A Proposal of the Information Retrieval System Based on the Generalized One-Sided Concept Lattices

Peter Butka[1], Jana Pócsová[2], and Jozef Pócs[3,4]

[1] Faculty of Economics, Technical University of Kosice,
Bozeny Nemcovej 32, 040 01 Kosice, Slovakia
peter.butka@tuke.sk
[2] Institute of Control and Informatization of Production Processes,
BERG Faculty, Technical University of Kosice,
Bozeny Necovej 3, 043 84 Kosice, Slovakia
jana.pocsova@tuke.sk
[3] Mathematical Institute, Slovak Academy of Sciences,
Gresakova 6, 040 01 Kosice, Slovakia
[4] Institute of Control and Informatization of Production Processes,
BERG Faculty, Technical University of Kosice,
Bozeny Nemcovej 3, 043 84 Kosice, Slovakia
pocs@saske.sk

Abstract. One of the important issues in information retrieval is to provide methods suitable for searching in large textual datasets. Some improvement of the retrieval process can be achieved by usage of conceptual models created automatically for analysed documents. One of the possibilities for creation of such models is to use well-established theory and methods from the area of Formal Concept Analysis. In this work we propose conceptual models based on the generalized one-sided concept lattices, which are locally created for subsets of documents represented by object-attribute table (document-term table in case of vector representation of text documents). Consequently, these local concept lattices are combined to one merged model using agglomerative clustering algorithm based on the descriptive (keyword-based) representation of particular lattices. Finally, we define basic details and methods of IR system that combines standard full-text search and conceptual search based on the extracted conceptual model.

1 Introduction

One of the possible and useful applications of knowledge modelling is to use created conceptual model from unknown (non-annotated) set of documents for

R.-E. Precup et al. (Eds.): Applied Computational Intelligence in Engineering, TIEI 1, pp. 59–70.
springerlink.com

improvement of information retrieval results (against “classic” approaches). This process should include preprocessing of documents set, building of conceptual model (hierarchy of concepts, linkage of documents to hierarchy), and using of created model for improvement of IR, with or without the combination of conceptual and full-text search. This approach can be seen as problem decomposition method for building of conceptual model from text documents [2]. This conceptual model combines locally applied Formal Concept Analysis (FCA) and agglomerative clustering of particular models into one structure, which is suitable to support information retrieval process and can be easily combined with standard full-text search, as was described in [4].

Formal Concept Analysis (FCA, [7]) is a theory of data analysis that identifies conceptual structures among data sets. FCA produces concept lattice among the data that can be understand as knowledge-based model. Standard FCA works in two basic approaches - binary (crisp) and fuzzy case. In crisp case FCA is based on binary data tables (object has/has not attribute). Due to fact that data tables from textual documents usually contain real-valued attributes, some fuzzification of classic crisp method is needed. In our previous work we have used one approach to one-sided fuzzification presented in [8]. In this chapter we will provide practical usage of slightly different approach (theoretically presented in [3]) based on the generalized one-sided concept lattice (with the algorithm for its creation), which is able to support creation of concept lattices for contexts (object-attribute tables) with non-homogenous attributes, which are represented as complete lattices and the resulting set of model attributes can be viewed as a mix of qualitative (binary), quantitative (discrete, interval-based) or fuzzy logically based (truth values of various logical systems) ones. This is the main difference to our previous work and will provide several benefits. Since it is not necessary to concern only one type of complete lattice for possible values of attributes, we are able to support any type of attributes in the current domain of documents. Therefore we are able to use not only terms weights, but also any lattice-based truth value structure like binary, nominal, or ordinal attributes for qualitative rating of documents in dataset. This is really interesting specially for today's popular social-based rating of web documents and can be helpful for deep analysis of documents, where keyword-based similarity is high, but qualitative attributes can help in their correct distinction.

