Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 17 (2013) 198 – 205

Procedia Computer Science

Information Technology and Quantitative Management (ITQM2013)

# Expanding database keyword search for database exploration

Binaya Balla, Zhengxin Chen

*College of Information Science and Technology, University of Nebraska at Omaha, Omaha, NE 68182,USA*

**Abstract**

Database keyword search (DB KWS) has received a lot of attention in database research community. Although much of the research has been motivated by improving performance, recent research has also paid increased attention to its role in database contents exploration or data mining. In this paper we explore aspects related to DB KWS in two steps: First, we expand DB KWS by incorporating ontologies to better capture users' intention. Furthermore, we examine how KWS or ontology-enriched KWS can offer useful hints for better understanding of the data and in-depth analysis of the data contents, or data mining.



Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management

*Keywords*: Database keyword search; ontology; query relaxation; data mining

## 1. Introduction

Recently keyword search on structured databases (particularly on relational databases, referred to as DB KWS below) has received a lot of attention in database community (e.g., [1,2,3,4,10,17]), and its potential contribution to in-depth exploration of database contents has also been addressed (e.g. [3,4]). Just like the flexible tube (with light) used in medical endoscope examination, DB KWS offers a unique opportunity to traverse various relational tables and lighten the hidden inside of databases relevant to the user-provided keywords, revealing a holistic view of database contents not available in the past, allowing researchers to see the hidden connections of data scattered throughout various tables in the database. Many efforts have been conducted in DB KWS for database exploration (as exemplified in studies cited in the ICDE tutorial [3,4]), but more systematical studies are still needed. In this paper, we report our on-going effort of conducting a concrete investigation on DB KWS for database exploration (or data mining) The unique feature of our approach is motivated from the following simple observation: In order to make DB KWS *effectively* aids data exploration, the list of keywords provided by users may need to be revised or expanded, because in many cases users may not know the exact keywords available in the database to suit their information needs; while in some other cases, the list of keywords may have to be expanded before DB KWS is to be conducted because the original list is too narrow. We also believe that ontologies can be used to achieve the goal of automatically rewriting of the keyword list. Motivated from these considerations, we have designed a research plan around our proposed prototype system and we are now in the process of implementing various components of this system and conducting experiments.

Our contribution of this paper can be summarized as follows:

1. *Ontology-enriched DB KWS*: We examine how ontologies can be used to revise or expand the user-specified keywords search to better satisfy users' information needs.



Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management

doi:10.1016/j.procs.2013.05.027

2. *KWS and ontology-enriched DB KWS for database contents exploration*: Relationships between retrieval and inference have always been an interesting philosophical issue, and is critical for understanding the nature of intelligence. By better capturing semantics through automatic query rewriting, ontology-enriched DB KWS opens door for better opportunities of exploring database contents and conducting data mining.
3. *A practical approach for implementation*: Rather than re-invent the wheel, we take advantage of open source software such as WordNet [15] (for incorporating ontologies) and WEKA [9] (for data mining) in implementing our approach.

## 2. Related topics

Instead of listing a number of related papers, we discuss sample papers of related topics.

### *2.1. Keyword search for structured databases*

The field of DB KWS has been researched for some time. The research is largely motivated by improving efficiency, such as constructing queries based on shortest paths, involving minimum number of joins and results relevance ranking. There are many research papers that present different algorithms to perform DB KWS. Some of them are DBXplorer, BANKS, BLINKS, DISCOVER etc. [1,2,3,4,10]. All of these tools involve similar process of input and output i.e. user provides input query keywords, the system retrieves the relevant tuples from the database using foreign key primary key relation ranks the result and outputs to the user. One common feature among all the systems is that they all retrieve tuples by performing joins and the result shall have all the keywords. In other words, they implement the "AND" semantics. For example, DBXplorer involves two steps; Publish and Search. The publish step consist of creating a symbol table that serves as an intermediary for the retrieval process. The creation of a symbol table is basically indexing and compressing the database for efficient retrieval. It uses schema graph. The search step involves looking up the symbol table, enumerating join trees to ensure all the keywords are present, rank and present result to the user. A different prototype system BANKS makes use of graph data structure for database keyword search. All the tuples in the relational database are represented as nodes of graph G. To compute the query and answer model, for each of the keywords in the query, a set of relevant nodes are located. The result set is a rooted directed tree that consist at least one node from the relevant node set. This tree may also contain nodes that are not present in the relevant node set but are required for the join. A special feature implements the Steiner tree. More recent studies include top-k keyword queries, as exemplified in [8,13].

