OMT, a Web-Based Tool for Ontology Matching

João Rodrigues Gomes[1]
Centro ALGORITMI, Departamento de Informática, University of Minho, Braga, Portugal

Alda Lopes Gançarski
Centro ALGORITMI, Departamento de Informática, University of Minho, Braga, Portugal

Pedro Rangel Henriques
Centro ALGORITMI, Departamento de Informática, University of Minho, Braga, Portugal

Abstract

In recent years ontologies have become an integral part of storing information in a structured and formal manner and a way of sharing said information. With this rise in usage, it was only a matter of time before different people wrote distinct ontologies to represent the same knowledge domain. The area of Ontology Matching was created with the purpose of finding correspondences between different ontologies that represented information in the same domain area. This paper starts with a study of already existing ontology matching methods in order to understand the existing techniques, focusing on the advantages and disadvantages of each one. Then, we propose an approach and an architecture to develop a new web-based tool using the knowledge acquired during the bibliographic research. The paper also includes the presentation of a prototype of the proposed tool, called OMT.

2012 ACM Subject Classification Information systems → Web Ontology Language (OWL); Computing methodologies → Ontology engineering

Keywords and phrases Ontology, Ontology Matching, Ontology Alignment

Digital Object Identifier 10.4230/OASIcs.SLATE.2022.8

Funding This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

1 Introduction

In computer science, the concept of an ontology was firstly introduced in [7] as being a model to store data within a domain that allows the representation and definition of concepts, often referred to as classes, their properties and the existing relations between them. An ontology knowledge base is the result of populating an ontology with data that matches the concepts and properties that have been formally defined.

Ontology Matching methods were created with the purpose of relating information coming from multiple heterogeneous ontology sources into a common ontology model that encapsulates the entire knowledge base from all the sources [11]. On a simpler level, these methods have the task of finding correspondences between ontologies. The same concept or property can be defined using different terminologies on different ontologies.

There has already been a lot of research and work done in the field of Ontology Matching, and currently there are many approaches available that automatically generate matches between Ontologies. However, these techniques are still far from being perfect due to the fact that human input is still needed when the use case requires an accurate matching, making these methods impractical when dealing with big and complex ontologies [6].

This paper will focus, firstly, on the study of the already existing techniques, analysing the advantages and disadvantages of each one, and, secondly, on the presentation of a new web-based approach to create a tool that relies as less as possible on human input.

[1] corresponding author

To accomplish that twofold objective, the paper is structured in five sections. After the Introduction and before the Conclusion (Section 5), Section 2 defines the problem, Ontology Matching, presents the state-of-the-art in this area, and discusses related work found in the literature review. Then Section 3 proposes a solution to develop OMT, our web-based tool to match ontologies. The architecture of the proposed system is depict and its modules explained. In Section 4, OMT prototype developed is presented.

2 Ontology Matching, related work

Ontology matching is the process of relating information from heterogeneous sources into a common model that can be queried and reasoned upon. On an abstract level, ontology matching is the task of finding correspondences between ontologies identifying a similar concept, property or relationship defined in both ontologies however using different terminologies.

Despite the efforts that have already been made in creating approaches to match ontologies, there is still a lot of room to grow regarding their ability to work in an autonomous way. Usually, these approaches evaluate ontologies given as input and come out with a group of possible correspondences. These correspondences have then to be examined by a person to determine which ones are correct, remove the false positives and create additional correspondences that were missed [6], making this whole process impractical when dealing with big and complex ontologies.

2.1 Matching Techniques

The approaches developed to solve the matching problem all make use of different techniques that each, in its own way, aims to exploit the information present in the ontology.

There are multiple criteria of classification that one can use to group up these approaches [11].

One simple way to divide these approaches is based on whether they analyze the information orthographically (string similarity) or semantically.

On the one hand, analysing information orthographically means that only the strings are important and taken into account. Many string techniques are available for that approach. On the other hand, analysing information semantically means that the meaning and context of the words are also taken into account. Obviously these approaches are not mutually exclusive and are used together in many different tools to achieve better results.