In order to provide searchable structure of the whole dataset from more concept lattices, merging of them is needed. Therefore, particular FCA models are labelled by characteristic attributes and simple agglomerative algorithm is used for clustering of local models, with the metric based on these characteristic attributes. This approach was also presented in [5], where grid computing was used to achieve better computing times. Then we will show the proposal for usage of merged

conceptual model in IR systems due to fact that it is natively keyword-based (with facet-based search by qualitative attributes), distributed and decomposed. Also linkage of documents to this conceptual structure is straightforward (thanks to attribute-object duality in FCA-based methods). Then we will specify details, methods, and characteristics of proposal for IR system, which is able to reuse conceptual model in combination with standard search.

In the second section of this chapter we will present idea of FCA-based merged conceptual model. Next, we will specify proposal for IR system suitable to reuse created conceptual model, which can be implemented in the future.

2 Hybrid FCA-Based Model of Documents Dataset

In this section we will describe a proposal of hybrid FCA-based model of text documents dataset. The presented approach is based on the decomposition of starting datasets (based on partitional clustering algorithm), creation of local models (based on FCA), description of such models in the form suitable for merging of models, and agglomerative clustering algorithm for their composition into final merged hierarchical model.

2.1 Problem Decomposition Approach for Model Creation

FCA can be used for creation of hierarchy of concepts and relations between these concepts. A problem with the use of FCA in domain of text documents with larger datasets is in time-consuming computation of concepts and hard interpretability of huge amount of concepts. Solution can be seen in combination with other methods like clustering algorithms used in the way known as problem decomposition approach:

- Clustering of input set of documents I can be viewed as a reduction step, where n clusters C_1, C_2,..., C_n of similar documents are found. The reduction step is based on the filtering of related terms of these objects (within the cluster). This step is top-down problem reduction approach to conceptual model extraction phase.
- Every cluster C_i (i=1..n) is independent training set (with the reduced cardinality of weight's vector), for each one a small FCA-based conceptual model O_i is built using our FCA algorithm.
- Small conceptual models O_1, O_2,..., O_n are merged together and global conceptual model M from tested collection is finally created.

Therefore, we can see this process as a transformation between several steps as follows: $S \rightarrow (C_1, C_2,..., C_n) \rightarrow (O_1, O_2,..., O_n) \rightarrow M$, where n is number of components of decomposition. As possible clustering method partitional algorithms can be used for identification of small groups (e.g., less than 10-15) of similar

documents. In our preliminary tests in previous work we have used Growing Hierarchical Self-Organizing Maps (GHSOM) algorithm for the first step in process. It is SOM-based (SOM - Self-Organizing Maps) hierarchical algorithm, where each layer is composed of independent SOM(s) that adjust their size according to the requirements of the input data. More information about this algorithm and its predecessors can be found in [6].

2.2 Generalized One-Sided Concept Lattice

In this section we provide a generalization of one-sided fuzzy concept lattices (independently described by Krajci [8] and by Yahia and Jaoua [1]). For the purposes of this chapter we provide only basic information needed for description of incremental algorithm, more theoretical details with proofs can be found in [3].

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

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

In our case relation R represents data table for analysis - document-term matrix of dataset vector model combined with other attributes for particular documents in one object-attribute model. The main difference to previous approaches is the possibility to create concept lattice for tables with different types of attributes (truth value structures).

If (B,A,L,R) is a generalized one-sided formal context, then we are able to define a pair of mappings $\uparrow: 2^B \rightarrow \prod_{a \in A} L(a)$ and $\downarrow: \prod_{a \in A} L(a) \rightarrow 2^B$ as follows:

