Distributed Version of Algorithm for Generalized One-Sided Concept Lattices

Peter Butka, Jozef Pócs, and Jana Pócsová

Abstract. In this paper we provide the distributed version of algorithm for creation of model Generalized One-Sided Concept Lattices (GOSCL), special case for fuzzy version of data analysis approach called Formal Concept Analysis (FCA), which provide the conceptual model of input data based on the theory of one-sided concept lattices and was successfully applied in several domains. GOSCL is able to work with data tables containing the different attribute types processed as fuzzy sets. One problem with the creation of FCA model is computational complexity. In order to reduce the computation times, we have designed the distributed version of the algorithm and showed its applicability on the generated data set. The algorithm is able to work well especially for data where number of newly generated concepts is reduced (like for sparse input data tables).

Peter Butka
Technical University of Košice,
Faculty of Electrical Engineering and Informatics,
Department of Cybernetics and Artificial Intelligence,
Letná 9, 040 01 Košice, Slovakia
e-mail: peter.butka@tuke.sk

Jozef Pócs
Palacký University Olomouc, Faculty of Science,
Department of Algebra and Geometry,
17. listopadu 12, 771 46 Olomouc, Czech Republic, and
Mathematical Institute, Slovak Academy of Sciences,
Grešákova 6, 040 01 Košice, Slovakia
e-mail: pocs@saske.sk

Jana Pócsová
Technical University of Košice, BERG Faculty,
Institute of Control and Informatization of Production Processes,
Boženy Němcovej 3, 043 84 Košice, Slovakia
e-mail: jana.pocsova@tuke.sk

F. Zavoral et al. (eds.), *Intelligent Distributed Computing VII,*
Studies in Computational Intelligence 511,
DOI: 10.1007/978-3-319-01571-2_15,

1 Introduction

The large amount of available data and the needs for their analysis brings up the challenges to the area of data mining. It is evident that methods for different analysis should be more effective and understandable. One of the conceptual data mining methods, called Formal Concept Analysis (FCA, [7]), is an exploratory data analytical approach which identifies conceptual structures (concept lattices) among data sets. FCA has been found useful for analysis of data in many areas like knowledge discovery, data/text mining, information retrieval, etc. The standard approach to FCA provides the method for analysis of object-attribute models based on the binary relation (where object has/has-not particular attribute). The extension of classic approach is based on some fuzzifications for which object-attribute models describe relationship between objects and attributes as fuzzy relations. From the well-known approaches for fuzzification we could mention an approach of Bělohlávek [2], an approach of Krajči [12], the approach based on the multi-adjoint concept lattices [13], and also work by one of the authors generalizing the other approaches [15].

In practical applications, so-called one-sided concept lattices are interesting, where usually objects are considered as a crisp subsets (as in classical FCA) and attributes are processed as fuzzy sets. In case of one-sided concept lattices, there is strong connection with clustering (cf. [10]). As it is known, clustering methods produce subsets of a given set of objects, which are closed under intersection, i.e., closure system on the set of objects. Since one-sided concept lattice approach produces also closure system on the set of objects, one can see one-sided concept lattice approaches as a special case of hierarchical clustering. Several one-sided approaches to FCA were already defined, we mention papers of Krajči [11] and Yahia & Jaoua [1]. These approaches allow only one type of attribute (i.e., truth degrees structure) to be used within the input data table. In our previous paper [3] we have introduced the necessary details and incremental algorithm for model called GOSCL (Generalized One-Sided Concept Lattice), which is able to work with the input data tables with different types of attributes, e.g., binary, quantitative (values from interval of reals), ordinal (scale-based), nominal, etc.

