Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 43 (2015) 53 – 61

Procedia
Computer Science

ICTE in Regional Development, December 2014, Valmiera, Latvia

Assessment of Name Based Algorithms for Land Administration Ontology Matching

Imants Zarembo[a]*, Artis Teilans[a], Aldis Rausis[b], Jazeps Buls[b]

[a]*Faculty of Engineering, Rezekne University of Applied Sciences, Atbrivosanas aleja 115, Rezekne, LV-4601, Latvia*
[b]*The State Land Service of Latvia, 11. novembra krastmala 31, Riga, LV-1050, Latvia*

Abstract

The purpose of this paper is to tackle semantic heterogeneity problem between land administration domain ontologies using name based ontology matching approach. The majority of ontology matching solutions use one or more string similarity measures to determine how similar two concepts are. Due to wide variety of available general purpose techniques it is not always clear which ones to use for a specific domain.
The goal of this research is to evaluate several most applicable string similarity measures for use in land administration domain ontology matching. To support the research ontology matching tool prototype is developed, where the proposed algorithms are implemented. The practical results of ontology matching for State Land Service of Latvia are presented and analyzed. Matching of Land Administration Domain Model international standard, present Latvian Land Administration ontology is conducted.

Peer-review under responsibility of the Sociotechnical Systems Engineering Institute of Vidzeme University of Applied Sciences

Keywords: Domain ontology; Edit distance; Land administration; Ontology matching; String similarity.

1. Introduction

The main goal of this research is to develop a land administration domain ontology that could be used to integrate relational databases of the land administration domain and to develop new information systems focused on interoperability and data interchange within a land administration domain.

* Corresponding author.
E-mail address: imants.zarembo@gmail.com

Peer-review under responsibility of the Sociotechnical Systems Engineering Institute of Vidzeme University of Applied Sciences
doi:10.1016/j.procs.2014.12.008

The authors of this paper undertake the task of automatically matching two land administration domain ontologies to assess their similarity and to determine compatibility. The purpose of this task is to determine how similar or how different the selected ontologies are, and to provide insight on how time and resource intensive integration of these ontologies would be. Automation of such a task is an important problem due to how labour intensive manual ontology matching using valuable time of domain experts is.

The first ontology is State Land Service of Latvia (SLSL) text part of cadastral database transformed to ontology. The second ontology is international standard ISO 19152:2012 – Land Administration Domain Model (LADM)[1] ontology created from data model presented in the standard.

The main issue with automatic ontology matching is the wide variety of available matching techniques and algorithms designed with a specific purpose in mind. Which means that algorithms with an excellent performance in one field may perform poorly in another. Therefore there is a need to find a specific set of techniques or methods which would provide best results for land administration ontology matching problem.

This paper describes the process and the results of automatic matching of the two ontologies and may be of interest to researchers who are confronted with a similar task.

2. Ontologies

2.1. Land administration domain ontology

ISO 19152:2012 standard defines a reference covering basic information – related components of land administration, provides an abstract, conceptual model with four packages (Fig. 1), terminology for land administration, and a basis for national and regional profiles and enables the combining of land administration information from different sources in a coherent manner[1].

The standard provides guidelines for those who want to implement this standard in their land administration information systems. Before implementing the standard it is necessary to determine compatibility between the data model described in the standard and the information system in question. One of the approaches to achieve this goal is to build an ontology for each of the data models and then to use ontology matching techniques to compare them. Matching result can help to understand what needs to be added or changed in their information system to conform to the standard.

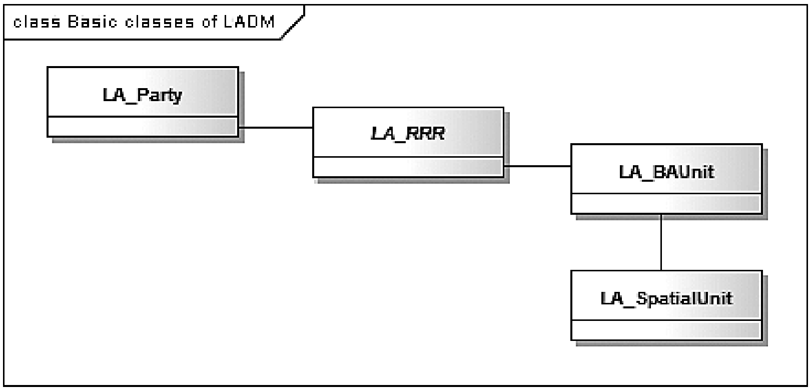

Fig. 1. LADM packages.

