SWApriori: A New Approach to Mining Association Rules from Semantic Web Data

Reza Ramezani, Electrical & Computer Engineering, Isfahan University of Technology, Iran
r.ramezani@ec.iut.ac.ir
Mohamad Saraee, School of Computing, Science and Engineering, University of Salford, Manchester, UK
m.saraee@salford.ac.uk
Mohammad Ali Nematbakhsh, Department of Computer Engineering, University of Isfahan, Iran
nematbakhsh@eng.ui.ac.ir

ABSTRACT

With the introduction and standardization of the semantic web as the third generation of the Web, this technology has attracted and received more human attention than ever and thus the amount of semantic web data is constantly growing. These semantic web data are a rich source of useful knowledge for feeding data mining techniques. Semantic web data have some complexities, such as the heterogeneous structure of the data, the lack of exactly-defined transactions, the existence of typed relationships between entities etc. One of the data mining techniques is association rule mining, the goal of which is to find interesting rules based on frequent item-sets. In this paper we propose a novel method that considers the complex nature of semantic web data and, without end-user involvement and any data conversion to traditional forms, mines association rules directly from semantic web datasets at the instance level. This method assumes that data have been stored in triple format (*Subject*, *Predicate*, and *Object*) in a single dataset. For evaluation purposes the proposed method has been applied to a drugs dataset that experiments results show the ability of the proposed algorithm in mining ARs from semantic web data without end-user involvement.

Keywords: Semantic Web, Association Rules, Semantic Web Mining, Data Mining, Information Retrieval, SWApriori

1. INTRODUCTION

Since the advent of RDF[1], RDFS[2] and OWL[3] standardization, people have a better understanding of the semantic web and thus the amount of semantic web data is constantly growing. This semantic web data contains information about people, geography, medicine and drugs, ontologies etc. With increasing data publishing from different sources, there is now a large amount of semantic web data.

Extending the scope of data mining research from traditional data to semantic web data allows us to discover and mine richer and more useful knowledge [1, 2]. The first reason is that the provision of ontological metadata by the semantic web improves the effectiveness of data mining. The second reason is that in traditional datasets, the data mining algorithms work with undefined data, such as structured and limited-feature datasets (RDB mining), unstructured and textual data (text mining) and unstructured web data (web mining) where entities do not have exact definitions. In contrast, in the semantic web, data are provided with

[1] http://www.w3.org/TR/rdf-concepts/
[2] http://www.w3.org/TR/rdf-schema/
[3] http://www.w3.org/TR/owl-features/

a well-defined ontology, and entities and the relationships between data are meaningful as well.

Association rule mining (ARM), as a major branch of data mining techniques, tries to find frequent itemsets and generate interesting association rules (ARs) based on these frequent itemsets. ARM techniques that have been introduced so far deal with traditional data in tabular format or graph-based structure.

In this paper we investigate the problem of association rule mining and semantic web data challenges and propose a new approach to mining ARs directly from semantic web data. This approach considers the complex nature of semantic web data as opposed to than traditional data and also, in contrast with existing methods, eliminates the need of data conversion and end-user involvement in the mining process.

In trying to apply ARM to semantic web data, we are faced with some problems and differences compared with traditional data, as follows:

- **Heterogeneous data structure:** traditional data mining algorithms work with homogeneous datasets in which instances are stored in a well-ordered sequence and each instance has predefined attributes. But in semantic web data, data are heterogeneous. This means that specific category/domain instances (such as people, cars, drugs and etc.) based on one ontology or multiple ontologies may have different attributes.
- **No exact definition of transactions:** in conventional information systems, data are stored in databases using predetermined structures, and by using these structures it's possible to recognize transactions and thus extract them from the dataset. Then traditional association rule mining algorithms work on these transactions [3]. For example in a market basket system, transactions are made of products that are purchased together and these products will have same TID as transaction identifier. In contrast, in the semantic web, different publishers may register different features for an instance at different times, and so an instance might perhaps have an attribute that another instance of the same type does not possess. Thus transactions have no exact definition in the semantic web.
- **Multiple relations between entities:** traditional ARM algorithms, in order to generate large itemsets[4], consider only entities' values and suppose there is only one type relation between entities (for example *bought together*). But in semantic web data, there are multiple relations between entities. In fact *predicates* are relations between two entities or between one entity and one value. These different relations must be considered in the ARM process [4].

Our proposed algorithm tries to solve the above problems. For dealing with a **heterogeneous data structure**, this algorithm uses a linked list based data structure. To solve the problem of **no exact definition of transactions**, the algorithm uses a new approach in ARM that without need to any transaction launces to generate *L-Large Itemsets* ($L \geq 2$) and finally for dealing with **multiple relations between entities**, in the proposed algorithm each *Item* is considered as an *Entity* and one *Relation*. Also in contrast to the existing methods of mining ARs from semantic web data, the proposed algorithm eliminates the need of end-user involvement in the mining process.

It is assumed data are stored in a dataset in triple structure and the provided dataset is a complete semantic web dataset, a subset of a complete semantic web dataset or a

[4] An itemset is a non-empty subset of items

concatenation of multiple semantic web datasets. In the paper the data structures used and proposed algorithms are described and discussed in details.

The rest of the paper is organized as follows. Section 2 introduces a number of related work. Section 3 briefly describes the concepts of association rules and the semantic web. Section 4 contains the general methodology and foundations of the proposed method and by using an example, clearly describes algorithm steps. Section 5 presents the proposed algorithm pseudo code. Section 6 gives the experimental evaluation and results and finally Section 7 concludes the paper and offers suggestions for future work.

2. RELATED WORK

In the past many machine-learning algorithms have been successfully applied to traditional datasets in order to discover useful and previously unknown knowledge. Although these machine learning algorithms are useful, the nature of semantic web data is quite different from traditional data. The majority of previous semantic web data mining work focus on clustering and classification [5-7]. Some of these work are based on inductive logic programming or ILP [8] which uses ontology encoded logics.

The ARM problem as first introduced [9, 10] has the aim of finding frequent itemsets and generating rules based on these frequent itemsets. Many ARM algorithms have been proposed which deal with traditional datasets [11-13]. These algorithms are classified into two main categories: Apriori based [14, 15] and FP-Tree based [16-18]. These algorithms usually work with discretized values, but in [19] an evolutionary algorithm was introduced for mining quantitative ARs from huge databases without any need to data discretization.

As will be seen, semantic web dataset contents are convertible to graph. Other related approaches in ARM are the use of frequent sub-graph and frequent sub-tree techniques for pattern discovery from graph structured data [16, 17, 20]. The logic behind these algorithms is to generate a tree/graph based on existing transactions and then mine the generated tree/graph. Although these methods are interesting, they are not appropriate for our work, because in semantic web data there is no exact definition of transactions, and also after converting dataset contents to graph, each vertex of the graph, independent of its incoming link, is not replicated in the whole graph more than once. On the other hand graph vertices are unique and thus discovering sub-graph/sub-tree redundancy is not possible.

Not all graph-based approaches are based on sub-graph techniques. In [21] an algorithm has been introduced that inputs data into a graph structure and then by a novel approach without the use of sub-graph redundancy, mines ARs from these data. This work is not useful for our problem because the algorithm finds only maximal frequent itemsets instead of all frequent itemsets and also, like other traditional ARM algorithms, this algorithm works only with well-defined transactions.

All the above work deals with traditional data. In [22] an algorithm has been introduced that by using a mining pattern which the end-user provides, mines ARs from semantic web data. This algorithm uses dynamic and graph-based structure data that must be converted to well-structured and homogeneous datasets so that traditional ARM methods can use them. To convert data, users must state the target concept of analysis and their involved features by a mining pattern following an extended SPARQL syntax. This work, similarly to other related approaches in mining ARs from semantic web data, requires that the end-user be familiar with ontology and dataset structure.

One of the recent work on semantic data mining is LiDDM [23]. LiDDM is a piece of software which is able to do data mining (clustering, classification and association rules) on linked data [24]. The working process behind LiDDM is as follows. First, the software acquires required semantic web data from different datasets by using user-defined SPARQL queries. Then it combines the results and converts them to traditional data in tabular format. At the next level, some pre-processing will be done on these data and finally traditional data mining algorithms is applied. The main limitation of LiDDM is that the end-user must be involved in the entire mining process and he/she has to be aware of ontologies and datasets structure and, based on this awareness, guides the mining process step by step.

RapidMiner semweb plugin [25] is a similar approach to LiDDM which applies data mining techniques on semantic web data or linked data. In addition to basic operations of data mining, the authors proposed methods for reformatting set-valued data, such as converting multiple values of a feature into a simple nominal feature to decrease the number of generated features and thus the approach scales well. As with LiDDM, in the RapidMiner semweb plugin the end-user has to define a suitable SPARQL query for retrieving interested data from linked data datasets.

In RDF structure, each data statement names a triple and is identified with three values: *subject*, *predicate* and *object*. In order to generate transactions, it is possible to use one of these three values to group transactions (transaction identifier) and use one of the remaining values as transaction items. Six different combinations of these values along with their usage are shown in Table 1 [26]. For example, grouping triples by predicate and using objects for generating transactions has usage in clustering. This approach eliminates one part of triples parts and doesn't consider it in mining process that isn't interested.

Table 1 - Combinations of triple parts

	Context	**Target**	**Use Case**
1	Subject	Predicate	Schema discovery
2	Subject	Object	Basket analysis
3	Predicate	Subject	Clustering
4	Predicate	Object	Range discovery
5	Object	Subject	Topical clustering
6	Object	Predicate	Schema matching

SPARQL-ML [27] is another approach to mining semantic web data that provides special statement as an extension to SPARQL query language to create and learn a model for specific concept of retrieved data. It applies classification and regression techniques to data, but other data mining techniques such as clustering and ARM are not covered by this approach. Another limitation is that this technique is applicable only on those datasets for which the SPARQL endpoints support SPARQL-ML, which is currently not very widespread. Our proposed algorithm can deal with all kinds of datasets and ontologies.

3. PRELIMINARIES

This section briefly describes *Association Rules* and *Semantic Web* concepts which are related to our research area.

3.1. Association Rules

Frequent itemset mining and association rule induction are powerful methods for so-called market basket analysis, which aims at finding regularities in the co-occurred items, such as sold products or prescript biomedical drugs. The problem of mining association rules was first introduced in 1993 [9].

Let we denote each item with I_i, thus $I = \{I_1, I_2, \dots, I_m\}$ is set of all items which sometimes called the *item base*. Each transaction T_i is a subset of I and based on transactions we define database as collection of transactions denoted by $D = \{T_1, T_2, \dots, T_n\}$. Each itemset ($S$) is a non-empty subset of I and an association rule (R) is a rule in the form of $X \rightarrow Y$ which both X and Y are itemsets. This rule means that if in a transaction the itemset X occurs, with certain probability the itemset Y will appears in the same transaction too. We call this probability as confidence and call X as rule antecedent and Y as rule consequent.

