An Extension of ToscanaJ for FCA-based Data Analysis over Triple Stores

Frithjof Dau and Barış Sertkaya

SAP Research Center Dresden, Germany
`(frithjof.dau|baris.sertkaya)@sap.com`

Abstract. Classical Business Intelligence (BI) solutions provide different means like OLAP, data mining or case based reasoning to explore structured data. The data is usually stored in dedicated repositories like data warehouses and then explored by standard BI means, which are usually based on mathematical statistics and provide a quantitative analysis of the data. In this paper, we complement this approach in two respects. First of all, FCA, as a qualitative approach for data analysis, is used for analyzing the data. Second, the analyzed data is not stored in a data warehouse, but in a triple store instead. To make this possible, the existing FCA-tool ToscanaJ has been extended in order to act on triple stores. The approach in this paper is exemplified on a dataset of documents crawled from the SAP community network.

1 Introduction

Business Intelligence (BI) solutions provide different means like OLAP, data mining or case based reasoning to explore data. In the standard BI approach, this data is usually extracted from transactional databases, transformed into a unified schema which is tailored to BI needs, and stored in a dedicated repository like a data warehouse. The standard BI means which are used to explore this data are focusing on attributes which can be numerically measured, thus they provide a *quantitative analysis* of the data (aka "number crunching") based on mathematical statistics. To some extent, though arguably oversimplified, one can understand BI as acting on lists or tables filled with numbers.

Compared to number crunching, Formal Concept Analysis (FCA) [5] provides a complementing approach. The starting point of FCA is a *dyadic* formal context of of formal objects formal attributes and a incidence relation which (only) describes whether attributes apply to objects or not. As FCA builds meaningful clusters based on the objects and attributes, it is a means for *qualitative analysis* of the data, complementing the classical quantitative analysis. Moreover, the clusters (aka formal concepts) are from a hierarchy (a complete lattice) which can be naturally visualized, thus FCA qualifies as a *visual analytics* tool.

A general overview over the benefits of FCA in information science is provided by Priss in [11]. Relevant for this paper are the relationships between FCA and both Business Intelligence (BI) and Semantic Technologies (ST).

With respect to BI, FCA can be for example considered as a data mining technology, particularly for mining association rules [15, 10]. More relevant to this paper is the approach to explore data in relational databases with FCA. As described in the next section, a method called "conceptual scaling" allows transforming columns in a database, being filled with arbitrary values, into formal contexts. Such scales can be compared to dimensions in BI applications. The exploration of data in databases with FCA is for example described in [6, 8, 16, 14]. A number of tools for FCA have been developed. Most important for this paper is Toscana [17, 13, 12] , developed in C, and its Java-based successor ToscanaJ [1][1]. Moreover, it should be mentioned that FCA has been used for exploring data warehouses as well [7].

FCA targets a formalization of the philosophical understanding of concepts with their extensions and intensions, thus FCA indeed is a semantic technology. Though it does not belong to the core of Semantic Web technologies, FCA provides decent means to define and analyze concept hierarchies, so it comes as no surprise that FCA has been used in the realm of querying, browsing and visualization of ontologies (e.g. OWLFCAViewTab for Protege and OntoViz), ontology alignment (e.g. FCA-Merge and OntEx), ontology engineering (e.g. relational exploration or role exploration) and ontology learning (e.g., Text2Onto).[2] In this paper, we exemplify the benefits of FCA for (semantically enabled) BI by analyzing data in a triple store with FCA methods. In order to do so, the existing ToscanaJ tool has been modified such that it can retrieve data from triple stores instead of relational databases. A short introduction into ToscanaJ and its modifications are provided in Sec. 3. An often named benefit of ST compared to relational databases are the ST capabilities to better deal with unstructured data like text-documents. FCA has already been employed to create concept hierarchies out of the content of text documents, e.g. [2] targets the creation of taxonomies out of text corpora, and [3] uses FCA to analyze a corpus of html-pages about rental offers for flats and houses.

In this paper, we apply FCA on a dataset of documents crawled from the SAP community network[3] (SCN), but do not target to investigate the contents of the documents, but utilize meta-data of the documents (which have been created in the crawling process) for FCA-purposes. This use case is described in Sec. 4.