The authors of this paper have developed ontology using the conceptual model provided by ISO 19152[2]. Geographical profile UML diagrams, described in ISO 19103 - Geographic information - Conceptual schema language, were transformed to Web Ontology Language OWL 2 ontology using a specially developed tool.

The LADM ontology contains 68 classes, 131 object properties, 96 data properties and 135 individuals. For purposes of this research only ontology classes are used.

2.2. State Land Service of Latvia ontology

The State Land Service of Latvia text part of cadastral database contains many of the same concepts as LADM, but they often are implemented in different ways and named differently. Applying name based ontology matching techniques can help finding the same concepts that are simply named slightly differently.

The authors of this paper have developed ontology using cadastral information system data model provided by SLSL. Database data model specification was converted to OWL 2 ontology using a specially developed tool. An automatic solution was more feasible in terms of labor intensity due to size of the database.

The ontology contains 248 classes, 1001 object properties and 945 data properties. Only ontology classes are used for purposes of this research.

3. Related work

There is considerable amount of research available on the topic of name based ontology matching. Stoilos, Stamou & Kollias (2005) attempt to solve a similar problem of working with string distance metrics that have been developed for different applications[3]. The difference in application can often lead to poor performance when applied to a new domain.

Cohen, Ravikumar & Fineberg (2003) compare edit-distance, token-based, hybrid distance and pruning methods by using several datasets[4]. The datasets contain such fields as first name, last name, house number, street etc. Cheatham & Hitzler (2013) use wide range of metrics on six different datasets all describing the same domain of conference organization[5].

Importance of pre-processing ontologies before matching is emphasized in paper by Behkamal, Naghibzadeh & Moghadam (2013), where the authors propose pre-processing approach to get a better result from matching process[6].

4. The pre-processing

The aim of pre-processing is to improve matching results by making sure concepts to be matched are prepared for further use. During the pre-processing phase concept names are stripped of any unnecessary or redundant data for the given purpose.

Pre-processing approaches can be divided into two major categories[7]: syntactic and semantic. Syntactic pre-processing methods are based on characters in the strings. Semantic methods take into account meanings of the strings.

To improve matching results for ontologies with different concept naming styles, the authors of the paper propose to use syntactic pre-processing methods: normalization and tokenization.

4.1. Normalization technique

Normalization techniques are used to reduce strings to be compared to a common format. Given the task at hand the following normalization techniques are selected to be used in the matching process:

- Case normalization, to convert alphabetic characters into their lower case counterparts;
- Blank normalization, to remove all blank characters;
- Diacritics suppression, to replace characters with diacritic signs in them;
- Link stripping, to replace apostrophes, blank underlines with blanks;
- Punctuation elimination, to remove punctuation signs.

4.2. Tokenization technique

Tokenization is a linguistic pre-processing technique that consists of segmenting strings into sequences of tokens by dividing strings with blank characters, cases (camel case, Pascal case etc.), digits etc. Tokenization is the most useful when naming conventions differ between the ontologies. For example, in one of ontology words in concept names are separated using camel case, but in another using underscores.

The practical observations of authors indicate that adding pre-processing to ontology matching task can give up to around 20% percent better results with the researched ontologies. Using the tokenization in combination with a normalization allows to improve the pre-processing phase results.

5. Matching problem

Ontology matching is the problem of finding the semantic mappings between given ontologies by determining correspondences between concepts.

According to the classification of matching approaches[7] there are at least the following concrete matching techniques: formal resource-based, informal resource-based, string-based, language-based, constraint-based, graph-based, instance-based and model-based matching techniques. A concrete technique classification is based on the way in which techniques interpret the input information.

