

# A String Metric for Ontology Alignment

Giorgos Stoilos, Giorgos Stamou, and Stefanos Kollias

Department of Electrical and Computer Engineering,  
National Technical University of Athens,  
Zographou 15780, Greece

**Abstract.** Ontologies are today a key part of every knowledge based system. They provide a source of shared and precisely defined terms, resulting in system interoperability by knowledge sharing and reuse. Unfortunately, the variety of ways that a domain can be conceptualized results in the creation of different ontologies with contradicting or overlapping parts. For this reason ontologies need to be brought into mutual agreement (aligned). One important method for ontology alignment is the comparison of class and property names of ontologies using string-distance metrics. Today quite a lot of such metrics exist in literature. But all of them have been initially developed for different applications and fields, resulting in poor performance when applied in this new domain. In the current paper we present a new string metric for the comparison of names which performs better on the process of ontology alignment as well as to many other field matching problems.

## 1 Introduction

It is widely recognized today that ontologies are going to play a key role in the realization of almost all modern knowledge based application. They have already been successfully applied in fields like the World Wide Web [1], intelligent multimedia systems [2] and many more. Ontologies are used in order that distributed and disparate applications and systems overcome semantic heterogeneity and enable them interchange knowledge for the completion of more complex tasks. But, the various ways that different organizations conceptualize a domain or the fact that they purpose ontologies for different applications, thus modelling a different perspective of the world or the same but with different constraints and properties, results in heterogeneous ontologies which still have to be brought into mutual agreement.

To overcome this heterogeneity, scientist have developed methodologies and tools for assisting the (still) semi-automatic process of ontology alignment. This process provides us with semantic correspondences among the entities that exist within two heterogeneous ontologies. Nowadays, many techniques have been developed, or borrowed from other fields, in order to discover the semantic correspondences among entities. Among these methods a very popular one is the comparison of the class and property names of the ontologies using a string distance metric so as to produce a degree of similarity. Such a technique is referred

to as *terminological matching*. Even if the core methodology of a platform is not based on this technique an initial similarity extraction step is usually performed using this method. This technique is based on the fact that the same concepts are likely to be modelled using quite similar names. Many platforms that use such a methodology to compute similarities between ontologies exist in literature; some examples are, Anchor-PROMPT [3], QOM [4], Cupid [5] and many more.

Though powerful string metrics exist in literature, and have been used successfully in the past, from our experience with the development of a new ontology alignment platform, they don't perform well when used in this new demanding and complex domain. In this paper we present a new string metric which is created by paying special care to each different characteristic of the process of ontology alignment, thus leading to a metric with very good performance.

The rest of the paper is organized as follows. In section 2 we review the most frequently used string metrics found in literature. In section 3 we present the requirements that the field of ontology alignment introduces and explain why the usual string metrics fail to satisfy these requirements by giving small examples. These specifications will guide us during the definition of the new string metric which is introduced in section 4. In section 5, we present two evaluation experiments and strength our points about the bad behavior of the classical metrics. In section 6 we integrate our new metric in a new ontology alignment platform to see its behavior when it is used in a more complete ontology platform and at last section 7 concludes the paper.

## 2 Related Work

Today quite a lot of string metrics exist in literature. These string metrics have been developed and applied in different scientific fields like statistics, for probabilistic record linkage [6], database, for record matching [7], Artificial Intelligence, for supervised learning [8], and Biology, for identifying common molecular subsequences [9]. In the current paper we have considered the *Levenstein* [10] distance, which counts the insertions and deletions needed to match two strings, the *Needleman-Wunsch* [11] distance, which assigns a different cost on the edit operations, the *Smith-Waterman* [9], which additionally uses an alphabet mapping to costs and the *Monge-Elkan* [7], which uses variable costs depending on the substring gaps between the words. Moreover we used the *Jaro-Winkler* [12,6] similarity, which counts the common characters between two strings even if they are misplaced by a "short" distance, the *Q-Gram* [13], which counts the number of tri-grams shared between the two strings and the *sub-string* distance [14] which searches for the largest common substring.