The following subsections take a closer look at some techniques used in these approaches.

2.1.1 String Based Techniques

In this category all the techniques analyse the information orthographically. They all measure the similarity of the Strings used to represent the concepts, properties, labels, comments and relationships of the ontology.

Distance functions map a pair of strings (x, y) to a real number r, where the smaller the value of r, the greater the similarity between the string pair [3]. There are many different implementations of these functions. One of the most important ones is the group of edit distance functions which measure the distance between two strings by calculating the most efficient sequence of pre-established operations that convert one string into the other. These pre-established operations are usually character insertion, deletion and substitution.

Another type of distance functions are the ones based on tokenization which is convert every string into a set composed of all its words (tokens). Afterwards, the similarity of the two strings is calculated by comparing the set of tokens that corresponds to each one. A simple metric is the *Jaccard Index*. Being (S,T) the respective sets of tokens of the string pair (s,t), the Jaccard Index is obtained by the following formula:

$$Jac(s,t) = \frac{|S \cap T|}{|S \cup T|}$$

The Jaccard Index calculates the similarity of a pair of strings taking into account the relation between the number of common tokens in regards with the whole set of tokens.

One last type of distance functions are the called Hybrid functions which make use of different techniques. For example, the **Monge-Elkan Similarity Function** proposes a recursive approach to this problem, that is best used when comparing two long strings. This technique splits both strings into a set composed of all their sub-strings. It then compares every sub-string of one set with every sub-string of the other set, only saving the maximum value calculated through every iteration. It then computes the average of every maximum value as the value for the distance of two strings. Given a string pair (s,t), let K and L the amount of sub-strings respectively, and **sim** a generic function to calculate the similarity between two strings, the Monge-Elkan Similarity Function is given by the following expression:

$$ME(s,t) = \frac{1}{K}\sum_{i=1}^{K} max_{j=1}^{L} sim(S\_i, T\_j)$$

The Jac and ME are some of the many functions and approaches possible based on strings. This type of techniques is usually the first one to be used in a system in order to find the matching concepts, properties and relationships that are syntactically defined in a similar way.

2.1.2 Language Based Techniques

While, in the previous category, the techniques analyse the information of the ontologies syntatically, the techniques in the language based category analyse the information semantically. This means that the concepts, properties, labels, comments and relationships are not analysed as mere strings but are analysed as words that have a meaning in some language. Therefore, a lot of the techniques used rely on Nature Language Processing (NLP) and on the use of external sources such as dictionaries, lexicons and databases [11].

This approach makes use of different techniques such as **tokenization**, which, as has been explained previously, is the act of decomposing a String into smaller parts that compose it. In the specific case of language based techniques, tokenization is used to identify the words (tokens) that exist in the input string in order to better understand and exploit the possible meaning a word can have in the domain it is being expressed in.

Lemmatisation is another technique that is used. It is the process of finding the **lemma** that corresponds to the word that is being analysed. By definition, a lemma is the canonical form of a word and is considered the base form of a whole set of words that derive from it. For example, considering the set of words `walk`, `walking`, `walked`, `walkabout` all derive from the word `walk` and therefore **walk** is lemma of this set of words. One of the most common ways to find the lemma of a word is by looking it up in a dictionary.

Another technique also used is often referred to as **stop-word elimination**. This process is not as well defined as the other previous two and it depends on who is using it and in what context. It consists on creating a list composed of words that are considered not useful in the context of the problem and are therefore filtered out on the pre-processing stage of the string analysis phase.

These techniques are used to prepare the information being analysed. After they are applied, the resulting terms (`tokens`, `lemmas`) are then compared to check for their similarity. If they are not similar, then dictionaries and thesaurus are used to check if the terms have the same meaning, i.e if they are synonyms.