2 FCA and Conceptual Scaling

FCA per se can only deal with *binary* attributes. For real data, the situation is usually different: Attributes assign specific values (which might be strings, numbers, etc) to data. For example, RDF-triples (s, p, o) are exactly of this form: The attribute p - from now on we will use the RDF-term "property" instead- assigns the value o to the entity s. In FCA, a process called "conceptual scaling"

[1] `http://toscanaj.sourceforge.net`

[2] As this field is only weakly related to this paper, no references are given.

[3] `http://www.sdn.sap.com/irj/scn/index`

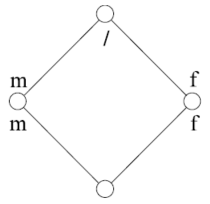
is used to deal with this issue. As ToscanaJ heavily uses scales, we recapitulate their core notions.

Let a specific property be given with a set of possible values. A *conceptual scale* is a specific context with the values of the property as formal objects. The choice of the formal attributes of the scale is a question of the design of the scale: The formal attributes are meaningful attributes to describe the values; they might be different entities or they might even be the values of the property again. To exemplify conceptual scaling, we reuse a toy example from [18], which is the following table provided on the right with two many-valued properties "sex" and "age". Note that empty cells are possible as well.

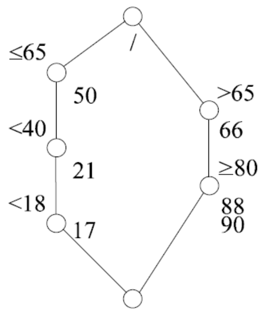
	sex	age
Adam	m	21
Betty	f	50
Chris		66
Dora	f	88
Eva	f	17
Fred	m	
George	m	90
Harry	m	50

Next, two conceptual scales for the properties "sex" and "age" and their line diagrams are provided.

S_1	m	f
m	×	
f		×



S_2	< 18	< 40	≤ 65	> 65	≥ 80
17	×	×	×		
21		×	×		
50			×		
66				×	
88				×	×
90				×	×



With the conceptual scales, the initial many-valued context can be transformed into a standard context, so that the corresponding concept lattice can be displayed. In [5], a number of standard scales is listed, like nominal scales, ordinal scales, and interordinal scales. An introduction into these scales is anyhow beyond the scope of this paper.

For conceptual scales, the following two points should be noted:

1. There is usually no standard or even necessary interpretation of an attribute: It has to be decided by the field expert which scale is appropriate. As discussed in [4], it can be argued that this is indeed not a drawback, but an advantage of FCA. On the other hand, particularly for repositories where the data schema is not stable but continuously changed and improved (which is

particularly the case for semantic repositories) it might not always be feasible to create all conceptual scales beforehand. For example, a nominal scale (like for the property "sex") could be created on the fly.
2. Conceptual scales do not depend on the real data, but only on the properties (and their values, of course) used in the data set. As one can see in the example, a realized context is derived from the scales and the real data in a later step after the scales have been created.

Both points are important for ToscanaJ, which is discussed in the next section.

3 ToscanaJ

There is a variety of software for FCA available. Most of them support the creation of contexts from scratch and the subsequent computation and display of the corresponding concept lattices. Contrasting this approach, Elba and ToscanaJ are a suite of mature FCA-tools which allow to query and navigate through data *in databases*. They are intended to be a *Conceptual Information System (CIS)*. CISs are "systems that store, process, and present information using concept-oriented representations supporting tasks like data analysis, information retrieval, or theory building in a human-centered way." Here, a CIS is an FCA-based system used to analyze data stored in one table of an RDBMS.

Similar to other BI-systems, in CIS we have to distinguish between a design phase and a run-time-phase (aka usage phase), with appropriate roles attached to the phases. In the design phase, a CIS engineer (being an expert for the CIS) together with a domain expert who has limited knowledge of a CIS) develops the CIS schema, i.e. those structures which will be later on used to access the system. This schema consists of manually created conceptual scales. Developing the scales is done with a CIS editor (Elba) and usually a highly iterative process. In the run-time phase, a CIS browser (ToscanaJ) allows a user to explore and analyse the real data in the database with the CIS schema. The diagram in Fig. reffig:ElbaToscanaJWorkflow depicts this overall workflow.

