

Using Market Basket Analysis to Find Semantic Duplicates in Ontology

Irina Astrova[1(✉)], Arne Koschel[2], and Su Ling Lee[2]

[1] Department of Software Science, School of IT, Tallinn University of Technology, Akadeemia tee 21, 12618 Tallinn, Estonia
irina@cs.ioc.ee

[2] Faculty IV, Department of Computer Science, University of Applied Sciences and Arts Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Germany
akoschel@acm.org

Abstract. This paper proposes a novel approach to detecting semantic duplicates in an ontology that was integrated from many other ontologies. The proposed approach is based on using: (1) a query log that was issued against the ontology; and (2) the Apriori and FP algorithms from market basket analysis, where sales transactions are viewed as queries and items in a sales transaction are viewed as search terms in a query. To prove the viability of the proposed approach, the paper also presents the results of four experiments that were conducted on the OntoLife ontology.

Keywords: Ontolgy · Semantic heterogeneity · Market basket analysis · Apriori algorithm · FP Growth (Frequent-Pattern Growth) algorithm · ICD (Iterated Contextual Distance) algorithm · Similar attributes · OntoLife ontology

1 Introduction

The main purpose of ontology-based data integration is to solve the semantic heterogeneity problem in data integration. Semantic heterogeneity refers to the ambiguous interpretation of similar terms, which are being used to describe the meaning of data in heterogeneous resources such as websites or databases. However, during an ontology-based data integration, the newly merged ontology can contain semantic duplicates, e.g., similar attributes. These attributes can cause incomplete query results.

As an example, consider a user who submits the following query against the Wikipedia ontology: "*Which performers were born in Chicago?*" In response to this query, the query-answering system will return only one result (viz. Michael Ian Black). However, if it were known that *actor* and *comedian* are subclasses of *performer* and that their attributes *birthplace*, *birth place*, *city of birth*, *place of birth* and *origin* are semantic duplicates of performer's location, the query-answering system could return 163 additional results [1]. Thus, the recall of query results can be greatly improved by finding similar attributes.

In this paper, we propose to use market basket analysis to find similar attributes. In market basket analysis, two items (or products) are considered to be similar if the purchase patterns of their customers are similar. Analogously, two attributes are

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12252, pp. 197–211, 2020.
https://doi.org/10.1007/978-3-030-58811-3_14

considered to be similar if the querying patterns of their users are similar [2]. Continuing the example with the Wikipedia ontology: If it were known that there are many users who have asked about the actor's *birth place* together with the actor's *name* and *birth date*, and that there are many users who have asked about the actor's *origin* together with the actor's *name* and *birth date*, we could conclude that attributes *birth place* and *origin* in a class *actor* are similar to each other.

2 Market Basket Analysis

Baskets are defined as sets of products bought together by customers in sales transactions during their visits to supermarkets or online shops. Market basket analysis examines the buying behaviour of customers and it is used to identify similar products based on the purchase patterns. A typical example of similar products are Becks and Heineken beers. At the first sight, these two products may appear dissimilar as they do not have many common customers: the customers buy either Becks or Heineken, but not both. However, it was observed that together with Becks and Heineken, many of the customers buy the same products such as chips, sausages and pretzels. Based on this observation, a similarity measure is defined as two products are found to be similar, if the purchase patterns of their customers are similar.

The aim of market basket analysis is that of finding sets of items (products) that appear together in (are related to) many baskets. Let products be attributes in an ontology (search terms in queries issued against the ontology), baskets be queries and a sales transaction database be a query log. While market basket analysis aims at finding frequent itemsets, this information is often presented as a set of "if–then" rules called association rules. For example, *if customers buy Heineken, then they also buy chips, sausages and pretzels*.

To prove the applicability of market basket analysis to finding similar attributes in an ontology, we experimented with three algorithms:

- **ICD (Iterated Contextual Distance) algorithm** [17]: This is an array-based algorithm, which returns the distances between attributes.
- **Apriori algorithm** [3]: This is an array-based algorithm, which returns association rules.
- **FP (Frequent Pattern) Growth algorithm** [15]: This is a tree-based algorithm, which returns association rules.