One of the problems of the FCA-based methods is computational complexity, which generally can be (in worst case) exponential. We have analyzed several aspects of GOSCL complexity. In [4] we have shown that for fixed input data table and attributes time complexity of GOSCL is asymptotically linear with the increasing number of objects. This is based on the fact that after some time (which is specific for the data set) new concepts are added linear to the increment of objects. Moreover, in [5] we have analyzed the significant reduction of computation times of the algorithm for sparse input data tables (i.e., input tables with many zeros). However, in order to achieve the objective to produce large-scale concept lattices on real data, which can be then used in retrieval tasks or text-mining analysis, it is possible to extend our approach and re-use distributed computing paradigm based on data distribution [9]. Therefore, the main aim of this paper is to describe the possibility to distribute the computation of GOSCL algorithm for larger context based on its decomposition to several subtables with the separated (and disjoint) subsets of rows,

i.e., data table is decomposed to several smaller tables, for which small lattices are created and these are then iteratively combined (by defined merging procedure) to one large concept lattice for the whole data table. The presented algorithm is also analyzed according to the randomly generated data with different sparseness.

In the following section we provide necessary details for definition of generalization of one-sided concept lattices and the algorithm for their creation. Section 3 is devoted to the introduction of distributed approach for creation of model, also with the detailed description of the algorithm. In next section, some basic experiments are provided in order to show the potential of the algorithm for its future usage on the real data.

2 Generalized One-Sided Concept Lattices

In this section we provide necessary details about the fuzzy generalization of classical concept lattices, so called generalized one-sided concept lattices, which was introduced in [3].

The crucial role in the mathematical theory of fuzzy concept lattices play special pairs of mappings between complete lattices, commonly known as Galois connections. Hence, we provide necessary details regarding Galois connections and related topics.

Let $(P,\leq)$ and $(Q,\leq)$ be complete lattices and let $\varphi\colon P \to Q$ and $\psi\colon Q \to P$ be maps between these lattices. Such a pair (φ,ψ) of mappings is called a *Galois connection* if the following condition is fulfilled:

$$p \leq \psi(q) \quad \text{if and only if} \quad \varphi(p) \geq q.$$

Galois connections between complete lattices are closely related to the notion of closure operator and closure system. Let L be a complete lattice. By a *closure operator* in L we understand a mapping $c\colon L \to L$ satisfying:

(a) $x \leq c(x)$ for all $x \in L$,
(b) $c(x_1) \leq c(x_2)$ for $x_1 \leq x_2$,
(c) $c(c(x)) = c(x)$ for all $x \in L$ (i.e., c is idempotent).

Next we describe mathematical framework for one-sided concept lattices. We start with the definition of generalized formal context.

A 4-tuple (B,A,L,R) is said to be a *one-sided formal context* (or *generalized one-sided formal context*) if the following conditions are fulfilled:

(1) B is a non-empty set of objects and A is a non-empty set of attributes.
(2) $\mathsf{L} : A \to \mathsf{CL}$ is a mapping from the set of attributes to the class of all complete lattices. Hence, for any attribute a, $\mathsf{L}(a)$ denotes the complete lattice, which represents structure of truth values for attribute a.
(3) R is generalized incidence relation, i.e., $R(b,a) \in \mathsf{L}(a)$ for all $b \in B$ and $a \in A$. Thus, $R(b,a)$ represents a degree from the structure $\mathsf{L}(a)$ in which the element $b \in B$ has the attribute a.

Then the main aim is to introduce a Galois connection between classical subsets of the set of all objects $\mathbf{P}(B)$ and the direct products of complete lattices $\prod_{a\in A}\mathsf{L}(a)$ which represents a generalization of fuzzy subsets of the attribute universe A. Let us remark that usually fuzzy subset are considered as function from the given universe U into real unit interval $[0,1]$ or more generally as a mappings from U into some complete lattice L. In our case the generalization of fuzzy subsets is straightforward, i.e., to the each element of the universe (in our case the attribute set A) there is assigned the different structure of truth values represented by complete lattice $\mathsf{L}(a)$.

Now we provide a basic results about one-sided concept lattices.

Let (B,A,L,R) be a generalized one-sided formal context. Then we define a pair of mapping $^{\perp}:\mathbf{P}(B)\to\prod_{a\in A}\mathsf{L}(a)$ and $^{\top}:\prod_{a\in A}\mathsf{L}(a)\to\mathbf{P}(B)$ as follows:

$$X^{\perp}(a)=\bigwedge_{b\in X}R(b,a), \tag{1}$$

$$g^{\top}=\{b\in B:\forall a\in A,\ g(a)\le R(b,a)\}. \tag{2}$$

Let (B,A,L,R) be a generalized one-sided formal context. Then a pair $(^{\perp},{}^{\top})$ forms a Galois connection between $\mathbf{P}(B)$ and $\prod_{a\in A}\mathsf{L}(a)$.