- **Support of an itemset**

The absolute support of the itemset S is the number of transactions in D that contain S. Likewise, the relative support of S is the fraction (or percentage) of the transactions in D which contain S.

More formally, let S be an itemset and U the collection of all transactions in D that contain all items in S. Then

$$Sup_{abs}(S) = |U|$$

$$Sup_{rel}(S) = (|U|/|D|) * 100\%$$

For brevity we call $Sup_{rel}(S)$ as $Sup(S)$.

- **Confidence of an Association Rule**

The confidence of an association rule $R = X \rightarrow Y$ is the support of the set of all items that appear in the rule divided by the support of the antecedent of the rule. That is,

$$Conf(R) = \big(Sup(\{X \cup Y\})/Sup(X)\big) * 100\%$$

Rules are reported as strong association rules if their confidence reaches or exceeds a given lower limit (minimum confidence, to be specified by a user). In this paper, we call this association rules as strong association rules.

- **Support of an Association Rule**

As mentioned in [9, 14], the support of the rule is the (absolute or relative) number of cases in which the rule is correct. For example in the association rule $R: A, B \rightarrow C$, the support of R is equal to support of $\{A, B, C\}$.

- **Frequent Itemsets**

Itemsets with greater Support than a certain threshold, so-called minimum support are frequent itemsets. The goal of frequent itemset mining is to find all frequent itemsets.

- **Maximal Itemsets**

A frequent itemset is called maximal if no superset is frequent, that is, has a support exceeding the minimum support.

3.2. Semantic Web

The Semantic Web (or Web of Data), sometimes called the third generation of the Web, emerges in distinction to the traditional web of documents. The goal of the Semantic Web is to standardize web page formats so that the data becomes machine readable. This data is described by ontologies. A well-known definition by T.R.Gruber in 1995 is "*An ontology is an explicit specification of a conceptualization*" [28]. The main purpose of the semantic web

is to be machine readable so this feature needs to make entities meaningful and also describe entities by standard methods.

In order to describe entities, some means of entity representation and entity storing are needed. There are several methods for representing and storing semantic web data. The first method is RDF[5] which is based on XML structure. XML is a powerful standard and also is flexible for transmitting structured data. In fact, the RDF documents are descriptions of semantic web data so this data becomes machine readable. Each RDF statement is a triple and each triple consists of three parts: *subject*, *predicate* and *object*. Subjects and predicates are resources that are identified by URI. Objects can be resources and shown by URI or can be constant values (literals) and represented as strings. In each triple, one relation or typed link exists between either two resources or between one resource and one literal. A similar concept to the URL is the IRI, which has been introduced to represent non-Latin text items in order to internationalize DBPedia [29].

RDFS is an extension of RDF which allows to define entities over classes, subclasses and properties. Hence it's possible to apply some inference rules on these RDFS structure entities.

Due to RDF and RDFS limitations the OWL[6] has been introduced which has more powers of deduction. OWL, which is based on DAML[7] and OIL [30], is the most well-known language that applies description logic to the semantic web data. The first version of this language has three versions, *OWL Lite*, *OWL DL* and *OWL Full*, which differ in expressive ability and deductive power. This language also allows transitive, symmetric, functional and cardinality relations between entities.

These three OWL flavors (Lite/DL/Full) are a bit old-fashioned. New profiles have been designed as OWL2 [31]. OWL 2 profiles are defined by placing restrictions on the structure of OWL 2 ontologies. Syntactic restrictions can be specified by modifying the grammar of the functional-style syntax and possibly giving additional global restrictions. OWL 2 has three subsets (EL, QL and RL). *OWL 2 EL* is particularly useful in applications employing ontologies that contain very large numbers of properties and/or classes and has polynomial time reasoning complexity with respect to the size of the ontology. *OWL 2 QL* is aimed at applications that use very large volumes of instance data, and where query answering is the most important reasoning task. This profile is designed to enable easier access and query to data stored in databases. *OWL 2 RL* is aimed at applications that require scalable reasoning without sacrificing too much expressive power. It is designed to accommodate OWL 2 applications that can trade the full expressivity of the language for efficiency, as well as RDF(S) applications that need some added expressivity.

As with traditional databases, which in order to retrieve information, need an endpoint language (SQL), semantic web datasets need such a language too. For this purpose, the SPARQL[8] [32, 33] language has been introduced which is able to extract information and knowledge from semantic web datasets. DBPedia [34] is an example of a semantic web dataset. SPARQL can be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware. SPARQL has capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions.

---

[5] Resource Description Framework

[6] Ontology Web Language

[7] http://www.daml.org/

[8] http:// www.w3.org/TR/rdf-sparql-query/

SPARQL also supports extensible queries based on RDF graphs. The results of SPARQL queries can be presented as result sets or RDF graphs.

4. MOTIVATION AND METHODOLOGY

In previous sections the importance of mining ARs from semantic web data was expressed and also some related work and preliminary concepts were illustrated. In this section we use an example to present a detailed view of our method along with the definitions that sustain it step by step. Finally the next section describes the data structures used and the proposed algorithms in detail.

4.1. Problems

In mining ARs from semantic web data, we face a number of issues as follows.

1- **Transactions**: In semantic web datasets, particularly generalized datasets such as DBPedia, unlike traditional data there is no exact definition of transactions. This means that if we verify existing data, we cannot determine how these data has been generated and also stored based on what model, what order and what process. This means we cannot regenerate transactions from existing data.

Let us take an example. Consider that ARs are based on frequent items. Frequent items are those items that have a **being together** relation to each other. For example in a market store, those goods that a customer buys in a single purchase at any time, construct a transaction. In Table 2 each transaction shows items that have been bought together.

Table 2 – Example of some together bought goods

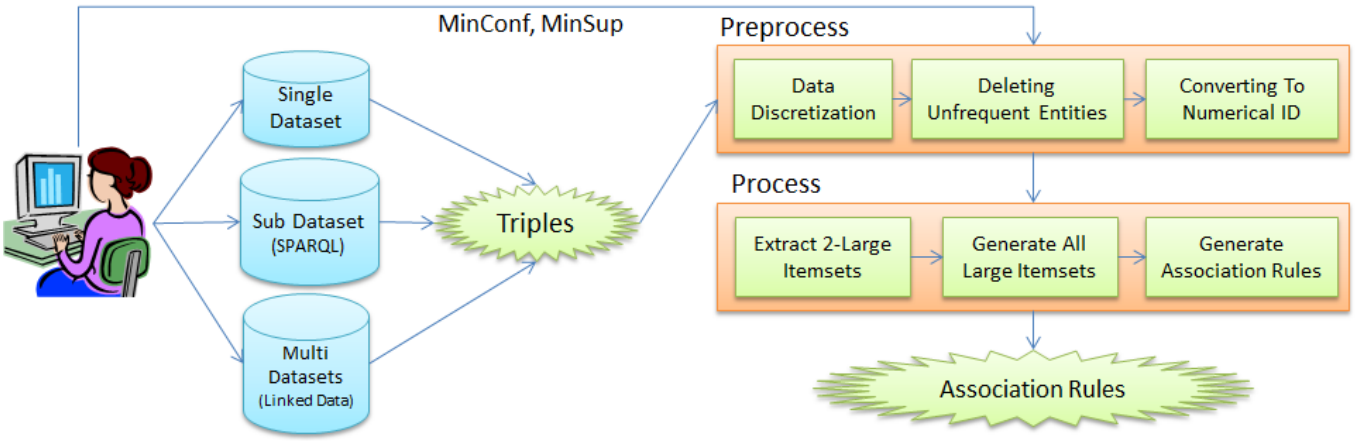
Transaction ID	Items Bought
100	Shirt
200	Jacket, Hiking Boots
300	Ski Pants, Hiking Boots
400	Shoes
500	Shoes, Shirt
600	Jacket

In contrast in semantic web data, there are many relations (not one relation: *being together* relation) between items and these relations hold for different individuals at different times. Thus constructing transactions from this data is difficult and also vague, because the meaning of transactions is not clear, unless the end-user defines this meaning, as was done in [22, 23].

There are two possible solutions to this problem. The first requires proposing a new concept of transaction in semantic web datasets by involving the end-user, and the second is to propose a new algorithm that doesn't deal with transactions within the ARM process. Our suggested algorithm is based on the second approach.

2- **Relations**: The concept that is new in semantic web data and does not exist in traditional data, is that of relationships or typed links between entities. Traditional ARM algorithms like Apriori do not consider these relationships in their mining process. Our proposed algorithm considers entity relationships when launches to generate large itemsets. In this algorithm an *Item* not only is an *Entity* but also consists of *Entity* + *Relationship*.

Figure 1 – Mining Process Workflow



3- **User Involvement**: In many existing semantic web data mining methods, in order to generate transactions the end-user must be aware of dataset and ontology structure [22, 23]. Here we have developed an algorithm that does not involve the end-user with the structure of dataset and ontology while mining process. Although the proposed method has no need for user involvement, if the end-user wants to, he/she can apply advanced ontology concepts and also restrict the mining process by manipulating input data (for example by using SPARQL).

4- **Heterogeneous Data**: In semantic web datasets, data is heterogeneous. This means that in one dataset you may observe two entities of the same type but with completely different attributes and vice versa. For example you may see two countries of which the first one only has *Population* and *NearBy* attributes and the second one only has *Capital* and *Language* attributes. The proposed algorithm uses special data structures that in addition to considering different relations between items, can deal with heterogeneous data. In fact as you will see later, the proposed algorithm looks at the heterogeneous data as a special graph.

The proposed algorithm in this paper solves these problems.

4.2. Outline of the proposed algorithm

The working logic of the proposed algorithm is similar to Apriori algorithm, in that both algorithms try to generate large itemsets and finally generate ARs based on these large itemsets. In contrast to Apriori algorithm, our proposed algorithm performs unsupervised mining of ARs from semantic web data directly, without end-user involvement and also without using transactions. As was mentioned earlier, in semantic web data there is no exact definition of transactions; thus the proposed algorithm has to be tuned in such a way that it doesn't need transactions. For this purpose, after receiving semantic web data in triple format, the proposed algorithm begins to generate **2-large itemsets** from input data at the instance level without the use of any transaction (in fact there is no transaction to be used) and then feeds the generated 2-large itemsets to the main algorithm. Afterwards, the algorithm generates **larger itemsets** based on these 2-large itemsets. These large itemsets are different than traditional large itemsets, in that each itemset's items consist of two parts: *Entity* and *Relation*, where Entity is an object and Relation is a predicate. Finally the **association rules** is generated from the large itemsets.