Language based techniques are some of the most important because they can find matches that, for example, the string based techniques do not. When dealing with ontologies written in different languages, these are the techniques that are mostly used, making use of dictionaries for example. Given that most matches in an ontology matching problem are not going to be orthographically equivalent, these techniques are in great use and hence is why tokenisation and lemmatisation are still being worked on and perfected nowadays.

2.1.3 Constraint and Instance Based Techniques

Constraint based techniques try to exploit the information of the contraints that can be placed on the properties and relationships of an ontology, such as the `Type` of data properties or the `Domain` and `Co-Domain` of relationships. This approach is based on the idea that, if properties and relationships on different ontologies match on the level of the constraints, then they are a potential match that needs to be studied. This technique is harder to be used alone and is mostly used in combination with other techniques to generate potential matches or analyse already generated matches.

Instance based techniques are considered an extension of all the techniques already presented, because they make use of the individuals when dealing with populated ontologies. Despite having the drawback of a much bigger quantity of information to analyze and consequently making the whole process take more time and use more resources, these type of techniques are great at both confirming already potential matches and generating new possible matches. When generating new matches, the general strategy is that if two individuals are alike, then the concepts they belong to are probably alike as well. When confirming potential matches, the general strategy is that if two concepts are alike and to be matched, then their individuals should also be alike [11].

2.2 Existing Ontology Matching Tools

In the following subsections, some of the many already existing ontology matching tools are presented focusing on the techniques each one uses as well as some of the restrictions the tools may have.

2.2.1 AROMA

AROMA (Association Rule Ontology Matching Approach) [5], is a tool for matching web directories, catalogs and ontologies designed in the OWL language.

The main technique used to find and reason matches is the **Association Rule Paradigm**. This paradigm focuses on the creation of implications/rules of the type x —> y that should be interpreted as: “if a term x is found on a given schema of an ontology, then that ontology may be associated with the concept y”.

Having this rule paradigm in mind, the first step in AROMA's algorithm is the extraction and selection of relevant terms associated to each concept. Then the process of creating and associating different rules to form matches in the different schemas is done. This is a simple overview of the methodology used when dealing with web directories or catalogs.

When the input schemas are those of an OWL ontology, the first step of the process had to be adapted. Rather than only using information contained in the schema of the ontology, the individuals data is also considered.

Although they do not go much in detail on the specific techniques used in each step of the algorithm, it is possible to deduce that language based techniques were probably used, especially tokenization to perform the extraction of the relevant terms.

This tool is not adequate to use in a general problem as it was developed with a specific target in mind, only allowing ontologies written in OWL to be used as input.

For a more in depth explanation of the methodology used in the AROMA tool, refer to [5].

2.2.2 AUTOMS

AUTOMS is a tool designed for the automatic matching of domain OWL ontologies [10]. As a tool, it integrates multiples methods that must be run in a particular sequence in order for the matching to be sucessful.

Firstly, a string based method is applied. It uses information concerning names, labels and comments of the ontologies concepts and properties and computes their similarity. The similarity between two terms is computed by making use of methods that compare the typographic similarities of Strings, sub Strings, sequences of ASCII characters and calculates how similar they are. It then proceeds to create pairs of possible matchings.

Secondly, WordNet is used to acess if any of the pairs created are synonyms.

Following these first two methods, the string based and language based methods are combined into a single structure to determine their similarities in regards to concepts and properties. The individuals of the ontology are then considered to help find matchings for concepts that have not been determined to be similar in the input ontologies.

This tool is a perfect example how some of the techniques explained in the previous sections can be used in combination to each other in order to create an algorithm that yields good results. It uses syntactic and semantic based techniques, followed by instance based techniques to generate and confirm possible matches.

For a more in depth explanation of the entire methodology or for any individual methods used, refer to [10].

2.2.3 Hertuda

Hertuda is an ontology matching tool [9] designed to only accept as input ontologies compatible with the Lite or DL versions of the OWL language. It was developed to be a very simple matcher that takes advantages of tokenization and string measure to obtain alignments. It handles concepts, relations and properties independently, which results in three different sets of results, one for which.