## 3 Desired Properties

Ontology alignment is a relatively new field in computer science. Thus, none of the classical string metrics has been created having the properties and charac-

teristics of this field in mind. Algorithms that are used in ontology alignment are very complex and contain many features and parameters that can affect the performance even of commonly accepted and “good” string metrics, when they are used in this new context. Features like the threshold (the value above which two pairs are considered identical), or the cardinality of mappings (“one-to-one, one-to-many” etc.) play a key role in ontology alignment and as we will see the metrics found in literature sometimes fail to give satisfactory results cause of the existence of these parameters. Thus, before we define our string metric we think that it is crucial to state the specifications that we want such a string metric to fulfill. More precisely we want the new metric to be:

1. **Fast:** Since ontologies are used in applications that demand processing in real-time, like the semantic web or intelligent retrieval tasks, the complexity of the string metric should be very low, leading to a fast matching procedure.
2. **Stable:** As we aforementioned, one very crucial parameter of ontology alignment algorithms is their threshold. When we will demand from alignment platform to automatically operate on the semantic web their threshold would probably be fixed at a value considered optimal by their authors. Though some methods that automatically adjust the threshold during runtime exist in literature [15] it cannot be proved that they select the optimum value for threshold each time an alignment is performed. Thus, we demand by the string metrics to be as “stable” as possible. By “stability” we define the ability of a string metric to perform almost optimal even if small diverges from the optimal threshold take place. As we will see all metrics fail to satisfy this crucial property. Even worst, classical metrics are really sensitive in small changes of the threshold, and while they can provide good results if the threshold is optimized, this performance can rapidly decrease if we slightly disturb the value of the threshold.
3. **Intelligent:** When operating for example in the semantic web context, it is likely that an ontology be compared to an irrelevant one, but with which string resemblances occur. In this case we want our metric to identify all the differences and provide us with correct results. But it is not uncommon the situation where usual string metrics fail to identify cases where two strings represent completely different concepts but resemble a lot. Consider for example the words “score” and “store”. They represent two completely different concepts. Though this is true the Monge-Elkan, Levenstein, SubString, Needleman-Wunsch, Q-Gram and Jaro-Winkler metrics rate the pair with a similarity degree of 0.68, 0.8, 0.6, 0.9, 0.57 and 0.88 which are relatively high values. In contrast our string metric assigns a value of 0.45.
4. **Discriminating:** One of the most usual cardinalities requested for alignment mappings is the “one-to-one” cardinality. As it is obvious in an “one-to-one” mapping if a string in a reference ontology is mapped with the same similarity degree to more than one in the second ontology it is very probable that the algorithm fails to pick the correct pair from the set of pairs. Hence, we would like from our similarity metric to rarely assign the same similarity degree when we compare one particular string to several others.

Many times during the experiments we faced situations where several runs of an alignment between two ontologies using the same configuration produced different precision and recall [16] values cause of this phenomenon.

From the above analysis it is obvious that ontology alignment is indeed a demanding and delicate process that adds many constraints to the string metrics used in it.

## 4 Definition of the String Metric

In the current section we will introduce the new string metric, using as our guide the properties and features introduced in the previous section.

The new metric is based on the intuitions presented in [17] about the similarity between two entities. We argue that the similarity among two entities is related to their commonalities as well as to their differences. Thus, the similarity should be a function of both these features. This feature also appears, sometimes implicitly, in other measures as well. For example, in those measures that perform string editing, such operation can be considered as a form of difference counting, while non-editing can be considered as similarity counting. Thus, our metric is defined by the following equation:

$$Sim(s_1, s_2) = Comm(s_1, s_2) - Diff(s_1, s_2) + winkler(s_1, s_2) \quad (1)$$

where  $comm(s_1, s_2)$  stands for the commonality between  $s_1$  and  $s_2$ ,  $diff(s_1, s_2)$  for the difference and  $winklerImpr(s_1, s_2)$  for the improvement of the result using the method introduced by Winkler in [6]. We now have to define the functions of commonality and difference.

