Information Retrieval and

Knowledge Discovery with FCART

A.A. Neznanov, S.O. Kuznetsov

National Research University Higher School of Economics,
Pokrovskiy bd., 11, 109028, Moscow, Russia
ANeznanov@hse.ru, SKuznetsov@hse.ru

Abstract. We describe FCART software system, a universal integrated environment for knowledge and data engineers with a set of research tools based on Formal Concept Analysis. The system is intended for knowledge discovery from big dynamic data collections, including text collections. FCART allows the user to load structured and unstructured data (texts and various meta-information) from heterogeneous data sources, build data snapshots, compose queries, generate and visualize concept lattices, clusters, attribute dependencies, and other useful analytical artifacts. Full preprocessing scenario is considered.

Keywords: Data Analysis, Knowledge Extraction, Text Mining, Formal Concept Analysis

1 Introduction

We introduce a new software system for information retrieval and knowledge discovery from various data sources (textual data, structured databases, etc.). Formal Concept Analysis Research Toolbox (FCART) was designed especially for the analysis of unstructured (textual) data. The core of the system supports knowledge discovery techniques, including those based on Formal Concept Analysis [1], clustering [2], multimodal clustering [2, 3], pattern structures [4, 5] and others. In case studies we applied FCART for analyzing data in medicine, criminalistics, and trend detection.

FCART is based on DOD-DMS (The Dynamic Ontology-driven Data Mining System) software platform. In case studies we applied DOD-DMS for analyzing data in the fields of medical informatics and trends detection. The core of the system complements a traditional knowledge extraction process with methods of clustering, multimodal clustering, Formal Concept Analysis, Hidden Markov chains, pattern structures and others.

Currently, there are several well-known open source FCA-based tools, such as ConExp [6], Conexp-clj [7], Galicia [8], Tockit [9], ToscanaJ [10], FCAStone [11], Lattice Miner [12], OpenFCA [13], Coron [14]. These tools have many advantages. However, they cannot completely satisfy the growing demands of the scientific com-

munity. One of the common drawbacks of these systems is poor data preprocessing. It prevents researchers from using the programs for analyzing complex big data without additional third party preprocessing tools.

For example, Coron has some tools for filtering objects and attributes, merging and transforming contexts (http://coron.wikidot.com/pre:filterdb), but Coron does not provide flexible tools for importing external data.

2 Methodology

The DOD-DMS is a universal and extensible software platform intended for building data mining and knowledge discovery tools for various application fields. The creation of this platform was inspired by the CORDIET methodology (abbreviation of Concept Relation Discovery and Innovation Enabling Technology) [15] developed by J. Poelmans at K.U. Leuven and P. Elzinga at the Amsterdam-Amstelland police. The methodology allows one to obtain new knowledge from data in an iterative ontology-driven process. The software is based on modern methods and algorithms of data analysis, technologies for processing big data collections, data visualization, reporting, and interactive processing techniques. It implements several basic principles:

1. Iterative process of data analysis using ontology-driven queries and interactive artifacts (such as concept lattice, clusters, etc.).
2. Separation of processes of *data querying* (from various data sources), *data preprocessing* (of locally saved immutable snapshots), *data analysis* (in interactive visualizers of immutable analytic artifacts), and *results presentation* (in report editor).
3. Extendibility on three levels: customizing settings of data access components, query builders, solvers and visualizers; writing scripts (macros); developing components (add-ins).
4. Explicit definition of analytic artifacts and their types. It allows one to check the integrity of session data and provides links between artifacts for end-user.
5. Realization of integrated performance estimation tools.
6. Integrated documentation of software tools and methods of data analysis.

FCART uses all these principles, but does not have an ontology editor and does not support the full C-K cycle. The current version consists of the following components.

- Core component including
  - multidocument user interface of research environment with session manager,
  - snapshot profiles editor (SHPE),
  - snapshot query editor (SHQE),
  - query rules database (RDB),
  - session database (SDB),
  - main part of report builder;
- Local XML-storage for preprocessed data;
- Internal solvers and visualizers;
- Additional plugins and scripts.