For each concept, all its labels, comments and URIs are extracted, forming a set. To then compare concepts, its respective sets are compared, resulting in a similitary measure value. Before this comparison takes place, all the terms in all the sets are subjected to a pre-processing step, where tokenization occurs.

This is a much simpler tool compared to the others already studied. Its purpose was to push to its limits the string based techniques and the String similarity functions in order to obtain matches. It can be used for comparison and analysis but it will fail in cases where the terminology used in the matching ontologies is very different.

2.2.4 AgreementMaker

AgreementMaker [4] is one of the most developed and matured tools in this area as since its initial publication, it has been enhanced and improved upon multiples times, having been written follow up articles every single time.

One of its advantages is the amount of customization is offers the users in terms of its matching methods used, conceptual or structural methods, the amount of user interaction they require and if only the schema should be considered or the individuals as well. It also has built in metrics such as precision and runtime to present to the user.

The first group of matching algorithms include string-based and language based methods that compare labels, comments, annotations and instances. The second group of algorithms try to exploit the information present in the structure of the ontologies, such as the concepts, properties and the relations that exist amongst them. The final group combines all the results obtained in the first two groups with the goal of obtaining a unique final matching.

After the process of matching is done, Agreement Maker also offers automatic methods of evaluating the final matches. It considers the most effect evaluation technique to be the comparison of the matching found by the tool against a *gold standard* of the domain the ontologies are included in, preferably built by domain experts.

In the Figure 1 can be seen the interface of this tool.

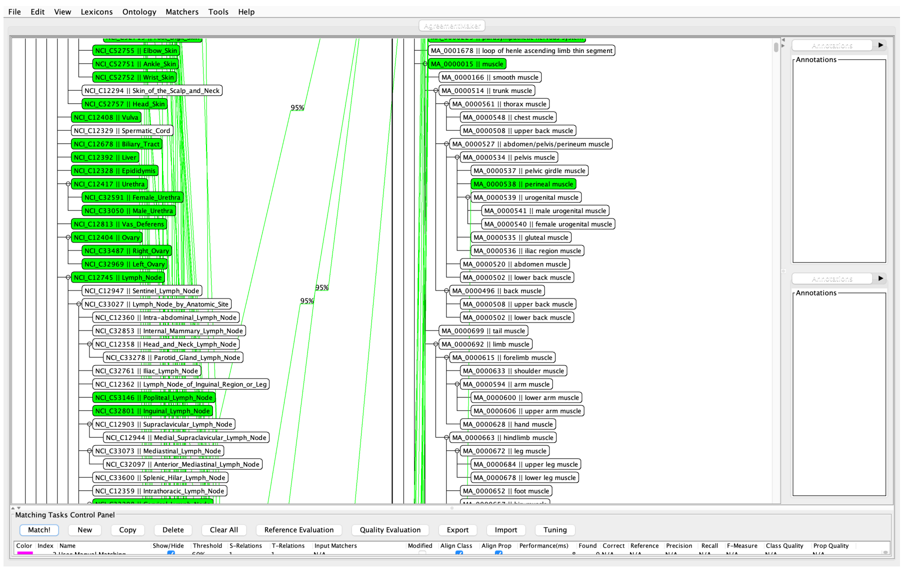


Figure 1 AgreementMaker Tool.

For further detail on this tool, refer to its original publishing article [4] and its source code is publicly available in its github repository.

2.2.5 MEDLEY

MEDLEY [8] is an OWL ontology matching system that takes a different approach from the tools that have been covered. It transforms every ontology triple into a graph structure, with the nodes being the ontology's concepts, properties and individuals and links being the relations that links them.

The techniques used are mostly based on string based techniques and language based techniques, such as the similarity between strings, tokenization and stemmatisation while also making use of dictionaries to find equivalences.

The base logic of this method is that, if a given entity is alike an entity that is already in the graph structure, then the neighbours of that entity must also be neighbours of the given entity, generating possible matches.

For more details on this system, refer to [8].