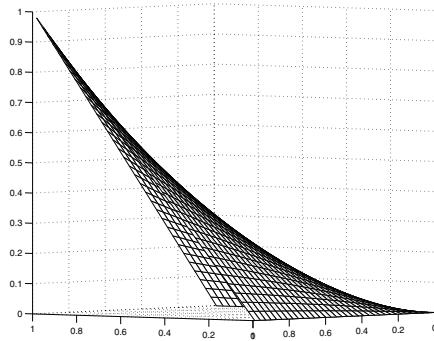
The function of commonality is motivated by the substring string metric. In the substring metric the biggest common substring between two strings is computed. This process is further extended by removing the common substring and by searching again for the next biggest substring until no one can be identified. The sum of the lengths of these substrings is then scaled with the length of the strings. The intuition behind this extension of the substring metric is the following. In the field of Computer Science researchers tend to use descriptive names for their variables or the units that represent real world entities. In other cases they tend to concatenate words and create new ones. For example in order to represent the concept of the number of pages of a book it is likely that someone uses the word “numberOfPages” or someone else might use the word “numPages”. As one can see these two strings share not one but two common substrings which is very crucial to identify in order to approximate their real similarity as much as possible. Moreover, we can now distinguish cases like the above with cases where the substring “Pages” is shared but the rest of the strings are quite different, thus satisfying the specification for an intelligent metric. Hence, the function of commonality is given by the following equation:

$$Comm(s_1, s_2) = \frac{2 * \sum_i length(maxComSubString_i)}{length(s_1) + length(s_2)} \quad (2)$$

As for the difference function, this is based on the length of the unmatched strings that have resulted from the initial matching step. Moreover, we believe that difference should play a less important role on the computation of the overall similarity. Our choice was the Hamacher product [18], which is a parametric triangular norm. This leads us to the following equation:

$$Diff(s_1, s_2) = \frac{uLen_{s_1} * uLen_{s_2}}{p + (1 - p) * (uLen_{s_1} + uLen_{s_2} - uLen_{s_1} * uLen_{s_2})} \quad (3)$$

where  $p \in [0, \infty)$ , and  $uLen_{s_1}$ ,  $uLen_{s_2}$  represent the length of the unmatched substrings from the initial strings  $s_1$  and  $s_2$  scaled with the string length, respectively. Observe that the parameter  $p$  can be adjusted at will giving a different importance on the difference factor. From experiments we performed we concluded that a value of 0.6 gives very good results. In Fig. 1 a three dimensional plot of the Hamacher function is illustrated.



**Fig. 1.** Hamacher t-norm curve

As it is obvious from (1), our similarity measure takes values from the interval  $[-1, 1]$ . The majority of string metrics found in literature range over  $[0, 1]$ , but other metrics that extend this interval can be found, like the Resnik similarity [19] for taxonomies.

The incorporation of a difference factor to the overall similarity is the key feature for the satisfiability of the rest of specifications introduced in section 3. The dissimilarity stretches the range of our similarity metric over the interval  $[-1, 1]$  performing a somewhat “clean-up” of the crucial space between 0.4 to 0.7, where a threshold can range, by discarding highly dissimilar pairs close to -1. In other string metrics this interval is cramped with false and true positive values. This fact makes these metrics very sensitive on the choice of the threshold. Moreover, this stretching makes it less probable to get same values for a string when it is compared to a large set of other strings, and thus satisfying the property of discrimination. At last the complexity of our string is polynomial to the size of the input strings, satisfying the property of a fast metric.

## 5 Evaluation

We have conducted two kinds of experiments. The first one is about ontology alignment, where we used a terminological matcher in order to compare the various string metrics. The second one is performed with classical benchmarks found in literature for data integration and retrieval [20].

