# Analyzing Social Networks Services Using Formal Concept Analysis Research Toolbox

A.A. Neznanov, A.A. Parinov

National Research University Higher School of Economics,
20 Myasnitskaya Ulitsa, Moscow, 101000, Russia
ANeznanov@hse.ru, AParinov@hse.ru

**Abstract.** Nowadays social data analysts use a complicated mix of languages, methods and technologies for analyzing social networks services (SNS) data. In this article we describe approaches and technologies for extracting, analyzing and visualizing social data using Formal Concept Analysis Research Toolbox (FCART). Integrated process of analyzing SNS data with a set of research tools based on Formal Concept Analysis is considered with examples on datasets from Russian segment of LiveJournal.

**Keywords:** Social Network Analysis, Formal Concept Analysis, Graph Mining, Knowledge Extraction, Data Mining, Software.

## 1 Introduction

Social network analysis (SNA) is a popular field with many applications [1]. Today social network analyst faces several challenges:

- Working with real social networks services (SNS) means processing Big Data.
- SNS web-services have different data models and APIs.
- It is hard to develop scaling distributed program architecture for communication with different social networks and data sources.
- It is hard to visualize social data to get insights.

Considering those challenges, data analyst has to learn many data formats, programming languages and systems. Obviously, there is a lack of transparent and flexible tools for SNS data analyst.

Despite the fact that Formal Concept Analysis (FCA) [2, 3] has many applications in different fields, including applications for analyzing social data [4, 5, 6], by now there are no approaches or program platforms which allow an analyst to use FCA methods with other tools for solving real life problems.

FCART is an integrated environment for knowledge and data engineers with a set of research tools based on Formal Concept Analysis [6]. FCART is built as a distributed web-based system with a thick client. It allows FCART to easily integrate various SNA tools.

In previous papers we described the architecture of FCART, main workflow and tools allowing to incorporate third-party programs into analyzing workflow [7]. In this paper we describe how FCART can be used for analyzing data from LiveJournal SNS.

## 2 Distributed FCART architecture

Before discussing data analysis of the specific SNS features using FCART, we need to describe the FCART architecture. SNS data analysis is closely connected to BigData analysis. Usually client computer has relatively small amounts of RAM and relatively slow processor to process Big Data, so distributed architecture is needed. In current distributed version, FCART consists of four parts.

1. FCART Thick Client for interactive data processing and visualization in integrated environment.
2. FCART Intermediate Data Storage (IDS) for storage and preprocessing (initial converting, indexing of text fields, etc.) of big datasets.
3. FCART Web-based solvers (Web-Solvers) for implementing independent resource-intensive computations.
4. FCART Auth Server for authentication and authorization, as well as integration of algorithmic and storage resources.

Further, we call the FCART Thick Client just Client and all FCART Web-services accessed by REST-interface are called just Server. From the user point of view, Client is responsible for user interaction, plugin development, data visualizing, etc. Server is responsible for storing and preprocessing big amounts of data.

Interaction between Client and Server goes through the main Server web-service. Client constructs http-requests. New commands are described in Section 4 of the paper.

## 3 Main FCART data workflow

Social data analysis is a part of main data workflow, which is described in detail in our previous articles [7, 8]. From the analyst point of view, basic FCA workflow in FCART has four stages (see Fig. 1).

1. Filling Intermediate Data Storage of FCART from various external SQL, XML or JSON-like data sources (querying external source described by *External Data Query Description* (EDQD).
2. Loading a data snapshot from local storage into current analytic session (snapshot described by Snapshot Profile). Data snapshot is a data table with annotated structured and text attributes, loaded in the system by accessing external data sources.
3. Transforming the snapshot to a binary context (transformation described by Scaling Query).
4. Building and visualizing concept lattices and other artifacts based on the binary context in a scope of analytic session.