2.2.6 Summary

In this subsection, already existing ontology matching tools are presented. A summary of the analysis made can be seen in Table 1. A last column was added to compare our tool, explained in the next section, with existing ones.

Table 1 Ontology Matching Tools – Comparison.

	Ontology Matching Tools					
	AROMA	AUTOMS	Hertuda	AM	MEDLEY	OMT
multiple input formats	x			x		x
syntactic techniques		x	x	x	x	x
semantic techniques	x	x		x	x	x
association rules techniques	x					x
graphs structure					x	
evaluation methods				x		x

3 OMT, proposal and architecture

This section provides an initial architectural design of the ontology matching tool we propose as well as a more detailed explanation of the different components that make it up.

Figure 2 represents the architecture of the web-based ontology matching tool desired.

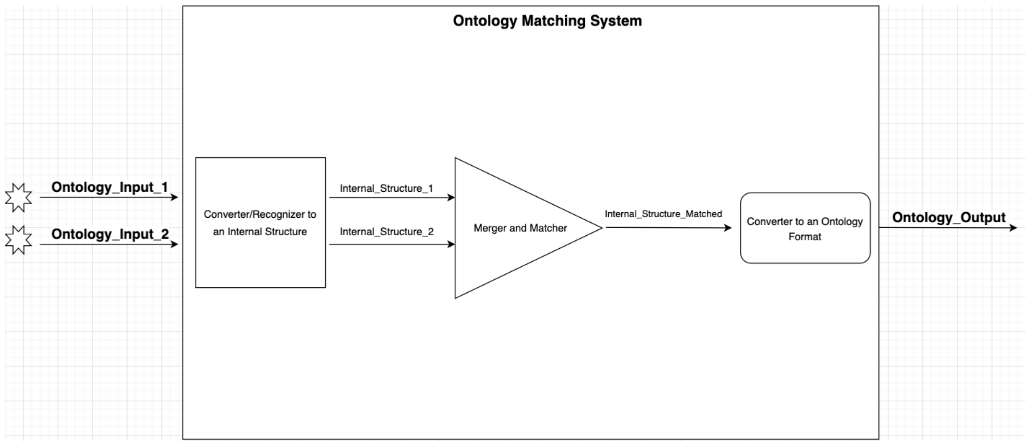


Figure 2 General overview of the system architecture.

The two most important components of the matching tool are the initial converter (Step 2) and the Merger/Matcher (Step 3), as they are the ones that convert the initial ontologies into an homogeneous format to be analysed in order to find matches.

In the following sections, it will be given a more in depth look at each module of the process.

3.1 Ontology Matching tool inputs

The inputs of an ontology matching tool are a critical part of it, as they determine what and who can use that tool. The majority of the tools already created for this purpose restrict the format of the input ontologies, limiting the group of potential users of the tool. One of the goals for this tool is to be adaptable and easily accessible. Adaptable in the sense that the tool applies different techniques depending on the contents of the ontologies and easily acessible since it will be a web-based tool, making it available to anyone that wishes to use it. With that mind, we intend for the tool not to restrict the format of the input ontologies to a specific language and instead accept different formats such as the OWL language family [1] and Turtle [2]. The input ontologies can also be written in two different natural languages: english or portuguese.

3.2 Converter/Recognizer

The Converter/Recognizer is the first module of the process this tool is going to use. As mentioned before, the two input ontologies may be written in different ontology or natural languages. Therefore, this step is crucial in homogenizing the information present in the ontologies into an internal structure (an intermediate neutral ontology representation) that will hold the information relevant of each input ontology. If the ontology content is written in portuguese, an extra step is necessary to translate the respective information to english before starting the next step.

3.3 Matcher and Merger

The Matcher and Merger is the most important block. It is where the matching techniques are chosen and applied in order to find alignments in the two input ontologies.

