Full-text Search in Intermediate Data Storage of FCART

Alexey Neznanov, Andrey Parinov

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

Abstract. The speed of full-text search directly affects the process of text analysis. Search engine creates a text index, which is used for fast full-text search. Solr and ElasticSearch are two popular search engines. A text analysis system requires fast implementing searching and indexing at the same time. This paper describes preprocessing workflow of the analysis system called Formal Concept Analysis Research Toolbox (FCART) and experiment of searching and indexing social networking service data at the same time. Results of the experiment show which search engine is better as the core of FCART search subsystem.

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

1 Introduction

Formal Concept Analysis Research Toolbox (FCART)[1] is a data analysis system, which supports texts knowledge discovery techniques, including those based on Formal Concept Analysis[2], clustering [3], multimodal clustering [4, 5], pattern structures[6]. The system supports iterative methodology of knowledge discovery. The goal of developing FCART is to create a system for handy texts analysis from external data sources, e.g. SQL databases, NoSql databases and Social Network Services. In previous papers, we have described the system architecture, main workflow and stages of data extraction from various external sources.

Fast search of relevant documents in a big dataset is an important function of a text analysis system. Development of search engine from scratch is time-consuming and unnecessary since many search engines are known so far, Solr [7] and ElasticSearch [8] being among most popular ones [9].

In this paper we describe key features of search engines mentioned above, describe preprocessing workflow of FCART, requirements to the full-text search engine as a part of an analytic system and then describe the experiment of searching and indexing at the same time of data that was gathered from social network service LiveJournal [10]. Based on the results of the experiment, we choose a search engine that would better serve as the core of FCART full-text search subsystem.

2 FCART architecture and preprocessing workflow

Formal Concept Analysis (FCA) has many applications in different fields [11,12]. "Formal Concept Analysis Research Toolbox" (FCART) is an integrated environment for knowledge and data engineers with a set of research tools based on Formal Concept Analysis. FCART is built as a distributed web-based system with a thick client. In new distributed version FCART consists of four parts:

1. FCART *Intermediate Data Storage* (IDS) for storage and preprocessing (initial converting, data preprocessing, etc.) of big datasets;
2. FCART *Web-based solvers* (Web-Solvers) for independent resource-intensive computations;
3. FCART *Auth Server* for authentication and authorization;
4. FCART *Thick Client* for interactive data processing and visualization in the integrated environment.

From an analyst point of view, basic FCA workflow in FCART has five stages (see Fig. 1).

1. Filling Intermediate Data Storage of FCART from various external SQL, XML or JSON-like data sources (querying external source is described by External Data Query Description (EDQD);
2. Data indexing and preprocessing;
3. Loading a data snapshot from a local storage to the current analytic session (snapshot described by Snapshot Profile). Data snapshot is a data table with annotated structures and text attributes, loaded in the system by accessing external data sources;
4. Transforming the snapshot to a binary context (transformation described by Scaling Query);
5. Building and visualizing concept lattices and other artifacts arising from formal contexts (binary data tables) within an analytic session.

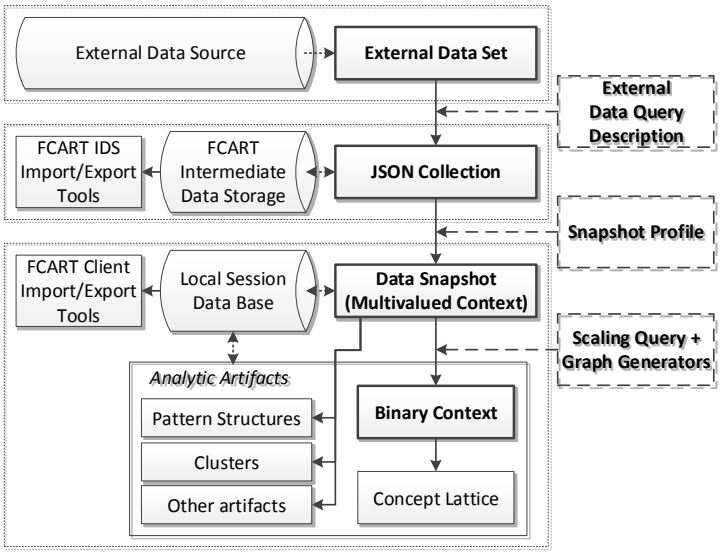

Fig. 1. Main FCA workflow of FCART

All steps of data analysis are accomplished with the use of Client program. In the background, Client interacts with Server using Server REST interface commands. In the previous articles, we described Server REST interface in detail.

In the current version of FCART the preprocessing workflow consists of two steps:

1. Normalizing incoming documents
2. Creating index

At normalizing step, FCART creates a document in JSON format, selects one text field as identifier and calculates document metrics (size, word count, etc.).

At index creating step, FCART creates index on one or multiple text fields. This index is used for fast data search.

3 Comparison of popular full-text libraries and systems

Full-text indexing library adds content to a full-text index. It then allows performing queries on this index, returning results either ranked by the relevance to the query or sorted by an arbitrary field such as document's last modified date. The library gets fast search responses because -- instead of searching the text directly -- it searches the index. The library cannot be used independently, it can be used as a database plugin or as a part of another software system.