From the author's point of view, ToscanaJ is that tool which allows best (if only) to apply a BI-like approach to analyze data with FCA-methods, thus –to some extent– it comes closest to the envisioned CUBIST-system (though the envisioned CUBIST-system will provide different functionalities than ToscanaJ). Anyhow, the original Ebla/ToscanaJ-suite has been developed to analyze data in a *relational table*, i.e. a table in a RDBMS or an excel-file. We have extended the suite in order to be able to access data in a *triple store*. The reasons for doing so are twofold:

- On the one hand, with the extended version of ToscanaJ we have a first FCA-system which is capable to apply FCA-based visual analytics on top of a triple store.
- On the other hand, analyzing the pros and cons of ToscanaJ w.r.t. the envisioned CUBIST-system will help to develop a CUBIST prototype which will overcome the shortcomings (with respect to BI-applications) of existing FCA-tools.

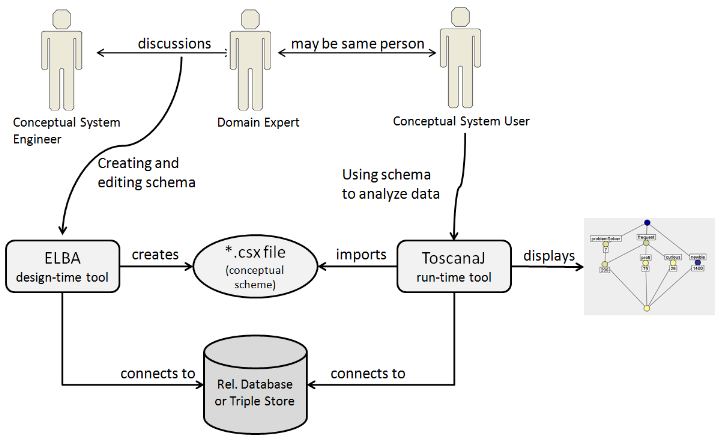


Fig. 1. Elba and ToscanaJ workflow

This extended version of the suite uses the Sesame framework[4] for accessing a triple store and and querying the RDF data therein. It provides two ways of connecting to a triple store over Sesame. One of them is over HTTP via Apache Tomcat[5], the other one is over the SAIL API[6]. Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies by the Apache Software Foundation. The SAIL API (Storage And Inference Layer) is a low level system API for RDF stores and inferencers. It is used for abstracting from the storage details, allowing various types of storage and inference to be used.

In a triple store we do not directly have the notions of tables and columns like in databases. As table information we use the type information in the triple store: we treat the objects of triples with the predicate `rdf:type` as tables. As column information, we use the predicates relating the subjects of the selected type to any object. More precisely, in order to detect the columns we get those subjects of the selected type and retrieve all distinct predicates that relate these subjects to an object.

The Elba/ToscanaJ-suite provides different kinds of conceptual scales. We have extended three of them –namely nominal scales, attribute scales and context tables– in order to act on triple stores.

Nominal scales are the simplest type of scales one can automatically create in Elba. They are used for properties like `gender` or `country` with mutually exclusive values. The formal attributes of the nominal scale are selected values of that property. As each object is assigned at most one of these values, the attribute concepts form an anti-chain, thus the scale cannot reveal any insight into attribute dependencies.

[4] see http://www.openrdf.org/doc/sesame2/system
[5] see http://tomcat.apache.org/
[6] see http://www.openrdf.org/doc/sesame2/system/ch05.html

Attributes scales offer an attribute-centered view, being close to "classical" formal contexts and allowing to create complex scales in an intuitive manner. Here each attribute is a property-value pair, which is manually selected from the triple store. Moreover, the CIS engineer can choose between a) "use only combinations existing in the database" and b) "use all possible combination". For option a) the diagram will only consist of concepts that could be derived from the data in the triple store, thus the diagram will reveal insights into dependencies between property-value pairs. If b) is chosen, a diagram of a Boolean lattice of all listed property-value pairs will be created independently of whether there exists objects in the triple store for each property-value combination or not.

Context table scales offer the most freedom to the CIS engineer. In context tables, arbitrary labels act as formal attributes. In contrast to the last two types of scales, no property-value pairs are chosen as attributes, thus now it has explicitly to be specified which objects of the data set fulfil the formal attributes. This is done by entering SPARQL expressions, which act as formal objects, and by entering the incidence relation as well, i.e. the relation which here relates the formal objects (SPARQL expressions) to the attributes (labels).