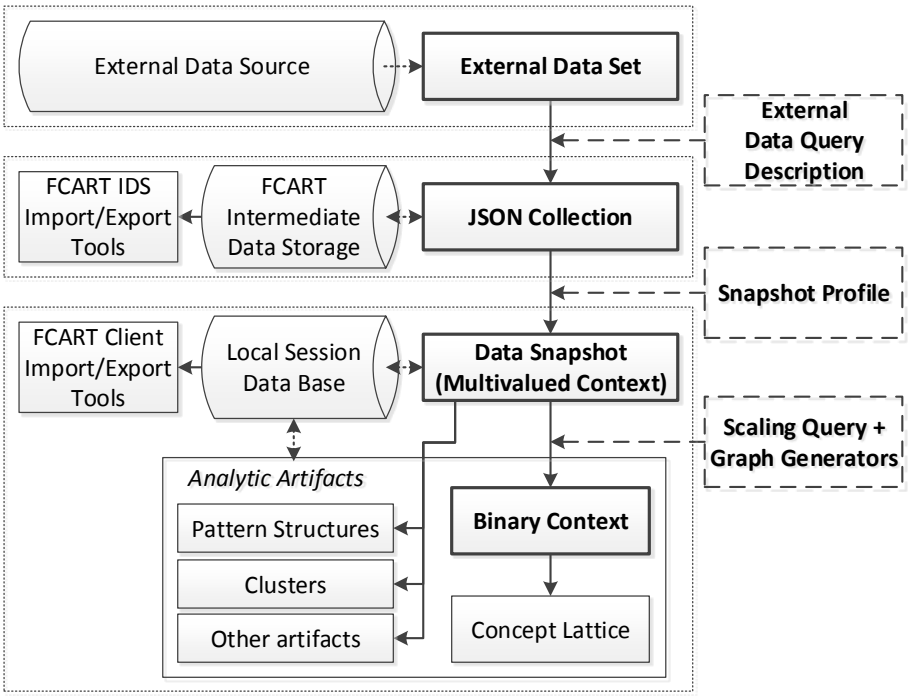


**Fig. 1.** Main workflow of FCART

Requirements of the processing SNS data lead to extension of the middle stage of the workflow. Let us consider data analysis workflow in details.

## 4 Social data analysis workflow in FCART

SNS treated as a data source has several specific properties:

1. Continuous update of data.
2. Special role of data elements timestamps in precedence relations.
3. Unique data model because of unique goal of SNS.
4. Denormalized data model for representing connectivity of nodes.
5. Tricky API suitable for specific SNS task, but not for bulk download of data.
6. Most of SNS have a relatively long time delay between responses to data requests. This delay slows down collection of data.

Incorporating these properties requires methodological, architectural and technological improvements. There are significant differences between the main FCA data workflow and the social data analysis workflow (see Fig. 2).

From the analyst point of view, social data analysis workflow consists of the following steps:

1. Get list of available SNS.
2. Get SNS node's fields that can be collected.
3. Create a work on the Server for extracting SNS data.
4. Start the work execution.
5. Check an execution status of the work.
6. Explicitly build graphs as subgraphs of SNS.
7. Start predefined data analysis algorithms (registered Web Solvers).
8. Create a binary or a triadic context from graphs.
9. Get information to the client side and visualize lattices, clusters, etc.

