IEEE 2013, IEEE International Conference on Fuzzy Systems, Hyderabad, India, July 7–10 2013, Proceedings. IEEE, 2013.

[24] Sosnowski Ł., Ślęzak D., *How to design a network of comparators*, [in:] Brain and Health Informatics, 2013, 389-398.

[25] Stawicki S., Ślęzak D., Recent advances in decision bireducts: *Complexity, heuristics and streams*, [in:] Lingras P., Wolski M., Cornelis C., Mitra S., Wasilewski P., eds.: RSKT, Volume 8171 of Lecture Notes in Computer Science., Springer 2013, 200-212.

[26] Szczuka M., Sosnowski Ł., Krasuski A., Kreński K., *Using Domain Knowledge in Initial Stages of KDD: Optimization of Compound Object Processing*, Fundamenta Informaticae, 2014.

[27] Ślęzak D., Sosnowski Ł., *SQL-Based Compound Object Comparators: A Case Study of Images Stored in ICE*, [in:] Proc. of ASEA, CCIS 117, Springer 2010, 304-317.

[28] Szczepaniak P., *Computational Intelligence and Applications*, Springer-Verlag, Heidelberg, New York 1999.

[29] Zadeh L., *Towards a theory of fuzzy information granulation and its certainty in human reasoning and fuzzy logic*, Fuzzy Sets Systems, 90, 1997, 111-127.

[30] Zadeh L., *Key roles of information granulation and fuzzy logic in human reasoning, Concept formulation and computing with words*, Proceedings of IEEE 5th International Fuzzy Systems.