For the purpose of the research string-based ontology matching techniques are used. String-based methods use the structure of the string to determine the similarity.

The following matching techniques are used to solve the automatic ontology matching task of SLSL and LADM ontologies:

- Levenshtein distance[8];
- Jaro–Winkler distance[9];
- Monge Elkan algorithm[10;]
- Longest Common Substring[11].

These techniques are well suited for ontology matching, due to their property to consider ontology entities or instances in isolation with other entities or instances.

Aforementioned four techniques will be examined in this chapter.

5.1. Levenshtein distance

Levenshtein distance (also known as edit distance) is a string metric for measuring the differences between two strings. The distance is the smallest number of insertions, substitutions and deletions required to change one string into another. The larger the Levenshtein distance, the more different the strings are.

Mathematical representation of Levenshtein distance is:

$$d_{(a,b)}(i,j) = \begin{cases} max\,(i,j) & if\ min(i,j) = 0 \\ min \begin{cases} d_{a,b}(i-1,j) + 1 \\ d_{a,b}(i,j-1) + 1 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & otherwise \end{cases} \tag{1}$$

where

- $1_{(a_i \neq b_j)}$ – indicator function equal to 0 if $a_i = b_j$, otherwise 1.

Levenshtein distance was selected because it is widely known and used string metric to serve as a baseline for other metrics used in this research.

Levenshtein distance has several restrictions:

- The distance is 0 if two strings are equal;

- The distance is at most the length of the longest of two strings;
- The minimum distance is at least the size of the difference between two string lengths.

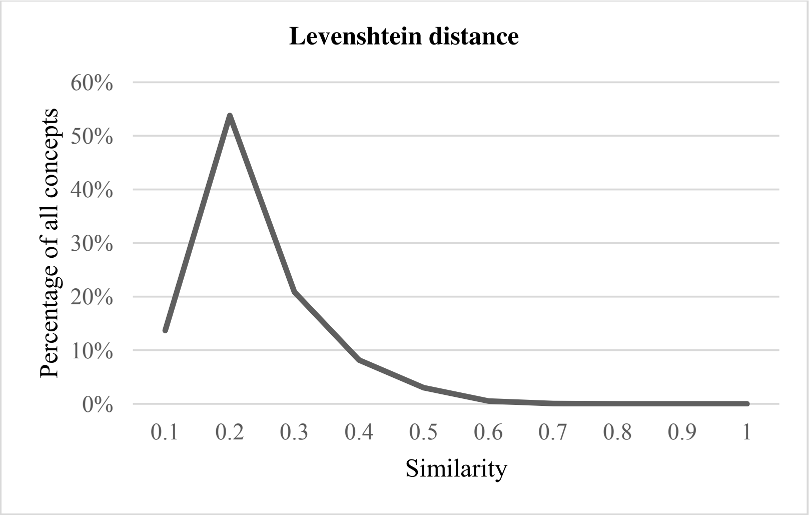

Fig. 2. Matching result using Levenshtein distance.

Levenshtein distance showed average similarity of 0.2 and similarity for the most concepts was 0.2 (Fig. 2).

5.2. Jaro-Winkler distance

Jaro-Winkler distance is a measure of similarity between two strings, which was developed for data linkage and duplicated detection.

The Jaro measure is the weighted sum of percentage of matched characters from each set of characters and transposed characters. Winkler increased this measure for matching first characters in the string, then rescaled it by multiple sub functions. Intervals and weights of the sub functions depend on the type of the string.

If c > 0 (if $c = 0$, then $\Phi = 0$), the Jaro string comparator mathematical representation is

$$\Phi = W_1 \times \frac{c}{d} + W_2 \times \frac{c}{r} + W_t \times \frac{c - \tau}{c} \tag{2}$$

where

- Φ – distance between two strings;
- W_1 – weight associated with characters in the first of two sets;
- W_2 – weight associated with characters in the second of two sets;
- W_t – weight associated with transpositions;
- d – length of string in first set;
- r – length of string in second set;
- τ – number of transpositions of characters;
- c – number of characters in common in pair of strings.