Figure 1 shows the workflow of the proposed mining process.

4.3. Example

Let us look at an example in order to illustrate the proposed algorithm behavior. For this example, some facts from our real world have been collected and converted to semantic web data. Then the proposed algorithm tries to mine ARs directly from this data. The data scope is from the educational system of *"Isfahan University of Technology"* and *"University of Isfahan"*.

In Table 3 you can see the dataset contents in triple format along with entities description at the end of table. Figure 2 also depicts Table 3 contents in a graph. Figure 10 shows Figure 2 in different way.

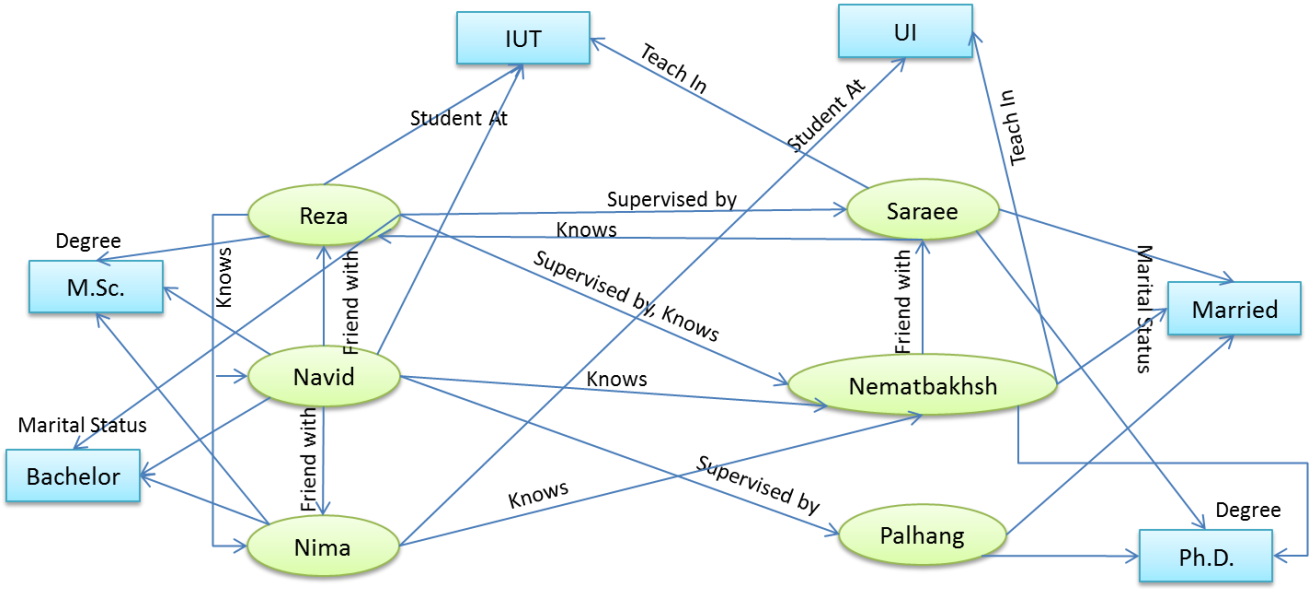
In order to simplify the example, some triples have been eliminated from the graph and also only a few of the relations between entities have been shown. Also only people have been used as subjects.

Table 3 - Input dataset contents (Example)

Subject	**Predicate**	**Object**
Reza	Supervised by	Saraee
Reza	Supervised by	Nematbakhsh
Reza	Marital Status	Bachelor
Reza	Student at	IUT
Reza	Knows	Nematbakhsh
Reza	Knows	Nima
Reza	Knows	Navid
Reza	Degree	M.Sc.
Navid	Supervised by	Palhang
Navid	Marital Status	Bachelor
Navid	Student at	IUT
Navid	Degree	M.Sc.
Navid	Knows	Nematbakhsh
Navid	Friend with	Reza
Navid	Friend with	Nima
Nima	Supervised by	Mirzaee
Nima	Marital Status	Bachelor
Nima	Student at	UI
Nima	Friend with	Reza
Nima	Knows	Nematbakhsh
Nima	Degree	M.Sc.
Ayoub	Supervised by	Saraee
Ayoub	Marital Status	Married
Ayoub	Student at	IUT
Ayoub	Degree	Ph.D.
Saraee	Marital Status	Married
Saraee	Teach in	IUT
Saraee	Knows	Reza
Saraee	Knows	Ayoub
Saraee	Degree	Ph.D.
Nematbakhsh	Friend with	Saraee
Nematbakhsh	Marital Status	Married

Nematbakhsh	Teach in	UI
Nematbakhsh	Knows	Reza
Nematbakhsh	Degree	Ph.D.
Palhang	Teach in	IUT
Palhang	Marital Status	Married
Palhang	Degree	Ph.D.
Entities: **Reza**, **Navid**, **Nima** and **Ayoub** are students (Type: Person) **Saraee**, **Nematbakhsh**, **Palhang** and **Mirzaee** are teachers (Type: Person) **IUT** (Isfahan University of Technology) and **UI** (University of Isfahan) are University. **M.Sc.** and **Ph.D.** are educational degree. *Relations:* All predicates are self-descripting		

Figure 2 - contents of Table 3 in graph



4.4. 2-Large Itemset

In the proposed algorithm, after preprocessing data and weaving ontology concepts into data elicitation, the first step of mining ARs from semantic web data is to generate 2-large itemsets, namely two entities that co-occurred abundantly. To identify these entities, the algorithm traverses all objects in the triples, combines large objects two by two and finally generates all possible object sets that have length of 2. Large objects are those objects that are appeared in more than *MinSup* triples.

In this example, *"Saraee"*, *"Nematbakhsh"* and *"IUT"* are large objects, because they have been appeared in many triples. By these three objects, candidate object sets with length of 2 are:

- *{Saraee, Nematbakhsh}*
- *{Saraee, IUT}*
- *{Nematbakhsh, IUT}*

Afterward, the algorithm verifies that the two entities of each set (as objects) based on two relations among their incoming relations (as predicates) have been referenced by sufficiently many entities (as subject). This process is repeated for all combinations of the incoming relations (predicates) of these two entities (objects). If the references count (the count of subjects that refer to both entities with both relations) is equal to or greater than predefined *MinSup* value, these two entities (objects) along with these two relations make a 2-large itemset.

Consider entities and relations presented in Figure 10. In this figure, *Nematbakhsh* is an object and *Knows* is one of its incoming relations. A similar situation exists for *IUT* as object and *Student at* as predicate. (*Knows* and *Student at* are incoming relations of *Nematbakhsh* and *IUT* respectively). Now suppose the algorithm compares (*Nematbakhsh + Knows*) with (*IUT + Student at*). As the Figure 2 and Figure 10 show, *Reza, Navid* and *Nima* refers to *Nematbakhsh* by *Knows* relation and *Reza, Navid* and *Ayoub* refer to *IUT* by *Student at* relation. Intersecting from (*Reza, Navid, Nima*) and (*Reza, Navid, Ayoub*) returns (*Reza, Navid*) as result. Thus if 2 (the count of intersection result) is equal to or greater than *MinSup* value, {(*Nematbakhsh + Knows*), (*IUT + Student at*)} is identified as a 2-large itemset. This 2-large itemset means those students that satisfy *Student at Isfahan University of Technology*, and also *Knows* Dr. *Nematbakhsh*. Based on this logic, in this example {(*Nematbakhsh + Knows*), (*M.Sc. + Degree*)} are identified as a 2-large itemset too.

- {(*Nematbakhsh + Knows*), (*IUT + Student at*)}
- {(*Nematbakhsh + Knows*), (*M.Sc. + Degree*)}

As another example (*Saraee + Supervised by*) and (*IUT + Student at*) is a candidate for 2-large itemset. Because the first one has been referenced by "*Reza, Ayoub*" and the second one has been referenced by "*Reza, Navid, Ayoub*". Intersecting of "*Reza, Ayoub*" and "*Reza, Navid, Ayoub*", returns "*Reza, Ayoub*" as a result that has 2 members. As in the previous example, if 2 (the count of intersection result) is equal to or greater than *MinSup* value, {(*Saraee + Supervised by*), (*IUT + Student at*)} is identified as a 2-large itemset that means for many of the persons that are student in *IUT*, their supervisor is Dr.*Saraee*.

Itemsets can have common entities. Based on this definition, an entity like *"Paper1"* can lie in both items of a 2-large itemset as entity so that the first item has the *"Write"* relation and the second one has the *"Cite"* relation. On the other hand {(*Paper1 + Write*), (*Paper1 + Cite*)} can be a 2-large itemset.

Finally after making all 2-large itemsets, the algorithm begins to generate larger itemsets.

4.5. Larger Itemsets

The Apriori algorithm to generate a (L+1)-candidate itemset, combines two L-large itemsets with L-1 first equal items and makes a candidate set with length of L+1. A candidate set is large when its occurrence becomes equal to or greater than predefined *MinSup* value and also all of its subsets are large itemsets too. Our proposed algorithm combines those two L-large itemsets if their L-1 first items have equal entities value and equal relations value respectively. Namely in generating large itemsets, the proposed algorithm considers that each entity has been referenced via what relation (predicate).

In the above example, the combination of both generated 2-large itemsets can make an itemset with length three, because the first item of them are equivalent and are equal to (*Nematbakhsh + Knows*). As the result **{**(*Nematbakhsh + Knows*), (*IUT + Student at*), (*M.Sc. + Degree*)**}** is a candidate itemset with length three. Suppose that all subsets of this 3-large

itemset are large. If the number of subjects that refer to these three objects via corresponding relation is equal to or greater than *MinSup*, these items will appear as a 3-large itemset.

- **{**(*Nematbakhsh* + *Knows*), (*IUT* + *Student at*), (*M.Sc.* + *Degree*)**}**

Generating larger itemsets will continue until making new candidate itemsets are not possible.

4.6. Association Rules

Finally the algorithm begins to generate ARs based on these large itemsets. As you saw, the generated large itemsets hold only the values of objects and predicates and the values of subjects that refer to objects via predicates are discarded. Here only the number of subjects is important, not the value of them, exactly like what happens in Apriori algorithm, because subjects value are similar to customer names that in ARM are not important. Thus the generated rules contains only objects and subjects. The algorithm also generates rules with only one item in the consequent part. The logic behind this is that usually the generated rules count is enormous, thus with only one item in the consequent part the generated rule count is reduced. Additionally, by generating complex rules (rules with multiple items in the consequent part) it is too hard to use the generated rules in the real world applications. Finally the rules with equal or greater confidence than *MinConf* value are identified as strong rules.