3 Current software properties and future work

Now we introduce version 0.8 of DOD-DMS as a local Windows application and version 0.4 as a distributed Web-based application. Those versions use local XML-storage for accumulating snapshots and integrated research environment with snapshot profiles editor, query builder, ontology editor, and some set of solvers (artifact builders) and visualizers (artifact browsers). The main solvers for this time can produce clusters, biclusters, concept lattice, sublattices, association rules, and implications, calculate stability indexes, similarity measures for contexts and concepts, etc. The set of solvers, visualizers, and scripts specifies a subject field of DOD-DMS edition.

We use Microsoft and Embarcadero programing environments and different programing languages (C++, C#, Delphi, Python and others). For scripting we use Delphi Web Script [16] and Python [17].

4 Data preprocessing in FCART

4.1 Obtaining initial artifacts

There are several ways to obtain a binary context, the basic FCA artifact:

- Load from ready data files of supported formats like CXT or CSV,
- Generate by plugin or script,
- Query from data snapshots.

Loading contexts from ready data files is supported by most FCA-tools. The most interesting way to obtain a context is querying from snapshots. Let us look to all steps needed to convert external data into some set of objects with binary attributes.

4.2 Access to external data sources and generating snapshots

Local storage of FCART can be filled from various data sources. System supports SQL, XML and JSON sources, so it can load data from most databases and Web-services.

Data snapshot (or snapshot) is a data table with structured and text attributes, loaded in the local storage by accessing external data sources. Snapshot is described by a set of metadata: snapshot profile, local path, link to external data source, time stamp, author, and comment. FCART provides one with a *snapshot profile editor* (SHPE). *Profile* consists of definitions of fields. Each element of a snapshot is a record: array of values of fields. Each field is defined by the following main properties:

- Id (identifier of field)
- Path (path in initial XML or JSON – may be empty)
- Name (user-friendly name of field)
- Group (for visual grouping of fields)

- Comment
- Data type (Boolean / Integer / Float / Text / Binary / DateTime)
- Is Unstructured? (field can be interpreted as unstructured text)
- Is Multivalued? (for sets / arrays)
- Type of multivalued presentation (delimited content / same path / path in form of "name + number")

Consider the following example of XML file:

```
<?xml version="1.0" encoding="utf-8"?>
<Data>
    // ...
  <Genre>Lounge</Genre>
  <Genre>Easy listening</Genre>
    // ...
</Data>
```

In this example field "Genre" is multivalued and have multivalued presentation type "same path" (Path = "<Data>/<Genre>"). But in other source we can have type "name + number" (Path = "<Data>/<Genre%d>"):

```
<Data>
    // ...
  <Genre01>Lounge</Genre01>
  <Genre02>Easy listening</Genre02>
    // ...
</Data>
```

Unstructured field definition additionally contains the following properties:

- Language (main language of text)
- SW (list of stop words)
- Stemmer (not required now because we use snowball stemmer from Lucene).

It is very useful for dealing with dynamic data collections, including texts in natural language, and helps to query full-text data more effectively. There is a sample of unstructured and multivalued field description in JSON format:

```
{       "Id": "02",
        "Path": "object\/author",
        "Caption": "Artwork Creators",
        "Group": "Common",
        "Comment": "The sequence of authors",
        "DataType": "Text",
        "Unstruct": {
                "Is": true,
                "Language": "English",
                "StopWords": [ ],
                "Stemmer": "Snowball"       },
        "MV": {  "Is": true,
                "MVType": "Vector",
                "MVRepresentation": "NameNumber",
                "MVFormat": "author%d"    }
```

}

Fig. 1 shows a variant of snapshot profile editor (SHPE) for XML filtering. The left pane “XML Structure” displays a sample of an XML-document from a dataset. A user can select any element from the document, add it to profile as a new field and set properties of the field.