In our experiments, we looked for sets of attributes that appear together in many queries, by analysing many-to-many relationships between the querying patterns and the user behaviours. We assumed that users who have similar questions in mind tend to submit similar queries, which have similar querying patterns.

In our previous work [2, 16], we used the ICD and Apriori algorithms to detect semantic duplicates in an ontology. In this paper, we employ the FP Growth algorithm, which is an improvement of the Apriori algorithm. The Apriori algorithm finds frequent itemsets and consequently querying patterns, by generating candidate itemsets. In contrast, the FP Growth algorithm finds querying patterns without the need for

candidate generation, rather by constructing an FP tree, which maintains the associations between frequent itemsets.

3 OntoLife

OntoLife [4] is an ontology for semantically managing personal information. This ontology has 110 attributes called datatype properties in the OWL terminology. It was developed by Intelligent System and Knowledge Processing Research Group (ISKP) by merging 10 other ontologies:

1. **Person** [5]: This ontology provides a formal vocabulary description of people. Furthermore, it defines constraints (e.g., a person should have exactly one name). It does not have any attributes.
2. **Friend of a Friend (FOAF)** [6]: This ontology describes the people's activities and the relationships such as friendships between them. It does not have any attributes.
3. **Family Tree** [7]: This ontology models the application domain of family that defines people's relationships such as children and parents. It does not have any attributes.
4. **Relationship** [8]: This ontology defines other relationships between people such as friends, colleagues and tutors. It does not have any attributes.
5. **ISO lists for Countries and Languages** [9]: This ontology models the application domain of people's countries and their languages. It does not have any attributes.
6. **Project** [10]: This ontology describes research projects and people who are involved into those projects. It has 6 attributes such as project title, start and end dates.
7. **Research** [11]: This ontology describes the hierarchy of research projects (e.g., related projects can include cloud computing and data mining). It does not have any attributes.
8. **Publication** [12]: This ontology models the application domain of people's publications. It has 27 attributes such as book title, edition of publication and publication date.
9. **PersonProjectAssociation** [13]: This ontology models the relationships between people and publications or research projects. Furthermore, it defines constraints (e.g., a person should have at least one role in a research project). It does not have any attributes.
10. **Biography** [14]: This is the largest ontology. It has 77 attributes. It describes the people's demographics, their profiles and contacts. Furthermore, it describes the CV-related information such as people's education and qualifications, skills and working experiences as well as injuries that people have got at work.

We selected OntoLife for our experiments because it is from an easy-to-understand application domain. Furthermore, the ontology has many attributes and is merged out of many other ontologies from the same application domain. Thus, it is likely to have semantic duplicates.

4 Experiment Results

4.1 Assumptions

To run the Apriori and FP Growth algorithms against the OntoLife ontology, we made a number of assumptions about how data are stored and manipulated when searching for frequent itemsets (similar attributes):

- In market basket analysis, a basket contains only a small set of products from all products available in the supermarket or online shop. Similarly, it is also assumed that users do not inquire about all attributes in the ontology at once in a single query.
- In market basket analysis, a basket can contain any number of products. Similarly, it is also assumed that a query can contain any number of attributes (i.e., the query can be of any size).
- In market basket analysis, the details on sales transactions such as the date of purchase, product price and product quantities are ignored. Similarly, the values of attributes in the ontology are ignored.
- In market basket analysis, the order of products in a sales transaction is ignored. Similarly, the order of attributes in a query is ignored.
- In market basket analysis, sales transactions are mined for a long period of time (at least for 6 months). As a result, the sales transaction database is large. Similarly, it is also assumed that the log of queries issued against the ontology is large.
- All queries are taken into account, regardless of whether they return results or not.