Now we are able to define one-sided concept lattice. For formal context (B,A,L,R) denote $C(B,A,\mathsf{L},R)$ the set of all pairs (X,g), where $X\subseteq B$, $g\in\prod_{a\in A}\mathsf{L}(a)$, satisfying

$$X^{\perp}=g \quad \text{and} \quad g^{\top}=X.$$

Set X is usually referred as *extent* and g as *intent* of the concept (X,g).

Further we define partial order on $C(B,A,\mathsf{L},R)$ as follows:

$$(X_1,g_1)\le(X_2,g_2) \ \text{ iff } \ X_1\subseteq X_2 \text{ iff } g_1\ge g_2.$$

Let (B,A,L,R) be a generalized one-sided formal context. Then $C(B,A,\mathsf{L},R)$ with the partial order defined above forms a complete lattice, where

$$\bigwedge_{i\in I}(X_i,g_i)=\Big(\bigcap_{i\in I}X_i,\ ((\bigvee_{i\in I}g_i)^{\top})^{\perp}\Big)$$

and

$$\bigvee_{i\in I}(X_i,g_i)=\Big(((\bigcup_{i\in I}X_i)^{\perp})^{\top},\ \bigwedge_{i\in I}g_i\Big)$$

for each family $(X_i,g_i)_{i\in I}$ of elements from $C(B,A,\mathsf{L},R)$.

At the end of this section we briefly describe an incremental algorithm for creating one-sided concept lattices. By the incremental algorithm we mean the algorithm which builds the model from the input data incrementally row by row, where in every moment current model from already processed inputs is correct concept lattice for such subtable (and addition of new increment means to provide another row and update the last model, i.e., concept lattice). Let (B,A,L,R) be a generalized one-sided formal context. We will use the following notation. For $b\in B$ we denote by

$R(b)$ an element of $\prod_{a\in A} \mathsf{L}(a)$ such that $R(b)(a) = R(b,a)$, i.e., $R(b)$ represents b-th row in data table R. Further, let 1_L denote the greatest element of $L = \prod_{a\in A} \mathsf{L}(a)$, i.e., $1_L(a) = 1_{\mathsf{L}(a)}$ for all $a \in A$.

Algorithm 1. Incremental algorithm for GOSCL

Require: generalized context (B, A, L, R)
Ensure: set of all concepts $C(B, A, \mathsf{L}, R)$
$L \leftarrow \prod_{a\in A} \mathsf{L}(a)$ ▷ Direct product of attribute lattices
$I \leftarrow \{1_L\}$ ▷ $I \subseteq L$ will denote set of intents
$C(B, A, \mathsf{L}, R) \leftarrow \emptyset$
for all $b \in B$ **do**
  $I^* \leftarrow I$ ▷ I^* represents "old" set of intents
  for all $g \in I^*$ **do**
    $I \leftarrow I \cup \{R(b) \wedge g\}$ ▷ Generation of new intent
  end for
end for
for all $g \in I$ **do**
  $C(B, A, \mathsf{L}, R) \leftarrow C(B, A, \mathsf{L}, R) \cup \{(g^{\top}, g)\}$
end for
return $C(B, A, \mathsf{L}, R)$ ▷ Output of the algorithm

3 Distributed Algorithm for Generalized One-Sided Concept Lattices

In this section we provide the theoretical details regarding the computation of GOSCL model in distributed manner. It means that division to the partitions is defined. The main theorem is proved, which shows that concept lattices created for the partitions are equivalent to the concept lattice created for the whole input formal context. Then the algorithm for our approach to distribution is provided. It is based on the division of data table into binary-like tree of starting subsets (with their smaller concept lattices), which are then combined in pairs (using specified merge procedure) until the final concept lattice is available.