4 Use case

In order to evaluate our approach, we have used a dataset crawled from the SAP Community Network (SCN). SCN contains a number of forums for SAP users and experts to share knowledge. Our dataset is taken from the forum *Service-Oriented Architecture (SOA)*, which contains 2600 threads and 10076 messages. The dataset is annotated by the crawler using ontologies from the NEPOMUK project. The used ontologies and short descriptions[7]. are provided below.

– NEPOMUK Information Element Ontology (NIE): The NIE Framework is an attempt to provide unified vocabulary for describing native resources available on the desktop.
– NEPOMUK file ontology (NFO): The NFO intends to provide vocabulary to express information extracted from various sources. They include files, pieces of software and remote hosts.
– NEPOMUK Message Ontology (NMO): The NMO extends the NIE framework into the domain of messages. Kinds of messages covered by NMO include Emails and instant messages.
– NEPOMUK Contact Ontology (NCO): The NCO describes contact information, common in many places on the desktop.

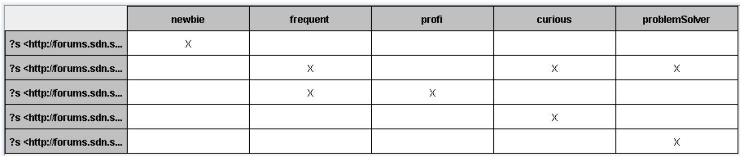
From these ontologies, our dataset uses the following classes as types:

– nie#DataObject: A unit of data that is created, annotated and processed on the user desktop. It represents a native structure the user works with. This may be a file, a set of files or a part of a file.
– nfo#RemoteDataObject: A file data object stored at a remote location.

[7] taken from the project website `http://www.semanticdesktop.org/ontologies`

– nie#InformationElement: A unit of content the user works with. This is a superclass for all interpretations of a DataObject.
– nco#Contact: A Contact. A piece of data that can provide means to identify or communicate with an entity.
– nmo#Message: A message. Could be an email, instant messanging message, SMS message etc.

For analyzing experience levels of the users of the SOA forum, we used the *Contact* type above and created a scale based on the number of posts, number of questions, number of resolved questions information provided in the data. We have named users that have less than 50 posts as *newbie*, users that have more than 400 posts as *frequent*, users that have more than 1000 posts as *profi*. Note that by definition, every profi is a frequent user as well, but no one can be both newbie and frequent (or profi) user. Moreover, we label users that have asked more than 200 questions as *curious* and people that have resolved more than 300 questions as *problem solver*. Note that this scale uses different measures (number of posts, number of questions, numbers of answers). In Fig. 2 it is shown how the context table in Elba is used to create the appropriate scale, and in Fig. 3 we see in Elbe the corresponding lattice. Note how SPARQL-queries are utilized to describe set of objects.



	newbie	frequent	profi	curious	problemSolver
?s <http://forums.sdn.s...	x				
?s <http://forums.sdn.s...		x		x	x
?s <http://forums.sdn.s...		x	x		
?s <http://forums.sdn.s...				x	
?s <http://forums.sdn.s...					x

Fig. 2. Designing in Elba a context table for contacts based on number of posts, questions, resolved questions

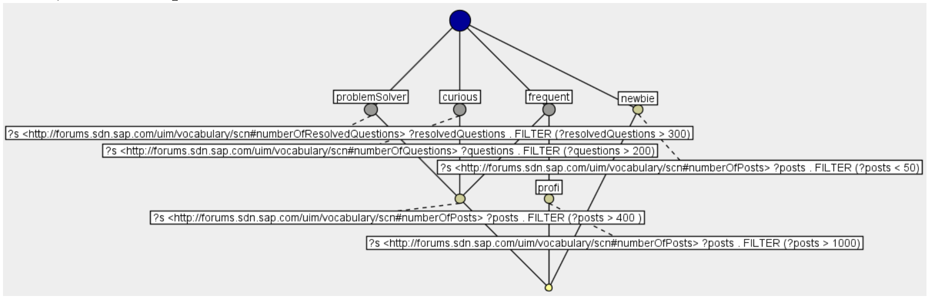
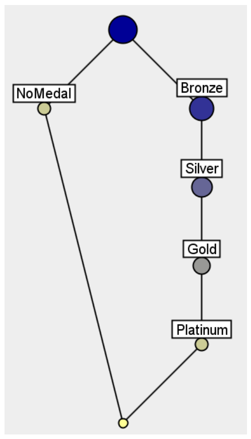