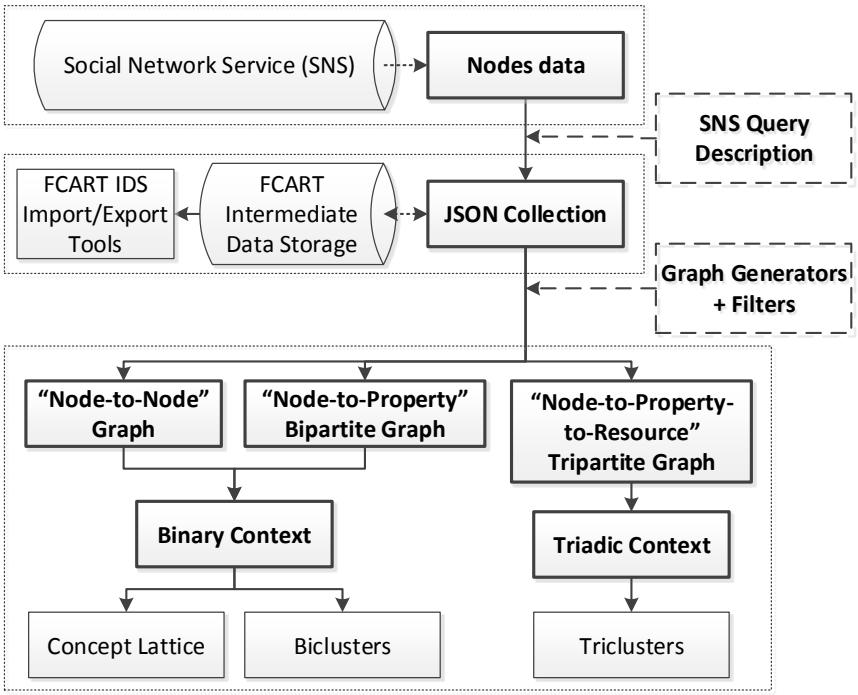


**Fig. 2.** SNS workflow of FCART

All steps of social data analysis are accomplished with the use of Client. In the background, Client interacts with Server using Server REST interface commands. In the previous articles [7,8] we described Server REST interface in detail. Next section of the paper describes new SNS commands of the interface.

### 4.1 Social networks REST interface commands

This section of the paper contains Server REST interface commands. Using new commands the analyst extracts data from available SNS and executes predefined algorithms. Each http-request is constructed from two parts:

1. *Prefix part* contains domain name and local path (e.g. http://zeus2.hse.ru:8443 for our main test installation).
2. *Command part* carries command, its parameters and data.

New commands are described in Table 1. Some commands expand iteration abilities and other commands deal with SNS Query.

Table 1. Interface commands for social network services

| № | Command | Description |
|---|---|---|
| 1 | sns.works | Return current works and its status in JSON document |
| 2 | sns.create_work\| <SNSQ> | Create new work for extraction data using social network services query (SNSQ) file |
| 3 | sns.<work_id>\| [start, stop, pause] | Commands to start, stop and pause work with work_id |
| 4 | sns.newid | Get new identifier |
| 5 | sns.names | Get names of supporting social network services |
| 6 | alg.names | Get names of existing algorithms |
| 7 | alg.start?alg=<Algorithm Name>& source=<SourceCollection>& source_fields=[<Field Name>]& target=<Target Collection> | Start predefined algorithm specifying source collection name and fields. |
| 8 | <dbName>.<collectionName>.niter\|INDEX, COUNT | Create new Iterator for enumerating Count elements in Collection starting from Index |
| 9 | <dbName>.<collectionName>.niter\|next | Get next element from last created Iterator |
| 10 | <dbName>.<collectionName>.niter\|cur | Return index of iterator current element in Collection |

For now Server interface is not fully RESTful [9], but we are working on it.

To create a Server work for extraction data the Client sends to the Server a special file, a social network services query (SNSQ) file, which contains necessary information about types of data that should be extracted. Syntax of the SNSQ is described in the next section.

### 4.2 Social networks services query

Described properties of SNS workflow prevent from developing universal program suitable for extraction from every SNS. To unify extraction data from various social SNS we use *social networks services query* (SNSQ) as an additional class of *External Data Query*. SNSQ is a JSON formatted file. It describes data fields that data should be extracted from SNS. SNSQ is transferred to IDS by REST interface and is defined by the following syntax:

```
{
```

```
"work_id": <Number>,
"sns" : <sns_Name>,
"type": "bulk" | "connected",
"start_node" :  [<String>],
"extract" : "info"| "foaf" |"full"
}
```

where "*work_id*" is a work identifier which gets from REST interface by requesting command; "*sns*" is an SNS name which gets from REST interface by requesting command; "*type*" is a downloading mode ("bulk" or "connected"); "*start node*" is an identifier of a first extracting node; "*extract*" is a type of extracting content of the nodes. Type of extraction can be "*info*", i.e., profile information such as date of birth, school, list of interests, etc.; "*foaf",* the list of friends nicknames; "*full"*, profile information and the list of friends.

### 4.3 Download social networks services data

Using the previous release of FCART an analyst could work with quite small external data storage because all data from external storage were converted to JSON files and saved to Client's data storage. In the current release, we have improved the strategies of working with external data. Now the analyst can choose between loading all data from external data storage to Server's data storage and accessing external data by chunks using paging mechanism. FCART analyst should specify the way of accessing external data at the property page of the generator. All these tasks are accomplished by means of the Client's interface.

