NYSOL: A User-Centric Framework for Knowledge Discovery in Big Data

Stephane Cheung, Masakazu Nakamoto, and Yukinobu Hamuro

***Abstract*—NYSOL is an integrated framework of knowledge discovery leveraged by a host of data processing and data mining tools, which is underpinned by innovative research activities. Our framework is designed for end-users to integrate the process of managing large-scale information assets and knowledge discovery on one platform to improve interoperability between processes. The fundamental principle of the framework is derived from direct processing of text-based data by a set of user customizable commands for data management, data processing, and data analysis, which greatly simplifies the software architecture. The NYSOL framework facilitates the knowledge discovery process in an efficient manner for novice and expert users. This paper discusses the historical development of NYSOL rooting from basic data processing commands at command line, to the recent growth of the NYSOL software ecosystem to extend additional components for data mining based on efficient machine learning algorithms. Initial experiments on NYSOL's GGP large-scale information processing architecture with NYSOL distributed file system (NDFS) are also presented. Observed performance of GGP demonstrates reduced overhead for inter-processing time and improvements in overall processing time.**

***Index Terms*—Big data, data mining, distributed processing, information processing, knowledge discovery.**

I. Introduction

Knowledge discovery in Databases (KDD) is a dynamic process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [1]. The steps constituting the process are data selection, data pre-processing, data transformation, repeated analysis, and interpretation and evaluation of patterns to extract useful knowledge. The KDD process is often centered around defined problems by end users, where multidisciplinary professionals and students are increasingly involved in the process for performance reporting and applied research. However, the process of obtaining information involves a slow import and query process from structured database systems from experts, and perplexing pre-processing and standardization of various data formats, from disparate systems. When transformed data is ready for analysis, users are challenged to create custom programs and sophisticated analysis tools for model fitting and to determine pattern results. Thus, it is important to narrow the gaps within the information realm for both novice and expert users to actualize a more user-centric and efficient KDD process.

Gartner defined big data as high-volume, high-velocity, and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision-making. The rapid emergence of big data has amplified the need to identify useful information from petabytes of data in an efficient manner. Besides click streams and transaction histories, researchers are increasingly integrating assorted sources of open data stored in separated spaces in different data structures [2].

This paper describes the software ecosystem of NYSOL, and its efficient framework designed to integrated all information in one place by using the set of commands to process text-based data. This methodology provides users a holistic view of information with high level of control of information throughout the KDD process. The present work includes the addition of data mining components to the existing array of data processing commands, to strengthen seamless processing of data throughout the KDD process and enhance time and computational efficiency.

II. Concept of the NYSOL Project

A. User-Centric Approach of NYSOL

The concept of NYSOL can be traced back to the early 1990s. Yasuyuki Matsuda invented this data processing methodology and a set of commands (M-COMMAND) as a simple, special-purpose programming language for the development of large-scale information system development projects. The philisophy is based on a bottom-up approach as he recognized that typical information system does not always fits specific business processes across all industries, and operation staff who are the actual users of information systems potentially understands what is needed for an efficient business information system. Nevertheless, typical information contains functions and which are too complex for users. As a result, users are often overwhelmed and need to spend additional time to understand and manage the system. Matsuda's philosophy simplifies the application layer and database layer of typical systems by applying a set of commands with specific functions to process plain text data efficiently. This framework empowers operations staff to customize a simple yet highly functional system to accumulate of all information from its business processes and operations. All information stored can be used for process improvements at operations level, and decision-making at

Manuscript received April 20, 2015; revised August 21, 2015. This work was supported in part by JST-ERATO Minato Discrete Structure Manipulation System Project.

Stephane Cheung and Masakazu Nakamoto are with JST ERATO Minato Discrete Structure Manipulation System Project, Japan. They are now with Kwansei Gakuin University, Japan (e-mail: stephane@erato.ist.hokudai.ac.jp, nain0606@gmail.com).

Yukinobu Hamuro is with Kwansei Gakuin University, Japan (e-mail: hamuro@kwansei.ac.jp).

doi: 10.18178/ijke.2015.1.3.037

management level. The methodology was applied in a few nation-wide chain retailers in Japan, and the M-COMMAND was implemented for large scale system development projects in a drug store chain [3], and subsequently in a meat retailer and wholesaler in Japan. Matsuda's invention is not just a technology breakthrough, it fundamentally examines how "information" system can be customized for business use. The key concepts for creating this efficient system framework are as follows:

1) Specialization

Data processing is carried out by a group of specialized commands where each command is assigned a specific function such as aggregate, extract, join, etc. Its modular nature allows users to build customized information systems by solely using a combination of commands and UNIX utilities with shell scripts. This framework has low computation requirements, cost effective, and is efficient for processing hundred million rows of data.