problemSolver
curious
frequent
newbie
?s <http://forums.sdn.sap.com/uim/vocabulary/scn#numberOfResolvedQuestions> ?resolvedQuestions . FILTER (?resolvedQuestions > 300)
?s <http://forums.sdn.sap.com/uim/vocabulary/scn#numberOfQuestions> ?questions . FILTER (?questions > 200)
?s <http://forums.sdn.sap.com/uim/vocabulary/scn#numberOfPosts> ?posts . FILTER (?posts < 50)
profi
?s <http://forums.sdn.sap.com/uim/vocabulary/scn#numberOfPosts> ?posts . FILTER (?posts > 400)
?s <http://forums.sdn.sap.com/uim/vocabulary/scn#numberOfPosts> ?posts . FILTER (?posts > 1000)

Fig. 3. Diagram of the context in Fig. 2

Next, for analyzing experience levels based on the *number of points* information we created another scale. This time, as labels we took contributor types that are officially defined by SCN as *bronze*, *silver*, *gold* and *platinium* contributors, which have more than 250, 500, 1500 and 2500 points respectively. Of course, there might be persons who do not have any of these ”medals” at all. We have two possibilities to understand the medals, as depicted in Fig.4.



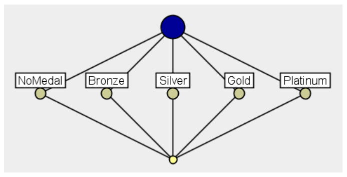


Fig. 4. Different scales for the status of contributors

The two approaches for modelling the scales shall be exemplified with gold owners. A gold owner has at least 1500 points. Either we consider a gold owner to be a silver owner and bronze owner as well (as he has more than 500 points to reach silver status and more than 250 points to reach bronze status), or we consider him to be not a silver or bronze owner (that is, a gold owner has *between* 1500 and 2499 points). Thus the two scales in Fig. 4 capture the different notions (aka meanings or semantics) of the status.

Besides these three scales, more scales have been created in Elba, which are not described here due to space limitations. We show next how the scales are utilized in ToscanaJ. First, in Fig. 5 and Fig. 6, we show how the diagrams we have created in Elba now appear with actual numbers in ToscanaJ.

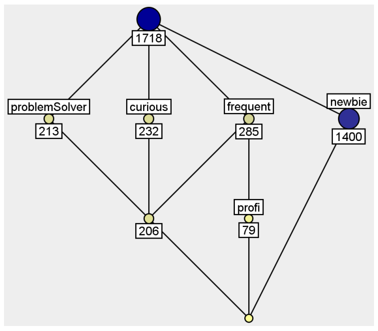


Fig. 5. Diagram of the scale based on number of posts, questions, resolved questions