Let $\pi = \{B_i\}_{i\in I}$ be a partition of object set, i.e., $B = \bigcup_{i\in I} B_i$ and $B_i \cap B_j = \emptyset$ for all $i \neq j$. This partition indicates the division of objects in considered object-attribute model, where R_i defines several sub-tables containing the objects from B_i and which yields several formal contexts $C_i = (B_i, A, \mathsf{L}, R_i)$ for each $i \in I$. Consequently we obtain the system of particular Galois connections $(^{\perp_i}, ^{\top_i})$, between $\mathbf{P}(B_i)$ and $\prod_{a\in A} \mathsf{L}(a)$. Next we describe how to create Galois connection between $\mathbf{P}(B)$ and $\prod_{a\in A} \mathsf{L}(a)$. We use the method of creating Galois connections applicable for general fuzzy contexts, described in [15]. Let $X \subseteq B$ be any subset and $g \in \prod_{a\in A} \mathsf{L}(a)$ be any tuple of fuzzy attributes. We define

$$\uparrow(X)(a) = \bigwedge_{i\in I} (X \cap B_i)^{\perp_i} \quad \text{and} \quad \downarrow(g) = \bigcup_{i\in I} g^{\top_i} \tag{3}$$

Theorem 1. *Let $\pi = \{B_i\}_{i\in I}$ be a partition of object set. Then the pair of mappings $(\uparrow,\downarrow)$ defined by (3) and the pair $(^{\perp},^{\top})$ defined by (1) represent the same Galois connection between $\mathbf{P}(B)$ and $\prod_{a\in A} \mathsf{L}(a)$.*

Proof. Let $X \subseteq B$ be any subset of object set and $a \in A$ be an arbitrary attribute. Using (3) we obtain

$$\uparrow(X)(a) = \bigwedge_{i\in I} (X \cap B_i)^{\perp_i} = \bigwedge_{i\in I} \Big(\bigwedge_{b\in X\cap B_i} R_i(b,a)\Big) = \bigwedge_{i\in I} \Big(\bigwedge_{b\in X\cap B_i} R(b,a)\Big).$$

Since π is the partition of the object set B, we have $X = \bigcup_{i\in I}(X \cap B_i)$. Involving the fact, that meet operation in lattices is associative and commutative, we obtain

$$X^{\perp}(a) = \bigwedge_{b\in X} R(b,a) = \bigwedge_{b\in \bigcup_{i\in I}(X\cap B_i)} R(b,a) = \bigwedge_{i\in I} \Big(\bigwedge_{b\in X\cap B_i} R(b,a)\Big),$$

which yields $\uparrow(X) = X^{\perp}$ for each $X \subseteq B$.

Similarly, we have

$$\downarrow(g) = \bigcup_{i\in I} g^{\top_i} = \bigcup_{i\in I} \{b \in B_i : \forall a \in A,\ g(a) \le R_i(b,a)\}.$$

Easy computation shows that this expression is equal to

$$\{b \in B : \forall a \in A,\ g(a) \le R(b,a)\} = g^{\top},$$

which gives $\downarrow(g) = g^{\top}$ for all elements $g \in \prod_{a\in A} \mathsf{L}(a)$.

Algorithm 2. Distributed algorithm for GOSCL

Require: generalized context (B,A,L,R), $\pi = \{B_i\}_{i=1}^{2^n}$ - partition of B with 2^n parts
Ensure: merged concept lattice $\mathcal{C}(B,A,\mathsf{L},R)$

1: **for** $i = 1$ to 2^n **do**
2: $\quad C_i^{(n)} \leftarrow \mathrm{GOSCL}(B_i,A,\mathsf{L},R_i)$ ▹ Application of Algorithm 1 on all subcontexts
3: **end for**
4: **for** $k = n$ down to 1 **do**
5: $\quad$ **for** $i = 1$ to 2^{k-1} **do**
6: $\quad\quad C_i^{(k-1)} \leftarrow \mathrm{MERGE}(C_{2i-1}^{(k)}, C_{2i}^{(k)})$ ▹ Merging of adjacent concept lattices
7: $\quad$ **end for**
8: **end for**
9: **return** $\mathcal{C}(B,A,\mathsf{L},R) \leftarrow C_1^{(0)}$ ▹ Output of the algorithm

