Undefined 1 (2013) 1–5
IOS Press

A Systematic Survey of Point Set Distance Measures for Link Discovery

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Mohamed Ahmed Sherif [a] and Axel-Cyrille Ngonga Ngomo [a]
[a] *Universität Leipzig, Institut für Informatik, AKSW, Postfach 100920, D-04009 Leipzig, Germany*
E-mail: {lastname}@informatik.uni-leipzig.de

Abstract. With the growth of the Data Web, large amounts of geo-spatial information have been made available. While discovering links between resources on the Data Web has been shown to be a demanding task, discovering links between geo-spatial resources proves to be even more challenging. This is partly due to the resources being described by the means of vector geometry. Especially, discrepancies in granularity and error measurements across datasets render the selection of appropriate distance measures for geo-spatial resources difficult. In this paper, we survey existing literature for point-set measures that can be used to measure the similarity of vector geometries. We then present and evaluate the ten measures that we derived from literature. We evaluate these measures with respect to their time-efficiency and their robustness against discrepancies in measurement and in granularity. To this end, we use samples of real datasets of different granularity as input for our evaluation framework. The results obtained on three different datasets suggest that most distances approaches can be led to scale. Moreover, while some distances are significantly slower than other measures, distances based on means, surjections and sums of minimal distances are robust against the different types of discrepancies.

Keywords: Link discovery, Geographic Distances

1. Introduction

The Web of Data has grown significantly over the last years. In particular, very large datasets pertaining to different domains such as bio-medicine (e.g., LinkedTCGA with now 20+ billion triples [27]) and geo-locations (e.g., LinkedGeoData (LGD) with 1+ billion triples [4]) have been made available. Implementing the fourth Linked Data principle (i.e., the creation of links between these knowledge bases and other knowledge bases) for these knowledge bases has been shown to be a difficult problem in previous works [5]. Most of the existing solutions (see [21] for an overview) address this problem by using a complex similarity or distance function to compare instances from two (not necessarily distinct) knowledge bases. The result of the function is then compared to a threshold. The result of the comparison is finally used to suggest the existence of a link between instances.

While previous works have compared a large number of measures with respect to how well they perform in the link discovery task [9], measures for linking geo-spatial resources have been paid little attention to. Previous works have yet shown that domain-specific measures and algorithms are required to tackle the problem of geo-spatial link discovery [22]. For example, 20,354 pairs of cities in DBpedia share exactly the same label. For villages in LinkedGeoData, this number grows to 3,946,750. Consequently, finding links between geo-spatial resources requires devising means to distinguish them using their geo-spatial location. On the Web of Data, the geo-spatial location of resources is most commonly described using either points or more generally by means of vector geometry. Thus, devising means for using geo-spatial infor-

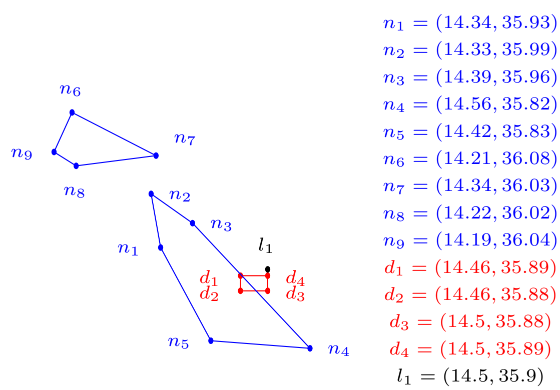


Fig. 1. Vector description of the country of *Malta*. The blue polygons shows the vector geometry for *Malta* in the *Nuts* dataset, the red polygon shows the same for the *DBpedia*, while the black point shows the location of the same real-world entity according to *LinkedGeoData*.

mation to improve link discovery requires providing means to measure distances between such vector geometry data.

Examples of vector geometry descriptions for the country of Malta are shown in Figure 1. As displayed in the examples, two types of discrepancies occur when one compares the vector descriptions of the same real-world entity (e.g., Malta) in different datasets: First, the different vector descriptions of a given real-world entity often comprise different points across different datasets. For example, Malta's vector description in DBpedia contains the point with latitude 14.46 and longitude 35.89. In LGD, the same country is described by the point of latitude 14.5 and longitude 35.9. We dub the discrepancy in latitude and longitude for points in the vector description *measurement discrepancy*. A second type of discrepancy that occurs in the vector description of geo-spatial resources across different datasets are discrepancies in *granularity*. For example, Malta is described by one polygon in DBpedia, two polygons in Nuts and a single point in LGD.