This component can be broken into two smaller parts, the Matcher and the Merger. The Matcher finds the correspondences between the two input ontologies. The Merger takes all the concepts and relations from the given ontologies that match (i.e., all the elemenst that are discovered as common or belonging to both inputs) and joins them with all the elements that have not match, i.e., that belong to one of the input ontologies but do not belong to the other. This way it is possible to create in the internal structure a new ontology that is the merge of the given ontologies. This can be better understood in Figure 3.

3.4 Converter to an Ontology

The Converter to an ontology is the last block of the proposed architecture that implements the last task. It receives as input the internal structure generated by the Matcher/Merger component and converts the information on that structure into an ontology language. This process is the inverse of the process executed by the converter/recognizer. As it was said regarding that component, one of the goals is to allow diversity of ontology formats, so the objective concerning this module is to allow the output ontology to be written in different possible languages and it is up to the user to decide in what language he would like their matched ontology to be written. As is the case with the input ontologies, we are aiming to at least allow the output ontology to be written in OWL language family and Turtle.

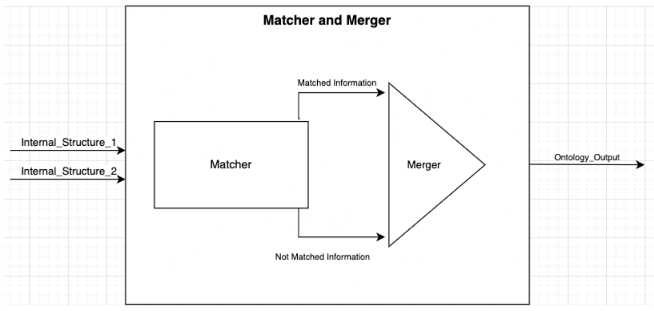


Figure 3 Detailed Matcher/Merger component.

4 OMT, the tool

This section presents a prototype of the proposed tool, OMT. The example shown corresponds to the analysis of two ontologies extracted from the Ontology Alignment Evaluation Initiative (OAEI) site[2], the first one describing the parts of the Human Body, and the second one describing the parts of the Mouse Body. Figure 4 shows how the users choose and upload the input ontologies to be matched, *human* and *mouse* ontologies as examples. After uploading an ontology, OMT identifies the natural language used to write its elements (concepts, properties and relations), and computes the size of concepts, properties and relations subsets, counting the number of elements in each subset. Figure 5 shows the information computed for both ontologies presenting it in a table to the user before starting the matching process. As it was mentioned in Section 3.2, if the input ontologies are written in portuguese, an additional step of translating the information to english is required. Figure 6 shows an example of said translation. Last but not least, Figure 7 shows the results of comparing the concepts of the two input ontologies. The process of comparing concepts makes use of techniques introduced in Section 2.1, such as **stop-word elimination** and **tokenization** to prepare the information and then distance functions such as the **Jaccard Index** to compare the similarity. The **matching value** column in Figure 7 represents how similar the 2 concepts are. The user has the option of clicking the **Details** link, that presents him with a new page (on the right, in Figure 6) showing the concepts and corresponding descriptions defined in the ontologies.

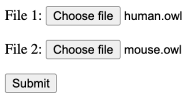
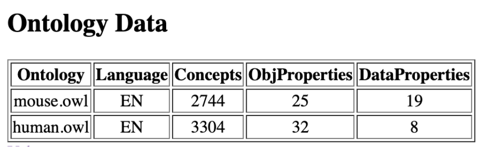


Figure 4 Input Form to submit the ontologies to be matched.

These are the functionalities that have been implemented so far. After improving the Matcher module, the Merger module will be worked out, as well as the Web application that will host the OMT tool will be implemented.