In current release, IDS supports two basic scenarios for accessing SNS data. The first scenario is to parallelize bulk download data from SNS. This operation is more time consuming than selective downloading. The second scenario is to extract data from SNS by selective nodes information download. This scenario is used for fast analyzing up-to-date data. In both scenarios, extracted data can be stored to the one of the Server data storage collections or stored to the cloud storage. Now FCART supports Amazon Simple Cloud. Saved to cloud storage data is obtained using paging mechanism.

The bulk download is useful for getting relatively big amounts of data which can take days and weeks for collecting. For reducing collecting time, we developed a parallel architecture of program system, which consists of two types of components: *Manager* and *Agent*. In general, social data download process has the following steps (Fig. 3):

1. Manager gets unique identifiers of nodes and saves nodes identifiers to Task Queue.
2. At the same time, Agents listen to the Task Queue. When an identifier appears in the Task Queue, an Agent gets identifier.
3. The Agent checks there is no such identifier in the Completed Task List and starts downloading information about a node to a file on the local computer.
4. The Agent appends the file to the IDS collection.
5. The Agent has downloaded the node information from SNS. It inserts the identifier to the Completed Task List and listens to the Task Queue for the next identifier.

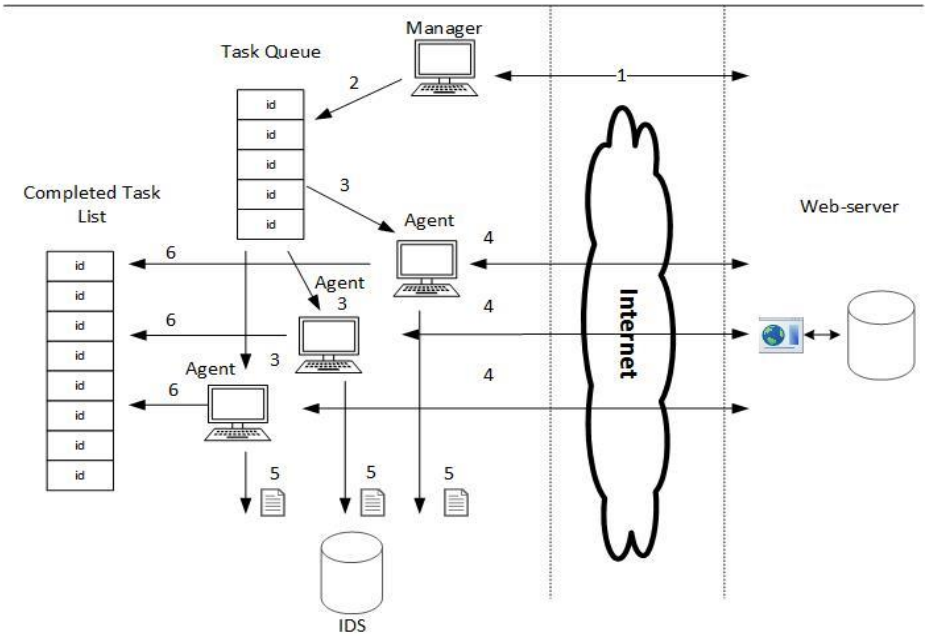


**Fig. 3.** Bulk downloading of SNS data

To access data from SNS an appropriate data adapter should be developed. The Manager and the Adapters use a data adapter to create SNS-specific queries. To date, we have developed data adapters for LiveJournal and Facebook.

Many SNSs limit number of http-requests per second. Gathering program has to wait for some time between sending requests to a SNS server. It is a big problem for fast social data gathering. FCART supports a selective download of nodes information. It is useful for fast gathering information about specific nodes and their connections. Using selective download analyst can get periodic information that is useful for incremental data update.

### 4.4 Traversing social graph

Traversing social graph is highly dependent on Server data storage system, which actually stores data and processing algorithm. After reviewing many systems, we have chosen the MongoDB [10] as a backbone of IDS. MongoDB is an open-source cross-platform high scaling document-oriented data storage system. MongoDB can process data in MapReduce parallel style. It uses JSON format (and BSON internally) as a native format for stored documents. Storing documents in JSON format and using unique nodes identifiers gives ability to connect documents with each other [11]. Example of traversing is described below.