In the above example **{**(*Nematbakhsh* + *Knows*), (*IUT* + *Student at*), (*M.Sc.* + *Degree*)**}** is a large itemset and the following are instances of generated rules from this large itemset. Bold words are relations (predicates) and italic words are entities (objects).

- **Student at** (*IUT*), **Knows**(*Nematbakhsh*) ➔ **Degree** (*M.Sc.*)
- **Knows**(*Nematbakhsh*), **Degree** (*M.Sc.*) ➔ **Student at** (*IUT*)
- **Student at** (*IUT*), **Degree** (*M.Sc.*) ➔ **Knows**(*Nematbakhsh*)

The first of the above rules means that for many of the *IUT* students that know *Nematbakhsh*, with a certain probability (rule confidence) their education degree is *M.Sc.*

These rules will be identified as strong rules if their *Confidence* becomes equal to or greater than *MinConf* value.

4.7. Ontology Usage

In the previous section, we provided an outline of the proposed algorithm and its steps. Here we pay attention to the question: *"What is the role of ontology in the proposed algorithm?"*

At first glance, it seems the proposed algorithm works barely at the instance level never considers semantic level, since it receives a semantic web dataset and directly mines ARs from the provided dataset. So when did the algorithm deal with ontology?

Ontologies have two aspects. Firstly they define the structure of data over classes and properties and secondly they define logic and relations between data. Since data obey data structures defined by ontologies, the proposed algorithm will be deal with the data's ontology implicitly. Also ontologies will appear as prefix for subjects, predicates and objects at the instance level so triple parts become distinct.

At the simplest level, the end-user does not need to be familiar with ontology and dataset structure and he/she only has to provide a desired dataset as algorithm input. But, if the end-user wishes, he/she can explicitly involve ontologies in the mining process at three phases.

- *Data providing phase*: at this phase, the end-user by using suitable SPARQL command, can provide more special data by considering ontology concepts. Smarter data will be lead to more interesting results. For example the end-user can determine that only subjects with special type (rdf:type) or other special features, attend in the mining process.
- *Preprocess phase*: at this phase, the end-user by using class relations such as *rdfs:subClassOf*, *rdfs:subPropertyOf*, *owl:equivalentClass*, *owl:equivalentProperty* and *owl:sameAs* can convert entities to each other, so results become more generalized. Also by using attributes such as *rdfs:Datatype*, *rdfs:range*, *rdfs:domain*, *owl:allValuesFrom* and *owl:someValuesFrom*, data discretization can be done in a smarter way. For example unit conversion can be done by using ontology concepts.
- *Postprocess phase*: by using ontology concepts, some meaningless results that are not compatible can be eliminated from the results set.

5. ALGORITHMS & DATA STRUCTURES

In this section we describe the data structures used and the proposed algorithms in detail. As was mentioned earlier, *subject*, *predicate* and *object* are parts of a triple. Each *entity* is a subject or an object. Here *Relation* is the same *Predicate* and also *Frequent Itemset* is the same *Large Itemset*.

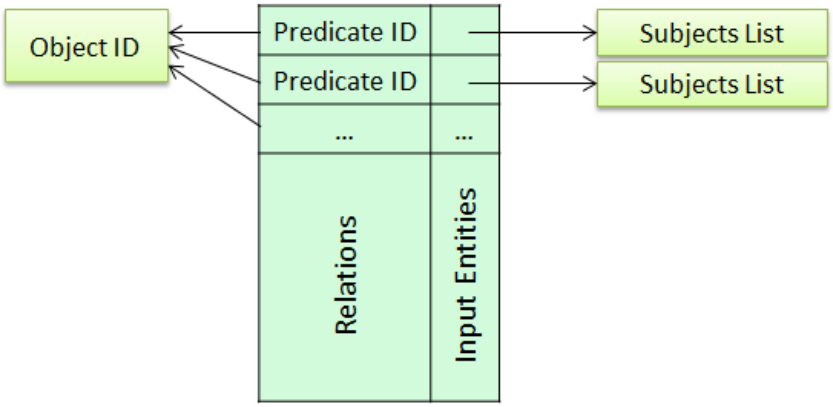
5.1. Data Structures

The algorithm input is a set of triples (*subject*, *predicate* and *object*). For the purpose of storing data in main memory, the simplest and the most efficient way is to use a cuboid (3D array) as data structure, in such a way that the first dimension stores source (subject), the second stores destination (object) and the third stores relation (predicate) between source and destination. For example in Figure 2 *"Reza"* is a source (subject), *"IUT"* is a destination (object) and *"Student at"* is a relation (predicate) between *"Reza"* and *"IUT"*. Each cuboid entry value is 0 or 1. If the $(i,j,k)_{th}$ entry value is equal to 1, this means there is a relation with k type from i_{th} entity (as subject) to j_{th} entity (as object). Although a cuboid structure is very fast and easy to use it requires a large amount of memory space. An alternative is to use a linked list data structure. To store each object scheme (predicates and subjects that are connected to the object), there is an **ObjectInfo** class with these attributes:

1- ***Object ID***: Object identifier
2- A ***Linked List*** that its entries have two parts:
   a. ***Predicate ID***: Predicate identifier
   b. ***Subjects List***: pointer to a list that contains subjects which refer to this O*bject ID* with this *Predicate ID*.

The **ObjectInfo** image has been depicted in Figure 3.

Figure 3 - ObjectInfo structure



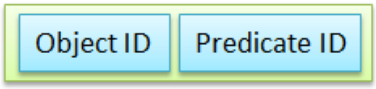
With this data structure policy, triples are in fact grouped based on objects, because for each object, the algorithm defines an *ObjectInfo* instance and then specifies that based on each predicate, what other subjects refer to this object. The purpose of this grouping is to increase the mining process speed based on the proposed algorithm.

Finally there is a list that has entries equal to the objects count. Each entry of this list refers to one of the *ObjectInfo* instances. Figure 10 shows the *ObjectInfo* data structure state after reading example dataset of Table 3. In this figure, for the reason of limited space, some entities such as Ph.D., UI, Mirzaei, and Palhang have been eliminated from the objects section.

As was mentioned earlier, this algorithm, in addition to entity values, considers relations between entities in the ARM process. Thus here each *Item* not only is equal to an entity but also each *Item* consists of an *Entity (Object)* and a *Relation (Predicate)* that is connected to that object. To store each *Item* there is an **Item** class that has *ObjectID* and *PredicateID* attributes.

Figure 4 shows the image of class *Item*.

Figure 4 - Item Structure

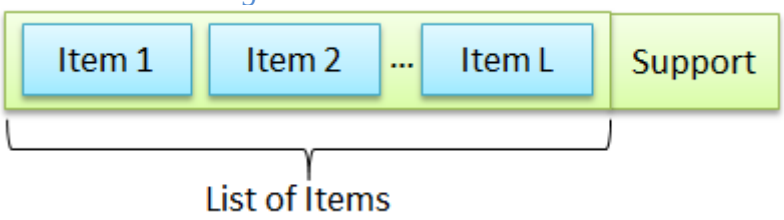


Generating ARs is based on large itemsets. Each itemset is non-empty set of *Items*. In order to storing generated (candidate/large) itemsets, there is an **Itemset** class that contains these attributes:

1- List of Items: that holds *L* items ($L \geq 2$).
2- Support: number of subjects that refer to all *Items* via correspond predicates. The Itemset is large if *Support* is equal to or greater than *MinSup* value.

Figure 5 shows the image of class *Itemset*.

Figure 5 - Itemset Structure

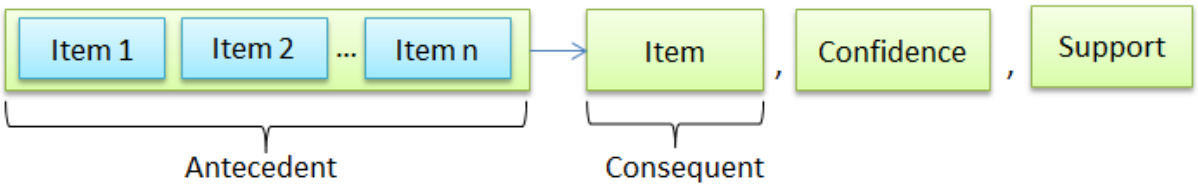


In section 4.5, it was said that ARs are constructed from *Items* and each rule has only one *Item* in the consequent part. To store generated ARs, there is a **Rule** class that contains these attributes:

1- List of Items as ***Antecedent***
2- An Item as ***Consequent***
3- Rule ***Confidence***
4- Rule ***Support***

In Figure 6 you can observe the **Rule** class image.

Figure 6 - Rule Structure



5.2. Algorithms

The proposed algorithm name is **SWApriori**. The algorithm workflow is as follows. After traversing triples, discretizing data and eliminating triples with less frequent subject, predicate or object, all triples parts (subjects, predicates and objects) must be converted to numerical IDs. This conversion is done to increase the mining process speed, because this algorithm focuses on comparing entities and relations and clearly comparing two numbers is faster than comparing two literals. After converting data into numerical values, the *Generate2LargeItemsets* algorithm is called by *SWApriori* algorithm and generates 2-large itemsets and feeds them to the main algorithm. Then the *SWApriori* algorithm launches to make larger itemsets. Finally the *GenerateRules* algorithm generates ARs based on these large itemsets.

These algorithms are as follows:

Algorithm1 (*SWApriori*) is the main algorithm that after calling *Generate2LargeItemsets* and generating 2-large itemsets, launches to generate L-large itemsets ($L \geq 3$) and finally calls *GenerateRules* to generate ARs. The pseudocode of this algorithm is shown in Figure 7.