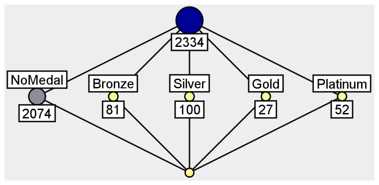


Fig. 6. Diagram of the scale based on number of points

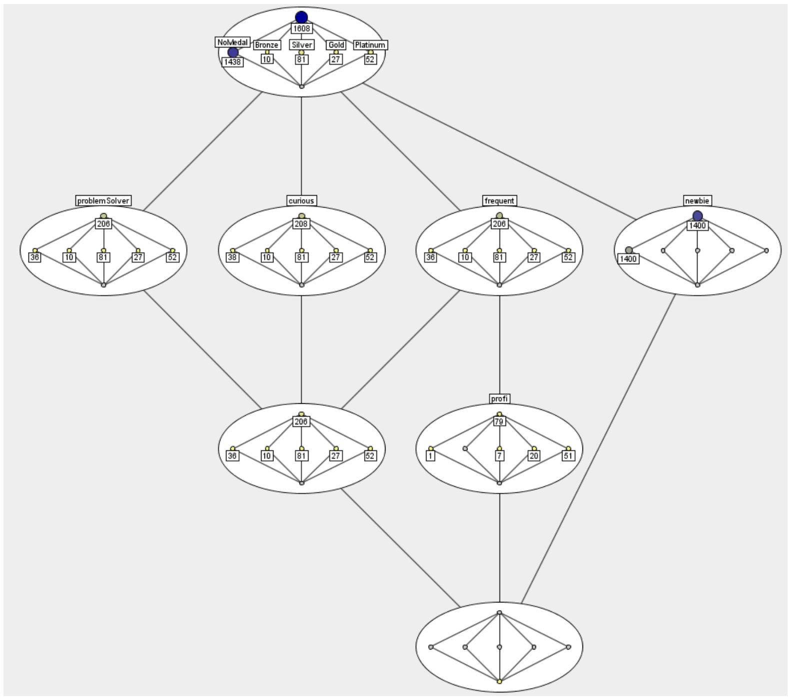


Fig. 7. Nested Diagram of two scales

The above displayed concept lattices are separately informative about the properties of forum users, i.e., the first one about experience levels based on number of posts, questions and resolved questions, and the second one about number of points. One of the most powerful techniques of FCA is to "combine" such lattices to give a combined view of several lattices together, which is called a *nested line diagram*. In its simplest form, a nested line diagram is a concept lattice whose concepts are themselves also concept lattices. Nested line diagrams allow the user to select a concept and zoom into it to see the lattice nested in that concept.

Figure 7 shows the nested line diagram of the diagrams in the Figures 5 and 6. Note that the outer diagram is actually the one in Figure 4:experienceLevels. The inner diagrams are the diagram in Fig. 6. Figure 8 shows an excerpt of the nested diagram that corresponds to the node *profi contributor*, and Figure 9 shows the inner diagram of this node. Note that the number of users corresponding to different levels of experience in this diagram differs from that of diagram in Figure 5. The reason is that, now we zoomed into the node gold contributor so the information in the inner diagram is restricted to the profi contributors only. For instance, as seen in this diagram that we have a high number of gold and platinum contributors (but no bronze contributors) amongst profi users, which is

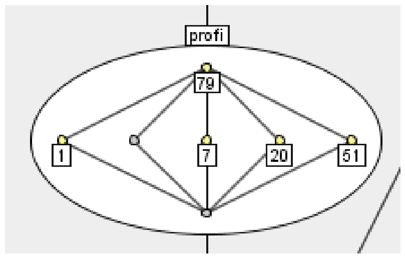


Fig. 8. Detail of Figure 7

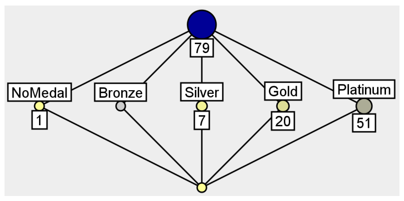


Fig. 9. Zooming into the node in Figure 8

quite natural. On the other hand, the one profi user with no medal is a surprising outlier which might deserve a dedicated investigation. In ToscanaJ, and thus in our extension of it to triple stores, one can nest an arbitrary number of diagrams and can browse nested diagrams easily by zooming in and out.

5 Conclusion and Further Research

This paper has shown how FCA can be applied data analysis methodology for data in triple stores, and the existing ToscanaJ suite has been extended in order to act not only on relational databases, but on triple stores as well.

As scales in the ToscanaJ workflow are manually crafted in the design phase of a CIS, this workflow is feasible for stable schemata. For ST, this is usually not the case: here the paradigm of agile schema development is prevalent. We plan to implement automatic or at least semi-automatic generation of scales based both on the schema information and the actual data in the triple store.

As stated in the introduction, FCA should be understood to *complement* existing BI approaches. ToscanaJ utilizes scales, thus lattices, for *all* kind of data, even for numerical attributes (which can be modelled as ordinal scales) and nominal attributes. The lattice structure of the scales do not reveal any structural insights, only the distribution of objects amongst the lattice nodes is interesting. Thus it can be argued that for these types of attributes, the standard visualization of BI like pie or bar charts are more appropriate. For this reason, another future research direction of CUBIST is the development of hybrid solutions, combining "classical" BI with FCA. This covers combinations of scales and their diagrams with BI diagrams for numerical data, like pie charts or sun-burst diagrams, and compared to nesting of scales, different approaches for using simultaneously several scales. In the long run, CUBIST intends to combine visualizations for both quantitative and qualitative analytics.

