JOURNAL OF APPLIED
COMPUTER SCIENCE
Vol. 21 No. 2 (2013), pp. 137-152

Evaluating Semantic Similarity With a New Method of Path Analysis in RDF Using Genetic Algorithms

Lukasz Strobin, Adam Niewiadomski

Lodz University of Technology
FTIMS/Insitute of Information Technology
ul. Wólczańska 215, 90-924 Lodz, Poland
800337@edu.p.lodz.pl
adam.niewiadomski@p.lodz.pl

Abstract. *This paper presents a novel method of evaluating semantic similarity by means of path analysis in RDF databases. Similarity is calculated by assignining each property (predicate in RDF terms) a weight, which is found using a genetic optimization algorithm. Presented method exhibits an advatage over existing methods, because of its flexibility and the fact that no prior knowledge of a particular database is necessary. This paper also presents an exemplary application of the method - recommendation engine. Proposed method is applied to a well known problem - music recommendation based on DBPedia. Results obtained in the experiment positively verify its advanntages and usefulness.*
Keywords: *RDF, Graph Databases, DBPedia, Genetic Algorithms, Path Analysis, Similarity.*

1. Introduction

Computing semantic similarity is an interesting problem, with applications in several domains, like recommendation systems, knowledge mining, knowledge

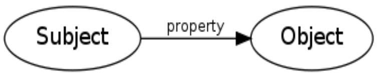


Figure 1. RDF triplet: object, predicate (property), subject

represenation and search algorithms. This paper presents a new method of computing semantic similarity by means of path analysis in RDF databases. Introduction to RDF is given in section 2 (e.g. see [1]). Computing semantic distance based on information from such databases is a recent research area (e.g. see [2, 3, 4], since huge amount of data is available in an LinkedData database [5]. An introduction to RDF is given in section 2. This paper presents methods of computing semantic similarity using path analysis in RDF databases, and proposes a novel method, based on genetic optimization of weights assigned to properties (edges) in paths between objects (section 3).

Application of proposed method to an area of item-based recommendation follows in section 3.3. Again, applying data from RDF to this field has been widely studied as well. Authors in [6, 7] discuss the usability and problems of using LinkedData in this domain.

Proposed method is used in item-based recommendation system [8], see section 3.3. Results of a concrete use-case of the proposed method, music recommendation using DBPedia, are discussed in detail in secion 4.

2. Evaluation of semantic similarity using RDF databases

This section gives an introduction to the RDF databases and presents possibilities of how to compute semantic similarity based on analysis of paths between objects.

2.1. RDF paths

RDF is a standarized data model designed to ease the exchange of information in the World Wide Web. Using RDF it is easy to seamlessly merge information from different sources, even if underlying database schemas are different [1]. RDF organizes data in triplets: subject, predicate and object. And object (last element) of one triplet can be a subject of a different triplet, which makes the data from an RDF database form a directed graph (see Figures 1 and 2)

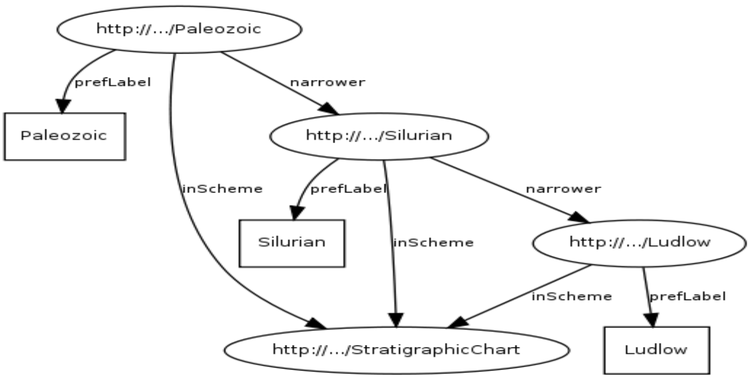


Figure 2. An example of a graph based on RDF [9]

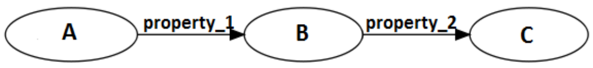


Figure 3. Example path between objects A and C

Since data in an RDF database forms a directed graph, paths may exist between two object from the database. These paths can be subject to further analysis. As an example, a path between object A and object C is shown in Figure 3.