The presented theorem is the base for our distributed algorithm. The main aim is effectively create closure system in $\prod_{a\in A} \mathsf{L}(a)$ which represents the set of all intents. Of course, there are more options how to process the distribution itself, i.e.,

Algorithm 3. Procedure MERGE for merging two concept lattices

Require: two concept lattices C_1 and C_2
Ensure: MERGE(C_1, C_2)
$I_1, I_2 \leftarrow \emptyset$ ▷ I_1, I_2 will denote list of intents for C_1, C_2
for all $(X, g) \in C_1$ **do**
  $I_1 \leftarrow I_1 \cup \{g\}$ ▷ Extraction of all intents from C_1
end for
for all $(X, g) \in C_2$ **do**
  if $g \notin I_1$ **then** ▷ Check for duplications of intents in I_1
    $I_2 \leftarrow I_2 \cup \{g\}$ ▷ Extraction of all intents from C_2
  end if
end for
$I \leftarrow \emptyset$ ▷ Set of all newly created intents
for all $g \in I_2$ **do**
  for all $f \in I_1$ **do**
    $I \leftarrow I \cup \{f \wedge g\}$ ▷ Creation of new intent
  end for
end for
MERGE(C_1, C_2) $\leftarrow C_1$
for all $g \in I$ **do**
  MERGE(C_1, C_2) $\leftarrow$ MERGE(C_1, C_2) $\cup \{(g^{\top}, g)\}$ ▷ Addition of new concepts
end for
return MERGE(C_1, C_2) ▷ Output of the merging procedure

how to merger sub-models during the process. We have designed the version of distribution which divides the starting set of the objects into 2^n parts, for which local models are created (in parallel) and then pairs of them are combined (in parallel) into 2^{n-1}, and then this process continues, until we have only one concept lattice at the end. This output should be isomorphic to concept lattice created by the sequential run, because we have shown that previous theorem is valid and we only make partitions accordingly to its assumptions and proof. It is easy to see that pair-based merging leads to the binary tree of sub-lattices, with n representing the number of merging levels. Now, we can provide the algorithm description. The Algorithm 2 is distributed version (with the idea described here) and merge procedure is provided separately in Algorithm 3.

4 Illustrative Experiments with the Generated Data Sets

For our illustrative experiments with the provided algorithm we have produced randomly generated input formal contexts with specified sparseness index of the whole input data table. The reason is simply the natural characteristic of FCA-based algorithms in general (with its exponential complexity in worst cases), i.e., whenever the number of concepts (intents) still grows fast with the number of inputs, this data table has still too much different combinations of values of attributes in it (this is problem especially for fuzzy attributes, which we have here) and merge procedure

is not effective in order to get better times in comparison with incremental algorithm. But in the cases with the set of intents for which number of concepts (intents) is strongly reduced, the provided approach to distribution will lead to the reduction of computation times (merge procedures will not need so much time to create their new lattices). Fortunately, as we are very interested in the application of GOSCL in text-mining domain, and sparse inputs generate much reduced concept lattices (more zeros will produce more similar combinations of values), our distributed algorithm seems to be efficient mainly for very sparse input matrices.

In order to simplify the data preparation (generation) process, we have used one type of attribute in the table (scale-based, with 5 possible values $\{0,1,2,3,4\}$, where 0 is lowest value). The values are generated from this scale. The generation of sparse data table is based on the sparse factor $s \in [0,1]$), which indicates the ratio of "zeros" (as a real number between 0 and 1) in data table, i.e., s indicates the level of sparseness of generated data table. For higher s, the number of zeros in input is also higher. Simple mechanism for generation was used with random number generator for decision on adding the zero or some non-zero value according to the selected sparseness. The generated data are then very close to the selected sparseness (especially for larger inputs). We can add that in text-mining domains the sparseness factor is very high (usually more than 0.99). The number of generated objects will be (for all our examples) 8192 (just to simplify its division to partitions which can be merged easily in particular levels, 2^{13} is 8192).