### 5.1 Ontology Alignment

The ontology test set used to perform the experiments was that of the EON ontology alignment contest [21,22]. This test set consists of one reference ontology (33 classes, 59 properties, 56 individuals and 20 anonymous individuals), for a bibliographic domain, to be compared with other ontologies. Most of these ontologies originate from the reference ontology by making some hand made changes. These changes were for example the extension, or shrinkage of the ontology hierarchy, the use of synonyms, foreign names, removal of class properties and many more. Most of these modifications are devised in order to evaluate ontology alignment platforms and algorithms, as a whole, that might use other ontology features, or external sources like multilingual dictionaries or lexicons as well and not just terminological matching. Thus from the initial test set we only used those that involved alternations of the strings of classes and properties, excluding synonyms, foreign languages and randomly generated strings. Clearly, such occasions cannot be handled by terminological matching and including them would not provide us with valuable results. The evaluation sets we used are the following:

1. **101:** In this test set the reference ontology is compared with itself.
2. **204:** In this test the reference ontology is compared with a modified one. These modifications involved naming conventions like the insertion of under-scores, abbreviations, upper-cased and lower-cased strings.
3. **301,302,303,304:** The reference ontology is compared with four real-life ontologies for bibliographic references found on the web and left unchanged.

**Table 1.** Precision and Recall of string metrics for various ontology alignment tests

Test	String Metrics															
	Levenstein		Sub-String		Jaro-Winkler		Monge-Elkan		Q-Gram		Smith-Waterman		Needleman-Wunsch		Sim	
	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.
101	1.0	1.0	1.0	1.0	1.0	1.0	.88	.88	1.0	1.0	.88	.88	1.0	1.0	1.0	1.0
204	.967	.967	.822	.804	.965	.923	.695	.695	.857	.847	.793	.75	.926	.829	.978	.978
301	.8	.786	.872	.786	.81	.557	.833	.737	.872	.786	.511	.754	.857	.786	.9	.786
302	.6	.645	.367	.666	.363	.666	.34	.625	.666	.666	.375	.625	.35	.687	.72	.645
303	.764	.812	.622	.791	.754	.833	.571	.66	.677	.833	.581	.666	.904	.791	.754	.833
304	.972	.947	.8	.96	.923	.947	.776	.776	.972	.947	.789	.789	.972	.947	.972	.947
312	.794	.968	.911	.968	.862	.781	.823	.875	.939	.968	.794	.843	.76	1.0	.911	.968
	Pre.=Precision, Rec.=Recall															

Furthermore, we performed an additional experiment, named **312**, aligning the two ontologies of experiments 301 and 302.

In order to perform the alignment and evaluation experiments we used the API for ontology alignment introduced by Euzenat [23]. We used the sample implementations found in the API after performing some slight modifications on them, in order to include all the metrics and exclude from the alignment references to external entities. In order to evaluate each metric we have used the classical measures from the field of information retrieval of precision and recall [16]. Algorithms to compute precision and recall, given a proposed and a correct reference alignment, can also be found in the API.

For each metric and for each experiment we were changing the threshold of the algorithm (the value below which a mapping between strings is discarded) in order to achieve the highest precision for the highest recall that is possible by a metric. The reason for giving maximum importance to the recall measure is the following. Since ontology alignment is likely to stay a semi-automatic process, human intervention will eventually be needed to complete an alignment. Since the burden of deleting false identified pairs by a platform is minimal compared to the burden of traversing two heterogeneous ontologies that might include thousands of concepts and attributes and identify similar entities, recall is a much more important measure. Furthermore, the choice of not keeping the threshold constant is that different metrics have different mathematical properties thus one optimal threshold for one metric could be a worst for all the others, and vice versa. The complete set of experiments and the values of precision and recall using several string metrics is depicted in Table 1, where our metric appears in the last column.