All of the research have been purely focused on searching the keywords in the database whether it uses the schema graph and symbol table or uses graph to represent the tuples as nodes, and improving the ranking system by using effective information retrieval (IR) style ranking methods. Different from these studies, our research aims at *understanding* the query keywords. It shall relax the keyword terms by expanding it with the use of ontologies [11], which provides a common vocabulary in the domains we are interested. The keywords used in search will not be limited to what the user inputs. The references to ontologies will provide more meaningful words pertaining to the keyword within the domain such as "car" in the "automobile" domain.

### *2.2. Query relaxation using ontologies*

Traditionally query relaxation has been used to rewrite relational queries to deal with failed queries. In this process, ontologies can play a critical role. Query relaxation refers to automatic generalization of the given query to better suit user's information needs, particularly dealing with the *failing query* problem: given a query that returns an empty answer, how can one relax the query's constraints so that it returns a non-empty set of tuples? By generalizing a query to capture "neighbouring" information, query relaxation has been an effective means for fully exploit users' information needs from databases and information retrieval systems.

Similar to query optimization, query relaxation requires a kind of rewriting of the original query. However, although query optimization is aimed at improved system performance, query relaxation is aimed at better user

satisfaction for the contents returned; in particular, in case of empty result, the system will try to find an approximate match. Unlike query optimization which makes use of a set of syntactical transformation rules to rewrite original queries, query relaxation pays attention to semantics via background knowledge for query rewriting. Therefore, query relaxation becomes an interesting "showcase" for intelligent query answering. Numerous studies have been conducting in database query rewriting, including [5,14,16,18], as well as an earlier work from our own research group [11]. However, query (i.e., keyword list) rewriting has not been extensively studied in in the field of DB KWS, and is a core focus of our current research.

### *2.3. Semantic queries for database exploration*

As noted in [5,6] although most studies in DB KWS are around performance improvement, DB KWS is not only about performance. A recent study [7] which compared several representative KWS systems revealed some problems of KWS: even researchers have paid a lot of attention on performance, there is no existing scheme which is best for search effectiveness, and the sets of results retrieved by different systems are not highly correlated, which indicates that performance is not the sole factor that differentiates these systems.

While focusing on performance issues, recent studies on KWS have presented many new innovative ideas, some also started addressing issues which are related to the intrinsic nature of KWS. For example, two recent tutorials [3] examined a wide range of interesting aspects of keyword-based search and exploration on databases. We believe that there is a need for re-examining DB KWS from a broader perspective: If KWS is just aimed to get rid of SQL syntax requirement, then the computational price we have paid may be too high for simply achieving "user-friendliness." We believe that KWS search should go beyond what SQL (or its superset) can achieve. Therefore, there is a need for re-examining DB KWS from a broader perspective. In particular, since KWS relieves the burden of the SQL syntax for the users, it offers them a better opportunity of focusing the interconnected contents of the database. This also opens the door for users to take a peep of what the data is actually about, thus has the potential of aiding data mining (DM), which is intended to find hidden knowledge patterns from the data.

Although very different algorithms are needed in KWS and DM, technically they can be complementary to each other. For example, by conducting DB search in a more natural way, KWS may reveal unexpected associations among values of different attributes. Such kind of associations cannot replace standalone association rule mining, cluster analysis or other data mining tasks, but can offer useful suggestions for how data mining can be conducted. In addition, KWS usually requires construction of intermediate data structures such as graphs or trees, based on which database search can be conducted. It is possible that some of these data structures also serve as the basis for certain data mining tasks, such as frequent subgraph mining. The use of ontology can also tie KWS and data mining together.