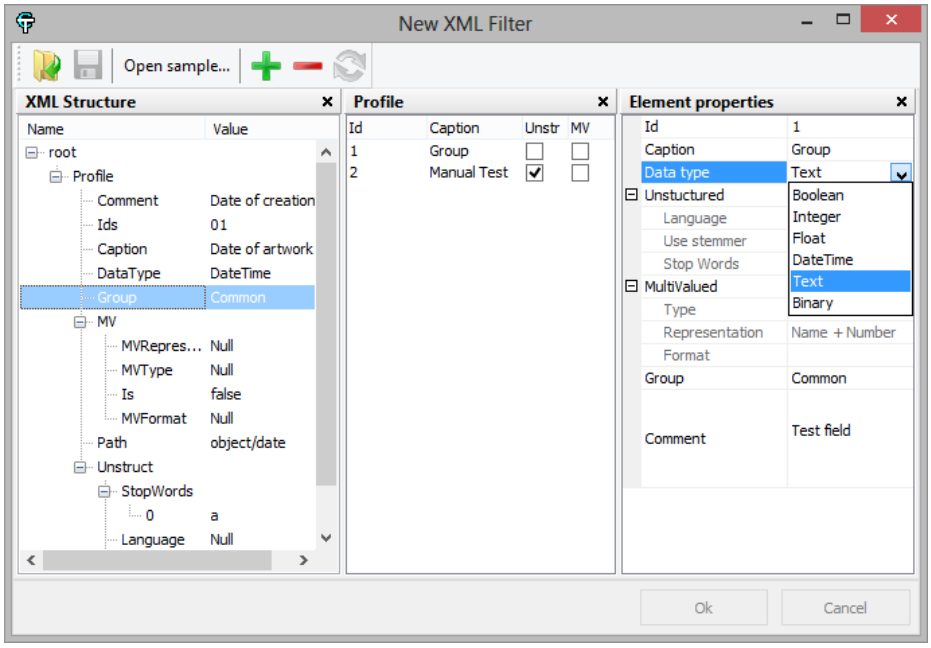

Fig. 1. Profile editor (profile by example)

4.3 Queries to snapshots and constructing binary contexts

The system has query language for transforming snapshots into binary formal context. This language describes so-called rules. Main rule types are the following:

- *Simple rule* generates one attribute from atomic fields of a snapshot. This rule type has syntax very similar to SQL WHERE clause
- *Scaling rule* generates several attributes from atomic fields based on nominal or ordinal scale
- *Text mining rule* generates one attribute from unstructured text fields.
- *Multivalued rule* generates one or many attributes from multivalued field (arrays and sets)
- *Compound rule* merges rules of all types into a single rule. This rule uses standard logical operations and brackets to combine elements.

We have also implemented additional rule types: *Temporal rules* are used for manipulating date and time intervals and *Filters* are used for removing objects with their intents from contexts.

In most cases, it is not necessary to write a query from scratch. One can select some entities in rules DB (RDB) and automatically generate a query. It is possible because the RDB is aware of dependencies between rules. Each rule type has XML presentation, so every query (or full RDB) can be imported and exported as an XML-file.

The following XML file is a sample of the scaling rule:

```
<scale name="Age" ScaleType="Order" DataType="Integer"
Ends="Open" id="t34">
  <Offset1>8</Offset1>
  <Offset2>16</Offset2>
  <Offset3>35</Offset3>
  <Offset4>60</Offset4>
</scale>
```

The application of this rule to snapshot generates 5 binary attributes: “Age < 8”, “8 <= Age < 16”, …, “60 <= Age”.

FCART uses Lucene full text search engine [18] to index the content of unstructured text fields in snapshots. The resulting index is later used to validate quickly whether the text mining or compound rule returns true or false.

5 Interactive visualization of concept lattice

The *concept lattice visualizer* is an example of interactive visualizer. It can be used to browse the collection of objects with binary attributes given as a result of query to snapshot (with structured and text attributes). The user can select and deselect objects and attributes and the lattice diagram is modified accordingly. The user can click on a concept. In a special window the screen shows names of objects in the extent and names of attributes in the intent. Names of objects and attributes are linked with initial snapshot records and fields. If the user clicks on the name of an object or an attribute, the content of the object or attribute description is shown in a special window according to the snapshot profile.

Fig. 2 demonstrates the result of building a sublattice from a concept lattice. The multi-document interface allows us to inspect several artifacts, so a sublattice will be opened in a new window.