Analysing the behaviour of different measures with respect to these two types of discrepancies is central to detect the measures that should be used for geo-spatial link discovery. In this paper, we address this research gap by first surveying existing measures that can be used for comparing vector descriptions. We then compare these measures in series of experiments on samples extracted from three real datasets with the aim of answering the following questions:

- Q_1: Which of the existing measures is the most time-efficient measure?
- Q_2: Which measure generates mappings with a high precision, recall, or F-measure?
- Q_3: How well do the measures perform when the datasets have different granularities?
- Q_4: How sensitive are the measures to measurement discrepancies?
- Q_5: How robust are the measures when both types of discrepancy occur? Geospatial Web Data integration and enrichment (workflows)

The remainder of this paper is structured as follows: Section 2 introduces some basic assumption and notations that will be used all over the rest of the paper. Then, in Section 4 we give a detailed description of each of point set distance functions, as well as their mathematical formulation and different implementations. Thereafter, in Section 5 we introduce evaluation of our work for both scalability and robustness. Finally, we conclude the paper with a brief overview of related work (Section 6), as well as a conclusion and future work (Section 7). All measures and algorithms presented herein were integrated into the LIMES framework.[1]

2. Preliminaries and Notation

We assume the link discovery problem as being formulated in a way akin to [22]: Given two sets S and T of resources as well as a predicate p, compute the set $M = \{(s,t) \in S \times T :< s,p,t > holds\}$, where $< s,p,t >$ is the RDF triple with the subject s, the predicate p and the object t. Computing M directly is commonly a non-trivial task. State-of-the-art link discovery systems thus most commonly aim to compute an approximation M' of M with $M' = \{(s,t) : \delta(s,t) \leq \theta\}$, where δ is a complex distance function and θ is a distance threshold. δ most commonly consists of a combination of atomic measures which can be used to compare property values of the resources s and t. For example, the edit distance is an atomic measure that can be used to compare the labels of two resources.

In addition to bearing properties similar to those bared by other types of resources (label, country, etc.), geo-spatial resources are commonly described by means of vector geometry.[2] Each vector descrip-

[1] `http://limes.sf.net`

[2] Most commonly encoded in the WKT format, see `http://www.opengeospatial.org/standards/sfa`.

tion can be modelled as a set of points. We will write $s = (s_1, \ldots, s_n)$ to denote that the vector description of the resource s comprises the points $s_1, \ldots, s_n$. A point s_i on the surface of the planet is fully described by two values: its latitude $lat(s_i) = \varphi_i$ and its longitude $lon(s_i) = \lambda_i$. We will denote points s_i as pairs (φ_i, λ_i). Then, the distance between two points s_1 and s_2 can be computed by using the *orthodromic distance*

$$\delta(s_1, s_2) = R\cos^{-1}\big(\sin(\varphi_1)\sin(\varphi_2) + \cos(\varphi_1)\cos(\varphi_2)\cos(\lambda_2 - \lambda_1)\big), \quad (1)$$

where $R = 6371km$ is the planet's radius.[3] Computing the distance between sets of points is yet a more difficult endeavour. Over the last years, several measures have been developed to achieve this task. Most of these approaches regard vector descriptions as ordered set of points. In the following sections, we present such measures and evaluate their robustness against different types of discrepancies.

3. Systematic Survey Methodology

We carried out a systematic study of the literature on distances for point sets according to the approach presented in [18,20]. The systematic survey results were retrieved from seven different scientific search engines (see Table 1 for details). The search engines returned 19,869 distinct publications. We then excluded publications that were not in English or did not discuss algorithms for finding distances between vector geometries. Moreover, we excluded all publications that were not peer-reviewed as well as works that were published as posters or abstracts. Overall, 21 publications were deemed relevant and contained measures that could used to compare point sets. We discarded those measures that are only relevant for convex polygons given that this condition does not necessarily hold for the descriptions of geo-spatial entities. Overall, 10 different distances for point sets could be retrieved from the relevant publications. In the following, we present our survey approach in more detail.

3.1. Research question formulation

We began by defining research questions that guided our search for measures. These questions were as follows:

Q_1: Which of the existing measures is the most time-efficient measure?
Q_2: Which measure generates mappings with a high precision, recall, or F-measure?
Q_3: How well do the measures perform when the datasets have different granularities?
Q_4: How sensitive are the measures to measurement discrepancies?
Q_5: How robust are the measures when both types of discrepancy occur?

3.2. Eligibility criteria

To direct our search process towards answering our research questions, we created two lists of inclusion/exclusion criteria for papers. Papers had to abide by all inclusion criteria and by none of the exclusion criteria to be part of our survey:

- Inclusion Criteria
  * Work published in English between 2003 and 2013.
  * Studies on geographic terms based link discovery.
  * Algorithms for finding distance between point sets.
  * Techniques for improving performance of some will-known point sets distance Algorithms.
- Exclusion Criteria
  * Work that were not peer-reviewed or published.
  * Work that were published as a poster abstract.
  * Distance functions that focused on finding distance only between convex point sets.