## 3. Methodology overview

The process flow involves the several steps, as shown in Figure 1. After user input, a "normal" DB KWS proceeds, with an option for keyword expansion. In case the option is taken, ontology extended DB KWS takes place. In addition, there is another option of DM with the result

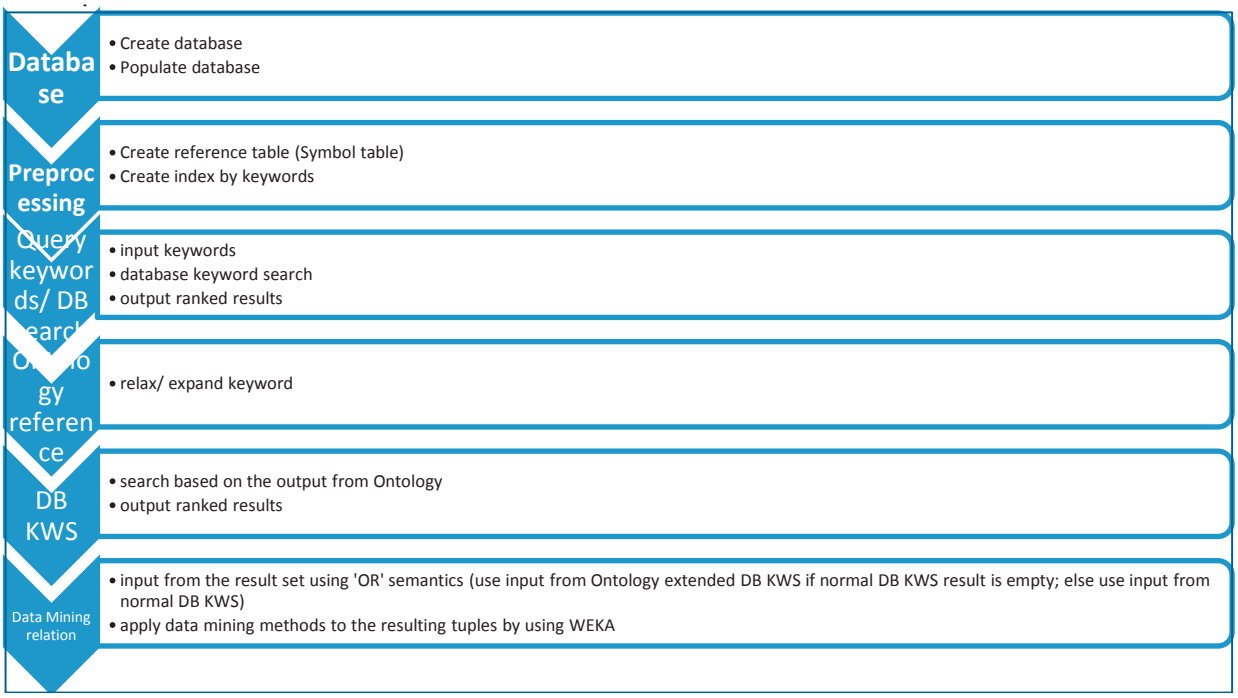


Fig. 1. Process overview

## 4. Implementation

We use Eclipse IDE and Java language for this project. The application takes single/multiple keywords as input and outputs relevant tuples from the database. The keywords shall be used to search the symbol table for all possible matches. SQL statement will be generated for all possible matches to retrieve tuples from all tables in the database. There are two sets of results as follows:

1. For DB KWS (4.1), only the tuples that consist all the keywords, either from a single table or joined by foreign to primary key relationship from multiple tables are ranked and output to the user.
2. For KWS for database exploration (4.2), the application retrieves all relevant tuples for each input keyword and does not limit to retrieving tuples that consist all keywords. In other words, it relaxes the enumeration of join trees, which basically ensures the 'and' semantic is applied.

### *4.1. Keyword extension and query relaxation using WordNet*

For database: We use XAMPP to create MySQL database. XAMPP is a free and open source cross-platform web server solution stack package. It is to install and consist of Apache HTTP Server, MySQL database, SQLite, PHP and Perl programming languages.

For WordNet Ontology: WordNet Ontology is used to expand the query keywords using Java APIs that are available for use on the Internet. There are two APIs available: JAWS and JWNL. (http://lyle.smu.edu/~tspell/jaws/index.html#using and http://sourceforge.net/projects/jwordnet/)

JAWS can be used as follows:
1) Obtain a copy of the WordNet database files, which can be accomplished by downloading and installing WordNet (you must download the full version of WordNet and not just the database files).