## 5 Use case: Extracting data from LiveJournal

LiveJournal is a social blogging system [12] very popular in Russia, it is one of the most popular social networking services after VKontakte (vk.com), Odnoklassniki (ok.ru) and Facebook. Using LiveJournal Web API we extracted more than 180000 person's profiles. Then we carried out some classical SNA experiments and tested improved subsystems of FCART.

### 5.1 Nodes content as LiveJournal profile

Initially IDS takes five initial profiles and initiates process of downloading the neighbourhood of these persons by relation "has a friend". Each profile consists of several blocks of fields [13]:

1. Nickname ("*nick*" – unique identifier).
2. List of friends ("*friends*" – array of nicknames).
3. List of users who checked this profile as a friend ("*friendsOf*" – array of nicknames).
4. List of interests ("*interest*" – array of tags).
5. List of watching communities ("*watching*" – array of community names).
6. List of memberships in communities ("*memberOf*" – array of community names).
7. List of posts (not used in next stages).
8. Other personal information.

In LiveJournal, friendship relation is not mutual by default. Mutual friendship is a situation when $b \in a.friends$ and $a \in b.friends$.

For each profile, LiveJournal SNS Query extracts its friends list and adds new nicks into downloading queue.

Time for extracting 15 thousands profiles is about 19 hours (average time evaluated after three extraction sessions in January 2015 without any special access rights).

### 5.2 Executing SNS query and Client graph query

Let us open "IDS Query" window, login to server and refresh IDS structure available for us. In our case, we can see database "lj" (for experimental datasets from LiveJournal). After selecting this database the user can select one of database collections and discover its content. Each document from collection is represented in form of raw text and tree. Both forms support full text search.

After selecting a collection the user has two possibilities:

1. Using "snapshot profile builder" (or ready profile) and generate snapshot as a multivalued context.
2. Using new tool called "Graph query".

Fig. 4 shows parameters of "node-to-node" graph query. Such query produces graph based on the relation between identifiers of nodes. In our case we select "Nick" tag as identifier of node and "Friends" tag as array of links to other nodes in the data

collection. Of course, if nodes do not have such links array, the user needs above mentioned extension of IDS interface or can try to build intermediate snapshots.

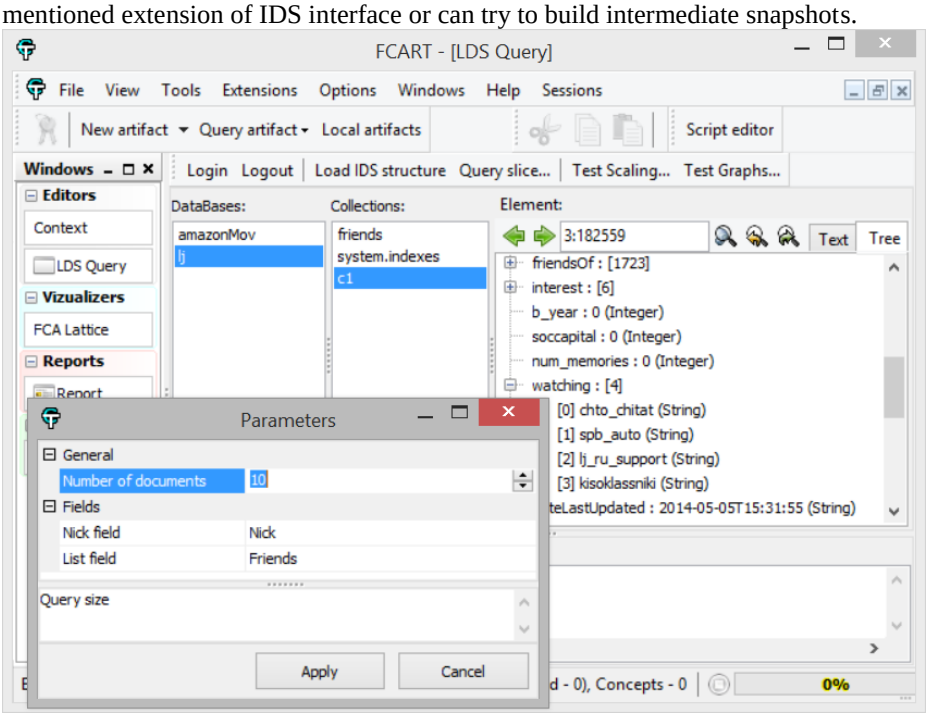


**Fig. 4.** IDS query builder and parameters of "node-to-node" graph query

The result of "node-to-node" graph query is a digraph, which is saved as an artifact in FCART and can be transformed into binary context. FCART supports interactive browsing of concept lattices. Only ten random person's profiles from LiveJournal dataset (and all their friends) have brought about 5357 vertices in resulting graph and objects and attributes in corresponding context (see Fig. 5). Time of building this graph is only 0,3 minutes (on Intel i7 4600 2 GHz).