There are several key features of the full-text search library:

1. Search, initial indexing and reindexing speed;
2. Support of languages and dialects;
3. List of supported programming languages;
4. Format of supported documents;

5. Stemming algorithms;
6. List of metrics and ranking methods;

The full-text indexing system is a standalone, independent product. It is based on full-text indexing library and provides more features. For example, Solr and ElasticSearch are based on same library called Lucene[13]. Besides the features mentioned above, the full-text system provides the following ones:

1. Support of multiple indexes based on different fields
2. Scalability
3. Support for complex search expressions
4. Ranking and grouping of search results
5. Ease of use and ease of integration with data storage (e.g. MongoDb)

The final value of each feature of a particular software system depends on CPU, number of cores, total amount of memory, etc. Solr and ElasticSearch provide very similar sets of features [14]. These sets of features are enough to satisfy our current needs. We deploy ElasticSearch and Solr on our server and execute similar experiment. In the next section we describe an experiment on indexing data gathered from the social network service Liverjournal.

4 Indexing data description

LiveJournal is a popular in Russia social blogging system. Using LiveJournal Web API we have extracted 1000,100 000, 1000000 person's profiles. Then we carried out some classical SNA experiments and tested improved subsystems of FCART.
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 :

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

Table 1 contains time results of gathering data. Figure 2 illustrates results of gathering data.

Number of parallel	Time for gathering, minutes			
	1000	10000	100000	1000000

threads	Profiles	Profiles	Profiles	Profiles
1	18	195	3671	42710
10	2	21	291	4483
100	0.2	3	32	471

Table 1. Time for extracting data from LiveJournal

Fig. 2. Time for extracting data from LiveJournal

5 Experiment of searching and indexing data

Solr 5.4 and ElasticSearch 2.0 have been deployed on a computer with installed INTEL Xeon Processor E5-2670 (2,60 GHz, 4 cores) 8Gb of RAM [15]. We have compared time of search the systems spend on indexing data. Table 2 contains results of the experiment. Fig 2 illustrates results of the experiment.
Experiment shows advantage ElasticSearch over Solr from 3 to 7 times.

Indexing system	Time for indexing, milliseconds			
	1000 profiles	10000 profiles	100000 profiles	1000000 profiles
Solr	3000	5700	6200	6900
ElasticSearch	5	16	29	42

Table 2. Solr and ElasticSearch time of searching while performing indexing

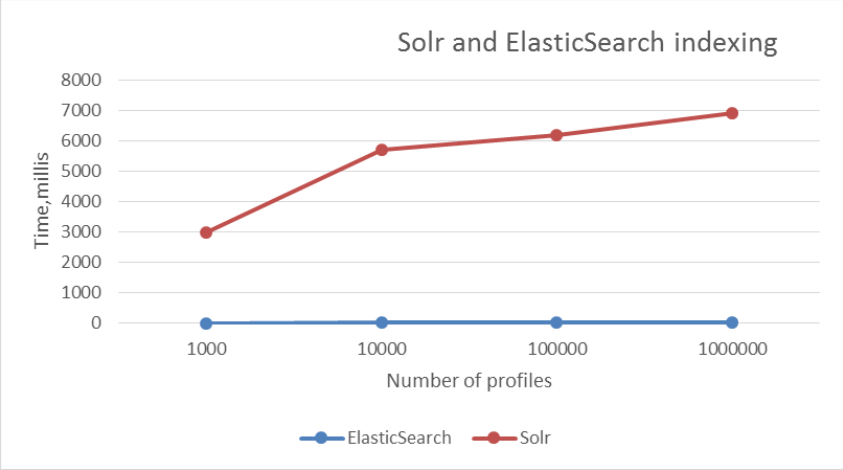

Fig. 3. Solr and ElasticSearch time of searching while performing indexing

Conclusion

In this paper, we have described key features of two popular full-text search engines, architecture and preprocessing workflow of FCART software. The case of searching data and indexing at the same time was considered. Our experiment have shown significant advantage ElasticSearch over Solr. In future work we will use ElasticSearch as the core of the search subsystem. Next FCART release will be integrated with ConceptCloud[16] and expanded by the function of indexing texts using parsed thickets index[17]

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. 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.
2. Ganter, B., Wille R. Formal Concept Analysis: Mathematical Foundations, Springer, 1999.

3. 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.
4. Mirkin, B. Mathematical Classification and Clustering, Springer, 1996.
5. 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.
6. 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.
7. Apache Solr (http://lucene.apache.org/solr/)
8. ElasticSearch (https://www.elastic.co/)
9. Ranking of Search Engines (http://db-engines.com/en/ranking/search+engine)
10. LiveJournal (http://livejournal.com)
11. Jonas Poelmans, Sergei O. Kuznetsov, Dmitry I. Ignatov, Guido Dedene, Formal Concept Analysis in knowledge processing: A survey on models and techniques. In: Expert Systems with Applications, Vol. 40. No. 16, pp. 6601-6623, 2013.
12. 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.
13. Apache Lucene (https://lucene.apache.org/)
14. Apache Solr vs Elasticsearch (http://solr-vs-elasticsearch.com/)
15. Zeus (http://zeus.hse.ru:8080)
16. Gillian J. Greene and Bernd Fischer. 2014. ConceptCloud: a tagcloud browser for software archives. In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014). ACM, New York, NY, USA, 759-762.
17. Galitsky B., Ilvovsky D., Kuznetsov S. Text integrity assessment: Sentiment profile vs rhetoric structure, in: Computational Linguistics and Intelligent Text Processing. 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part II. Vol. 9042. Springer International Publishing, 2015. P. 126-139.