2) Download the Java Archive (JAR) file containing the compiled JAWS code (available at http://lyle.smu.edu/~tspell/jaws/index.html#using).
3) When starting your application you must:
a) Include in your Java Virtual Machine's class path the JAR file you downloaded.
b) Use the wordnet.database.dir system property to specify where the WordNet database files are located.

JWNL can be used as follows:
1) Call JWNL.initialize() somewhere in the initialization code of your program.
2) Call Dictionary.getInstance() to get the currently installed dictionary. The only dictionary methods you should really ever need to call are lookupIndexWord(), lookupAllIndexWords(), and getIndexWordIterator().
3) The other methods that may be useful are Relationship.findRelationships() and PointerUtils. Relationship.findRelationships() allows you to find relationships of a given type between two words (such as ancestry). Another way of thinking of a relationship is as a path from the source synset to the target synset. The methods in PointerUtils allow you to find chains of pointers of a given type. For example, calling PointerUtils.getHypernymTree() on the synset that contains "dog," returns a tree with all its parent synsets ("canine"), and its parents' parents ("carnivore"), etc., all the way to the root synset ("entity").
4) JWNL provides support for accessing the WordNet database through three structures - the standard file distribution, a database, or an in-memory map. Utilities are provided to convert from the file structure to an SQL database or in-memory map, and a configuration file controls which system the library uses.

For DB KWS: We closely follow DBXplorer [1] for implementing the DB KWS. The KWS can be broken down into a Publish step and a Search step.

The Publish step prepares the database for KWS by identifying the database along with the set of tables and columns within the database to be published. A reference 'Symbol' table is created that is used during search time to identify the locations of query keywords in the database such as the tables, columns and rows they occur in.

The Search process can be further broken down into the following steps:
a) The symbol table is looked up to identify the tables, and columns/rows of the database that contain the query keywords.

b) Once all the potential subsets of tables are identified. They are joined only if they are connected in the schema graph. By enumerating all the joins, it eliminates the subsets that do not contain all the keywords in the query.

c) SQL statement is created for each join tree that is the result of the pervious step. The final rows are ranked and presented to the user.

### *4.2. Keyword search for database exploration*

For database: We use XAMPP to create MySQL database. XAMPP is a free and open source cross-platform web server solution stack package. It is to install and consist of Apache HTTP Server, MySQL database, SQLite, PHP and Perl programming languages.

For input dataset for DM: We relax the requirement (b) i.e. enumerate join trees (in section 4.1 in 'for DB KWS'). Doing so, the number of intermediate results of the subset of tables shall be much bigger. In other words, the results satisfy the "OR" semantics. More result is better for this project because the main focus is to incorporate data mining in KWS. For all the subset of tables, SQL statements are created to retrieve data from the database. The retrieved data is further analyzed using data mining techniques.

For applying DM techniques:
The result set is input into WEKA, a well-known open-source data mining tool [9]. The input file is preprocessed to identify numeric and nominal attributes because they are important to establish data mining connection. WEKA uses numeric and nominal attributes to discover hidden patterns by applying data mining techniques. It is a collection of machine learning algorithms for data mining tasks. The algorithms can be applied directly to a dataset. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. WEKA was able to successfully produce clusters and association rules based on my input files.

## 5. Experiments

We have used user_restaurant_review database in the experiments. It consists of 9 tables such as ratings, user profile, restaurant info, hours and so on. The database schema is shown in Fig.2.

### *5.1. Keyword extension and query relaxation using Wordnet*

The KWS for DB exploration/ data mining could be a behind the scene process which shall work in conjunction to the normal KWS process. The users may be notified if they wish to see result for data exploration based on their keywords. This may be a similar function to the keyword expansion using Ontology in the sense that, if any keyword retrieves 0 results, the users may be prompted if they wish to relax the keyword by referring the keyword to the Ontology. WordNet Ontology is used to expand the query keywords using Java APIs that are available for use on the Internet.