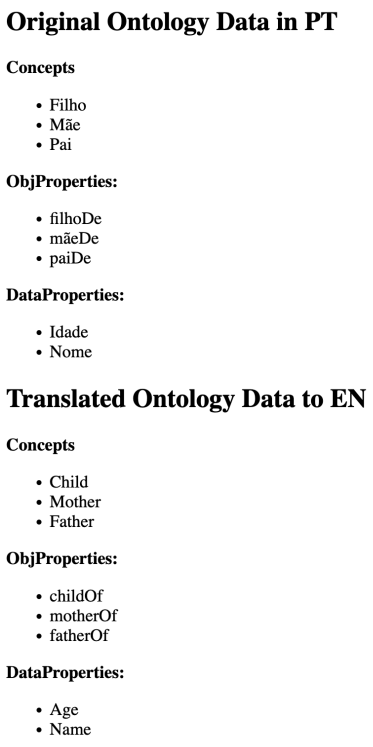
[2] OAEI is accessible at `http://oaei.ontologymatching.org/`, and contains a large amount of different ontologies to be used as benchmarks to test Ontology Matching Tools.



Ontology Data

Ontology	Language	Concepts	ObjProperties	DataProperties
mouse.owl	EN	2744	25	19
human.owl	EN	3304	32	8

Figure 5 Input ontologies data.



Original Ontology Data in PT

Concepts

- Filho
- Mãe
- Pai

ObjProperties:

- filhoDe
- mãeDe
- paiDe

DataProperties:

- Idade
- Nome

Translated Ontology Data to EN

Concepts

- Child
- Mother
- Father

ObjProperties:

- childOf
- motherOf
- fatherOf

DataProperties:

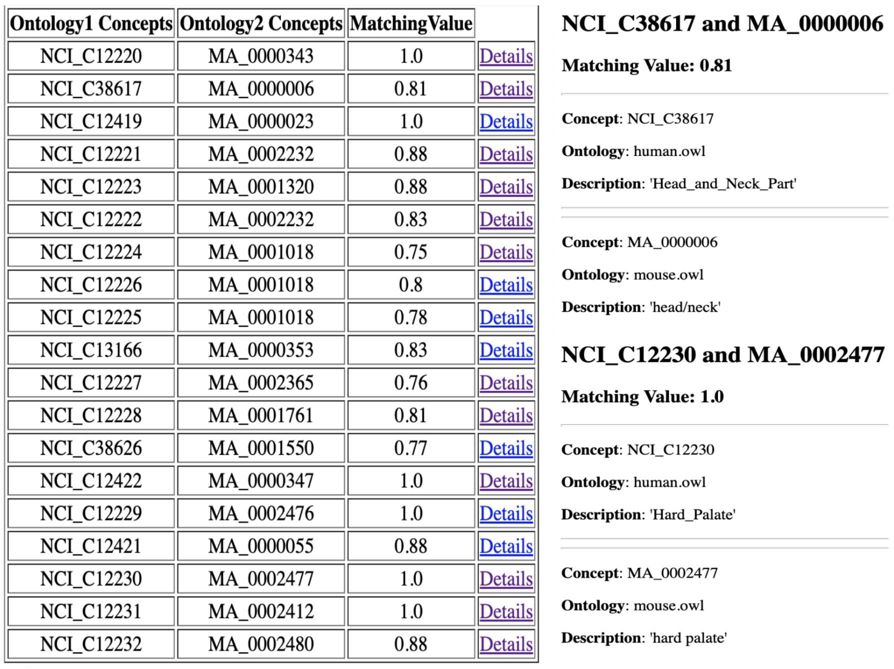
- Age
- Name

Figure 6 Translation of the elements of a PT ontology to an EN counterpart.

5 Conclusion

Ontology matching is an important functionality in many applications for relating information, e.g., from heterogeneous sources into a common model that can be queried and reasoned upon. The work in progress we present in this paper contributes to the ontology matching research issue proposing a new adaptable and easily accessible system called OMT. We summarize the study based on the extensive literature review done on already existing ontology matching methods. This study gave rise to new ideas we integrate in our system. An overview of the architecture of the system is then presented in the paper. The Matching and Merge component of the system integrates techniques coming from String, Language and Contraint and Instance based techniques.