Figure 7 – SWApriori: Mining association rules from semantic web data directly

1. **Algorithm 1. Mining association rules from semantic web data**
2. **SWApriori(DS, MinSup, MinConf)**
3. **Input:**
4. *DS*: Dataset that consists of triples (Subject, Predicate, and Object)
5. *MinSup*: Minimum support
6. *MinConf*: Minimum confidence
7. **Output:**
8. *AllFIs*: Large itemsets
9. *Rules*: Association rules
10. **Variables:**
11. *FIs*[9], *Candidates*: List of Itemsets
12. *IS*[10], IS_1, IS_2, IS_3: Itemset (multiple items)

[9] FIs = Frequent Itemsets
[10] IS = Itemset

```
13.  ObjectInfoList: List of ObjectInfo
14. Begin
15.  Traverse triples and discretize objects
16.  Delete triples which their subject, predicate or object has frequency less than MinSup value
17.  Convert input dataset's data to numerical values
18.  Store converted data into ObjectInfo instances
19.  ObjectInfoList = ObjectInfo instances
20.  FIs = AllFIs = Generate2LargeItemsets(ObjectInfoList, MinSup)
21.  L = 1
22.  Do
23.    L = L + 1
24.    Candidates = null;
25.    For each IS1, IS2 in FIs
26.      If IS1[1..L-1].ObjectID    = IS2[1..L-1].ObjectID      and
27.         IS1[1..L-1].PredicateID = IS2[1..L-1].PredicateID Then
28.         IS3 = CombineAndSort(IS1,IS2)
29.         Candidates = Candidates ∪ IS3
30.      End If
31.    End For
32.    FIs = null;
33.    For each IS in Candidates
34.      If Support(IS) ≥ MinSup AND all subsets of IS are large Then
35.        FIs = FIs ∪ IS
36.      End If
37.    End For
38.    AllFIs = AllFIs ∪ FIs
39.  While (FIs.Lenght > 0)
40.  Rules = GenerateRules(AllFIs, MinConf)
41.  Return AllFIs, Rules
42. End
```

Let us explain the **SWApriori** algorithm in detail. This algorithm accepts a dataset that contains triples along with minimum support (*MinSup*) and minimum confidence (*MinConf*) values as input parameters. The preprocess step is done in lines 15 to 19. In line 20 all 2-large itemsets are generated by calling **Generate2LargeItemsets** algorithm. The loop between lines 22 to 39 generates all large itemsets and will continue until generating larger itemsets is no longer possible. In each iteration of this loop, all large itemsets with length of L are verified and new candidate itemsets with length of L+1 are generated. Each loop iteration (lines 25-31), uses previous loop iteration results which have been stored in *FIs*. Line 25 states that all large itemsets with length of L must be compared two by two, and this comparison is done in lines 26 and 27. If two large itemsets with length of L are combinable (their L-1 first items are equal) they will be combined by the **CombineAndSort** function and will generate a new candidate itemset with length of L+1. The items of this new candidate itemset are sorted by *Object ID* and then by *Relation ID*. In line 29 the new candidate itemset is added to the candidate itemsets collection. After generating all candidate itemsets with length of L+1, in lines 33 to 35 all large itemsets are selected from the candidate itemsets collection and then added to the large itemsets collection (*FIs*). Finally line 37 adds generated large itemsets with length of L+1 to the collection of all frequent itemsets (*AllFIs*). After generating all possible large and frequent itemsets, the ARs are generated by calling **GenerateRules** in line 40.

Calculating the exact time complexity of **SWApriori** algorithm is not easy, because as the number of *L* increases, the number of generated frequent itemsets first is increased and then is

decreased. In the worst case **SWApriori** is in the order of $O(I^2L^3)$, if I is the number of large itemsets and L is the length of the largest itemset.

Algorithm2 (*Generate2LargeItemsets*) is called by **SWApriori** and by traversing all *ObjectInfo* instances generates all possible object sets that have length two. Finally if many subjects by two arbitrary predicates refer to both objects of the generated object set, the object set along with these two predicates are identified as a 2-large itemset. The pseudocode of this algorithm is shown in Figure 8.

Figure 8 - Generate2LargeItemsets: Generating 2-Large itemsets from ObjectInfo instances

```
1. Algorithm 2. Generating 2-Large itemsets from ObjectInfo instances
2. Generate2LargeItemsets(ObjectInfoList, MinSup)
3. Input:
4.    ObjectInfoList: List of ObjectInfo instances
5.    MinSup: Minimum support value
6. Output:
7.    LIS: List of Itemsets with two in length
8. Variables:
9.    Ob1, Ob2: ObjectInfo
10.   SS1[11], SS2: Subject Set //subjects that refer to an object via special predicate
11.   R1[12], R2: Value corresponds to RelationID //refers to predicates
12. Begin
13.   For each Ob1, Ob2 in ObjectInfoList
14.     For each R1 in Ob1.Relations
15.       For each R2 in Ob2.Relations
16.         SS1 = R1.SubjectsList
17.         SS2 = R2.SubjectsList
18.         IntersectionCount = IntersectCount(SS1, SS2)
19.         If IntersectionCount ≥ MinSup Then
20.           LIS = LIS ∪{(Ob1.ObjectID + R1), (Ob2.ObjectID + R2)}
21.         End If
22.       End For
23.     End For
24.   End For
25.   Return LIS
26. End
```

This algorithm accepts all *ObjectInfo* instances and minimum support value as input parameters. *ObjectInfo* instances store objects information as it was shown in Figure 3. Each *ObjectInfo* instance is related to an object and reveals what subjects by what predicates refer to the object. This algorithm generates all possible 2-large itemsets. In line 13 all *ObjectInfo* instances are traversed and compared two by two. In lines 14 and 15 all input relations (*Relation* attribute of *ObjectInfo* class) of these two instances are traversed and compared two by two. In line 16 the list of all subjects that refer to object Ob_1 by predicate R_1 is extracted from $Ob_1.R_1.SubjectsList$ and then added to SS_1 list. This operation will be repeated for Ob_2 and R_2 and the result is added to SS_2 in line 17. In line 18 an intersection is taken from SS_1 and SS_2. This intersection reveals what subjects refer to both objects by both predicates. If the intersection count is equal to or greater than *MinSup* value, both objects along with their

[11] SS = Subject Set
[12] R = Relation

corresponding predicates generate a 2-large itemset. This algorithm finishes when all objects, for each their incoming predicates, are compared to each other.

The complexity of **Generate2LargeItemsets** is in the order of $O(B^2R^2S)$, if B is the number of large entities (large *ObjectInfo* instances), R is the maximum number of relations of *ObjectInfo*s and S is the maximum number of subjects concerned to an *ObjectInfo* (S is the required time for intersecting by using hash set)

Algorithm3 (*GenerateRules*) traverses all generated large itemsets and proceeds to generate candidate rules with one item in the consequence part. If the candidate rule confidence is equal to or greater than *MinConf* value, the rule is identified as strong rule. The pseudocode of this algorithm is shown in Figure 9.

Figure 9 – GenerateRules: Generating association rules based on large itemsets

```
1. Algorithm 3. Generating association rules based on large itemsets
2. GenerateRules(AllFIs, MinConf)
3. Input:
4.    AllFIs: All large itemsets
5.    MinConf: Minimum confidence
6. Output:
7.    Rules: Association rules
8. Variables:
9.    IS[13]: Itemset
10.   Itm: Item
11.   Consequent: Item that is appeared in rule consequent part
12.   Antecedent: List of Items that are appeared in rule antecedent part
13. Begin
14.    For each IS in AllFIs
15.       For each Itm in IS
16.          Consequent = Itm
17.          Antecedent = IS – Consequent
18.          Confidence = Support(IS) ÷ Support(Antecedent)
19.          If Confidence ≥ MinConf Then
20.             Rules = Rules ∪ (Antecedent, Consequent)
21.          End If
22.       End For
23.    End For
24.    Return Rules
25. End
```

This algorithm accepts frequent and large itemsets and a minimum confidence value as input parameters. In line 14, the large itemsets are selected one by one. In line 15 all *Items* of the selected large itemset are traversed. Line 16 and 17 construct a rule body based on the selected large itemset and selected item, and then line 18 calculates the confidence of this new rule. Line 19 verifies the rule confidence. If the confidence value is equal to or greater than *MinSup* value, that is this rule is a strong rule and then it is added to the strong rules collection in line 20. Notice that the algorithm in line 16 selects only one *Item* as consequent part.

The complexity of **GenerateRules** is in the order of $O(IL)$, if I is the number of all large itemsets and L is the length of the largest itemset.

[13] IS = Itemset

6. EXPERIMENTAL RESULTS

In order to evaluate usefulness of the **SWApriori** algorithm and to prove its ability to mine ARs from semantic web data directly and without the end-user involvement, some experiments on *Drugbank* dataset have been made that show the proposed method is able to make 2-large itemsets and larger itemsets without regard to transactions and finally generates ARs based on these large itemsets. This method does not involve the end-user in the mining process in the sense that he/she does not need to be familiar with the ontology and dataset structure.

6.1. Dataset

In order to test the proposed algorithm *Drugbank* dataset was used which is a detailed database on small molecules and biotech drugs. Each drug entry ("DrugCard") has extensive information on properties, structure, and biology (what the drug does in the body). Each drug can have 1 or more targets, enzymes, transporters, and carriers associated [35].

The Drugbank dataset has heterogonous semantic annotations and contains 772,299 different triples; from these triples, 249,967 distinct entities (subject and object) and 110 distinct relations are extractable. In this dataset each subject has 34 relations on average.

6.2. Experimental set-up

To extract 2-large itemsets, the input data must be converted to the algorithm standard format. This conversion is done automatically by the algorithm so that all subjects, predicates and objects are converted to equivalent numerical IDs. On the other hand, each triple is expressed by *SubjectID*, *PredicateID* and *ObjectID*.

The input data may be a complete dataset or a subset of a complete dataset. The input dataset can also be a concatenation of multiple datasets that has been made using SPARQL language and linked data standards [24]. That is if the end-user wants, he/she can select a subset of the entire dataset by a SPARQL query and then feed this sub dataset to the algorithm or can concatenate multiple datasets and then feed this super dataset to the algorithm.

Finally after generating large itemsets and strong rules, in order to interpret the results, the numerical IDs is converted to the equivalent text values. In semantic web datasets, because there is no exact definition of transactions, the end-user or the expert himself/herself has to interpret the generated rules and use them in real world applications.

6.3. Previous Work

Since there are fundamental differences between SWApriori and previous work and hence comparing generated results may not show the advantages of methods over each other, in this subsection SWApriori and its generated results is structurally compared with [22] and [26]. Some results obtained by applying SWApriori on *Drugbank* dataset will be presented in next subsection.

SWApriori employs itemsets which have many immediate common subjects to generate larger itemsets. Immediate subject means an object concerned to a subject, both should be located at one triple. In contrast, the proposed algorithm in [22] employs objects that are directly or indirectly connected to a common subject to generate transactions. This means there is one or more edges between subject and the employed object in the input graph. Hence SWApriori could not generate all ARs that the proposed algorithm in [22] could. In addition since the end user in [22] by knowing the structure of the dataset and ontology

determine which objects should be used to generate transaction, the generated ARs are specified to special objects, but in contrast since the generated ARs by SWApriori are general and encompass all relations and objects, a filtration, such as [36], should be done on the generate rules to extract interested and useful ARs.

As it was mentioned earlier, the proposed method in [26] uses only two parts of triples to generate transactions and hence it loses a lot of information. In addition since one part of triples is used as TID and this TID is employed by Apriori just for identifying transactions items, the generated rules are ambiguous, because no information about TIDs is presented in the generated rules. SWApriori can generate all ARs generateable by [26] when objects are used as *Target* (rows #2, #4 in Table 1).