First illustrative experiment shows that the efficiency is different for different sparseness. Therefore, we have tried random contexts for 5 and 7 attributes and different sparseness (from 0.5 to 0.9). The results are shown in the left part of the Figure 1. As we can see, with the higher the sparseness of the data, the ratio of computation times between distributed version and sequential (reduction ratio T_{dis}/T_{seq}) is lower (which means better reduction). Another experiment is based on the analysis of fixed sparseness (0.9), where number of attributes is changing from 5 to 20 and also N (the number of levels in merging) has three different settings (5, 8 and 10). The result is shown on the right part of the Figure 1, where we can see that number of merging levels (and therefore also number of starting partitions 2^n) has not big influence on the reduction ratio. On the other hand, with the increasing number of attributes reduction ratio should increase due to higher number of intents produced by the combination of values for more attributes.

One of our main interests is to analyze data from text-mining domains, which are usually represented by very sparse input tables (e.g., sparseness 0.998 for Reuters dataset with basic preprocessing settings). Therefore, we also add some experiments with the very sparse inputs. First, we have analyzed the reduction ratio for fixed sparseness 0.998 and number of merging levels 5, where number of attributes are changing (from 100 to 500). As we can see on the left side of the Figure 2, reduction ratio increased with the number of attributes, but is still quite significant even for 500 attributes. Moreover, this illustrative experiments were not realized with specialized sparse-based implementation, which will probably keep the ratio under 1 for more attributes. For reference see [6], where we have provided and analyzed specialized sparse-based version of GOSCL with significant reduction of computation times.

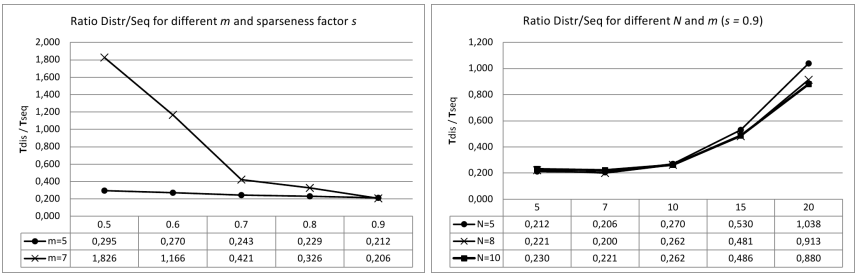

Fig. 1 Experiments with the distributed version of GOSCL - left side: analysis for different sparseness; right side: analysis for different number of attributes and merging levels

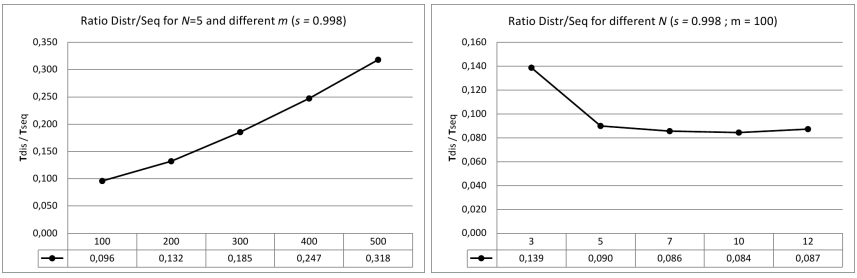

Fig. 2 Experiments with the distributed version of GOSCL on very sparse inputs - left side: analysis of reduction ration with changing number of attributes; right side: analysis of reduction ratio for different number of merging levels

If merge procedure will be implemented (in the future) in similar way, we will be probably able to work with even more attributes with still good reduction ratio. In last illustrative experiment, we have used same sparseness of data (0.998) with 100 attributes and analyze the reduction ratio for different merging levels (N). The result is shown on the right side of the Figure 2, where we can see now more evidently that higher number of levels can help in better times, but it seems that it is not needed to make very small partitions (e.g., for $N = 12$, which leads to starting with two-object sets, ratio is little higher than for previous values). The best number of merging levels can be estimated on real datasets and then used for better results.

At the end of this section we should say that this merging distributed approach seems to be applicable in sparse domains and can lead to computation times reduction, but there are several limitations which should be investigated more on the real data sets. It is expected that also for real data, where randomness is not the way how the table values are produced, the provided algorithm can be useful (but it will depend on data), especially for cases which relatively do not produce too much intents (e.g., very sparse inputs). The first experiments with the Reuters dataset proved the same behavior of the algorithm. Moreover, the usage of specialized sparse-based

implementations should be analyzed in our next experiments. This will show the limits of our distributed GOSCL in creation of large-scale concept lattices from real textual datasets or other sparse-based domains [14].