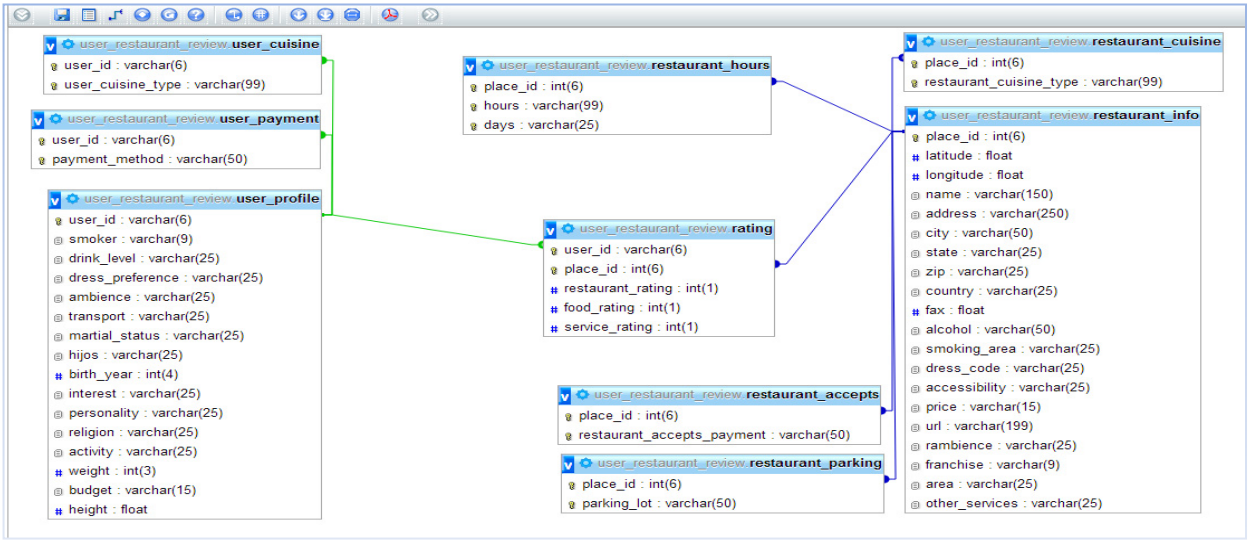


Fig. 2 Restaurant DB schema

As an example, suppose the user enters keyword ‘evaluation’ for DB KWS, the search would yield empty. It shall ask the users if they would like to refer to WorldNet, if yes, it shall get the expanded keyword ‘rating’ from Ontology reference and search the database for ‘rating’. The results shall be displayed to the users. Keyword expanded referring to the WordNet, new keyword: ‘rating’ The DB KWS for key ‘rating’ retrieves tuples from ‘restaurant_rating’ table. More specifically, it finds three keyword matches for columns ‘restaurant_rating’, ‘food_rating’ and ‘service_rating’ in the table. This is a special case where the keyword is the column name in one of the table in the database. In such cases, the users may be prompted to enter additional keywords. The user shall be able to take advantage of the expanded keyword. Consequently, the user may enter more specific keyword for the search such as ‘3 stars’, ‘restaurant rating 4 star’ or ‘food rating 5 star’ etc.

### *5.2. Keyword search for database exploration.*

Using WEKA, a number of data mining rules can be obtained. As an example, a classification model can be built by using the decision tree algorithm to predict the class attribute ‘marital status’. This time it shows >86% correctly classified instances. A decision tree is shown in Fig. 3.