The ARs generated by SWApriori contain one item and one of its concerned relations. But in contrast the ARs generated by [22] and [26] contains only one item and they suppose there is "being together" relation among items.

6.4. Results

In this subsection, the acquired results will be described. The proposed method in this paper is a new approach to mining ARs from semantic web data that in contrary to other existing methods [22, 23] does not require that the end-user be familiar with the structure of dataset and ontology and also does not convert semantic web data to traditional tabular data and does not use traditional ARM algorithms. The results obtained show the effects of applying **SWApriori** algorithm with different *MinSup* values on *Drugbank* dataset. The obtained rules prove this new approach is able to mine ARs from semantic web data directly without the need for transactions and end-user involvement.

In the following you can see some results of mining ARs from the *Drugbank* dataset. In these results, the *MinConf* value is 0.7 and the *MinSup* values range is between 0.02 and 0.33. In the provided *Drugbank* dataset, *MinSup* values less than 0.02 would cause to generate a huge amount of ARs which need a great time to be processed and *MinSup* values more than 0.33 would not generate any ARs.

Table 4 shows some extracted rules along with their confidence and support values that the proposed algorithm has discovered from *Drugbank* dataset. In each rule, the first sentence identifies a predicate (relation) and the inter parentheses word identifies an object. These extracted rules prove the ability of the proposed algorithm in mining ARs from semantic web data directly and without the end-user involvement and also any need for transactions. For example the 3rd rule in Table 4 indicates that %81 of drugs that has goal to catalytic activity, their effect process is physiological.

Figure 11 to 17 show the algorithm behavior and its effects on *Drugbank* dataset from different aspects. In these figures, the X-axis denotes *MinSup* values.

For different *MinSup* values, Figure 11 shows the number of covered objects as large entity, i.e. how many *ObjectInfo* instances have been known as large and frequent entities. This number has an B^2 effect on the time complexity of the **Generate2LargeItemsets** algorithm. In this Figure, objects are considered regardless to their incoming relations.

Figure 12 shows the covered 2-large itemsets count, namely for different *MinSup* values, how many 2-large itemsets has been produced by **Generate2LargeItemsets** algorithm that have length two. These generated 2-large itemsets are fed to the main algorithm to generate larger itemsets. The number of 2-large itemsets has an I^2 effect on the time complexity of the main algorithm and is dependent to the number of large *ObjectInfo* instances. In the worst

case the number of 2-large itemsets is equal to B^2R^2, if B is the number of large *ObjectInfo* and R is the maximum number of relations of *ObjectInfo*s.

Figure 13 and Figure 14 show large itemsets count that have been caused by different *MinSup* values. Since the variation in these counts is great, they are shown in two figures. As these two figures show, these counts are dependent to the number of input 2-large itemsets and has an I effect on the time complexity of the **GenerateRules** algorithm and the number of generated rules as well.

Figure 15 and Figure 16 also show the strong rules counts that have been generated by different *MinSup* values. These figures show how this count is related to *MinSup* and *MinConf* values and the number of large itemsets as well. Due to the non-existence of an exact definition of transactions and also since we didn't guide the mining process (e.g. filtering input data by SPARQL commands to show the ability of the proposed algorithm in mining ARs without the end-user involvement), the generated ARs count is usually high. A large number of generated rules are meaningless or uninteresting, hence proposing a method to distinguish useful ARs from uninteresting ones is suggested for future work. Similarly to the large itemsets figures, due to the great differences between the counts of generated rules, these counts are shown in two figures.

Finally Figure 17 shows the average rules confidences arising from different *MinSup* values. *MinConf* value has been kept at 0.7. This figure shows that the rules confidence values are independent of the *MinSup* value and the large itemsets count. Independent of the generated rules count, the average confidence value usually is high and is between 0.864 and 0.967.

7. CONCLUSIONS AND FUTURE WORK

In this paper the importance of mining association rules from semantic web data and related challenges was discussed and a new algorithm was proposed that can deal with and solve these challenges. The proposed algorithm name is **SWApriori**. This algorithm can discover ARs from semantic web datasets directly, particularly generalized datasets which do not belong to special domain. On the other hand the algorithm can handle all kinds of datasets and ontologies regardless of the dataset domain. The rationale behind the developed method is that the algorithm after receiving a semantic web dataset, proceeds by applying ontology semantics (if needed), data discretization, infrequent data elimination and finally converting triples to numerical IDs. At the first level of the ARM process, the algorithm identifies large objects and then generates all large objects sets of length two. Afterwards the algorithm generates 2-large itemsets through large objects sets regardless to transactions. Here each itemset consists of multi items and each item consists of an object and a predicate (relation). After generating all 2-large itemsets, the algorithm continues by generating L-large itemsets ($L \geq 3$) based on (L-1)-large itemsets. Finally ARs are discovered by using all large itemsets. Discovered rules contain only one item in the consequent part.

The most sensible features of the proposed algorithm are as follow:

- There is no need to convert semantic web data to traditional data. The input data are used in their original format, triple format, by the algorithm.
- Traditional association rule mining algorithms (like Apriori) are not used.
- There is no need for a transactions concept: in fact with semantic web datasets, there is no transaction.

- There is no need for user involvement in the mining process: here the main user role is to provide input dataset and the values of *MinSup* and *MinConf*. That is the end-user doesn't need to be aware of dataset and ontology structure. But if the end-user wishes, he/she can filter input dataset by SPARQL language or extend the input dataset by assembling linked datasets. Also the end-user can tune the pre-process and the post-process of the proposed algorithm by using ontology concepts for smarter results.
- The algorithm considers different relations between entities: in this algorithm each item consists of an object and a predicate. These items are considered in the mining process.
- The algorithm handles heterogeneous data structures.
- The proposed algorithm can be easily adapted to use other binary combinations of subjects, predicates and objects in generating ARs

And there are some drawbacks in the proposed method as:

- The proposed method is not intelligent enough to involve meaning of data (provided by ontology) in the mining process to guide the process intelligently and generate only interested and useful rules.
- If the content of the input data is general and the end-user does not filter it, the number of generated ARs would be enormous and a large part of them may be uninteresting.

In fact, this algorithm is very similar to the Apriori algorithm but with different strategies and based on these strategies, the algorithm is able to mine ARs from specialized and generalized semantic web datasets directly. We believe that this kind of learning will become important in the future and will have an effect on the machine learning research area especially the area of semantic web research. The acquired results show the usefulness of the proposed method.

As future work, we intend to apply this method to linked data [24] as the algorithm collects data which are related to an entity from multi datasets automatically, and mine ARs from these connected data. This work require such concepts as ontology alignment, otology mapping, broken links and etc.

Another possible topic for future work is to use encoded knowledge in the ontologies in order to filter the generated association rules.

Other interested possibilities are to cluster entities based on generated frequent itemsets [37]. It is possible to apply this clustering to subjects or objects.

Usually there are hierarchically structure and inheritance rules involved in ontologies [38, 39]. Considering these concepts will lead to a reduction in the generated association rules and improve obtained results quality.

Figure 10 - ClassInfo instances state (Example)

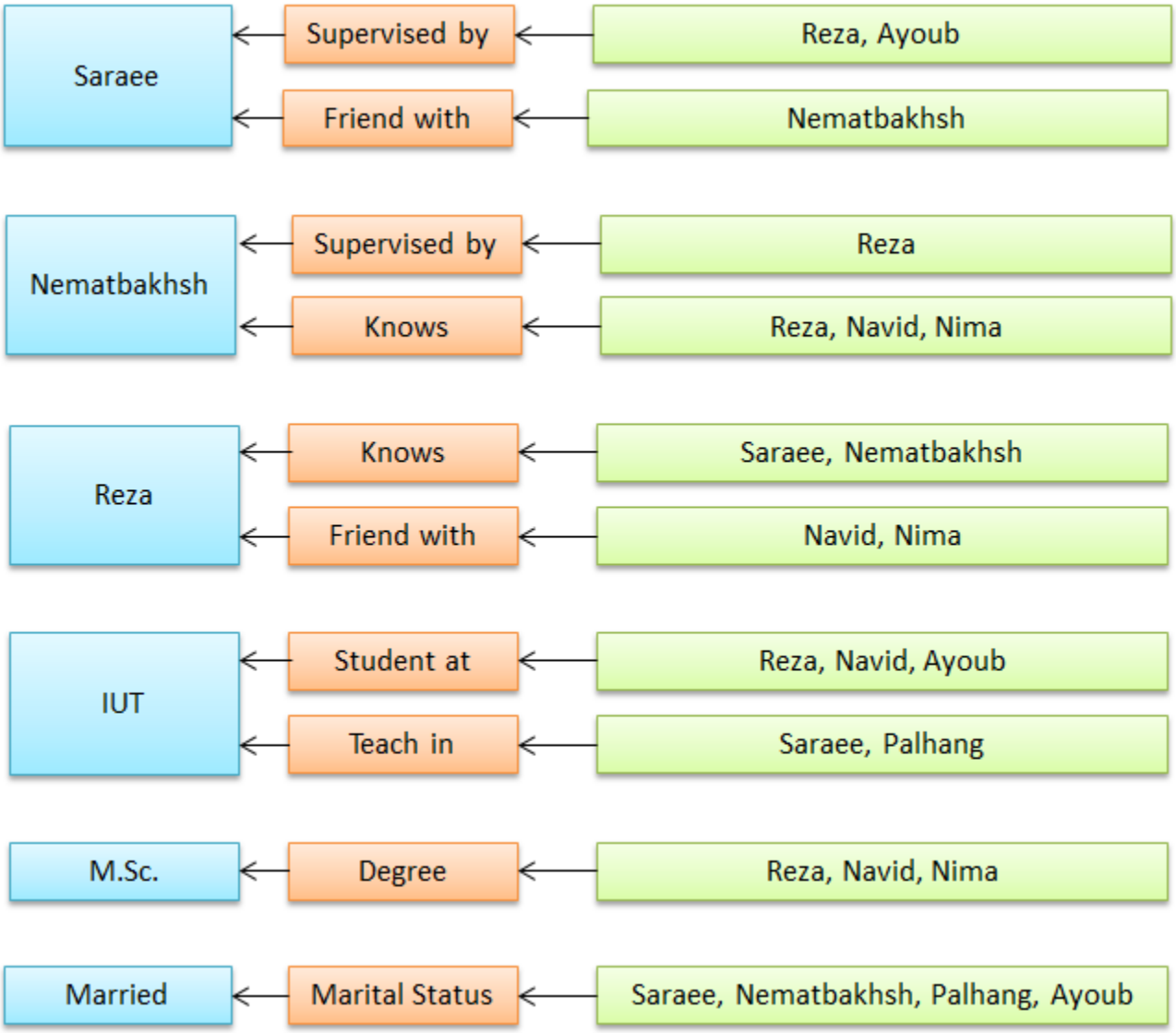


Table 4 - Some discovered association rules along with their confidence and support