An important factor about RDF data should be considered - one object may have more properties than other object (this can be compared to article length in Wikipedia). An **object centrality** is defined as the number of properties an object has. This characteristic is important for evaluating semantic similarity, and will be used in section 3.

2.2. Similarity evaluation using RDF path analysis

The simplest method of path analysis (originating in graph theory) is to calculate the similarity as a inverse of a length of a shortest path, which is expressed by equation 1 (e.g. see [10]).

$$S(A,B) = \frac{1}{min(l_i)} \tag{1}$$

where S is similarity between objects A and B, and l_i is the path length

However such simple analysis is not relevant to the characteristics of a typical RDF database. According to eq. 1, the similarity between two objects in the database does not depend on the number of paths of given length. Similarity evaluation should differentiate if there is one or a hundred paths of the same length.

Another approach of measuring distance using data in LinkedData is in given in [6], eq. 2. Here, the distance measurement is based on links, and the weight assignedd to a link type is based on the frequency of its occurence - if a certain link type exist rarely, it is regarded as more important, hence weight assigned to it is higher.

$$LDSD(a,b) = \frac{1}{1 + \sum_i \frac{D_d(l_i,r_a,r_b)}{1+log(D_d(l_i,r_a,n))} + \frac{D_d(l_i,r_a,r_b)}{1+log(D_d(l_i,r_b,n))}} \tag{2}$$

However, method expressed in equation 2 is completely agnostic of the links types present between resources.

Another method of path analysis is presented in [11]. Each predicate has an assigned weight corresponding to the importance of a given property in a domain (e.g. music). For every path the weights of all predicates are summed. Also, since the number of paths increases with increate of a path length, authors of [11] propose to divide a sum obtained for each path by the path length raised to the power of 3. This method can be summarized by equation 3 shown below. Equation 3 is introduced by the author of this paper, because in [11] only a computation algorithm is given. Eq. 3 presentes a method of measuring relatedness measure based on path analysis, while eq. 4 normalizes this measure to interval [0, 1].

$$M(A,B) = \sum_{i=1}^{n_{AB}} \left(\frac{\sum_{j=1}^{l_i} w_{ij}}{(l_i)^3}\right) \tag{3}$$

$$S(A,B) = \frac{1}{1 + M(A,B)} \tag{4}$$

where n_{AB} is a number of paths, i is path index, l_i is path i's length (number of edges in path), and w to $w_i j$ the weight of property in position j in path i (e.g weight equal to 3 for property *color*).

However, this method has three main disadvantages:

- necessity to manually define weights for properties, e.q. an expert in the given field, and in the used database (which properties are present, and how many of them, etc) must evaluate each problem

- hard to predict results, because weights are assigned arbitrarily, without analysis of the structure of data in the database (for example, number of paths). Also the fact that weight sum for each path is divided by path length to the power 3 is a general choice, not tailored for a particular case
- some objects in the database may be better described than others, which means some objects may have more properties. It follows that more paths may exist between two objects not becasue they are more related to each other, but because there objects have more properties associated with them

Because of these disadvantages a different method for path analysis is proposed, which is described in section 3.

3. Proposed method: RDF path analysis using genetic algorithms

The main idea presented in this chapter is inspired by the method presented in [11] (see equation 3), but addresses the disadvantages listed in section 2.2. Instead of using fixed predefined parameters, all are found using a genetic optimization algorithm. Additionally, object cetrality (see section 2.1) is incorporated into the equation, so the calcuated relatedness is not higher just because there are more properties describing an object.

3.1. Advantages of using genetic optimization for path analysis

The main contributions of the proposed method are listed below.

Automatic property and weight association No weights or properties (predicates) have to be set ad-hoc (manually preset by an expert). Instead, necessary function parameters are found using a genetic optimization algorithm, optimized on the basis of a training vector (a set of pairs with associated similarity). Thanks to this approach, there is no need to analyze what properties are important and with what weight.