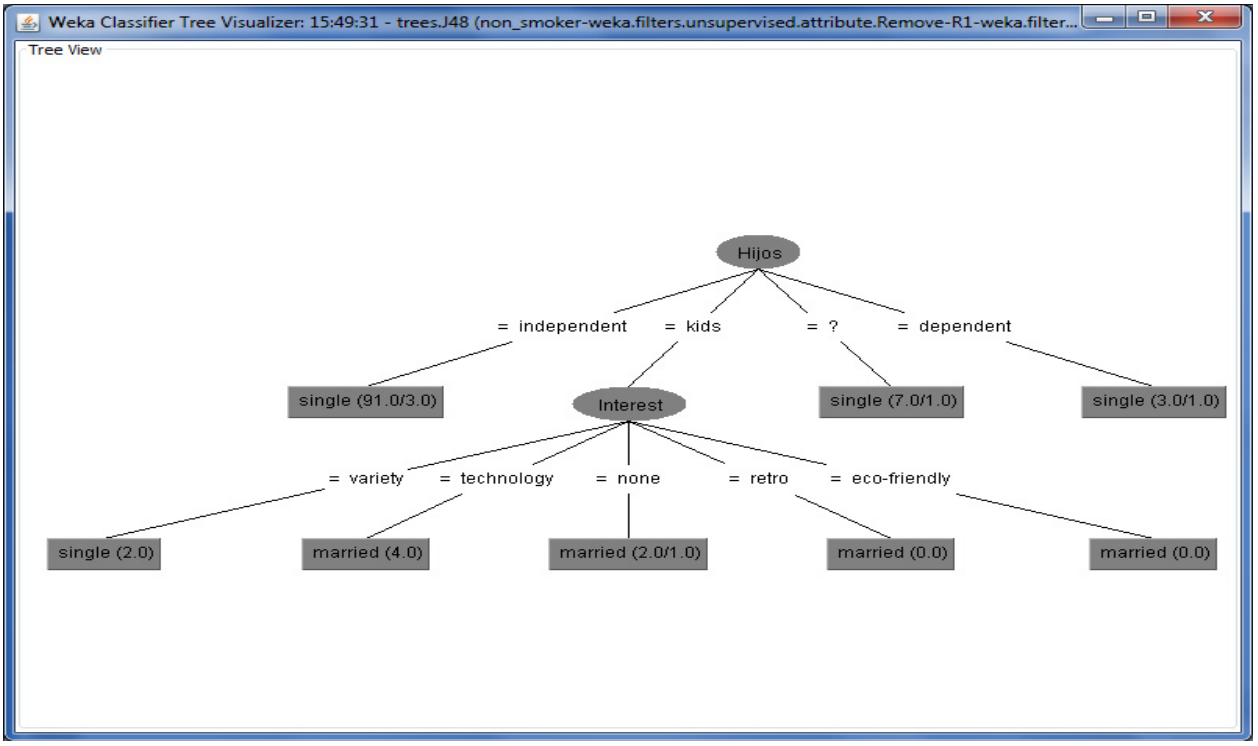


Fig. 3, A decision tree produced by WEKA

. Some classification rules based on the model are shown below:

1. Independent non-smokers are more likely to be single.
2. Non-smokers with kids and that are interested in technology are more likely to be married.

Next, we explore Aprioi algorithm to generate association rules in WEKA. Fig. 4 depicts the result.

The top-most rule states:
Hijos = independent Activity= student 78 ==> Marital Status= single 77 conf:(0.99)
The above rule states that 'students' with 'independent' hijos are 'single'. This rule has a 99% confidence.
(Hijos in Spanish translates to offspring in English)

## 6. Conclusion

Continuing and expanding recent research of keyword search on structured databases, in this paper we have explored aspects related to DB KWS in two steps: First, we expand DB KWS by incorporating ontologies to better capture users' intention. Furthermore, we have examined how KWS or ontology-enriched KWS can offer useful hints for better understanding of the data and in-depth analysis of the data contents, or data mining. We have identified key elements in a prototype system, and are completing our implementation.

## References

1. S. Agrawal, S. Chaudhuri, G. Das (2002). DBXplorer: A System for Keyword-Based Search over Relational Databases. International Conference on Data Engineering (ICDE 02).
2. Bhalotia, G.; Hulgeri, A.; Nakhe, C.; Chakrabarti, S.; Sudarshan, S.; , Keyword searching and browsing in databases using BANKS,

*Proceedings. 18th International Conference on Data Engineering (ICDE),* 431-440, 2002.