Rule	Confidence	Support
goClassificationProcess(cellular metabolism) ➔ goClassificationProcess(physiological process)	0.95	0.16
massSpecFile(0) ➔ state(Solid)	0.88	0.19
goClassificationFunction(catalytic activity) ➔ goClassificationProcess(physiological process)	0.81	0.26
drugType(experimental) ➔ state(Solid)	0.85	0.14
drugType(smallMolecule) ➔ state(Solid)	0.91	0.20
state(Solid) ➔ structure(1)	0.85	0.20
goClassificationFunction(catalytic activity) , goClassificationProcess(metabolism) ➔ goClassificationProcess(physiological process)	0.78	0.26
state(Solid) , massSpecFile(0) ➔ structure(1)	0.91	0.19
structure(1) , massSpecFile(0) , drugType(smallMolecule) ➔ state(Solid)	0.89	0.19

Figure 11 - Covered entities (ObjectInfo) count by different MinSup values

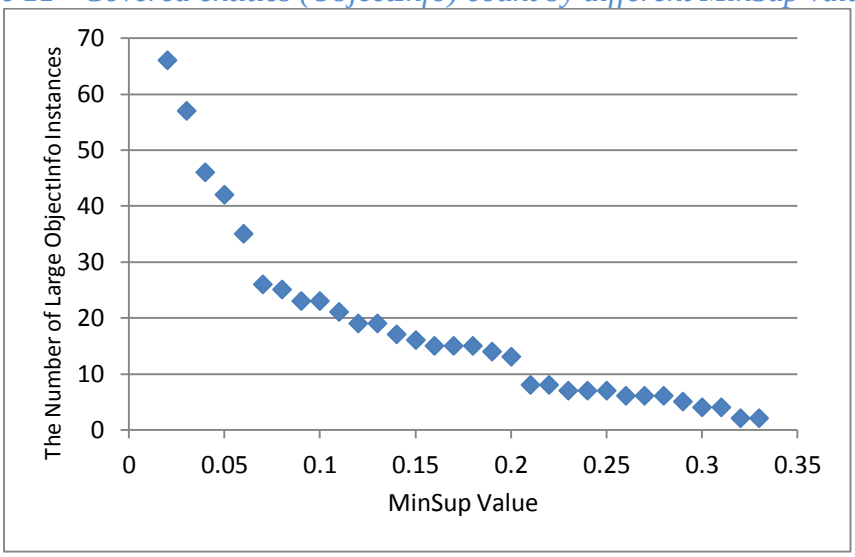


Figure 12 - Covered 2-latge itemsets count by different MinSup values

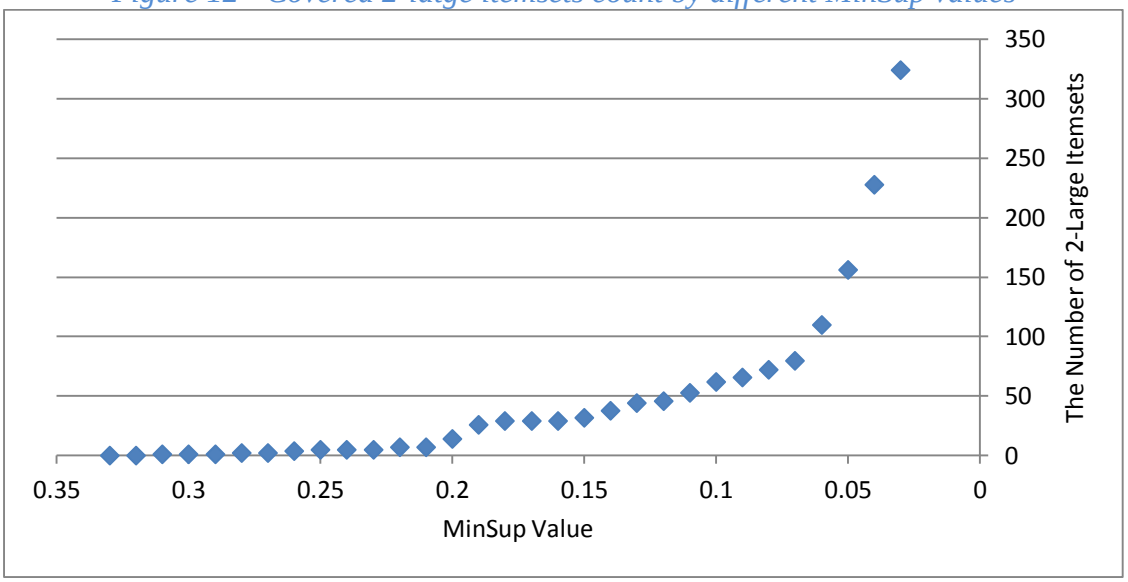


Figure 13 - Generated large itemsets count (MinSup = 0.02 ... 0.19)

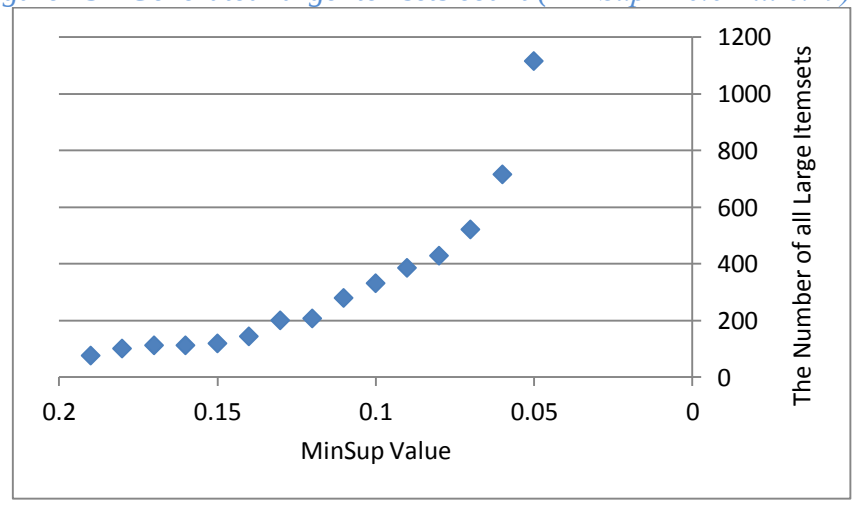


Figure 14 - Generated large itemsets count (MinSup = 0.20 ... 0.33)

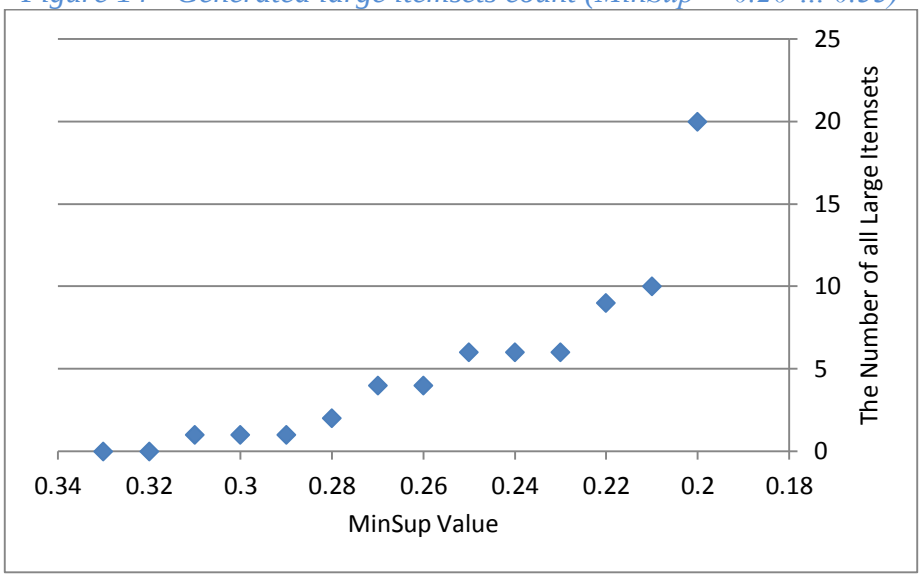


Figure 15 - Generated strong ARs count (MinSup = 0.02 ... 0.19)

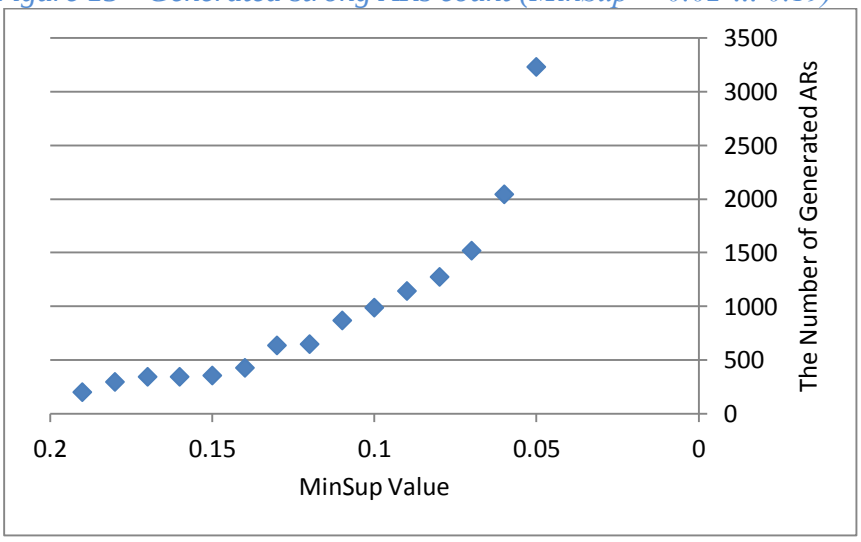


Figure 16 - Generated strong ARs count (MinSup = 0.20 ... 0.33)