Different weights of properties for different path lengths Properties (predicates) in each path length are treated independenly for evey path length. This means that the property may have different weights for different path length.

Associating weights to path lengths Weights are also asssigned for path lengts, which means that the method automatically detects which path lengths are most useful for relatedness calculation (e.g. it may turn out that paths of length 4 or more are useless for evaluation of similarity)

Incorporating object centrality into relatedness calculation As stated in previous chapter, existing methods of path analysis for RDF databases do not take the number of properties for an object into consideration, which leads to higher similarity between objects that are better described in the database. In the proposed method, each calcuated path weight is divided by an average of 'object centrality' (see section 2.1) of two analyzed objects

3.2. Two-phase genetic optimization of function parameters

Equation 5 presents the function used to evaluate relatedness measure based on path analysis between two objects in an RDF database. Eq. 6 normalizes this measure to ensure that similarity values are in the interval [0, 1].

$$M(A,B) = \sum_{i=1}^{n_{A,B}} w^P_{l_i}\left(\frac{\sum_{j=1}^{l_i} w^E_{l_i}(P^E_i(j))}{c_{A,B}^{\frac{l_i}{2}}}\right) \tag{5}$$

$$S(A,B) = \frac{1}{1 + M(A,B)} \tag{6}$$

where:
- l_i is the path length (number of properties in a path),
- $n_{A,B}$ is the number of paths between object A and B,
- $w^P_{l_i}$ is the weight assigned to a path length l_i,
- P^E_i is the path with index i,
- $P^E_i(j)$ is the property at index j of the path P^E_i, $(j = 1, 2, ..l_i)$
- $w^E_{l_i}(P^E_i(j))$ is the weight assigned to a property n the jth position in path for path length l_i

- $c_{A,B}$ is the mean object centrality of A and B

All required parameters are found using a two-phase genetic algorithm [12].Therefore a training vector stating the similarity between objects a_i and b_i is required. Each element in the training vector has the composition (s is similarity) $< a_i, b_i, s_i >$. Such general training vector U is shown in equation 7:

$$U = \{< a_1, b_1, s_1 >, < a_2, b_2, s_2 >, ..., < a_n, b_n, s_n >\} \tag{7}$$

Phase 1 The first phase of the genetic optimization algorithm is to find the weights for properties, so the parameters denoted by w_i^E in equation 5. Note that this optimization phase has to be done each time for each path length that is considered.
Therefore, in phase 1 of the genetic optimization algorithm the chromosome has the following form:

$$chr = \{\langle p_1, w_1^E \rangle, \langle p_2, w_2^E \rangle, ..., \langle p_i, w_i^E \rangle\} \tag{8}$$

Fitness function of the genetic algorithm is defined as minimalization of a variance between obtained similarity values and similarity values given in training vector U.

$$ff(chr) = \sum_{i=1}^{|U|} (U(a_i, b_i) - S_{chr_k}^{j}(a_i, b_i))^2 \tag{9}$$

where:
- $U(a_i, b_i)$ is the similarity from the training vector U for objects a_i and b_i
- $S_{chr_k}^{j}(a_i, b_i)$ is the calculated similarity values using function parameters from chromosome chr_k. Note that only one path length j is taken into cosideration in this case.

Phase 2 The second phase of the optimization algorithm is to find the weights of the path lengths, so the parameters denoted by w^P from equation 5. Therefore, the chromosome has the following form:

$$chr = \{\langle len_1, w_1^P \rangle, \langle len_2, w_2^p \rangle, ..., \langle len_i, w_i^P \rangle\} \tag{10}$$

where:
- len_i is the path length for which the weight has to be found

The fitness function for the second phase of optimization is defined the same as for the first phase, so by equation 9. Again the training vector U is used, and the function parameters w_i^E found in phase 1.

3.3. Application area: item-based recommendation

The proposed method of evaluating of similarity can be used in several areas. In this paper the application of evaluated simiarity for computing recommendation is described.