4.2 Query Log

As input, the Apriori and FP Growth algorithms took the query log. This log was the result of the past user interactions with the OntoLife ontology. Figure 1 shows an excerpt of the query log, which comprises 30 queries. These queries are written in SPARQL and contain 41 attributes, which constitute approximately 37% of the total attributes in the ontology. The shortest query has one attribute, whereas the longest query has 5 attributes. Not only can queries be distinguished by their size, but also by their frequency occurrence in the query log. A unique query appears only once in the query log. An example of a unique query is: "*Who have "rain man" as their nickname?*" In contrast, repeated queries are the queries, which appear more than once in the query log. They queried on the same attributes but possibly with different values or in a different order. An example of repeated queries is: "*Who were injured on November 22, 2009, were 50% disabled by that injury and worked in the public sector?*" and "*Who were injured on January 2, 1989, worked in the government sector and were 20% disabled by that injury?*".

The query log was converted into a sales transaction database, which contains 30 sales transactions (one sales transaction per query) represented as rows. Figure 2 shows an excerpt of the sales transaction database. In this database, each query is identified by TID (transaction identifier). For each row, if an attribute does not appear in the query, it will not appear in the corresponding row either.

4.3 Minimum Support Threshold and Minimum Confidence Threshold

The Apriori and FP Growth algorithms find frequent itemsets, by pruning those itemsets whose support is lower than a user-defined minimum support threshold. After that, the algorithms generate strong association rules from the frequent itemsets, by pruning those association rules whose confidence is lower than a user-defined minimum confidence threshold. We conducted four experiments with different values for the minimum support threshold and the minimum confidence threshold because these values had impact on the results of our experiments.

A support reveals the probability of occurrence of an attribute set X in queries. A frequent itemset has a support, which is greater than or equal to the minimum support threshold, where P denotes the probability. A confidence refers to a conditional probability. This is the probability that an attribute set Y occurs in queries given that an attribute set X has already occurred in the same queries. A confidence is used to generate strong association rules from the frequent itemsets that meet the minimum confidence threshold.

$$\textbf{Confidence}\ (X \Rightarrow Y) = \textbf{Support}(X \Rightarrow Y)/\textbf{Support}(\mathbf{X}) = \mathbf{P}(X \cap Y)/\mathbf{P}(\mathbf{X}) = \mathbf{P}(Y|X)$$

Like in market basket analysis, false positives and false negatives are inevitable when the thresholds are set either too low or too high, respectively. For example, when the minimum support threshold was too low, false positives tended to occur as those originally infrequent attributes were falsely found as frequent. When the minimum support threshold was too high, false negatives tended to occur where those originally frequent attributes were falsely found as infrequent.

4.4 Experiment 1

Figure 3 shows the results of our first experiment, where the minimum support threshold was 12% and the minimum confidence threshold was 55%. In this experiment, 24 strong association rules were generated by the algorithms. Two pairs of attributes were found similar:

1. *specificDate* and *sickness_injuryDesc*;
2. resource's *title* and resource's *name*.

Among the strong association rules, we looked for sets of attributes that frequently appeared together in the consequents (right-hand sides) of those rules. When we found such rules, we had candidates for similar attributes that appeared in the antecedents (left-hand sides) of those rules. For example, the consequents of Rule 17 and Rule 20 (marked in blue in Fig. 3) suggested a pair of similar attributes: resource's *title* and resource's *name*. Furthermore, the consequents of Rule 11 and Rule 14 (marked in red in Fig. 3) suggested another pair of similar attributes: *specificDate* and *sickness_injuryDesc*. However, intuitively these attributes were not similar and thus, such a finding was a false positive. One possible reason for this false positive was the small size of the query log (which contained only 30 queries).

The size of the query log was a crucial factor that could affect the experiment results. This is because the two algorithms assume that a sales transaction database is large. It was observed that there was yet another pair of similar attributes but it was not found (false negative): publication's *description* and biography's *description*. Again, due to the small size of the query log, there were no common query patterns, which inquired both attributes.

There was no significant difference in the runtime between the algorithms, as both took about 16 ms, but in the made count. In particular, the Apriori algorithm made 70 counts on the itemsets, whereas the FP Growth made 30 counts.