Other type of graph query is a "node-to-property" query for producing bipartite graph (formal context). For example, in LiveJournal data we can select person "interests" as a property and build context where objects are users and attributes are their interests. However, the most beneficial type of graph query is "node-to-property-to-resource" [14]. Now we test the implementation of this new feature as a different approach to solve prediction tasks, like in [15]. For tripartite graphs FCART does not build a trilattice, but constructs tricontext and set of triclusters.

### 5.3 Interactive part of social media mining with FCA

The most interesting part of workflow for the analyst is the interactive work with artifacts in multiple-document user interface of FCART client. We will illustrate the workflow with real examples of SNS data and working with graph data.

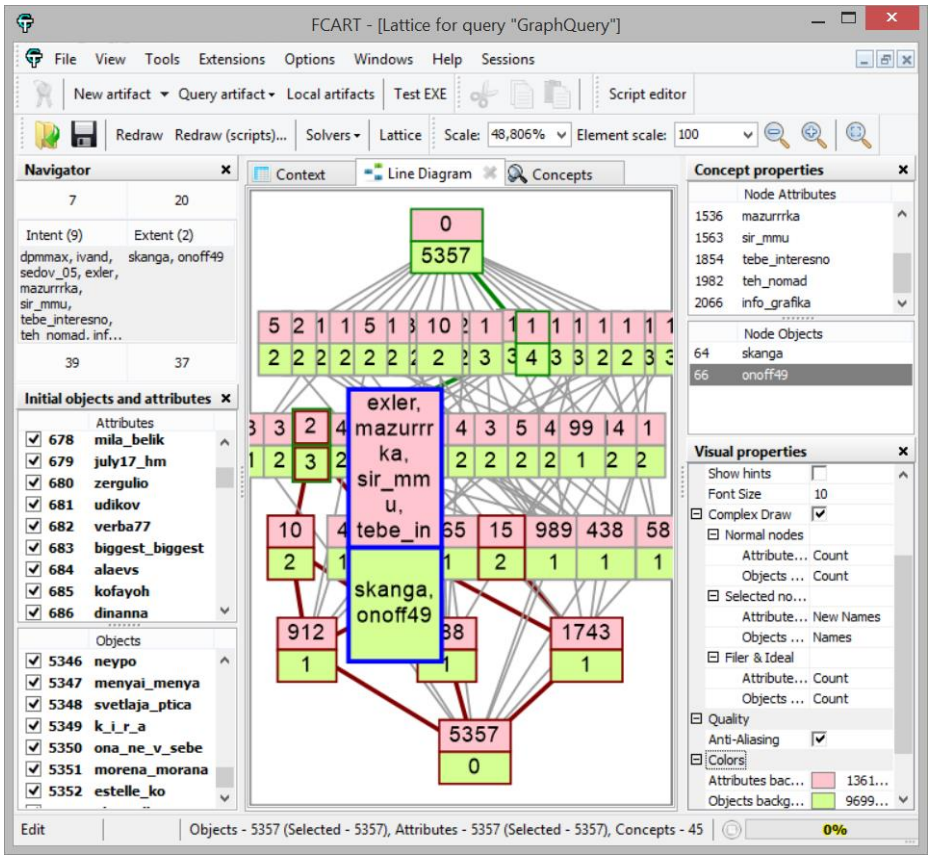


**Fig. 5.** Concept lattice browser

The system has special instruments to visualize and navigate big lattices: scaling of lattice element sizes, parents-children navigator, filter-ideal selection, separation of focused concepts from other concepts. FCART has several drawing techniques for visualizing fragments of big lattices: for rendering focused concept neighborhood and for accenting (by layout and highlighting) selected concepts, "important" concepts [16], and filter-ideal with different sorting algorithms for lattice levels.