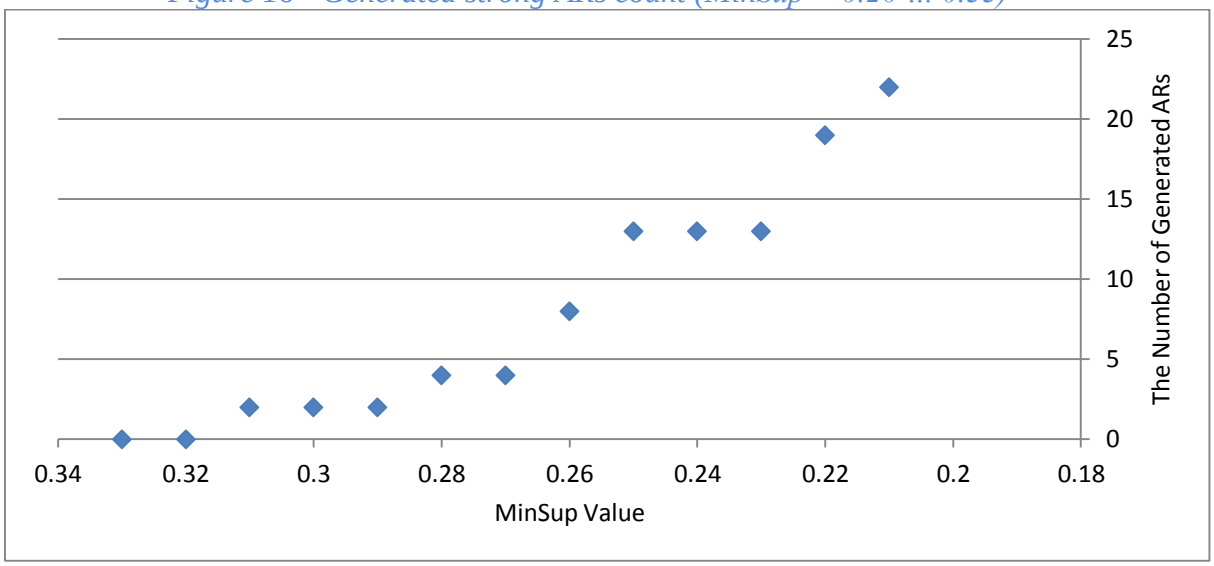
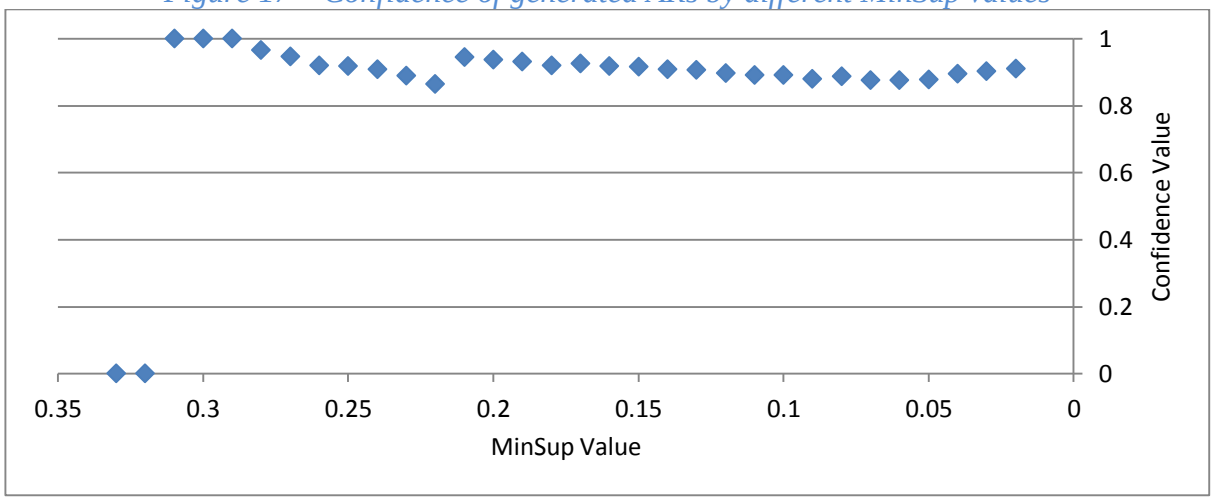


Figure 17 – Confidence of generated ARs by different MinSup values



8. REFERENCES

[1] A. H. G.Stumme, B.Berendt, "Semantic web mining: state of the art and future directions," *Web Semantics: Science, Services and Agents on the World Wide Web*, pp. 124-143, 2006.

[2] N. G.-P. J.M.Benitez, F.Herrera, "Special issue on "New Trends in Data Mining" NTDM," *Knowledge-Based Systems*, pp. 1-2, 2012.

[3] J. Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh, "Algorithms for association rule mining—a general survey and comparison," ACM SIGKDD Explorations Newsletter 2, no. 12000.

[4] H. W. J.Zhang, Y.Sun, "Discovering Associations among Semantic Links," presented at the Web Information Systems and Mining, 2009. WISM 2009. International Conference on, 2009.

[5] Y. S. Stephan Bloehdorn, "Kernel methods for mining instance data in ontologies," presented at the The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, 2007.

[6] C. d. A. N.Fanizzi, F.Esposito, "Metric-based stochastic conceptual clustering for ontologies," *Information Systems*, pp. 792-806, 2009.

[7] L.Getoor, "Link mining: a new data mining challenge," presented at the SIGKDD Explorations, News, 2003.
[8] L. D. R. S.Muggleton, "Inductive logic programming: theory and methods," *The Journal of Logic Programming*, vol. 19-20, pp. 629-679, 1994.
[9] T. I. R.Agrawal, A.N.Swami, "Mining association rules between sets of items in large databases," presented at the SIGMOD '93 Proceedings of the 1993 ACM SIGMOD international conference on Management of data 1993.
[10] W. F. Gregory Piateski, *Knowledge Discovery in Databases* MIT Press Cambridge, MA, USA, 1991.
[11] U. G. Hipp Jochen, and Gholamreza Nakhaeizadeh, "Algorithms for association rule mining—a general survey and comparison," presented at the ACM SIGKDD Explorations Newsletter, 2000.
[12] C. Zhang, and Shichao Zhang, *Association rule mining: models and algorithms*: Springer-Verlag, 2002.
[13] C. Hidber, *Online association rule mining* vol. 28: ACM, 1999.
[14] R. S. R.Agrawal, "Fast algorithms for mining association rules," presented at the In Proceeding of 20th international conference in large databases, 1994.
[15] K. Z. X.Liu, W.Pedrycz, "An improved association rules mining method," *Expert Systems*, pp. 1362-1374, 2012.
[16] G. K. M.Kuramochi, "Frequent Subgraph Discovery," presented at the Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, 2001.
[17] S. N. Y.Chi, R.R. Muntz, J.N.Kok, "Frequent Subtree Mining - An Overview," *Fundamenta Informations*, vol. 66, pp. 161 - 198, 2005.
[18] A. R. Islam, and Tae-Sun Chung, "An Improved Frequent Pattern Tree Based Association Rule Mining Technique," presented at the Information Science and Applications (ICISA), International Conference on, 2011.
[19] J. M. V. V.Pachón Álvarez, "An evolutionary algorithm to discover quantitative association rules from huge databases without the need for an a priori discretization," *Expert Systems with Applications*, pp. 585-593, 2012.
[20] V. V. Rao, and E. Rambabu, "Association rule mining using FPTree as directed acyclic graph," presented at the Advances in Engineering, Science and Management (ICAESM), International Conference on, 2012.
[21] V. T. Vivek Tiwari, S.Gupta, R.Tiwari, "Association Rule Mining: A Graph Based Approach for Mining Frequent Itemsets," presented at the Networking and Information Technology (ICNIT), 2010 International Conference on, 2010.
[22] R. B. V.Nebot, "Finding association rules in semantic web data," *Knowledge-Based Systems*, pp. 51-62, 2012.
[23] R. I. V.Narasimha, O.P.Vyas, "LiDDM: A Data Mining System for Linked Data," presented at the Proceedings of the LDOW2011, Hyderabad, India, 2009.
[24] T. H. C.Bizer, T.Berners-Lee, "Linked data - the story so far," *International Journal on Semantic Web and Information Systems*, pp. 1-22, 2009.
[25] M. A. Khan, Gunnar Aastrand Grimnes, and Andreas Dengel, "Two pre-processing operators for improved learning from semantic web data," presented at the In First RapidMiner Community Meeting And Conference (RCOMM), 2010.
[26] F. N. Ziawasch Abedjan, "Context and Target Configurations for Mining RDF Data," presented at the SMER '11 Proceedings of the 1st international workshop on Search and mining entity-relationship data 2011.
[27] A. B. Christoph Kiefer, André Locher, "Adding data mining support to SPARQL via statistical relational learning," in *ESWC'08 Proceedings of the 5th European semantic*

web conference on The semantic web: research and applications methods, 2008, pp. 478-492

[28] T. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," *Human–Computer Studies*, pp. 907-928, 1995.

[29] C. B. Dimitris Kontokostasa, Sören Auerb, Sebastian Hellmannb, Ioannis Antonioua, George Metakides, "Internationalization of Linked Data: The case of the Greek DBpedia edition," *Web Semantics: Science, Services and Agents on the World Wide Web*, pp. In Press, Corrected Proof, 2012.

[30] F. V. H. D.Fensel, I.Horrocks, D.L.McGuinness, P.F.Patel-Schneider, "OIL: An Ontology Infrastructure for the Semantic Web," *IEEE Intelligent Systems*, vol. 18, pp. 38 - 45, 2001.

[31] B. Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, Carsten Lutz, "Owl 2 web ontology language: Profiles," W3C Recommendation2009.

[32] M. J. Krys J. Kochut, "SPARQLeR: Extended Sparql for Semantic Association Discovery," presented at the The Semantic Web: Research and Applications, 4th European Semantic Web Conference, ESWC 2007, 2007.

[33] E. Prud'Hommeaux, and Andy Seaborne, "SPARQL query language for RDF," presented at the W3C recommendation 15, 2008.

[34] J. L. C.Bizer, G.Kobilarov, S.Auer, C.Becker, R.Cyganiak, S.Hellmann, "DBpedia - A crystallization point for the Web of Data," *Web Semantics*, pp. 154-165, 2009.

[35] Drugbank. (2012/09/11). *Drugbank documentation: http://www.drugbank.ca/documentation*.

[36] G. Yang, S. Mabu, K. Shimada, and K. Hirasawa, "A novel evolutionary method to search interesting association rules by keywords," *Expert Systems with Applications*, vol. 38, pp. 13378-13385, 2011.

[37] K. W. B.C.M.Fung, M.Ester, "Hierarchical document clustering using frequent itemsets," presented at the Proceedings of the Third SIAM International Conference on Data Mining, SIAM, 2003.

[38] N.Lavraˇc, "Using Ontologies in Semantic Data Mining with SEGS and g-SEGS," presented at the Discovery Science, 14th International Conference, Espoo - Finland, 2011.

[39] A. H. T. T.Jiang, "Mining RDF Metadata for Generalized Association Rules: Knowledge Discovery in the Semantic Web Era," presented at the WWW '06 Proceedings of the 15th international conference on World Wide, 2006.