The user can customize settings of lattice browsing in various ways. The user can specify whether nodes corresponding to concepts show numbers of all (or only new) objects and all (or only new) attributes in extent and intent respectively, or names of all (or only new) objects and all (or only new) attributes. Separate settings can be specified for the selected concept, concepts in the order filter, and the remainder of the lattice. The visual appearance can be changed: zooming, coloring, and other tools are available.

Right clicking on the name of an attribute user can choose several options: one can build a sublattice containing only objects with selected attribute; build a sublattice containing only objects without selected attribute; or find the highest concept with a selected attribute. Right clicking on the name of an object allows one the same actions.

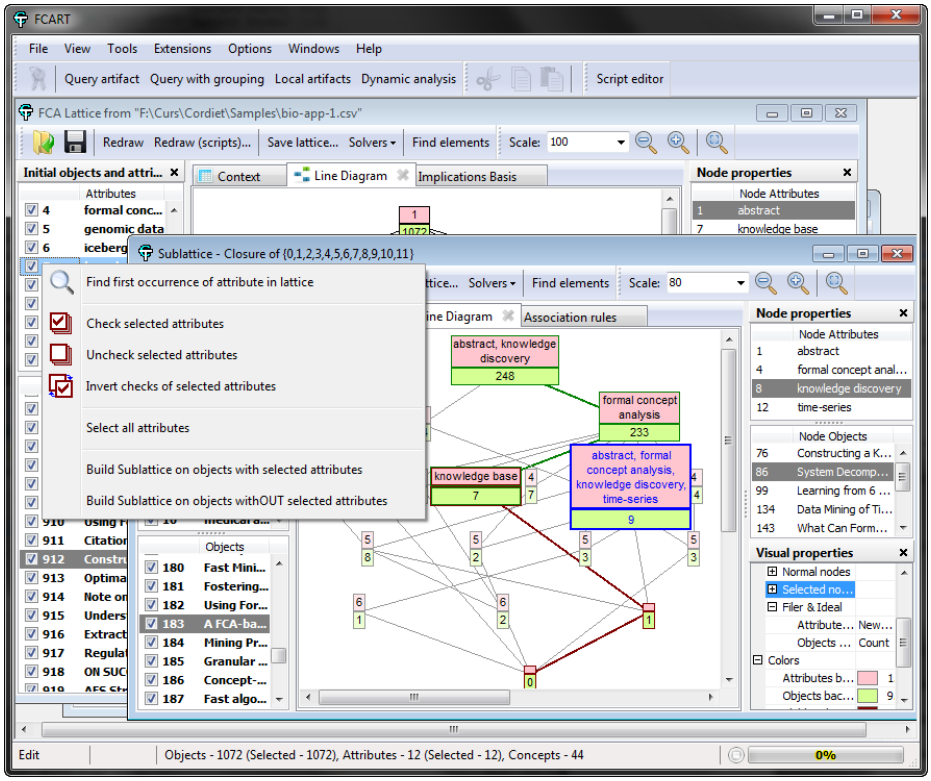

Fig. 2. Concept lattice visualizer

If we built a formal context using a query to a snapshot then we can simply look for a definition of each attribute (in form of a query rule from RDB) and a source of each object (in form of XML- or JSON-file) in left part of the visualizer window. If a filter rule is present in query then we can see comparison between sets of objects in the context and records in the snapshot.

Linking analytic artifacts with snapshots are very important for subsequent analysis of the same data collection. Researcher can simply interpret results of the analysis by viewing initial pieces of data.

6 Conclusion and future work

FCART is a powerful environment being in active development. The next major release of the local version 0.8 is planned for March 2013 and after that the system will be freely available to the FCA community. In this article we considered in details the powerful preprocessing tools of the system.

We intend to improve methodology, extend the set of solvers, optimize some algorithms, and use the proposed system for solving various knowledge discovery problems. We already have tested new solvers based on concept stability [19, 20] and other indices [21]. In the preprocessing queue we will try to simplify writing queries to external data sources by introducing SQL- and XML-explorer of databases and web-services.

Acknowledgements

This work was carried out by the authors within the project “Mathematical Models, Algorithms, and Software Tools for Intelligent Analysis of Structural and Textual Data” supported by the Basic Research Program of the National Research University Higher School of Economics.