From Table 1 we can immediately see that the Monge-Elkan and Smith-Waterman metrics perform worst than any other metric. In experiment 101 we can see the point made in section 3 about the use of an “one-to-one” mapping and the optimistic behavior of some metrics. In all other experiments their performance is kept in low levels. On the other hand our metric, on average, performs better compared to the other metrics. It manages to achieve high precision while retaining recall at high levels. Even in cases where a slightly better recall was achieved by other measures, the price to pay was a dramatically low precision, such that in some cases even up to 50 or 40 false positive pairs were retrieved. For example, Needleman-Wunsch metric achieves a better recall in experiments 302 and 312 but the precision is so poor that up to 56 and 10 false negative pairs, respectively, have to be removed later, possibly by human intervention. The same thing happens with Sub-String (Jaro-Winkler) distance, in experiments 302 and 304 (302), where the price for the slightly better recall are 55 and 18 (56) false positive pairs, respectively.

As we mentioned before, achieving a very high recall is crucial for the process of creating mappings between ontologies. But at the same time precision must be kept at high levels since having to discard up to 50 pairs, as it happened in the experiments when a good recall was achieved for such small ontologies, is not desired at all. It would be interesting to see in the cases where other metrics

achieve better recall but substantially lower precision what happens if we drop recall at the same level as the one achieved by our metric. Even if this happens most metrics still achieve substantially low precision. More precisely in 302, for the same recall, Needleman-Wunsch achieves 0.57 precision, Jaro-Winkler 0.607 and Sub-String 0.62. Only Q-Gram achieves 0.775. At last Sub-String, in 304, achieves the same precision (0.972), and Needleman-Wunsch in experiment 312, 0.861.

In order to give a more intuitive view of the strength of the new metric, we have computed the precision and recall for each experiment and for each experiment for nine different threshold values, ranging from 0.1 to 0.9. Then we computed the average precision and recall of all experiments for all these different thresholds and we have created an average precision vs. average recall chart. This can be depicted in Fig. 2. We have excluded from these charts Monge-Elkan and Smith-Waterman metrics since their performance is very poor when applied to our field of interest.

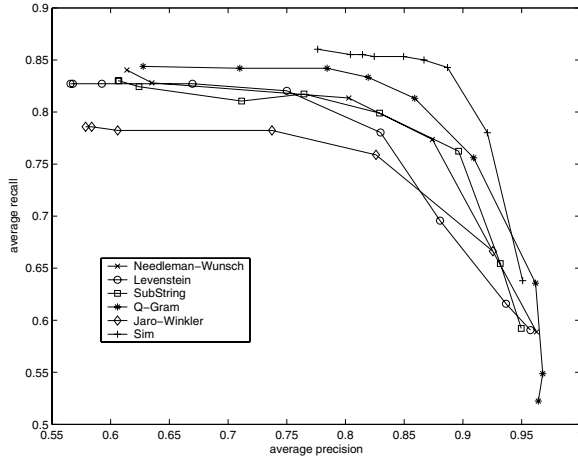


Fig. 2. Average Precision vs. Average Recall values

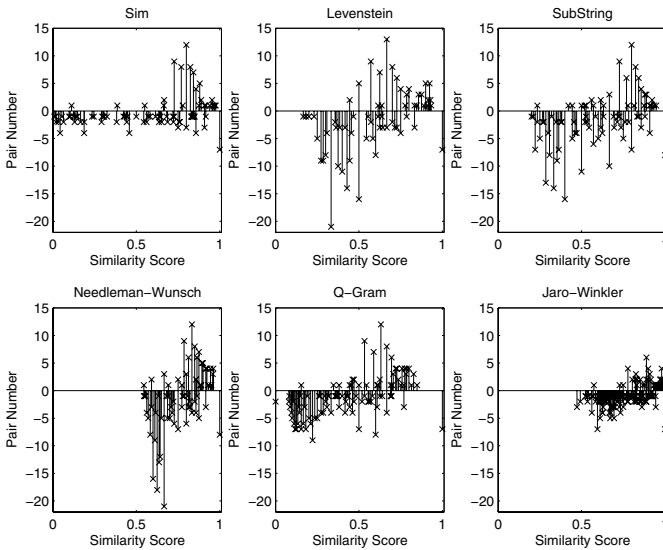
In Fig. 2 we can immediately see that our metric achieves at the same time a substantially better precision and recall. While other metrics have scored their best recall when precision is at a very low levels (and subsequently decrease recall to increase precision), our metric can simultaneously achieve high values on both these measures.