2) NoSQL

Data is stored in a simple, NoSQL mechanism in a standardized text format that is non-relational, open-source, and scalable. M-COMMAND directly carries out specific processes on CSV tabular data files without the need to access multiple relational databases. This framework vastly reduces time required for database design, data retrieval from multiple databases, and data processing on different formats. In addition, data volumes can be easily scaled for big data.

3) UNIX philosophy

The commands follow the UNIX standard and can be executed at UNIX shell. Data stream is redirected from one command to the next through standard streams. UNIX platform is ideal for processing big data, information can be transmitted on the UNIX client/server architecture in a cloud environment, and it supports multi-threading and parallel processing of big data.

4) Open source

Libraries used in the NYSOL tools are open source, which allows customization and integration with other systems. The output from NYSOL tools is compatible with open source data mining software such as R, Weka, Gephi, etc. The total cost of ownership for users is therefore reduced.

Based on the above concepts, the NYSOL team developed a host of flexible data processing commands since late 1990s for big data and information management. Robust data mining tools underpinned by machine learning algorithms are eventually added to the ecosystem to enhance the knowledge discovery functionality. These tools are user friendly for multidisciplinary users. Its flexibility allows users to enforce data consistency, manage and fine tune their own data processes, and understand results from data mining models to convey processed information in various forms for interpretation.

B. Evolution of NYSOL

The first generation known as M-COMMAND (MCMD) is developed in C in the late 1990s anchored by Yukinobu Hamuro. In 2004, MCMD was jointly developed into MUSASHI by Naoki Katoh, Takashi Washio, and Yukinobu Hamuro [4]. MUSASHI processes data structured in XML Table with enhanced data definition, and the commands are implemented with new algorithms to enhance processing efficiency. Basic analysis tools are also introduced to enhance analysis capability. MUSASHI was awarded the best prize at the Fundamental Artificial Intelligence Workshop by Japanese Society for Artificial Intelligence (JSAI). During the Open Information Project from 2007 to 2009, Business Mining Research Center (BMRC) was established to collaborate with businesses and formulate marketing and business strategy. Software performance was also enhanced with the development of KGMOD library as the foundation for development of various data processing applications [5]. The new MCMD commands are developed using the KGMOD libraries and BOOST C++ libraries with improved algorithms for high speed data processing of CSV text based data.

C. Current Developments during The NYSOL Project

The NYSOL Project is launched in 2012 supported by JST-ERATO Minato Discrete Structure Manipulation System Project based in Hokkaido, Japan. "NYSOL" is an Ainu terminology indigenous to Hokkaido, which means "cloud". Thus, the term "NYSOL" entails two meanings – "Hokkaido" and "cloud era". The NYSOL Project aims to drive technology innovation in KDD for community of novice and expert users. To date, NYSOL has substantiated new initiatives towards the expansion of the NYSOL software ecosystem by adding new data mining and data manipulation tools to the NYSOL software package to extend processing functionality throughout the KDD processes from conversion, processing, modeling to visualization.

For instance, core data processing functionality of KGMOD libraries was refined, with the development of five additional data mining and visualization components to expand information processing and knowledge discovery functionality. These components also broaden compatibility with text, graph, and zero-suppressed binary decision diagram (ZDD) data structures. At the same time, efficient machine learning algorithms are continuously refined. Details of the components are presented in section III. The data mining tools are developed with Ruby, in addition Ruby wrappers are developed for existing commands, for in-memory computation of data objects to increase processing efficiency, and users can customize data processing by integrating the commands with other Ruby functions and UNIX based processes. Systems-based research initiatives include the development of distributed data processing architecture for NYSOL tools known as GGP aimed for big data distributed processing in a cloud environment.

This project is built upon a close collaboration structure between academia and businesses. NYSOL was incorporated in 2013 to facilitate the implementation of applied research. Core tools and algorithms are developed through research, seeding innovative applications and analysis methodologies for use in business practices. Real business transaction data and open data from news and social media are used for the development of innovative analysis methodologies and validation of data mining tools. For instance, the results of graph mining and text mining research for the analysis of

financial news were successfully implemented into a business model to determine the effects of news sentiments on stock performance [6].