$$\uparrow(X)(a) = \inf_{b \in X}(R(b,a)),$$

$$\downarrow(g) = \{\, b \in B : \ \forall a \in A, g(a) \leq R(b,a)\}.$$

A pair of mappings $(\uparrow,\downarrow)$ forms a Galois connection between 2^B and $\prod_{a \in A} L(a)$. Galois connections [9] are fundamental for FCA, therefore almost all known approaches of fuzzifying the classical FCA are based on such mappings and it is also true in our case. For formal context (B,A,L,R) denote $C(B,A,L,R)$ the set of all pairs (X,g), where $X \subseteq B$, $g \in \prod_{a \in A} L(a)$, satisfying $\uparrow(X) = g$ and $\downarrow(g) = X$. Set X is usually referred as *extent* and g as *intent* of the concept (X,g). If we define a partial order on $C(B,A,L,R)$ as $(X_1,g_1) \leq (X_2,g_2)$ iff $X_1 \subseteq X_2$ iff $g_1 \geq g_2$, then $C(B,A,L,R)$ with this partial order forms complete lattice, which is called *generalized one-sided concept lattice*.

Now we are able to provide an incremental algorithm for creation of generalized one-sided concept lattice. Let (B,A,L,R) be a generalized one-sided formal context. For $b \in B$ put $R(b)$ an element of $\prod_{a \in A} L(a)$ such that $R(b)(a) = R(b,a)$, i.e., $R(b)$ represents b-th row in data table R. Let 1_L denote the greatest element of $L = \prod_{a \in A} L(a)$, i.e., $1_L(a)=1_{L(a)}$ for all $a \in A$. Then pseudocode of algorithm for creation of generalized one-sided concept lattice can be written as follows.

Algorithm (Generalized One-Sided Concept Lattice)
Input: (B,A,L,R) – generalized formal context
begin
create lattice $L := \prod_{a \in A} L(a)$
$C := \{1_L\}, C \subseteq L$ is the set of all intents
while$(B \neq \emptyset)$
{
choose $b \in B$
$C^* := C$
for each $c \in C^*$
$C := C \cup \{c \wedge R(b)\}$
$B := B \setminus \{b\}$
}
for each $c \in C$
$C(B,A,L,R) := C(B,A,L,R) \cup \{(\downarrow(c),c)\}$
end
Output: $C(B,A,L,R)$ – set of all concepts

Let us remark that step for creation of the lattice $L := \prod_{a \in A} L(a)$ can be done in various ways and it is up to programmer. For example, it is not necessary to store all elements of $\prod_{a \in A} L(a)$, but it is sufficient to store only particular lattices $L(a)$, since lattice operations in L are calculated component-wise.

2.3 Description of Local Models

Use of the algorithm from previous subsection leads to creation of one generalized one-sided concept lattice for every subset from clustering phase within the presented problem decomposition approach. Particularly, if we will use hierarchical algorithm like GHSOM, then we will get one concept lattice for documents from every 'leaf' cluster (neuron without sub-map, in the end of expansion) in the hierarchy of GHSOM maps.

Every concept in particular lattice is characterized by extent and intent. Extent is set of objects (documents) and intent is set of corresponding attributes, which are weights of terms (words) and other attributes defined on documents. So, the concept can be viewed as a set of documents characterized by minimal value of its

attributes. Before the creation of lattices documents can be tested through attributes - if value of some attribute is lower then some threshold, new value of attribute is set to zero. This is inspired by work presented in [10] and helps in reduction of number of terms in concept lattice description.

If we have higher concept in hierarchy of lattice, the number of concept's attributes is smaller. Attributes with nonzero weights can be used as characteristics of actual concept (set of documents in concept). Every concept lattice then can be presented as hierarchy of concepts characterized by some attributes. Because we need some description of lattice for merging of lattice to one model, we need to extract attributes from particular lattices and create their representation based on these attributes. A weight of descriptive terms is based on level of terms in hierarchy (of course, important is highest occurrence of term). Other attributes (binary, ordinal, nominal) can be also characterized in similar way using some simple function for getting value (map value from its lattice structure to interval) from <0,1>. Then all attributes can be used for characterization of particular lattices and for clustering based on metric. Attributes from the input documents collection, which are not presented in concept lattice, have zero weight.