In summary, using the first experiment results, the OntoLife ontology could be refined. In particular, OWL has a construct *owl:equivalentProperty* to specify that resource's *title* is a semantic duplicate of resource's *name*.

4.5 Experiment 2

Figure 4 shows the results of our second experiment, where the minimum support threshold was 12% and the minimum confidence threshold was 90%. In this experiment, 16 strong association rules were generated by the algorithms. One pair of attributes was found similar: *specificDate* and *sickness injuryDesc*. However, it was the same false positive discovered before.

Since the minimum support threshold was still 12%, the algorithms showed the same runtime and made the same count on the itemsets as they did in the first experiment.

4.6 Experiment 3

Figure 5 shows the results of our third experiment, where the minimum support threshold was 10% and the minimum confidence threshold was 90%. In this experiment, 51 strong association rules were generated by the algorithms. Four pairs of attributes were found similar:

1. *specificDate* and *sickness injuryDesc*;
2. *disab percentage* and *hasGovernmentalInvolvement;*
3. *hasGovernmentalInvolvement* and *sector;*
4. *disab percentage* and *sector.*

However, all these pairs were false positives. The first pair was caused by the small size of the query log. The second pair was caused by redundant association rules: Rule 13 and Rule 30. (An association rule is redundant if a more general rules with the same or a higher confidence exists, i.e., if it has the same consequent but one or more items were removed from the antecedent.) The third pair was also caused by redundant association rules: Rule 17 and Rule 33. The fourth pair was discovered due to the transitivity. Since *disab percentage* is similar to *hasGovernmentalInvolvement* and *hasGovernmentalInvolvement* is similar to *sector*, so is *disab percentage* similar to *sector*.

In the third experiment, the FP Growth algorithm was again faster than the Apriori algorithm. In particular, the former ran 16 ms, whereas the latter ran about 31 ms.

During its run, the Apriori algorithm made 131 counts on the itemsets, whereas the FP Growth made 30 counts.

4.7 Experiment 4

In the fourth experiment, the minimum support threshold was 28% and the minimum confidence threshold was 55%. Here no strong association rules were generated by the algorithms. Therefore, we concluded that this minimum support threshold was too high – it pruned all the frequent itemsets.

In the fourth experiment, the FP Growth algorithm was again faster than the Apriori algorithm. In particular, the former ran about 1 ms, whereas the latter ran about 3 ms. During its run, the Apriori algorithm made 3 counts on the itemsets, whereas the FP Growth made 30 counts.

5 Related Work

Different approaches to finding similar attributes in an ontology have been proposed in the literature [18]. These approaches fall into four main categories:

- **Term-based approach:** In this approach, two attributes are considered to be similar if their names are similar. However, many false negatives and false positives can be found in the presence of synonyms and homonyms. Synonyms are different terms used to name same attributes (e.g., "bear" and "support"). Homonyms are same terms used to name different attributes. They have the same spelling but different meanings (e.g., "bear" can refer to an animal or can mean "support", depending on the contexts they are used in).
- **Value-based approach:** In this approach, two attributes are considered to be similar if their values are similar. This approach can detect semantic duplicates in an ontology more accurately than the term-based approach. However, without information about data instances, this approach is not feasible.
- **Structure-based approach**: In this approach, two attributes are considered to be similar if their structures are similar. Structures can be either internal or external. Examples of internal structures are data types, cardinalities, domains and transitivity of the attributes. However, internal structures are the subject of false negatives and false positives. For example, many attributes can have the same data types, but they are not necessarily similar. In case of external structures, an ontology is viewed as a graph, where vertices represent concepts and edges represent semantic relations between those concepts. This approach attempts to find similar classes and attributes, by identifying the structural isomorphism between sub-graphs of an ontology. For example, three types of relations can be identified between classes: taxonomic, mereologic and all the involved relations. In the taxonomic structure, the graph depicts the subClassOf relations. These relations indicate that the connected concepts are similar. One of the common similarity measures is to count the number of edges in the taxonomy between two classes to determine their similarity. In the mereologic structure, the graph depicts the PartOf relations. In all the

involved relations structure, classes are related via their attribute definitions. External structures are also the subject of false negatives and false positives. For example, it is difficult to distinguish the PartOf relations from the subClassOf relations.