III. Components of NYSOL

The building blocks of NYSOL are designed to reinforce a user-centric process in data mining by streamlining connections throughout the KDD process.

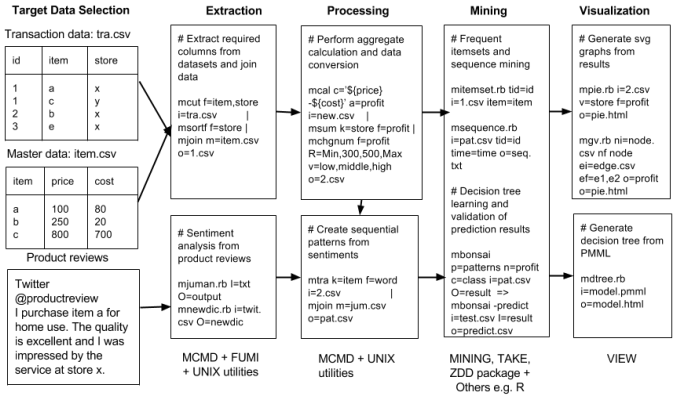


Fig. 1. Relationship of NYSOL components in the KDD process.

Fig. 1 illustrates the relationship of the building blocks of NYSOL in the KDD process. The process of data extraction, processing, mining, and visualization are inter-connected, and the series of commands across various components can be executed in a single shell script file. Further, input and output data are stored and processed as text files across all components. Six key components are described in the following sections.

A. M-Command (MCMD)

MCMD is a set of commands, each with specific data processing function, applied as simplified programming language designed to support the processes of data management, data pre-processing and data aggregation on one platform. MCMD is the core component of NYSOL with a set of 76 data processing commands, each with specific functions designed for efficient information processing from large-scale raw data. For instance, *msortf* command can be used to sort data by user defined criteria, and *mcal* can carry out excel like functions. Refer to Fig. 1 for the process flow of commands, and Fig. 4 for sample script to execute six commands.

B. MINING

The MINING component comprised of commands incorporating data mining and machine learning algorithms, which simplifies the creation and interpretation of resulting models. Users can apply various data mining techniques with customizable parameters on pre-processed data, including classification with naive Bayes classifier (mnb.rb), detection of burst state with hidden Markov model in time series data (*burst.rb*), graph partitioning with METIS (*mgpmetis.rb is developed as a Ruby wrapper on the original METIS command*) [7], and sequential patterns classification with decision tree (*mbonsai*) [8]-[10].

C. TAKE

Itemset mining and graph mining tools are implemented in the TAKE component. TAKE is based on a high-speed pattern enumeration algorithm Linear time Closed itemset Miner (LCM)[1] developed by Takeaki Uno [11]. LCM is commonly applied to discover frequently appearing patterns, it can also enumerate maximal patterns and closed patterns. Features such as emerging patterns and taxonomy for hierarchical classification are added to TAKE tools. TAKE includes powerful graph mining tools to compute similarity between graphs, enumeration of maximal cliques, graph polishing, and subgraph clustering.

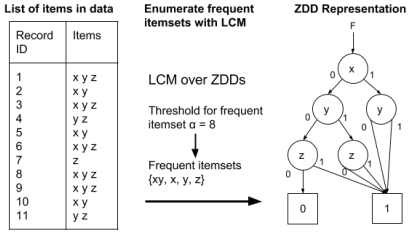


Fig. 2. Example of enumerating frequent itemsets using LCM over ZDD when minimum support frequency is set as 8.

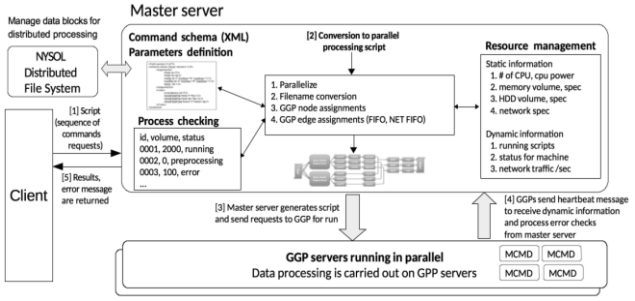


Fig. 3. Configuration of NYSOL GGP architecture.