Function for assigning of weights can be different, e.g., $f(w,x) = w.x$, where w is value of base level of concept in hierarchy (this can be level of individual objects as concepts, or we can use some higher level as 'zero' base level), x is level of higher concept with term/attribute occurrence. It is possible to optimize weights and importance of higher level terms using selection of specific 'zero' level (this also helps to reduce number of attributes in local hierarchy). ´Height´ of concept in lattice is based on number of concept's documents (cardinality of concept).

Example of descriptive node based on one concept lattice (on some Times newspaper articles), attributes (terms) are described according to their highest occurrence within the concept lattice (first number in brackets) and by weight for this level (second number in bracket from interval <0,1>):

Documents: Times 60s/72 Times 60s/141 Times 60s/164 Times 60s/211
Terms: african(3, 1.0), southern(3, 1.0), prime(2, 0.75), britain'(2, 0.75), white(2, 0.75), feder(2,0.75), central(1, 0.5), black(1, 0.5), confer(1, 0.5), africa(1,0.5), field(1, 0.5), northern(1, 0.5), nationalist(1, 0.5), draft(1, 0.5)

Nodes can be combined into hierarchical agglomerative clustering structure by averaging of their weights (or any other operation suitable for their combination).

2.4 Clustering of Lattices Based on Descriptions

First step for clustering is to create one node for every local hierarchy (for every concept lattice), which contains: 1) list of documents; 2) list of characteristic attributes (sorted by value of weights); 3) vector of attributes weight's values (normalized). Particular nodes are then compared using vectors of attributes weights, vectors are normalized using interval <0,1>. After this step differences between

numbers of documents in particular nodes are respected. Comparison of lattices is used in process of agglomerative clustering of these nodes.

The similarity of two nodes is computed as follows. Let node U_i is represented using list of normalized weights of attributes $U_i = (u_{i1}, u_{i2}, ..., u_{in})$, where n is number of attributes in collection (with 0 implicitly used for attributes not included in the current node), and similarity of two nodes U_1 and U_2 is counted as:

$$sim(U_1, U_2) = \frac{\sum_{k \in K} \frac{\min(u_{1k}, u_{2k})}{\max(u_{1k}, u_{2k})}}{|K|}, \tag{1}$$

where K is a greatest subset of all attributes for which weight value is non-zero at least for one of the nodes U_1, U_2:

$$\max(u_{1k}, u_{2k}) > 0, \ \forall k \in K . \tag{2}$$

The proposed agglomerative clustering method is based on merging of most similar pairs of nodes in every step of the algorithm. Whole process can be described by the following procedure, where *MAXSIM* is maximal similarity found between all pairs in U for current step and *joint* is function for creation of new joint node from some pair of nodes p (e.g., by averaging of their weights):

Algorithm (Agglomerative clustering of description nodes)
Input: U – set of all nodes representing particular concept lattices
begin
  $A := \{(0,U)\}$
  while($|U|>1$)
  {
    $m := m + 1$
    $Pairs := \{(U_i,U_j): sim(U_i,U_j) = MAXSIM\}$
    $P := \emptyset$
    $New := \emptyset$
    foreach $p = (U_{p1},U_{p2}) \in Pairs$
      $P := P \cup \{U_{p1},U_{p2}\}$
      $New := New \cup joint(p)$
    $U := (U \backslash P) \cup New$
    $A := A \cup \{(m,U)\}$
  }
end
Output: A – agglomerative hierarchical structure of clustered nodes

Output of the algorithm is agglomerative hierarchical structure - simple hierarchy where leafs are nodes for particular concept lattices and nodes in the higher

levels are combination of their descriptions. Similarity of documents is based on their vectors of weights. Joint node from similar nodes is created by merging of set of their documents and by combination of their vector of weights. Final hierarchy contains nodes with the list of documents in it and the sorted list of characteristic attributes of nodes. Every node has link to its upper node and lower nodes. This structure and particular concept lattices can be reused in IR process.