Additionally, since curves in Fig. 2 are interpolated, we have put “marks” on each one of these to denote the points where average precision and average recall is computed for different threshold values. Concerning our string metric we can identify an area of seven different threshold values where average precision is increasing while at the same time average recall still stays at high levels. This is the area where recall ranges from 0.86 to 0.85 while precision ranges from 0.77



to 0.88. In all the other curves, after the third or fourth threshold value (where average precision vs. recall is at acceptable levels), we can observe a very high decrease of recall when a higher threshold is set. In other words the area that our metric achieves a high recall combined with a very high precision is a highly unstable area for all the other metrics. If not a highly optimal threshold is set in an ontology platform that uses such a metric, there is a high risk that low results are obtained.

In order to give a more clear picture on the stability of our metric, in Fig. 3 we present the number of pairs assigned a particular similarity degree in all six experiments.



**Fig. 3.** Number of pairs assigned a particular similarity values

From Fig. 3 we can immediately deduce that the Needleman-Wunsch and Jaro-Winkler metrics are highly instable. Notice how many pairs have been accumulated within a small interval of the similarity degree axis. Clearly a small variation in the threshold of an alignment platform that uses one of these metrics could wreck the performance. Moreover, all the other metrics suffer too by a bad stability. Observe for example the diagram for the Levenstein distance. Within the interval of  $[0.4, 0.6]$  where, according to Fig. 2, average precision vs. average recall is at its best levels, the metric retrieves 16 true positives and 79 false negatives. As it is obvious a slight disturbance of the threshold might fetch many of the false pairs or loose many of the correct ones. Sub-String and Q-Gram metrics are a bit more stable. Sub-String has the interval of  $(0.4, 0.6]$  where 8 true and 46 false positives exist (without including 0.4 where 16 more false positives appear). Q-Gram, on the other hand, ranks 9 true positives and 44 false negatives within the interval  $[0.3, 0.5]$ , where obviously the same problem as before exists.

When it comes to our metric its stability is obvious. The highly dissimilar pairs have too early been discarded close to -1 and the danger of fetching too many of them if an optimal threshold is not chosen is very small. Into the interval [0.5,0.7], only 4 true positives and 17 false positives appear. Thus, with this property satisfied we can more easily choose a threshold where the maximum recall strength of the metric is being used and at the same time not worry if we are at a point where low precision is encountered. At last observe that in our case, under the interval where the majority of true positives is present, a very low number of false positives exist, which justifies the results of the high precision.

### 5.2 Census and Field Matching

Even though our metric was originally designed for the domain of ontology alignment it still is a string metric in the classical sense. Thus we could not resist but to evaluate it with classical benchmarks found in literature like the ones in [7,8,24,20]. The list of the datasets used can be found in Table 2 as well as the number of strings that each dataset includes.