The prototype, so far implemented as a proof of concept of our proposal, is also introduced. As explained, that prototype allows for the choice of the two ontologies to be merged. Before the matching process, if one of the input ontologies is written in portuguese, it is automatically converted to an english version. After converting the input ontologies into the same internal format, OMT tool outputs a results table with the concepts matching. This table exhibits the concepts identified as similar after analysing the two given ontologies, being also displayed the matching percentage evaluated for each pair.



Ontology1 Concepts	Ontology2 Concepts	MatchingValue	
NCI_C12220	MA_0000343	1.0	Details
NCI_C38617	MA_0000006	0.81	Details
NCI_C12419	MA_0000023	1.0	Details
NCI_C12221	MA_0002232	0.88	Details
NCI_C12223	MA_0001320	0.88	Details
NCI_C12222	MA_0002232	0.83	Details
NCI_C12224	MA_0001018	0.75	Details
NCI_C12226	MA_0001018	0.8	Details
NCI_C12225	MA_0001018	0.78	Details
NCI_C13166	MA_0000353	0.83	Details
NCI_C12227	MA_0002365	0.76	Details
NCI_C12228	MA_0001761	0.81	Details
NCI_C38626	MA_0001550	0.77	Details
NCI_C12422	MA_0000347	1.0	Details
NCI_C12229	MA_0002476	1.0	Details
NCI_C12421	MA_0000055	0.88	Details
NCI_C12230	MA_0002477	1.0	Details
NCI_C12231	MA_0002412	1.0	Details
NCI_C12232	MA_0002480	0.88	Details

NCI_C38617 and MA_0000006

Matching Value: 0.81

Concept: NCI_C38617
Ontology: human.owl
Description: 'Head_and_Neck_Part'

Concept: MA_0000006
Ontology: mouse.owl
Description: 'head/neck'

NCI_C12230 and MA_0002477

Matching Value: 1.0

Concept: NCI_C12230
Ontology: human.owl
Description: 'Hard_Palate'

Concept: MA_0002477
Ontology: mouse.owl
Description: 'hard palate'

Figure 7 Matching the Concepts of two input ontologies.

As future work, we will formalise the algorithms involved in the system components, giving a special focus on the Matching and Merge algorithm. The system will then be fully implemented and tested with several heterogenous data. Finally, an evaluation of the system will be conducted, comparing it with other existing tools.

References

1 Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah McGuinness, Peter Patel-Schneijder, and Lynn Andrea Stein. OWL Web Ontology Language Reference. Recommendation, World Wide Web Consortium (W3C), February 10 2004. See `http://www.w3.org/TR/owl-ref/`.

2 Gavin Carothers and Eric Prud'hommeaux. RDF 1.1 turtle, February 2014. URL: `http://www.w3.org/TR/2014/REC-turtle-20140225/`.

3 William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, 2003.

4 Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. Agreementmaker: Efficient matching for large real-world schemas and ontologies. *Proc. VLDB Endow.*, 2:1586–1589, 2009.

5 Jérôme David, Fabrice Guillet, and Henri Briand. Matching directories and owl ontologies with aroma. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, page 830–831, New York, NY, USA, 2006. Association for Computing Machinery. `doi:10.1145/1183614.1183752`.

6 Sean M. Falconer and Natalya F. Noy. *Interactive Techniques to Support Ontology Matching*, pages 29–51. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

7 Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

8 Walid Hassen. Medley results for oaei 2012. In *Proceedings of the 7th International Conference on Ontology Matching - Volume 946*, OM'12, pages 168–172, Aachen, DEU, 2012. CEUR-WS.org.

9 Sven Hertling. Hertuda results for oaei 2012. In *OM*, 2012.

10 Konstantinos Kotis, Alexandros G Valarakos, and George A Vouros. Automs: Automated ontology mapping through synthesis of methods. In *Ontology Matching*, page 96, 2006.

11 Lorena Otero-Cerdeira and Francisco Javier Rodríguez-Martínez and Alma María Gómez-Rodríguez. Ontology matching: A literature review. *Expert Syst. Appl.*, 42:949–971, 2015.