The number of transpositions is computed as follows[9]: The first assigned character in the first string is compared to the first assigned character in the other string. If the characters are different, it is assumed that half of a transposition has occurred. The same procedure is repeated on the following characters in both strings. The number of mismatched characters is divided by two to get the number of transpositions.

If two strings are the same, Jaro string comparator Φ is set to $W_1 + W_2 + W_t$ which equals to 1. If two strings have no characters in common Φ is equal to 0.

In Winkler's modified Jaro string comparator W_1, W_2 and W_t are arbitrarily set to $\frac{1}{3}$. The new string comparator metric checks whether the first few characters in the string being compared agree:

$$\Phi_n = \Phi + i \times 0.1 \times (1 - \Phi), \tag{3}$$

if the first i characters agree.

Mathematical representation of Jaro-Winkler distance is:

$$d_j = \begin{cases} 0 & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & otherwise \end{cases} \tag{4}$$

where

- d_j – Jaro distance between two strings;
- m – the number of matching characters;
- t – half the number of transpositions.

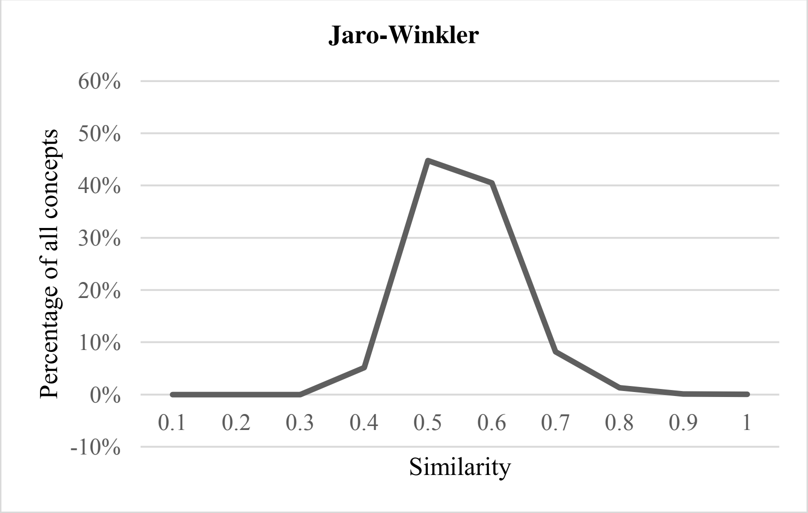

Fig. 3. Matching result using Jaro-Winkler distance.

Jaro-Winkler distance showed the best results amongst compared algorithms with average similarity of 0.5 and similarity for the most concepts was 0.5 – 0.6 (see Fig. 3). Jaro-Winkler distance was the only metric which identified some of the concepts to be equal (similarity of 1.0).

5.3. Monge-Elkan distance

Monge-Elkan is a general text string comparison method based on tokens and internal similarity function for tokens that finds the best match for each token. In this context tokens are character sequences that are split into words, as it is the case with most human languages.

The algorithm uses the recursive structure of typical textual fields[10]. Two strings match with degree 1.0 if they are the same atomic string or one abbreviates the other. Otherwise their degree of match is 0.0.

Each subfield of the first string is assumed to correspond to the subfield of the second string with which it has the highest score.
The score of matching two strings then equals to the mean of these maximum scores:

$$match(A,B) = \frac{1}{|A|}\sum_{i=1}^{|A|} \max_{j=1,|B|} match'(A_i, B_i) \tag{5}$$

where

- $|A|$ − the number of tokens in A;
- $match'$ − an internal matching algorithm.

If strings contain just one token each:

$$match(A,B) = match'(A,B) \tag{6}$$

Matching of abbreviations uses four patterns:

- The abbreviation is a prefix of its expansion;
- The abbreviation combines a prefix and a suffix of its expansion;
- The abbreviation is an acronym for its expansion;
- The abbreviation is a concatenation of prefixes from its expansion.