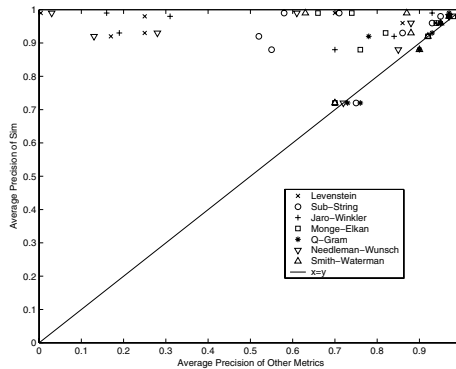


Fig. 4. Average Precision of string metrics vs. our metric

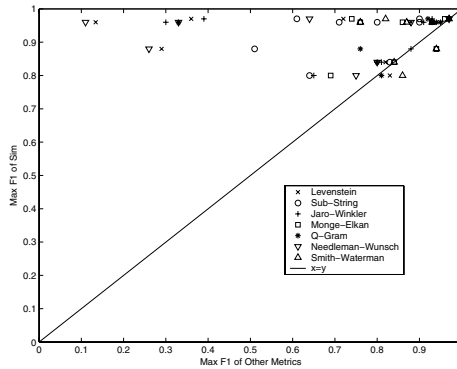
Each dataset contains two relations. Each relation contains a number of strings where each string is compared with all the strings of the other relation. Each comparison of two strings is assigned a similarity degree. Every entry for a string contains a key which is purposed for the identification of the correctness of a pair. In order to evaluate our metric against these datasets we used the SecondString open-source library [20]. This library contains all these datasets as well as algorithms to compute the average precision and maximum F1 measure for each test. The F1 measure is an aggregation of the precision and recall measures.

In Figs. 4 and 5 we can see two scatter plots. The former is about the average precision achieved in all experiments by all classical metrics, relative to our

**Table 2.** Datasets used in experiments

Name	Strings
bird1	377
bird2	982
bird3	98
bird4	719
park	654
restaurant	863
peopleMatch	90
census	841

metric, while the latter one is about the maximum F-measure. As we can see from these plots our metric performs better in the majority of experiments performed with the data sets presented in Table 2. Only in five cases in average precision and equal times in maximum F1 loses with a small difference.



**Fig. 5.** Maximum F1 of string metrics vs. our metric

At last these experiments also gave us results about the complexity of the metrics. Our metric comes second, very close to the substring metric, with which has a large resemblance. Both these metrics are far from competition, relative to time execution, from the other metrics from which perform about 5 to 20 times faster when it comes to experiments with big strings to be compared.

## 6 A Sample Implementation

At last, we have started the integration of the similarity metric in an ontology alignment platform. This way we can test the behavior of our metric when the

aggregation of more than one methods for computing similarity is involved. The platform features three kinds of matching methods, as described in [14]. These are the following:

- **Terminological Matching:** This method is the standard terminological method, which computes similarities based on the strings of class and property names.
- **Structural Internal Matching:** In this method we refine the similarity computed by terminological matching, for two classes, by a portion of the similarities between the names of their properties.
- **Structural External Matching:** In this method we again refine the similarity between two classes by a portion of the similarity computed for the superclasses of the two classes.

The similarities computed by the above methods are aggregated to produce an overall similarity. In this aggregation we give more importance to similarities computed for class and property names, by the terminological method, and less importance to the other methods.

**Table 3.** Precision, Recall and fallout for the experiments

	Precision	Recall	F-Measure
101	1.0	1.0	1.0
201	0.926	0.692	0.792
204	0.989	0.978	0.983
222	0.966	0.945	0.955
223	0.956	0.956	0.956
230	0.962	1.0	0.98
301	0.98	0.79	0.874
302	0.857	0.625	0.722
303	0.816	0.83	0.824
304	0.92	0.90	0.915
312	0.967	0.937	0.95

In order to evaluate the sample implementation we have again used the evaluation experiments from [21,22]. Additionally to the experiments used for the evaluation of the string matching method we have included experiments, 201, 222, 223 and 230. The reader is referred to [22] for a detailed description of the experiments.

In the following table we can see the precision, recall and f-measure achieved by the sample ontology alignment platform. In all experiments we have used a fixed threshold, set at value 0.65.

## 7 Conclusions

Ontology alignment platforms have been benefited a lot by the use of string distance metrics in order to discover semantic mappings between ontologies. Though

powerful metrics exist in literature they have been developed and purposed for different domains and applications. The delicate and demanding features of the process of ontology alignment, such as speed, threshold, cardinality, or the potentially short size of ontologies can badly affect the performance of classical metrics which usually are “optimistic”, to the degrees they assign, or accumulate values close to one another.

These demanding features has led us to the creation of a new string metric taking extra care to satisfy each one of them. Experiments has shown that the new metric performs better on average by the classical ones, when optimal configurations are used, and can greatly outperform them when no a priory knowledge for the alignment task is known. Furthermore, experiments with classical benchmarks for field and census matching has show that our metric is still very powerful to be used in such domains and for such tasks, too. At last, all this interestingly good performance comes with a low complexity.