A set of objects from which the recommendations will be generated is given and is denoted as T (its elements are denoted as T_i). A specified number of elements from T is returned as the result of recommendation. Input to the recommedation engine is a set of items a user has graded. Such input set is denoted as S. An input pair (denoted by P) composes of an item (denoted by $Item$) and a grade (denoted by g)

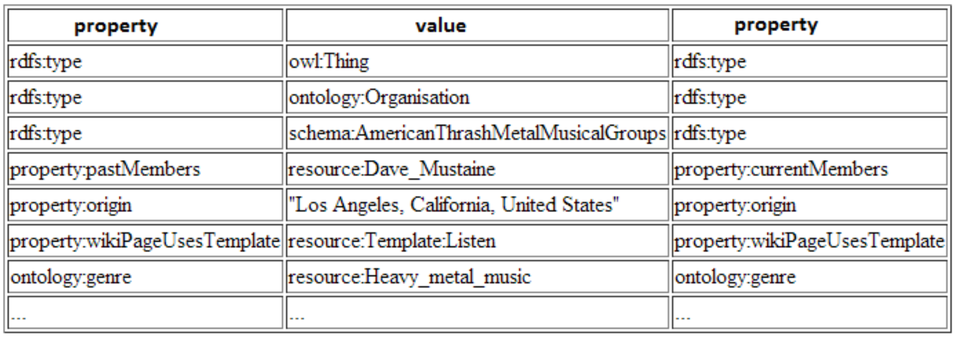
$$P_i = \langle Item_i, g_i \rangle$$

$$S = \{P_1, P_2, ..., P_n\}$$

For every element in T (set from which elements are recommended), the recommendation measure is calculated. Recommendation measure (denoted as M_i) is a weighted sum of similarity of items in T_i to all elements from the input set (E_I), where the weight is the grade (g_i):

$$M(T_i) = \frac{\sum_{j=1}^{|S|} S(T_i, P_j) \times g_j}{\sum_{i=1}^{|S|} g_j} \tag{11}$$

n elements from set T with the highest recommendation measure M are recommended.



property	value	property
rdfs:type	owl:Thing	rdfs:type
rdfs:type	ontology:Organisation	rdfs:type
rdfs:type	schema:AmericanThrashMetalMusicalGroups	rdfs:type
property:pastMembers	resource:Dave_Mustaine	property:currentMembers
property:origin	"Los Angeles, California, United States"	property:origin
property:wikiPageUsesTemplate	resource:Template:Listen	property:wikiPageUsesTemplate
ontology:genre	resource:Heavy_metal_music	ontology:genre
...	...	...

Figure 4. Some example paths of length 2 between bands Metallica (left side of the table) and Megadeth (right side of the table)

4. Case study - music recommendation using DBPedia

This section presents an example of application of proposed method for music recommendation using DBPedia. Firstly a the process of optimization, training data and obtained parameters is presented, afterwards a subset of obtained results are shown.

DBPedia is an extraction of structured information from Wikipedia's Infoboxes, which contain some data organized in the form key-value [13]. The main idea of DBPedia was that articles in Wikipedia are not easily processed by an artificial intelligence, while RDF is [14]. Therefore, this RDF database was used for the purpose of music recommendation, as in [2, 3, 4].

4.1. Evaluation of function parameters for similarity between musical artists

Paths between musical artists were extracted using an automatic generation of SPARQL queries, which were excuted in DBPedia's SPARQL endpoint. For each path length a set of queries had to be generated (section 2.1). As an example, some paths of lenght 2 for bands Metallica and Megadeth is shown in Figure 4.

Training set - LastFM extract Training set was obtained by extracting data from lastfm.com by means of a web crawler written in Selenium2 framework.

Table 1. LastFM class mapping

LastFM class	**associated similarity value**
Super similarity	0.95
Very high similarity	0.85
High similarity	0.75
Medium similarity	0.6
Lower similarity	0.4
Dissimilar	0

LastFM offers a possibility to get list bands that are similar to a given band. Similarity is given in 5 classes: Super similarity, Very high similarity, High similarity, Medium similarity and Lower similarity. For each of these classes a relatedness from a range 0-1 was associated. Additionally to 5 similarity classes given in LastFM, class called 'dissimilar' was introduced. This class to similarity mapping is presented in Table 1. The training set extracted from LastFM composed of 550 triplets.