A user also can build linked sublattices and compute standard association rules, implication bases, concept indices, etc. All artifacts can be commented, annotated and appended to one of reports in a current session.

Stability indices of concepts [17] have high importance in current research, because they allow filtering uninteresting concepts, e.g., selecting only non-primitive friendship concepts.

A special feature of FCART is separate mode of comparison of two lattices with nonempty intersection of their objects. This mode allows finding similarity between different contexts (and underlying graphs). It can be used for dynamic network analysis.

## 6 Conclusion and future work

In this paper, we have discussed extended workflow of social data extraction from SNS and analyzing such data with FCA methods in FCART software. The case of extracting profiles and friendship graphs from LiveJournal social blogging system was considered.

We plan to develop more SNS data providers, increase efficiency of sparse graph processing and implement tripartite graph building as the next improvement into graph querying subsystem for the use of triclustering methods.

## Acknowledgements

This work was carried out by the authors within the project "Data mining based on applied ontologies and lattices of closed descriptions" supported by the Basic Research Program of the National Research University Higher School of Economics.

## References

1. Prell, C. Social Network Analysis: History, Theory and Methodology, SAGE, 2011.
2. Ganter, B., Wille, R. Formal Concept Analysis: Mathematical Foundations, Springer, 1999.
3. Kuznetsov, S.O. Pattern Structures for Analyzing Complex Data // Proc. of 12th International conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC-2009), 2009, pp. 33-44.
4. Aufaure, M.-A., Le Grand, B. Advances in FCA-based Applications for Social Networks Analysis. International Journal of Conceptual Structures and Smart Applications, Vol. 1, Issue 1, 2013, pp. 73-89.
5. Poelmans J., Ignatov, D.I., Kuznetsov, S.O., Dedene, G. Formal concept analysis in knowledge processing: A survey on applications. Expert Systems with Applications, 40, 2013, pp. 6538-6560.
6. Ali, S.S., Bentayeb, F., Missaoui, R., Boussaid, O. An Efficient Method for Community Detection Based on Formal Concept Analysis // Proc. 21th International Symposium on Methodologies for Intelligent Systems (ISMIS-2014), LNCS, Vol. 8502, Springer, 2014, pp 61-72.
7. Neznanov A.A., Ilvovsky D.A., Kuznetsov S.O. FCART: A New FCA-based System for Data Analysis and Knowledge Discovery // Contributions to the 11th International Conference on Formal Concept Analysis, 2013, pp. 31-44.
8. Neznanov, A., Ilvovsky, D., Parinov, A. Advancing FCA Workflow in FCART System for Knowledge Discovery in Quantitative Data // 2nd International Conference on Information Technology and Quantitative Management (ITQM-2014), Procedia Computer Science, 31, 2014, pp. 201-210.
9. Fielding, R.T., Architectural Styles and the Design of Network-based Software Architectures. PhD dissertation, University of California, 2000 (http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm)
10. MongoDb (http://mongodb.org)

11. Model One-to-One Relationships with Embedded Documents (http://docs.mongodb.org/manual/tutorial/model-embedded-one-to-one-relationships-between-documents/ )
12. LiveJournal (http://livejournal.com)
13. LiveJournal XML-RPC Client/Server Protocol Reference (http://wh.lj.ru/s2/developers/f/LiveJournal_XML-RPC_Specification_(EN).pdf)
14. Gnatyshak, D.V., Ignatov, D.I., Semenov, A., Poelmans, J., Analysing Online Social Network Data with Biclustering and Triclustering // 2nd International Workshop Concept Discovery in Unstructured Data (CDUD 2012), 2012, pp. 30-39.
15. Hsu, W.H., Lancaster, J., Paradesi, M.S.R., Weninger, T., Structural Link Analysis from User Profiles and Friends Networks: A Feature Construction Approach // International Conference on Weblogs and Social Media, 2007, pp. 75-80.
16. Belohlávek, R., Trnecka, M. Basic Level of Concepts in Formal Concept Analysis // 10th International Conference on Formal Concept Analysis (ICFCA 2012), 2012, pp. 28-44.
17. Kuznetsov, S.O., On Stability of a Formal Concept. Annals of Mathematics and Artificial Intelligence, 40, 2007, pp.101-115.