- **Context-based approach:** In this approach, two attributes are considered to be similar if the contexts of their usage are similar. However, it can be difficult to identify similar contexts. In this paper, we proposed to use market basket analysis to solve this problem.

6 Conclusion

During an ontology-based data integration, the newly merged ontology such as OntoLife can contain semantic duplicates (e.g., similar attributes). These attributes can cause incomplete query results. In this paper, we focused on finding semantic duplicates in the OntoLife ontology. The approach we proposed is as follows: (1) the Apriori or FP Growth algorithm is applied to find strong association rules derived from a query log, where each query is a set of search terms being attributes in the ontology, and (2) candidates for semantic duplicates are determined as antecedents of the strong association rules having similar consequents.

In addition, we conducted a number of experiments on the query log, which was a list of queries posed by users to inquire the OntoLife ontology. The Apriori algorithm scanned the query log multiple times for generating candidate itemsets. In contrast, FP Growth algorithm scanned the query log only twice for constructing an FP tree. In all our experiments, the FP Growth algorithm made a constant of 30 counts on the itemsets. In contrast, the Apriori algorithm made a number of counts, which varied from 3 to 131 counts, depending on the minimum support threshold. The more counts the Apriori algorithm made, the longer time it ran. Despite the different runtime (the FP Growth algorithm was faster than the Apriori algorithm), it was observed that the algorithms returned the same results.

Overall, the results of our experiments showed that market basket analysis in general and the Apriori and FP Growth algorithms in particular are capable of finding similar attributes in an ontology, although user involvement is required to identify the optimal values for the minimum support threshold and the minimum confidence threshold.

Acknowledgement. Irina Astrova's work was supported by the Estonian Ministry of Education and Research institutional research grant IUT33-13.

Appendix

Nr.	Query in SPARQL:

01	PREFIX bio:≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻
	SELECT ?name
	WHERE {
	?person bio:awardTitle ?awardTitle .
	?person bio:awardDate ?awardDate .
	?person bio:shipmentDate ?shipmentDate.
	?person bio:internationalName ?internationalName .
	FILTER ((?awardTitle = "Best Bookseller") ||
	(?awardTitle="Oddest Booktitle Prize")
	&& (?awardDate = 2012-02-24)
	& (?shipmentDate = 10-Sept-2012) &&
	(?internationalName = "Springer"))
	}

02	PREFIX bio: ≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻
	SELECT *
	WHERE {
	?person bio:disab_type ?disab_type .
	?person bio:weight ?weight .
	FILTER ((?disab_type = "persistent delusion" &&
	?disab_type = "depression") && ?weight ≻ 80)
	}

03	PREFIX bio: ≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻
	SELECT ?name, ?nickname
	WHERE {
	?person bio:nickname ?nickname .
	FILTER (?nickname = "rain man")
	}

04	PREFIX bio: ≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻ SELECT ?name, ?nationality WHERE { ?person bio:nationality ?nationality . ?person bio:dateofBirth ?dateofBirth . ?person bio:militaryUnit ?militaryUnit . ?person bio:militaryGrade ?militaryGrade . FILTER (?nationality = "French" && ?dateofBirth = "2000-03-29" && militaryUnit = "Eagle" && militaryGrade = "Corporal")

Fig. 1. Query log for OntoLife.

05	PREFIX bio:≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻ SELECT ?name WHERE { ?person bio:name ?name . ?person bio:height ?height . ?person bio:disab_type ?disab_type . ?person bio:weight? weight . ?person bio:selfGrade? selfGrade . ?person bio:eyecolor? eyecolor . FILTER (?height ≺ "1.0 metres" && weight ≻= 80 kg && (eyecolor = "hazel" || eyecolor = "blue") && disab_type = "depression" && selfGrade = "good") } ORDER BY ?name
06	PREFIX bio: ≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻ SELECT ?name WHERE { ?person bio:name ?name . ?person bio:journal ?journal . ?person bio:note ?note . FILTER ((?journal = "Newsweeks") || (?journal="Times") && note = "Fifty cents") } ORDER BY ?name
....	....
....	....