Obtained function parameters for music similarity Table 2 presents some extracted properties and obtained weights for different path lengths, while table 3 show weights for path lengths

4.2. Evaluated similarity and recommendations

This section presents results for computation of semantic similarity between musical bands. Firstly, evaluated similarity from a well known musical artist, Rihanna, between other artists is presented (Table 4). Secondly, computed recommedations for a different set of bands is shown (Tables 5, 6). There were two groups of bands: input bands (set S, see section 3.3) ,composed of 3 bands (Metallica, Megadeth and Slayer) and target bands, composed of 200 bands (set T, see section 3.3). For every band from the input set a grade was assigned (see equation 11). Elements that exist in S are removed from T before starting computing recommendation.

Table 6 shows the generated recommendations for the input bands set and cal-

Table 2. Obtained weights for properties for given path length

path length	property (predicate)	weight
1	ontology/associatedMusicalArtist	22305
	ontology/associatedBand	165984
	property/associatedActs	110834
2	property/pastMembers	37
	ontology/formerBandMember	0
	property/currentMembers	1736
	background	9970
	genre	365
	property/associatedActs	890
	terms/subject	8588
	ontology/bandMember	2530
	ontology/associatedMusicalArtist	596
	property/genre	241
	ontology/recordLabel	9994
	property/background	9999
	type	5403
	associatedBand	170
3	...	...
	...	...

Table 3. Obtained path weights

path length	weight
1	0.01
2	0.67
3	0.29
4	0.03

culated recommendation measure (see section 3.3)

Table 4. Evaluated similarity from band Rihanna to other artists

artist1	**artist2**	**similarity**
Rihanna	Sean Paul	0.63
	Lady Gaga	0.56
	Britney Spears	0.48
	50 Cent	0.43
	Slayer	0.32
	Iron Maiden	0.03
	Sepultura	0
	Megadeth	0
	Pantera	0

Table 5. Input bands with grades, and evaluated similarity

input band (P_i)	**grade** (g_i) **for** P_1	**second band** (T_i)	**similarity R**(P_i, T_i)
Megadeth	9	MD.45	0.96
	9	Kreator	0.86
	9	Iron Maiden	0.85
	9	...	...
Metallica	8	Guns n' roses	0.93
	8	Iron Maiden	0.91
	8	Machine Head	0.86
	8	...	...
Slayer	7	Exodus	0.92
	7	Kreator	0.89
	7	Pantera	0.83
	7	...	...

4.3. Results discussion

As can be seen from section 4.1, weights calculated for particular properties may be surprising. Take for example the 'genre' property (from Table 2). Weight obtained for this property is very low (for both property types, ontology and property), while it could be expected that it is an important factor indicating if bands are similar or not (as assumed in [11]). The same conclusion can be applied to another property, *ontology/associatedMusicalArtist for path length 2* - obtained weight is

Table 6. A subset of computed recommendations

recommended band T_i	recommendation measure $M(T_i)$
Iron Maiden	0.87
Kreator	0.821
MD.45	0.713
...	...

small, but it could be expected that the weight would be high. The reason for such behavior is the construction of the database, and the number of properties objects have. This proves the advantage of optimizing weights other than using pre-defined ones - the database structure is unexpected, and assigning high weights to properties which may be regarded as important my not yield desired results.

An important conclusion can be drawn from Table 3 - the most relevant paths to analyze are paths of weight 2 and 3, while paths of length 1 and 4 can actually be neglected. The main reson why paths of length 1 are not informative is that there may actually be one type of path between such bands - *associatedBand* and alike. The number of such connections in the database is small, therefore this information is unreliable. Also, such connection cannot have assigned proper weight - it either is present or not. On the other hand, the main reason why paths of length 4 are not informative is that there is a lot of information noise in this case, and objects may be connected by very unmeaningful paths. The number of paths of length 4 between bands is very high, and the paths are often composed of some uninformative elements (e.g. 'locale_US').