3 A Proposal for Usage of Conceptual Model in IR System

The process of IR can be characterised by scheme represented in Fig.1. In control part it is important to connect blocks with control functionality like selecting of documents, choosing the method for indexing/analysis, and creation of index. The picture represents classic approach for IR system implementation. Only some of the blocks and edges are changed according to standard (full-text) search. In our case, we want to reuse standard architecture (process) with the minimum changes. Main differences to classic scheme are in five basic aspects, which are highlighted in Fig.1. It is possible due to fact that proposed conceptual model is keyword-based combined with some qualitative attributes (used for faceted search or as secondary ranking attribute) and is therefore compatible with standard textual operations and query format.

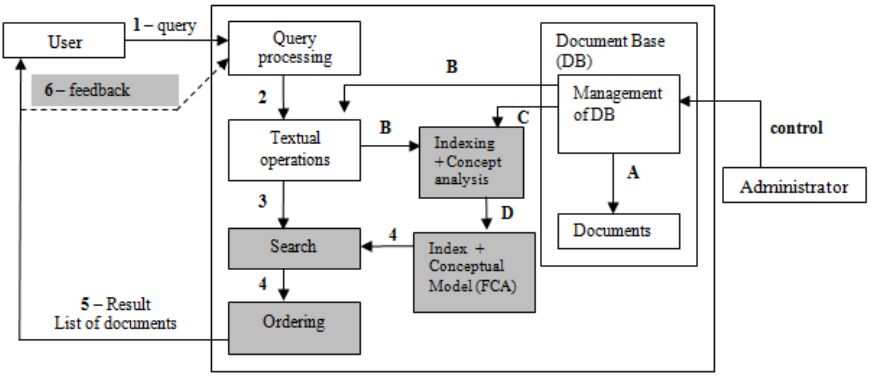

Fig. 1. Scheme of IR process with the combination of standard index and conceptual model.

Process of documents analysis (*Indexing + Concept analysis*) will be now combination of full-text analysis and proposed FCA-based method for text analysis. New generalized "index" (*Index + Conceptual model(FCA)*) is combination of full-text index and created hybrid FCA-based model (local FCA models and agglomerative clustering structure of their descriptions). For *Search* block, which returns unordered set of relevant documents to query, we need to define suitable method for returning the query results by full-text and conceptual search in specified combination. Therefore also *Ordering* block, which prepares ordered set of

documents from the previous step, needs to be revised in order to use the method for combination of full-text and conceptual ordering of results. Another place of possible change is within the process of query extension with automatic or manual (controlled by user) feedback based on the query expansion (step *6 - feedback*).

3.1 Use of Combined Index in Search

Use of our hybrid FCA-based model can be done in different ways, where difference is in application of query to the combined index. Classic approach is based on vector model of documents, while in our case we are able to combine *full-text search* (vector model based on standard part of index) and *conceptual search* (with the use of hybrid FCA-based model). By this combination we will get some specific modes of retrieval: 1) *Standard Full-text search* – query represented in vector model is used directly for search, only full-text part of combined index is used; 2) *Conceptual Model search* – vector representation of query is used for comparison with hybrid FCA-based model; 3) *Extended Full-text search* – full-text with automatic query expansion, where original query is used to search in conceptual model for expansion keywords; 4) *Combined Full-text and Conceptual search* – the result of search is provided in two separate lists (one for full-text and one for conceptual search), and lists are combined using ordering procedure in the next step of the process.

Full-text search is well-known method based on the analysis of similarity between vector representation of query and documents (query expansion do not change the method). More interesting is conceptual search in hybrid FCA-based model (conceptual part of combined index), where search follows three basic steps: 1) *Search for local models*; 2) *Search in selected local models*; 3) *Return of search result*. Details regarding possible options are summarized in next table.