Disclaimer: Parts of this work have been carried out in the CUBIST project, which is funded by the European Commission under the 7th Framework Programme of ICT, topic 4.3: Intelligent Information Management.

References

1. P. Becker, J. Hereth, and G. Stumme. ToscanaJ: An open source tool for qualitative data analysis. In V. Duquenne, B. Ganter, M. Liquiere, E. M. Nguifo, and G. Stumme, editors, *Advances in Formal Concept Analysis for Knowledge Discovery in Databases, (FCAKDD 2002)*, 2002.
2. P. Cimiano, A. Hotho, and S. Staab. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of Artificial Intelligence Research*, 24(1):305–339, 2005.
3. R. Cole and P. Eklund. Browsing semi-structured web texts using formal concept analysis. In *Proceedings of the 9th International Conference on Conceptual Structures (ICCS 2001)*, pages 319–332, 2001.
4. F. Dau and J. Klinger. From formal concept analysis to contextual logic. In B. Ganter, G. Stumme, and R. Wille, editors, *Formal Concept Analysis, Foundations and Applications*, volume 3626 of *Lecture Notes in Computer Science*, pages 81–100. Springer-Verlag, 2005.
5. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations.* Springer-Verlag, Berlin, Germany, 1999.
6. J. Hereth. Formale begriffsanalyse und data warehousing. Masters thesis, TU Darmstadt, Germany, 2000.
7. J. Hereth. Relational scaling and databases. In U. Priss, D. Corbett, and G. Angelova, editors, *Conceptual Structures: Integration and Interfaces, Proceedings of the 10th International Conference on Conceptual Structures, (ICCS 2002)*, volume 2393 of *Lecture Notes in Computer Science*, pages 62–76. Springer-Verlag, 2002.
8. J. Hereth, G. Stumme, R. Wille, and U. Wille. Conceptual knowledge discovery - a human-centered approach. *Journal of Applied Artificial Intelligence (AAI)*, 17(3):281–301, 2003.
9. W. H. Inmon, D. Strauss, and G. Neushloss. *DW 2.0: The Architecture for the Next Generation of Data Warehousing.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.
10. L. Lakhal and G. Stumme. Efficient mining of association rules based on formal concept analysis. In B. Ganter, G. Stumme, and R. Wille, editors, *Formal Concept Analysis: Foundations and Applications*, volume 3626 of *Lecture Notes in Artificial Intelligence*, pages 180–195. Springer-Verlag, 2005.
11. U. Priss. Formal concept analysis in information science. *Annual Review of Information Science and Technology*, 40, 2005.
12. M. Roth-Hintz, M. Mieth, T. Wetter, S. Strahringer, B. Groh, and R. Wille. Investigating snomed by formal concept analysis. In *Proceedings of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2000.
13. P. Scheich, M. Skorsky, F. Vogt, C. Wachter, and R. Wille. Conceptual data systems. In O. Opitz, B. Lausen, and R. Klar, editors, *Information and Classification*, pages 72–84. Springer, Berlin-Heidelberg, 1993.
14. G. Stumme. Conceptual on-line analytical processing. In K. Tanaka, S. Ghandeharizadeh, and Y. Kambayashi, editors, *Information Organization and Databases*, chapter 14. Kluwer, Boston-Dordrecht-London, 2000.
15. G. Stumme. Efficient data mining based on formal concept analysis. In A. Hameurlain, R. Cicchetti, and R. Traunm uller, editors, *Database and Expert Systems Applications. Proceedings of DEXA 2002*, volume 2453 of *Lecture Notes in Computer Science*, pages 534–546. Springer-Verlag, 2002.

16. G. Stumme, R. Wille, and U. Wille. Conceptual knowledge discovery in databases using formal concept analysis methods. In J. M. Zytkow and M. Quafofou, editors, *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'98)*, volume 1510 of *Lecture Notes in Artificial Intelligence*, pages 450–458. Springer-Verlag, 1998.
17. F. Vogt and R. Wille. Toscana — a graphical tool for analyzing and exploring data. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing*, pages 226–233. Springer-Verlag, 1995.
18. K. E. Wolff. A first course in formal concept analysis. In F. Faulbaum, editor, *Proceedings of Advances in Statistical Software 4*, pages 429–438, 1993.