30	PREFIX bio: ≺"http://users.auth.gr/ elkar/thesis/Biography.owl#" ≻
	SELECT *
	WHERE {
	?person bio:nickname ?nickname .
	?person bio:specificDate ?specificDate .
	?person bio:disab_percentage ?disab_percentage .
	?person bio: hasGovernmentalInvolvement
	?hasGovernmentalInvolvement .
	?person bio:sector ?sector .
	FILTER ((?specificDate = "2009-11-23" && sector ="private non-business" && disab_percentage = 10 && hasGovernmentalInvolvement = 0 && nickname = "flowerpot"))
	}

Fig. 1. (*continued*)

TID:	Items:
001	{*awardTitle, awardDate, shipmentDate, internationalName*}
002	{*disab_type, weight*}
003	{*nickname*}
004	{*nationality, dateOfBirth, militaryUnit, militaryGrade*}
005	{*height, disab_type, weight, selfGrade, eyecolor* }
006	{*journal, note*}
...	...
...	...
030	{*nickname, specificDate, disab_percentage, governmentInvolvement, sector*}

Fig. 2. Query log converted to sales transaction database.

Nr.	Association rules:	Confidence:
Support: 13.3%		
1)	$(specificDate \Rightarrow sector)$	100%
2)	$(specificDate \Rightarrow disab_percentage)$	100%
3)	$(sickness_injuryDesc \Rightarrow sector)$	100%
4)	$(sickness_injuryDesc \Rightarrow disab_percentage)$	100%
5)	$(resource'sname \Rightarrow isAlive)$	80%
6)	$(resource'sname \Rightarrow nickname)$	80%
7)	$(resource'stitle \Rightarrow isAlive)$	57%
8)	$(resource'stitle \Rightarrow nickname)$	57%
9)	$(specificDate, sector \Rightarrow disab_percentage)$	100%
10)	$(specificDate, disab_percentage \Rightarrow sector)$	100%
11)	$(specificDate \Rightarrow disab_percentage, sector)$	100%
12)	$(sickness_injuryDesc, sector \Rightarrow disab_percentage)$	100%
13)	$(sickness_injuryDesc, disab_percentage \Rightarrow sector)$	100%
14)	$(sickness_injuryDesc \Rightarrow disab_percentage, sector)$	100%
15)	$(nickname, resource'sname \Rightarrow isAlive)$	100%
16)	$(isAlive, resource'sname \Rightarrow nickname)$	100%
17)	$(resource'sname \Rightarrow nickname, isAlive)$	80%
18)	$(nickname, resource'stitle \Rightarrow isAlive)$	100%
19)	$(isAlive, resource'stitle \Rightarrow nickname)$	100%
20)	$(resource'stitle \Rightarrow nickname, isAlive)$	57%
Support: 26%		
21)	$(sector \Rightarrow disab_percentage)$	100%
22)	$(disab_percentage \Rightarrow sector)$	100%
23)	$(isAlive \Rightarrow nickname)$	88%
24)	$(nickname \Rightarrow isAlive)$	66%

Fig. 3. Experiment 1 results. (Color figure online)