3. Y. Chen, W. Wang, Z. Liu and X. Lin, Keyword search on structured and semi-structured data (tutorial), *Proc. SIGMOD 2009*.
4. Y. Chen, W. Wang and Z. L, Keyword-based search and exploration on databases (tutorial), *ICDE 2011*.
5. Z. Chen, J. Torson and S. Servisetti, Reexamining Database Keyword Search: Beyond Performance, Proc. ISIICT 2011.
6. Z. Chen, Database Keyword Search: Beyond Its Current Landscape, Int. J. ITDM 2012.
7. J. Coffman and A. C. Weaver, A framework for evaluating database keyword search strategies, *Proc. CIKM'10.*
8. J. Feng, G. Li, J. Wang; , Finding Top-k Answers in Keyword Search over Relational Databases Using Tuple nits, *IEEE Transactions on Knowledge and Data Engineering,*, 23(12), 1781-1794, Dec. 2011.
9. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA Data Mining Software: An Update; *SIGKDD Explorations*, 11(1), 2009.
10. H. He , H. Wang , J. Yang , P. S. Yu, BLINKS: ranked keyword searches on graphs, *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007.
11. J. Hill, J. Torson, B. Guo and Z. Chen, Toward Ontology-guided Knowledge-driven XML Query Relaxation, *Proc. IEEE CIMsim* 2010.
12. V. Hristidis , Y. Papakonstantinou, Discover: keyword search in relational databases, *Proceedings of the 28th International conference on Very Large Data Bases (VLDB)*, 670-681, 2002.
13. Y. Luo; W. Wang; X. Lin; X. Zhou; J. Wang; Kequi Li; , SPARK2: Top-k Keyword Query in Relational Databases, *IEEE Transactions on Knowledge and Data Engineering,*, 23(12), 1763-1780, Dec. 2011.
14. J. Mei, L. Ma and Y. Pan, Ontology Query Answering on Databases. In: *Proc. Int'l Semantic Web Conf.*, 445-458 (2006)
15. George A. Miller (1995). WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41.
16. I. Muslea, Machine learning for online query relaxation, *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining,* 246 – 255 (2004)
17. L. Qin, J. X. Yu. and L. Chang, Keyword search in databases: The power of RDBMS. *Proc. SIGMOD '09*, 681-693.
18. X. Zhou, J. Gaugaz, W.-T. Balke and W. Neidl, Query relaxation using malleable schemas. In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data,* 545 – 556 (2007)

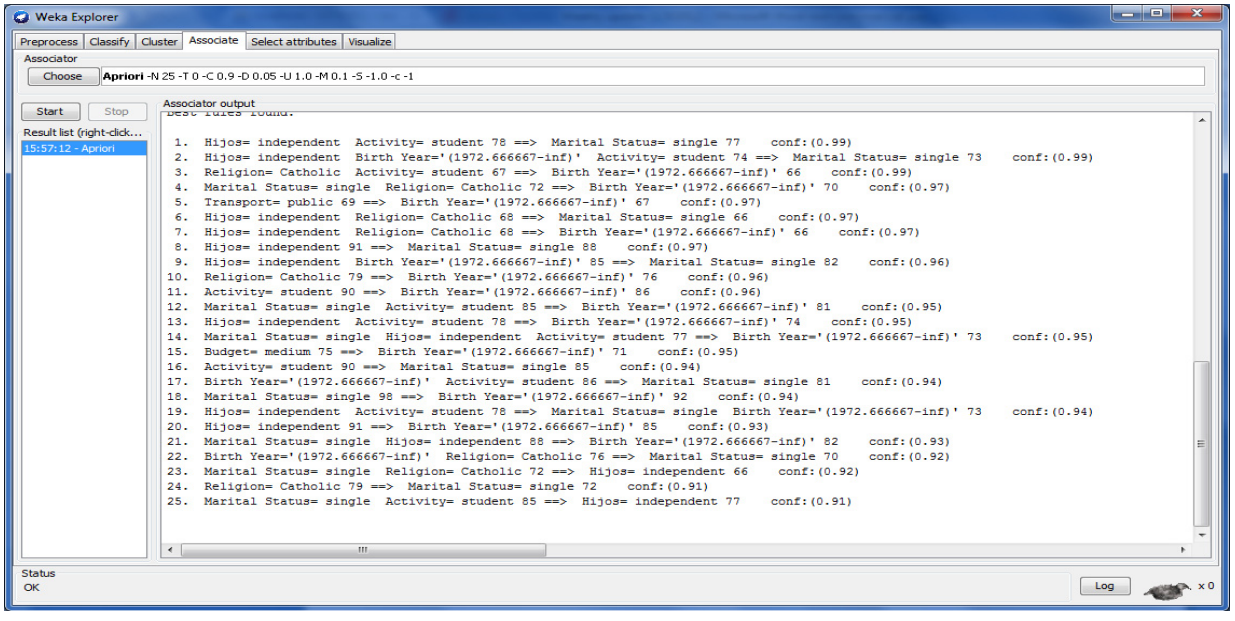


Fig. 4. WEKA result of association rule mining