## Acknowledgements

This work is supported by the FP6 Network of Excellence EU project Knowledge Web (IST-2004-507482).

## References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American* **279** (2001)
2. Benitez, A., Smith, J., Chang, S.F.: Medianet: A multimedia information network for knowledge representation. Volume 4210., IS&T/SPIE-2000 (2001)
3. Noy, N., Musen, M.: Anchor-prompt: Using non-local context for semantic matching. In: Proc. IJCAI 2001 workshop on ontology and information sharing, Seattle (WA US). (2001) 63–70
4. Ehrig, M., Staab, S.: Qom - quick ontology mapping. In: Proc. of the 3rd International Semantic Web Conference, Hiroshima (JP). Volume volume 3298 of LNCS. (2004) 683–697
5. Madhavan, J., Berstein, P., Rahm, E.: Generic schema matching using cupid. In: Proc. of the 27th VLDB, Roma (IT). (2001) 48–58
6. Winkler, W.: The state record linkage and current research problems. Technical report, Statistics of Income Division, Internal Revenue Service Publication (1999)
7. Monge, A., Elkan, C.: The field-matching problem: algorithm and applications. In: Proceedings of the second international Conference on Knowledge Discovery and Data Mining. (1996)
8. Tejada, S., Knoblock, C.A., Minton, S.: Learning object identification rules for information integration. *Information Systems* **26** (2001) 607–633
9. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. *Journal of Molecular Biology* **147** (1981) 195–197
10. Levenstein, I.: Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics and Control Theory* (1966)
11. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Molecular Biology* **48** (1970) 444–453

12. Jaro, M.: Probabilistic linkage of large public health data files (disc. p687-689). *Statistics in Medicine* **14** (1995) 491–498
13. Sutinen, E., Tarhio, J.: On using q-gram locations in approximate string matching. In: *ESA '95: Proceedings of the Third Annual European Symposium on Algorithms*, Springer-Verlag (1995) 327–340
14. Euzenat, J., Le Bach, T., Barrasa, J., Bouquet, P., De Bo, J., Dieng-Kuntz, R., Ehrig, M., Hauswirth, M., Jarrar, M., Lara, R., Maynard, D., Napoli, A., Stamou, G., Stuckenschmidt, H., Shvaiko, P., Tessaris, S., Van Acker, S., Zaihrayeu, I.: State of the art on ontology alignment. deliverable 2.2.3 (2004)
15. Ehrig, M., Sure, Y.: Ontology mapping - an integrated approach. In: *Proceedings of the First European Semantic Web Symposium*. Volume 3053. (2004) 76–91
16. Do, H., Melnik, S., Rahm, E.: Comparison of schema matching evaluations. In: *Proceedings of the 2nd International Workshop on Web Databases*. (2002)
17. Lin, D.: An information-theoretic definition of similarity. In: *Proc. 15th International Conf. on Machine Learning*, Morgan Kaufmann, San Francisco, CA (1998) 296–304
18. Hamacher, H., Leberling, H., Zimmermann, H.J.: Sensitivity analysis in fuzzy linear programming. *Fuzzy Sets and Systems* **1** (1978) 269–281
19. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: *Proceedings of the IJCAI-95*. (1995) 448–453
20. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string metrics for matching names and records. In: *Proc. KDD-2003 Workshop on Data Cleaning and Object Consolidation*. (2003)
21. Euzenat, J.: Evaluating ontology alignment methods. In: *Proc. Dagstuhl seminar on Semantic interoperability and integration*, Wadern (DE). (2004) 47–50
22. Sure, Y., Corcho, O., Euzenat, J., Hughes, T., eds.: *Proceedings of the 3rd Evaluation of Ontology-based tools (EON)*. (2004)
23. Euzenat, J.: An api for ontology alignment. In: *Proc. 3rd conference on international semantic web conference (ISWC)*, Hiroshima (JP). (2004) 698–712
24. Cohen, W.: Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems* **18** (2000) 288–321