Nr.	Association rules:	Confidence:
Support: 13.3%		
1)	$(specificDate \Rightarrow sector)$	100%
2)	$(specificDate \Rightarrow disab_percentage)$	100%
3)	$(sickness_injuryDesc \Rightarrow sector)$	100%
4)	$(sickness_injuryDesc \Rightarrow disab_percentage)$	100%
5)	$(specificDate, sector \Rightarrow disab_percentage)$	100%
6)	$(specificDate, disab_percentage \Rightarrow sector)$	100%
7)	$(specificDate \Rightarrow disab_percentage, sector)$	100%
8)	$(sickness_injuryDesc, sector \Rightarrow disab_percentage)$	100%
9)	$(sickness_injuryDesc, disab_percentage \Rightarrow sector)$	100%
10)	$(sickness_injuryDesc \Rightarrow disab_percentage, sector)$	100%
11)	$(nickname, resource'sname \Rightarrow isAlive)$	100%
12)	$(isAlive, resource'sname \Rightarrow nickname)$	100%
13)	$(nickname, resource'stitle \Rightarrow isAlive)$	100%
14)	$(isAlive, resource'stitle \Rightarrow nickname)$	100%
Support: 26%		
15)	$(sector \Rightarrow disab_percentage)$	100%
16)	$(disab_percentage \Rightarrow sector)$	100%

Fig. 4. Experiment 2 results.

Nr.	Association rules:	Confidence:
Support: 10%		
1)	$(hasGovernmentalInvolvement \Rightarrow sector)$	100%
2)	$(hasGovernmentalInvolvement \Rightarrow disab_percentage)$	100%
3)	$(hasGovernmentalInvolvement \Rightarrow nickname)$	100%
4)	$(publication'sdescription \Rightarrow cert_date)$	100%
5)	$(biography'sdescription \Rightarrow cert_date)$	100%
6)	$(biography'sdescription \Rightarrow resource'stitle)$	100%
7)	$(sector, hasGovernmentalInvolvement \Rightarrow disab_percentage)$	100%
8)	$(disab_percentage, hasGovernmentalInvolvement \Rightarrow sector)$	100%
9)	$(hasGovernmentalInvolvement \Rightarrow disab_percentage, sector)$	100%
10)	$(nickname, hasGovernmentalInvolvement \Rightarrow sector)$	100%
11)	$(nickname, sector \Rightarrow hasGovernmentalInvolvement)$	100%
12)	$(sector, hasGovernmentInvolvement \Rightarrow nickname)$	100%
13)	$(hasGovernmentalInvolvement \Rightarrow nickname, sector)$	100%
14)	$(nickname, hasGovernmentInvolvement \Rightarrow disab_percentage)$	100%
15)	$(nickname, disab_percentage \Rightarrow hasGovernmentInvolvement)$	100%
16)	$(disab_percentage, hasGovernmentInvolvement \Rightarrow nickname)$	100%
17)	$(hasGovernmentInvolvement \Rightarrow nickname, disab_percentage)$	100%
18)	$(resource'stitle, cert_date \Rightarrow biography'sdescription)$	100%
19)	$(biography'sdescription, cert_date \Rightarrow resource'stitle)$	100%
20)	$(resource'stitle, biography'sdescription \Rightarrow cert_date)$	100%
21)	$(biography'sdescription \Rightarrow resource'stitle, cert_date)$	100%
22)	$(nickname, nationality \Rightarrow isAlive)$	100%
23)	$(nationality, isAlive \Rightarrow nickname)$	100%
24)	$(nickname, sector \Rightarrow disab_percentage)$	100%
25)	$(nickname, disab_percentage \Rightarrow sector)$	100%

Fig. 5. Experiment 3 results.

26)	$(nickname, sector, hasGovernmentalInvolvement \Rightarrow disab_percentage)$	100%
27)	$(nickname, disab_percentage, hasGovernmentalInvolvement \Rightarrow sector)$	100%
28)	$(nickname, disab_percentage, sector \Rightarrow hasGovernmentalInvolvement)$	100%
29)	$(disab_percentage, sector, hasGovernmentalInvolvement \Rightarrow nickname)$	100%
30)	$(disab_percentage, hasGovernmentalInvolvement \Rightarrow nickname, sector)$	100%
31)	$(nickname, sector \Rightarrow disab_percentage, hasGovernmentalInvolvement)$	100%
32)	$(nickname, disab_percentage \Rightarrow sector, hasGovernmentalInvolvement)$	100%
33)	$(sector, hasGovernmentalInvolvement \Rightarrow nickname, disab_percentage)$	100%
34)	$(nickname, hasGovernmentalInvolvement \Rightarrow disab_percentage, sector)$	100%
35)	$(hasGovernmentalInvolvement \Rightarrow nickname, disab_percentage, sector)$	100%