The algorithm has quadratic time complexity. Given two strings, every subfield in the first string must be compared with every subfield in the second.

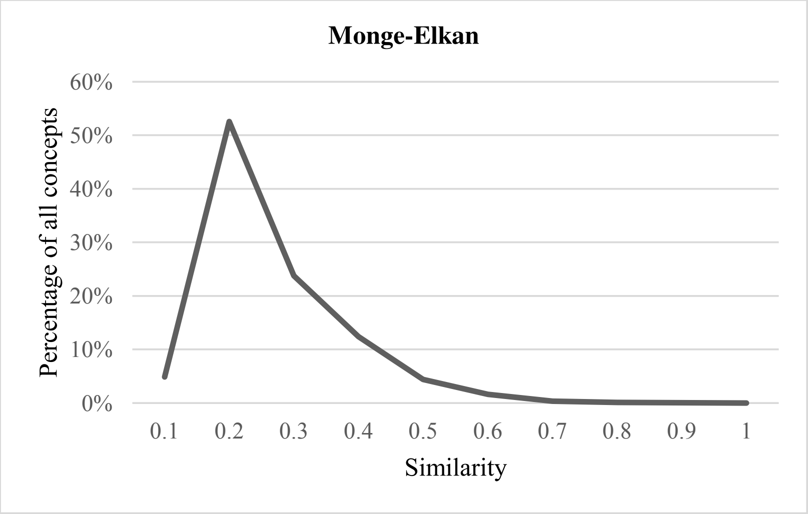

Fig. 4. Matching result using Monge-Elkan distance.

Monge-Elkan distance showed better results than Longest common substring or Levenshtein distance results, but was worse than Jaro-Winkler distance with average similarity of 0.2 and similarity for the most concepts was 0.3. As can be seen in figure 2 and 4, Monge-Elkan and Levenshtein distance results have very similar distribution graphs while being different from all other compared algorithms.

5.4. Longest common substring

Longest common substring is the longest substring of two or more strings. If two strings under consideration are identical substring similarity is 1 and if they are dissimilar substring similarity is 0.
To measure exact dissimilarity the following substring similarity measure can be used:

$$\sigma(x, y) = \frac{2|t|}{|x| + |y|} \tag{7}$$

where

- σ – substring similarity measure;
- t – the longest common substring;
- x, y – strings under consideration.

In the best case scenarios we are looking for string pairs, which contain one another or both strings are exactly the same.

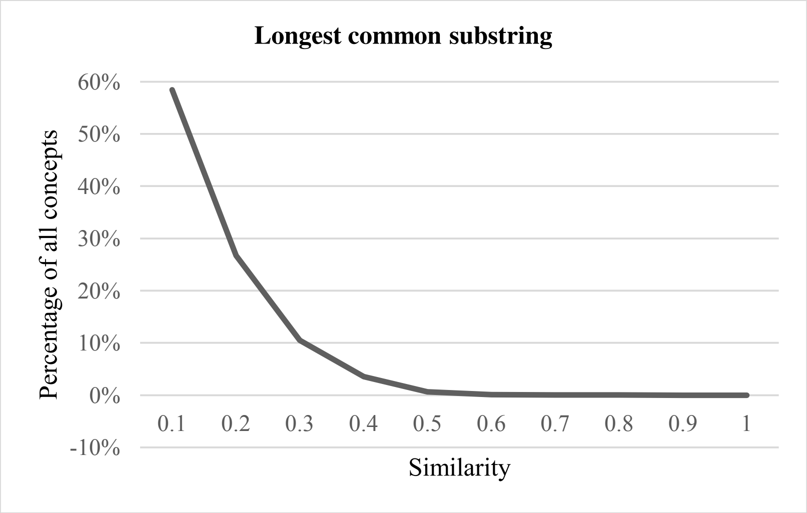

Fig. 5. Matching result using longest common substring technique

Longest common substring technique showed worse results than Levenshtein distance on average resulting in similarity of 0.1 and similarity for the most concepts was 0.1 (see Fig. 5).