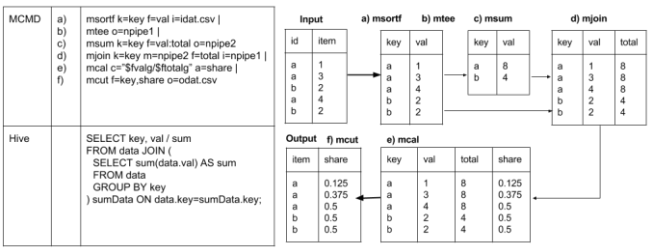


Fig. 4. Comparison of benchmark testing program with HiveQL and MCMD based on the example process flow for the calculation of ratio.

TABLE I: Details of the Five Types of Data Processing Tasks

Tasks	Details
Cut	Select all columns, equivalent to copy operation, which consumes low CPU usage.
Sum	One column is set as aggregate key and sorted in ascending order, sum is computed based on each key.
Sort	Sort all random records by key.
Share	Calcuate the composition ratio on pre-processed data.
Rule	Calculate association rules between 2 items. 18 commands are used, and aggregation key is assigned 3 times.

D. ZDD

The ZDD component is based on the algorithm developed by Shin-ichi Minato. The techniques of ZDD and BDD are the key research topic within the ERATO Project. ZDD is a

[1] Details of LCM implementation can be found at: http://research.nii.ac.jp/~uno/code/lcm_seq.html

variant of Binary Decision Diagram (BDD). It applies new reduction rules for manipulating combination sets, which is effective for handling a set of sparse combinations, and is more compact than ordinary BDDs [12]. ZDD can store a million combinations of itemsets enumerated by LCM in a compact graph-based structure on main memory, thus reducing overhead on I/O. This component contains 58 ZDD operations for various calculations for the manipulation of ZDD objects. An example of frequent itemset mining using LCM over ZDD algorithm is illustrated in Fig. 2. Input data contains 11 transaction records, itemsets *xy*, *x*, *y*, *z* occur in 8 transactions. Therefore, when we specify the minimum support for frequent itemsets where $\alpha = 8$, {*xy*, *x*, *y*, *z*} is obtained as the set of frequent itemsets. The result is stored in a compact graph-based structure which can be used for further computations between ZDD set objects.

E. VIEW

VIEW component allow users to convey text-based information into visual representation. For instance, *mgv.rb* converts CSV based graph data into DOT plain text graph description language. The DOT graphs can be rendered in open source graph visualization software packages such as GraphViz and Gephi [13]. The commands *mpie.rb* and *mbar.rb* generate basic pie charts and bar graphs from text data to SVG graphics; these charts are commonly used for basic visual analysis of data. In addition, results of nodes and splitting rules from the data mining command *mbonsai* represented in Predictive Model Markup Language (PMML) format can be visualized as a decision tree in SVG format.

F. FUMI

FUMI enables users to process atypical information such as natural text language for sentiment and qualitative analysis. This package contains commands to perform morphological analysis and parsing of Japanese case frame with KNP and JUMAN [14]. Output can be saved in CSV file allows for further analysis and data processing.

IV. Development of NYSOL GGP System Architecture

A. GGP System Architecture

Big data demands enormous computational power for processing and generating knowledge. Parallel computing framework is often used for distributed data processing tasks to increase computational speed for big data. An experimental NYSOL GGP system architecture is constructed for use with NYSOL commands with custom-built NYSOL distributed file system (NDFS), designed for the analysis of big data organized in CSV tabular text format in a distributed processing architecture [15] (see Fig. 3).

Current popular big data frameworks include MapReduce programming implementation with Hadoop's Distributed File System [16], [17]. Queries based on key/value pairs are split and distributed across parallel nodes for filtering and sorting by map tasks in a parallel manner. This is followed by a reduce function that merges the immediate results. Apache Hive is an alternative data warehouse infrastructure built on top of Hadoop for data summarization, query, and analysis using HiveQL as SQL-based query language and converts queries in Hive to map/reduce [18].