5 Conclusions

In the presented paper we have introduced the possible solution for distribution of creating the FCA-based model known as Generalized One-Sided Concept Lattice. As we have shown, it is possible to produce merged concept lattice from the smaller ones produced individually for disjoint subsets of objects. For the merging we have introduced simple merging procedure which is based on the partition similar to binary tree (lists are smallest concept lattices from the start of the distribution and root is merged final lattice). The illustrative experiments on the randomly generated data were added which shows its potential (especially for very sparse inputs) and limits. The future work should be done in realization of more experiments (on very sparse real data) and specialized sparse-based implementation of the merging procedure (very first experiments with real dataset proved the results of the paper).

Acknowledgements. This work was supported by the Slovak Research and Development Agency under contracts APVV-0035-10, APVV-0208-10 and APVV-0482-11; by the Slovak VEGA Grants 1/1147/12, 1/0729/12, 1/0497/11; by the ESF Fund CZ.1.07/2.3.00/30.0041.

References

1. Ben Yahia, S., Jaoua, A.: Discovering knowledge from fuzzy concept lattice. In: Kandel, A., Last, M., Bunke, H. (eds.) Data Mining and Computational Intelligence, pp. 167–190. Physica-Verlag (2001)
2. Bělohlávek, R.: Lattices of Fixed Points of Fuzzy Galois Connections. Math. Log. Quart. 47(1), 111–116 (2001)
3. Butka, P., Pócsová, J., Pócs, J.: Design and Implementation of Incremental Algorithm for Creation of Generalized One-Sided Concept Lattices. In: 12th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary, pp. 373–378 (2011)
4. Butka, P., Pócsová, J., Pócs, J.: On Some Complexity Aspects of Generalized One-Sided Concept Lattices Algorithm. In: 10th IEEE Jubilee International Symposium on Applied Machine Intelligence and Informatics, Herlany, Slovakia, pp. 231–236 (2012)
5. Butka, P., Pócsová, J., Pócs, J.: Experimental Study on Time Complexity of GOSCL Algorithm for Sparse Data Tables. In: 7th IEEE International Symposium on Applied Computational Intelligence and Informatics, Timisoara, Romania, pp. 101–106 (2012)
6. Butka, P., Pócsová, J., Pócs, J.: Comparison of Standard and Sparse-based Implementation of GOSCL Algorithm. In: 13th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary, pp. 67–71 (2012)
7. Ganter, B., Wille, R.: Formal concept analysis: Mathematical foundations. Springer, Berlin (1999)
8. Grätzer, G.: Lattice Theory: Foundation. Springer, Basel (2011)

9. Janciak, I., Sarnovsky, M., Tjoa, A.M., Brezany, P.: Distributed classification of textual documents on the grid. In: Gerndt, M., Kranzlmüller, D. (eds.) HPCC 2006. LNCS, vol. 4208, pp. 710–718. Springer, Heidelberg (2006)
10. Janowitz, M.F.: Ordinal and relational clustering. World Scientific Publishing Company, Hackensack (2010)
11. Krajči, S.: Cluster based efficient generation of fuzzy concepts. Neural Netw. World 13(5), 521–530 (2003)
12. Krajči, S.: A generalized concept lattice. Logic Journal of IGPL 13(5), 543–550 (2005)
13. Medina, J., Ojeda-Aciego, M., Ruiz-Calviño, J.: Formal concept analysis via multi-adjoint concept lattices. Fuzzy Set. Syst. 160, 130–144 (2009)
14. Paralič, J., Richter, C., Babič, F., Wagner, J., Raček, M.: Mirroring of Knowledge Practices based on User-defined Patterns. J. Univers. Comput. Sci. 17(10), 1474–1491 (2011)
15. Pócs, J.: Note on generating fuzzy concept lattices via Galois connections. Inform. Sci. 185(1), 128–136 (2012)