6. Results and discussion

Experiments were conducted on SLSL and LADM ontologies using specially developed ontology matching tool prototype with Levenshtein distance, Jaro-Winkler distance, Monge-Elkan distance and Longest common substring name based ontology matching techniques and normalization and tokenization techniques implemented.

In the first step both ontologies were prepared for matching using pre-processing. In this phase concept names were stripped of any unnecessary details using normalization and tokenization techniques. In the second step matching algorithms were executed on every concept pair in the first ontology and in the second ontology. In cases where the algorithm outputs edit distance between two strings, the results were converted to similarity. Finally the results were summarized and analyzed to determine performance of each algorithm using matching precision as the criteria. The measure is a real number in range between 0.0 (strings are different), and 1.0 (strings are the same).

When working with real world ontologies they can contain less than perfect structure and entity names. This is the case with SLSL ontology, which has been transformed from relational database model. For this reason name based ontology matching techniques are not guaranteed to produce the desirable threshold of similarity (for example, 0.75 of higher). Nonetheless the results showed that Jaro-Winkler distance was able to come the closest to the threshold. In SLSL database all entities were normalized in accordance with relational database construction techniques. Denormalizing the entities could lead to improved matching results.

7. Conclusion

In this paper the authors have proposed and approbated the methodology of matching two ontologies of the same domain: from relational database created ontology and from domain standard created ontology.

The results of the proposed automatic ontology matching showed that for land administration ontologies the best results were provided by Jaro-Winkler distance. The main obstacles in automatic SLSL and LADM ontology matching is the fact, that SLSL ontology was created from relational database and has normalized data structures in comparison with LADM ontology which consists of abstract data model entities without normalization. Another reason for differences is the use of inconsistent terminology across ontologies. Name based algorithm performance may be suboptimal when dealing with different words, which have the same or similar meanings.

Further work on this subject includes creation of land administration domain ontology for State Land Service of Latvia which would be able to answer the following questions: What data do we have? What implied relationships does our data have? Can we interchange our data with another system or standard?

References

1. ISO 19152:2012 - Geographic information - Land Administration Domain Model (LADM). Retrieved: 2014.10.30, URL: http://www.iso.org/iso/catalogue_detail.htm?csnumber=51206.
2. Zarembo I., Kodors S., Automatic generation of OWL Ontologies from ISO 19103 Geographic Information profile UML Class Diagrams. Virtual Multidisciplinary Conference (QUAESTI 2013), ISBN: 978-80-554-0826-2, Slovak Republic, 2013. p. 225-229.
3. Stoilos G., Stamou G., Kollias S., A String Metric for Ontology Alignment. ISWC'05 Proceedings of the 4th international conference on The Semantic Web, Berlin, Heidelberg, 2005, p. 624-637.
4. Cohen W. W., Ravikumar P., Fienberg S. E., A Comparison of String Distance Metris for Name-Matching Tasks. In Proceedings of IJCAI-2003 Workshop on Information Integration on the Web, 2003, p. 73-78.
5. Cheatham M., Hitzler P., String Similarity Metrics for Ontology Alignment. The Semantic Web–ISWC 2013, 2013, p. 294-309.
6. Behkamal B., Naghibzadeh M., Moghadam, A., Pre-processing Ontologies to Improve The Results of Matchers. IJST, Transactions of Electrical Engineering, Vol. 36, No. E2, p. 95-108.
7. Euzenat J., Shvaiko P., Ontology Matching, 2nd edition. Springer-Verlag, Berlin Heidelberg, ISBN: 978-3-642-38721-0, 2013.
8. Levenshtein V. I., Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10 (8), 1966, p. 707-710.
9. Winkler W. E., U.S. Bureau of the Census, String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. Proceedings of the Section on Survey Research Methods (American Statistical Association), 1990, p. 354–359.
10. Monge A. E., Elkan C. P., The webfind tool for finding scientific papers over the Worldwide Web. In Proceedings of the 3rd International Congress on Computer Science Research, 1996, p. 41-46.
11. Gusfield D., Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology. USA: Cambridge University Press, ISBN:0-521-58519-8, 1999.