TABLE II: CPU Processing Time (in Seconds) of Each Data Processing Task in Four Testing Environments

Tasks	Cut	Sum	Sort	Share	Rule
MCMD on single process	179.3	1553.4	1435.2	2797.5	709.8
Hadoop w/ Hive	436.6	316.0	2040.4	3229.2	1193.1
Hadoop w/ MCMD	494.2	270.0	2818.0	2866.2	1184.5
GPP w/ MCMD	137.1	69.5	598.9	2053.1	611.5

Experimental environment: Mac mini (CPU: Intel Core-i5 2.3GHz, memory: 16GB, OSX10.7). MCMD on single process shows the execution time by mcmd on a standalone machine (seconds). Results for other experiments are executed by parallel processing concurrently with five machines.
Input data for cut, sum, sort, share tasks includes 400 million lines at 10GB. Input data for rule task includes 40 million lines at 1GB.

Similar data summarization methods in MCMD is referred to as "key break process", which is the processing method of aggregation and joining common key fields with same value. The key field in key break process uniquely identifies individual rows or an entity in the data as the default field for sorting. In consideration of key break processing for distributed processing, the same key value is processed on the same processing unit the process is completed. This process is similar to the shuffle process in MapReduce. Key features of GGP architecture are as follows:

1) Implement automatic parallelization functionality to execute programs in distributed processing that are not written in multi-threaded code.
2) Use of NYSOL distributed file system (NDFS) to increase processing efficiency.
3) Use network pipe for streaming data between nodes.

The advantage of MCMD on GGP in comparison to Hadoop and Hive is the ease to optimize data structure and process data content for high-speed text processing, by leveraging the processing flexibility of the series of commands in MCMD. Nevertheless, the downside of GGP is that data must be structured in tabular format, and it does not use standard query language such as SQL. In addition, Hadoop stores data stream between nodes as temporary files, yet in GGP, even though the use of network pipe can reduce the overhead of file I/O to improve processing speed, the robustness of the GGP system is inferior to Hadoop for load balancing between nodes.

B. Computational Experiments

In order to illustrate the feasibility and effectiveness of GGP architecture, we simulated three types of environments including Hive with Hadoop, MCMD with Hadoop, and MCMD with NYSOL's GGP distributed processing against regular processing with MCMD on a standalone workstation as benchmark. Five types of data processing tasks (see Table I) are executed on these environments.

The comparison of equivalent processing program for Hive query language and MCMD, and an example of process flow of MCMD commands are shown in Fig. 4.

The experimental results in Table II show that the experimental MCMD with GGP framework outperforms Hive with Hadoop and MCMD with Hadoop in all five types of

data processing. MCMD with Hadoop performs better than Hive with Hadoop for calculation operations including sum, share, and rule processing. Given that both systems are tested on distributed processing structure, where the difference comes down to processing performance of mapper and reducer.

When comparing GGP against regular processing, dramatic efficiency close to 300% is achieved in sorting experiments, this is attributed to independent implementation of distributed processing in GGP, and independent execution of sort command. However, for more complex processing such as share and rule, improvements are reduced to ≈30% and ≈15% respectively. This is due to increased processing at each node which corresponds to increased network load. Further optimization is required to improve the efficiency of complex processes.

V. Conclusion

The NYSOL Project has expanded the ecosystem of KDD tools which balances the creation and management of information assets from big data. The user-centric nature of NYSOL tools empower users to create and customize programs, and to build a tailored information system for data management and analysis in a simplified and efficient mannner. The NYSOL package is available for public download distributed under the terms of GNU Affero General Public License[2] at http://www.nysol.jp/en/home. In addition, an experimental parallel processing GGP system that specializes in data table structure was developed, and the resulting benchmark achieved high processing efficiency. Moving forward, we aim to improve user interface for NYSOL tools, optimize processing efficiency with GGP architecture to enhance its robustness for extensive use in high performance research and commercial environment. We hope that The NYSOL Project could continue to benefit the community with innovative applied research and development in KDD, supported by educational initiatives.

Acknowledgment

This research and development work is supported by JST-ERATO Minato Discrete Structure Manipulation System Project based within Hokkaido University, Japan and JST CREST Project based within Kwansei Gakuin University, Japan. We would like to thank Shin-ichi Minato, Takeaki Uno, and Naoki Katoh for the joint development on machine learning algorithms used throughout the development of NYSOL commands.

References

[1] U. Fayyad, P. Piatetsky-Shapiro, and P. Smyth, "The KDD process of extracting useful knowledge from volumes of data," *Communications of the ACM*, vol. 39, no. 11, pp. 27-34, 1996.