Results in section 4.2 shows that the obtained results are meaningful. In Table 4 bands are sorted from most to least related. As can be seen in the table, there is only one partially wrong result - calculated similarity between artists Rihanna and Slayer is about 3, while it should be 0.

As presented in Tables 5 and 6, recommendation engine works properly. Based on out input set, so bands Megadeth, Metallica and Slayer, the top 3 of the recommended bands are Iron Maiden, Kreator and MD.45, which are of a very similar genre and music type.

5. Conclusions

This paper shows that data presented in RDF can be succesfully analyzed by means of adaptive path analysis. Paths were analyzed by assigning weights for different properties (predicates) in the graph. While such path analysis is not a new concept, this paper presents a method of genetic optimization of the assigned weights, based on a training vector. It was shown that such approach has two main adavantages: flexibility (the same algorithms can be used for any RDF database) and automation (only a training set is required, no weights or other function parameters have to be assigned manually). It is shown how such semantic relatedness can be used for computing recommendations.

A case study of computing relatedness and recommendation in the domain of music prooves usefulness of the proposed method. Obtained results, especially surprising weights of some properties (e.g. very low weight of 'genre' property), prove that is normally impossible to properly assign such weights manually.

Continuation of research may be focused on futher analysis of results (e.g. usability of function parameters in different domains), using more sophisticated genetic algorithms (like Hierachical Fair Competion, [15]) and investigating other methods of intelligent path analysis.

References

[1] Powers, S., *Practical RDF: Solving Problems with the Resource Description Framework*, O'Reilly, Beijing, 2003.

[2] Budanitsky, A. and Hirst, G., *Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures*, Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics, Pittsburgh, USA, 2001.

[3] Euzenat, J. and Shvaiko, P., *Ontology Matching*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[4] Rada, R., Mili, H., Bicknell, E., and Blettner, M., *Development and application of a metric on semantic nets.* IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 1, 1989, pp. 17–30.

[5] Bizer, C., Heath, T., and Berners-Lee, T., *Linked Data - The Story So Far*, International Journal on Semantic Web and Information Systems (IJSWIS), Vol. 5, No. 3, MarMar 2009, pp. 1–22.

[6] Passant, A., *Measuring Semantic Distance on Linking Data and Using it for Resources Recommendations*, 2010.

[7] Passant, A., *dbrec - Music Recommendations Using DBpedia.* In: International Semantic Web Conference (2), edited by P. F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Z. 0007, J. Z. Pan, I. Horrocks, and B. Glimm, Vol. 6497 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 209–224.

[8] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J., *Item-based collaborative filtering recommendation algorithms*, In: Proceedings of the 10th international conference on World Wide Web, WWW '01, ACM, New York, NY, USA, 2001, pp. 285–295.

[9] *Introduction to RDF, https://www.seegrid.csiro.au/wiki/Siss/SKOSandSISSvoc*, 2011.

[10] Alvarez, M., Qi, X., and Yan, C., *A shortest-path graph kernel for estimating gene product semantic similarity*, Journal of Biomedical Semantics, Vol. 2, No. 1, 2011, pp. 1–9.

[11] Leal, J. P., Rodrigues, V., and Queirós, R., *Computing Semantic Relatedness using DBPedia.* In: SLATE, edited by A. Simões, R. Queirós, and D. C. da Cruz, Vol. 21 of *OASICS*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012, pp. 133–147.

[12] Qing, L., Gang, W., Zaiyue, Y., and Qiuping, W., *Crowding clustering genetic algorithm for multimodal function optimization*, Appl. Soft Comput., Vol. 8, No. 1, Jan. 2008, pp. 88–95.

[13] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., and Ives, Z., *DBpedia: A Nucleus for a Web of Open Data*, In: In 6th Int'l Semantic Web Conference, Busan, Korea, Springer, 2007, pp. 11–15.

[14] Hebeler, J., Fisher, M., Blace, R., Perez-Lopez, A., and Dean, M., *Semantic Web Programming*, John Wiley & Sons Inc., Chichester, West Sussex, Hoboken, NJ, 2009.