In general, search within the descriptive nodes (search for local models) can be defined as some operator $H(q,A,V)$, which returns list of nodes from agglomerative structure A according to query q using search strategy V. Within the search in concept lattices we can define expansion operator $S(q,C,n, L)$, where C is most similar concept to query q in currently processed concept lattice L and n is specified size of neighbourhood, which is explored in order to return more concepts similar to C.

Use of particular approach for search can be switched manually or automatically. One problem with conceptual search is possibility that keyword is not available in FCA model due to preprocessing of documents dataset or pruning in concept lattice creation algorithm, but same term is (usually) still available in full-text part of combined index. Therefore, in case of automatic switch, we propose following procedure. If conceptual search found result, it is used for expansion of query in full-text. If not, then query is used in non-expanded form. Second choice is to manually switch modes (choosing the operators and their combinations) from the search approaches, e.g., using some user controlled switch in web browser application.

Table 1. Search possibilities for particular conceptual search steps in hybrid FCA-based model

Main Goal	Possible method	Details
1. Search for Local Models - use of agglomerative structure A	Search without hierarchical information	Only leaf nodes are compared to query
- representation of query q is compared to descriptive models of nodes - possible method = search strategy V	Strictly hierarchical search	Top-down approach where on every level most similar node(s) to query is(are) found and then is(are) used as a new "parent(s)" for search on lower level
Output: most similar leafs = concept lattices	Partially hierarchical search	Similarity is analysed to every node in structure - if node is higher in hierarchy, complete subtree under this node is used[1]
2. Search in Selected Local Models - use of particular concept lattices for search - representation of query q is compared to intent model of concepts - most similar concept to query q in current lattice L is C - possible method = use of expansion in search within the concept lattice; n is specified size of neighbourhood of concept C for expansion	Non-expanded search	$n = 0$, C is returned as result
	Expanded search	$n > 0$, result of search is concept C together with other concepts from the structure of current concept lattice L for which neighbourhood factor (path in concept lattice diagram) is less or equal to specified parameter n (conceptual expansion); first step of expansion is always up in hierarchy (if possible), because we want to find some new similar documents (from different subparts of diagram for current C)
3. Return of Search Results - concept(s) from conceptual search are returned as output from Search block	Return of results in conceptual search (used for strict version and for combined version)	Concepts with their extents (documents) and intents (attributes values) are returned
	Return of conceptual search results for full-text query expansion	If expansion of full-text search is expected, then intents are returned for automatic expansion of query

3.2 Ordering of Search Results

The output of ordering step is ordered list of documents according to some relevance function (to query). Difference in relevance evaluation comes from different modes of search. In case of full-text search relevance is directly based on vector

[1] The main idea here is to prefer leaf nodes (or lower level nodes) in order to avoid huge amount of concept lattices for exploring at the end. This can be achieved also by normalization of similarity measures using level of node in hierarchy (simply by the rule "lower is better").

model of IR (similarity of documents to query), for which we can define evaluation function E for full-text relevance („ranking“). If conceptual search is used only for full-text query expansion, function for ordering is still E.

For strict conceptual search we have list of concepts with their extents (lists of documents for every concept). Then conceptual ordering K is defined as follows: 1) documents from most similar concept (to query) are preferred, documents of other concepts are less preferred with increasing distance to most similar concept (according to neighbourhood), first occasion of document in concepts is used as its ordering base; 2) two documents with first occasion for same concept are ordered according to their vector-based similarity with query.

For combined search the input for ordering is in two lists, which can be provided separated (then ordering is based on previous approaches) or in one combined list. In second case we can define order differently, e.g., as weighted combination of full-text ordering E and conceptual ordering K using some parameters p and 1-p, where p is from <0,1>. If p is near 1, full-text search is dominant, if p is near 0, conceptual search is preferred. The whole ordering process can be then defined by operator $U(E,K,p)$.