[2] S. Madden, "From databases to big data," *IEEE Internet Computing*, vol. 16, no. 3, pp. 4-6, May-June 2012.

[3] Y. Hamuro, N. Katoh, Y. Matsuda, and K. Yada, "Mining pharmacy data helps make profits," *Data Mining and Knowledge Discovery*, vol. 2, no. 4, pp. 391-398, 1998.

[4] Y. Hamuro, N. Katoh, and K. Yada. "MUSASHI: Flexible & efficient data pre-processing tool for KDD based on XML," in *Proc. the First International Workshop on Data Cleaning and Preprocessing*, pp. 38-49, December 2002.

[5] S. Cheung, Y. Hamuro, H. Morita, and N. Katoh, "KGMOD: Powerful data processing and mining tool for KDD," *Asia Pacific Conference on Information Management (APCIM2009)*, March 2009.

[6] T. Azuma, K. Okada, and Y. Hamuro, "Is no news good news? The streaming news effect on investor behavior surrounding analyst stock revision announcement," *International Review of Finance*, vol. 14, no. 1, pp. 29-51, 2014.

[7] G. Karypis and V. Kumar, "Multilevel k way partitioning scheme for irregular graphs," *Journal of Paralleland Distributed Computing*, vol. 48, no. 1, pp. 96-129, 1998.

[8] S. Shimozono, A. Shinohara, T. Shinohara, S. Miyano, S. Kuhara, and S. Arikawa, "Knowledge acquisition from amino acid sequences by machine learning system BONSAI," *Transactions of Information Processing Society of Japan*, vol. 35, no. 10, pp. 2009-2018, 1994.

[9] J. R. Quinlan, *C4. 5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[10] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, CRC Press, 1984.

[11] T. Uno, T. Asai, Y. Uchida, and H. Arimura, "An efficient algorithm for enumerating closed patterns in transaction databases," *Discovery Science*, Springer Berlin Heidelberg, January 2004, pp. 16-31.

[12] S. I. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems," in *Proc. IEEE 30th Conference on Design Automation*, pp. 272-277, June 1993.

[13] DOT. (2014). [Online]. Available: http://en.wikipedia.org/wiki/DOT_(graph_description_language)

[14] D. Kawahara and S. Kurohashi, "A fully-lexicalized probabilistic model for Japanese syntactic and case structure analysis," in *Proc. the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, June 2006, pp. 176-183.

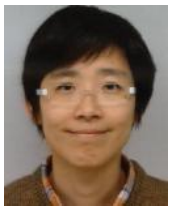
[15] Y. Hamuro and M. Nakamoto, "Development of distributed processing system for large-scale table structured data," presented at the 75th National Convention of IPSJ, March 2013.

[16] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107-113, 2008.

[17] Apache Hadoop. (2015). [Online]. Available: http://wiki.apache.org/hadoop

[18] Apache Hive TM. (2015). [Online]. Available: https://hive.apache.org

[2] Details of GNU Affero General Public License is available at http://www.gnu.org/licenses/agpl-3.0.html



Stephane Cheung obtained her master degree from Wasada University, Japan with a specialization in technology management and Nanyang Technological University, Singapore with a specialization in business administration in 2009. She worked at JST-ERATO Minato Discrete Structure Manipulation System Project at Hokkaido University from 2013 to 2015, and she joined Kwansei Gakuin Univesrity and participates in JST CREST Project since April 2015. Her research interests include micro-clustering, data mining, and business strategy.

Mazakazu Nakamoto graduated from the Department of Chemistry in Graduate School of Science, Osaka City University, Japan in 2000. After graduation, he joined Okayama Food Service Co., Ltd and was engaged in business systems research and development. He is working at JST-ERATO Minato Discrete Structure Manipulation System Project at Hokkaido University since 2010 to 2015. He joined Kwansei Gakuin Univesrity and participates in the JST CREST Project since April 2015. His area of specialization includes data mining analysis, programming, and information systems development for big data.

Yukinobu Hamuro is working as an associate professor of Institute of Business and Accounting, Kwansei Gakuin University, Japan. He served as a research advisor in JST-ERATO Minato Discrete Structure Manipulation System Project at Hokkaido University from 2011 to 2015. He developed the NYSOL Project and is currently anchoring the JST CREST Project since April 2015. He obtained his master's degree in management from Kobe University of Commerce. His research interests include data mining and development of data mining programs.