Support: 13.3%

36)	$(specificDate \Rightarrow sector)$	100%
37)	$(specificDate \Rightarrow disab_percentage)$	100%
38)	$(sickness_injuryDesc \Rightarrow sector)$	100%
39)	$(sickness_injuryDesc \Rightarrow disab_percentage)$	100%
40)	$(specificDate, sector \Rightarrow disab_percentage)$	100%
41)	$(specificDate, disab_percentage \Rightarrow sector)$	100%
42)	$(specificDate \Rightarrow disab_percentage, sector)$	100%
43)	$(sickness_injuryDesc, sector \Rightarrow disab_percentage)$	100%
44)	$(sickness_injuryDesc, disab_percentage \Rightarrow sector)$	100%
45)	$(sickness_injuryDesc \Rightarrow disab_percentage, sector)$	100%
46)	$(nickname, resource'sname \Rightarrow isAlive)$	100%
47)	$(isAlive, resource'sname \Rightarrow nickname)$	100%
48)	$(nickname, resource'stitle \Rightarrow isAlive)$	100%
49)	$(isAlive, resource'stitle \Rightarrow nickname)$	100%

Support: 26.6%

50)	$(sector \Rightarrow disab_percentage)$	100%
51)	$(disab_percentage \Rightarrow sector)$	100%

Fig. 5. (*continued*)

References

1. Astrova, I., Koschel, A.: Automatic detection of duplicated attributes in ontology. In: Cordeiro, J., Filipe, J. (eds.) ICEIS 2009: Proceedings of the 11th International Conference on Enterprise Information Systems, INSTICC, 2009, vol. DISI, pp. 283 – 286 (2009)
2. Astrova, I.: Improving query results with automatic duplicate detection. In: Ioannidis, Y., Manghi, P., Pagano, P. (eds.) Proceedings of the Second Workshop on Very Large Digital Libraries, VLDL 2009: A Workshop in conjunction with the European Conference on Digital Libraries 2009. Institute of Information Science and Technology, DELOS Association (2009)
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Stonebraker, M., Hellerstein, J.M. (eds.) Readings in Database Systems, 3rd edn., pp. 580 – 592. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998)
4. Kargiot, E., Kontopoulos, E.: OntoLife: An Ontology for Semantically Managing Personal Information. http://lpis.csd.auth.gr/ontologies/ontolist.html#ontolife
5. Person Ontology. http://ebiquity.umbc.edu/ontology/person.owl
6. Friend of a Friend (FOAF) Ontology. http://xmlns.com/foaf/spec/
7. Family Tree Ontology. http://users.auth.gr/elkar/thesis/FamilyTree.owl
8. Relationship Ontology. http://purl.org/vocab/relationship/
9. ISO lists for Countries and Languages Ontology. http://psi.oasis-open.org/iso/639/#
10. Project Ontology. http://ebiquity.umbc.edu/ontology/project.owl
11. Research Ontology. http://ebiquity.umbc.edu/ontology/research.owl
12. Publication Ontology. http://ebiquity.umbc.edu/ontology/publication.owl
13. PersonProjectAssociation Ontology. http://ebiquity.umbc.edu/ontology/association.owl
14. Biography Ontology. http://users.auth.gr/elkar/thesis/Biography.owl
15. Han, J., et al.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov. **8**(1), 53–87 (2004)
16. Astrova, I., Koschel, A., Lee, S.L.: How the Apriori Algorithm can help to find semantic duplicates in ontology. In: Knowledge-Based Software Engineering. Springer Nature (2020). to appear
17. Das, G., Mannila, H.: Context-based similarity measures for categorical database. In: Proceedings of PKDD, 2000, pp. 201–210 (2000)
18. Shvaiko, P., Euzenat, J.: Ontology Matching 2007. Springer, Heidelberg (2007)