3.3 Formal IR System and Possible Extensions

Standard formal description of IR System is defined as a tuple $(D, Q, F, R(q_i, d_j))$, where D is a set of documents representations, Q is a set of queries representations, F is a set of relations between representation of documents (index structure), and R is evaluation function for ordering of returned relevant documents to queries. In our case D and Q are not changed (vector-based model with some other attributes, if needed). F is structure of combined index (classic full-text index and hybrid FCA-based model). Evaluation function R is in our case combination of more parts, which are defined by particular operators: H for search in agglomerative hierarchical structure of nodes in conceptual model, S for extended search in concepts within the concept lattices, and U for ordering of searched results.

In general, system is designed and proposed with minimal change from the perspective of user. Ordering and presentation of results are provided in standard form as well as queries inputs (keywords with some facets and qualitative attribute entries). Also feedback is solved only using automatic query expansion, with possibility to have manual switch (if needed). We can still imagine many extensions to such system, which are basically of two types: 1) Improving the quality of retrieval results by updating the conceptual index and/or search algorithms without the involvement of users (includes extension of FCA-based model, improved quality of conceptual search, better model for combined index, etc.); 2) Extension based on involvement of users which include design and implementation of specific user interfaces with conceptual model in mind (choosing of concepts, use of GHSOM structure for search, etc.), manual feedback and lattice-based presentation of results.

4 Conclusions

In this chapter we have proposed method for creation of conceptual model based on the FCA (with the new method for creation of Generalized One-sided Concept Lattice) and agglomerative clustering and its usage within the system for information retrieval (IR). Automatic generation of such conceptual models can be strongly beneficial, especially for huge documents sets without semantic annotations. In such cases our proposal of conceptual analysis of documents and IR system can be simply applied and very useful, while it is not necessary to change keyword-based character of search and results (with conceptual information extracted directly from documents set).

Acknowledgments. This work was supported by the Slovak VEGA Grants No. 1/0685/12, No. 2/0194/10 and No. 1/0390/10, and also by the Slovak Research and Development Agency under contracts APVV-0208-10 and APVV-0035-10.

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, Heidelberg (2001)

[2] Butka, P.: Combination of problem reduction techniques and fuzzy FCA approach for building of conceptual models from textual documents. In: Paralic, J., Dvorsky, J., Kratky, M. (eds.) ZNALOSTI 2006, Proceedings of the 5th Annual Conference, Ostrava, Czech Republic, pp. 71–82 (2006)

[3] Butka, P., Pocs, J.: Generalization of one-sided concept lattices (2011) (Submitted)

[4] Butka, P., Pocsova, J.: Hierarchical FCA-based conceptual model of text documents used in information retrieval system. In: Proceedings of 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI 2011), Timisoara, Romania, pp. 199–204 (2011)

[5] Butka, P., Sarnovsky, M., Bednar, P.: One Approach to Combination of FCA-based Local Conceptual Models for Text Analysis – Grid-based approach. In: Proceedings of 6th International Symposium on Applied Machine Intelligence (SAMI 2008), Herlany, Slovakia, pp. 131–135 (2008)

[6] Dittenbach, M., Merkl, D., Rauber, A.: Using growing hierarchical self-organizing maps for document classification. In: Proceedings of European Symposium on Artificial Neural Networks (ESANN 2000), Bruges, Belgium, pp. 7–12 (2000)

[7] Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1997)

[8] Krajci, S.: Cluster based efficient generation of fuzzy concepts. Neural Netw. World 13, 521–530 (2003)

[9] Ore, O.: Galois Connexions. Trans. Am. Math. Soc. 55, 493–513 (1944)

[10] Quan, T.T., Hui, S.C., Cao, T.H.: A Fuzzy FCA-based Approach to Conceptual Clustering for Automatic Generation of Concept Hierarchy on Uncertainty Data. In: Proceedings of CLA conference (CLA 2004), Ostrava, Czech Republic, pp. 1–12 (2004)