References

1. Ganter, B., Wille R. Formal Concept Analysis: Mathematical Foundations, Springer, 1999.
2. Mirkin, B. Mathematical Classification and Clustering, Springer, 1996.
3. Ignatov, D.I., Kuznetsov, S.O., Magizov, R.A., Zhukov, L.E. From Triconcepts to Triclusters. Proc. of 13th International Conference on rough sets, fuzzy sets, data mining and granular computing (RSFDGrC-2011), LNCS/LNAI Volume 6743/2011, Springer (2011), pp. 257-264.
4. Ganter, B., Kuznetsov, S.O. Pattern Structures and Their Projections. Proc. of 9th International Conference on Conceptual Structures (ICCS-2001), 2001, pp. 129-142.
5. Kuznetsov, S.O. Pattern Structures for Analyzing Complex Data. Proc. of 12th International conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Conference (RSFDGrC-2009), 2009, pp. 33-44.
6. Yevtushenko, S.A. System of data analysis "Concept Explorer". (In Russian). Proceedings of the 7th national conference on Artificial Intelligence KII-2000, p. 127-134, Russia, 2000.
7. Conexp-clj (http://daniel.kxpq.de/math/conexp-clj/)
8. Valtchev, P., Grosser, D., Roume, C. Mohamed Rouane Hacene. GALICIA: an open platform for lattices, in Using Conceptual Structures: Contributions to the 11th Intl. Conference on Conceptual Structures (ICCS'03), pp. 241-254, Shaker Verlag, 2003.
9. Tockit: Framework for Conceptual Knowledge Processing (http://www.tockit.org)
10. Becker, P., Hereth, J., Stumme, G. ToscanaJ: An Open Source Tool for Qualitative Data Analysis, Proc. Workshop FCAKDD of the 15th European Conference on Artificial Intelligence (ECAI 2002). Lyon, France, 2002.

11. Priss, U. FcaStone - FCA file format conversion and interoperability software, Conceptual Structures Tool Interoperability Workshop (CS-TIW), 2008.
12. Lahcen, B., Kwuida, L. Lattice Miner: A Tool for Concept Lattice Construction and Exploration. In Suplementary Proceeding of International Conference on Formal concept analysis (ICFCA'10), 2010.
13. Borza, P.V., Sabou, O., Sacarea, C. OpenFCA, an open source formal concept analysis toolbox. Proc. of IEEE International Conference on Automation Quality and Testing Robotics (AQTR), 2010, pp. 1-5.
14. Szathmary, L., Kaytoue, M., Marcuola, F., Napoli, A., The Coron Data Mining Platform (http://coron.loria.fr)
15. Poelmans, J., Elzinga, P., Neznanov, A., Viaene. S., Kuznetsov, S.O., Ignatov D., Dedene G.: Concept Relation Discovery and Innovation Enabling Technology (CORDIET) // CEUR Workshop proceedings Vol-757, Concept Discovery in Unstructured Data, 2011.
16. Grange, E. DelphiWebScript Project (http://delphitools.info/dwscript)
17. Python Programming Language – Official Website (http://www.python.org)
18. Apache Lucene (http://lucene.apache.org)
19. Kuznetsov, S.O.: Stability as an Estimate of the Degree of Substantiation of Hypotheses on the Basis of Operational Similarity. In: Nauchno-Tekhnicheskaya Informatsiya, Ser. 2, Vol. 24, No. 12, pp. 21-29, 1990.
20. Kuznetsov, S.O., Obiedkov, S.A. and Roth, C., Reducing the Representation Complexity of Lattice-Based Taxonomies. In: U. Priss, S. Polovina, R. Hill, Eds., Proc. 15th International Conference on Conceptual Structures (ICCS 2007), Lecture Notes in Artificial Intelligence (Springer), Vol. 4604, pp. 241-254, 2007.
21. Klimushkin, M.A., Obiedkov, S.A., Roth, C.: Approaches to the Selection of Relevant Concepts in the Case of Noisy Data // 8th International Conference on Formal Concept Analysis (ICFCA 2010), pp. 255-266, 2010.