3.3. Search strategy

Based on the research question and the eligibility criteria, we defined a set of most related keywords. There were as follows: *Linked Data, link discovery, record linkage, polygon, point set, distance, metric, geographic, spatial, non-convex*. We used those keywords as follows:

- *Linked Data* AND (*Link discovery* OR *record linkage*) AND (*geographic* OR *spatial*)
- *Non-convex* AND (*polygon* OR *point set*) AND (*distance* OR *metric*)

A keyword search was applied in the following list of search engines, digital libraries, journals, conferences and their respective workshops:

[3] We assume the planet to be a perfect sphere.

– Search Engines and digital libraries:
  * Google Scholar[4]
  * ACM Digital Library[5]
  * Springer Link[6]
  * Science Direct[7]
  * ISI Web of Science[8]
– Journals:
  * Semantic Web Journal(SWJ)[9]
  * Journal of Web Semantics(JWS)[10]
  * Journal of Data and Knowledge Engineering(JDWE)[11]

3.4. Search Methodology Phases

In order to conduct our systematic literature review, we applied a six-phase search methodology:

1. Apply keywords to the search engine using the time frame from 2003–2013.
2. Scan article titles based on inclusion/exclusion criteria.
3. Import output from *phase 2* to a reference manager software to remove duplicates. Here, we uses *Mendeley*[12] as it is free and has functionality for deduplication.
4. Review abstracts according to include/exclude criteria.
5. Read through the papers, looking for some approaches that fits the inclusion criteria and exclude papers that fits the exclusion criteria. Also, retrieve and analyze related papers from references.
6. Implement point sets distance functions found in step 5.

Table 1 provides details about the number of retrieved articles through each of the first five search phases. Note that in the sixth phase we only implemented distance functions found in the articles resulted from phase 5.

Table 1

Number of retrieved articles during each of the search methodology Phases.

Search Engines	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
Google Scholar	9,860	21	19	10	4
ACM Digital Library	3,677	16	16	5	3
Springer Link	5,101	22	21	11	8
Science Direct	1055	21	18	10	4
ISI Web of Science	176	15	14	4	2
SWJ	0	0	0	0	0
JWS	0	0	0	0	0
JDWE	0	0	0	0	0

4. Distance Measures for Point Sets

In the following, we present each of the distances derived from our systematic survey and exemplify it by using the DBpedia and Nuts descriptions of Malta presented in Figure 1. The input for the distances consists of two point sets $s = (s_1, \ldots, s_n)$ and $t = (t_1, \ldots, t_m)$, where n resp. m stands for the number of distinct points in the description of s resp. t. W.l.o.g, we assume $n \geq m$.

4.1. Mean Distance Function

The mean distance is one of the most efficient distance measures for point sets. First, a mean point is computed for each point set. Then, the distance between the two means is computed by using the orthodromic distance. Formally:

$$D_{mean}(s,t) = \delta\left(\frac{\sum\limits_{s_i \in S} s_i}{n}, \frac{\sum\limits_{t_j \in T} t_j}{m}\right). \quad (2)$$

D_{mean} can be computed in $O(n)$. For our example, the mean of the DBpedia description of Malta is the point (14.48, 35.89). The mean for the Nuts description are (14.33, 35.97). Thus, D_{mean} returns $18.46km$ as the distance between the two means points.

4.2. Max Distance Function

The idea behind this measure is to compute the overall maximal distance between points s_i and t_j. Formally, the maximum distance is defined as:

$$D_{max}(S,T) = \max_{s_i \in S, t_j \in T} \delta(s_i, t_j). \quad (3)$$

[4]http://scholar.google.com/
[5]http://dl.acm.org/
[6]http://link.springer.com/
[7]http://www.sciencedirect.com/
[8]http://portal.isiknowledge.com/
[9]http://www.semantic-web-journal.net/
[10]http://www.websemanticsjournal.org/
[11]http://www.journals.elsevier.com/data-and-knowledge-engineering/
[12]http://www.mendeley.com/

For our example, D_{max} returns $38.59km$ as the distance between the points d_3 and n_6. Due to its construction, this distance is particularly sensitive to outliers. While the naive implementation of *Max* is in $O(n^2)$, [7] introduced an efficient implementation that achieves a complexity of $O(n \log n)$.

4.3. Min Distance Function

The main idea of the *Min* is akin to that of *Max* and is formally defined as

$$D_{min}(s,t) = \min_{s_i \in S, t_j \in T} \delta(s_i, t_j). \quad (4)$$

Going back to our example, D_{min} returns $7.82km$ as the distance between the points d_2 and n_5. Like D_{max}, D_{min} can be implemented to achieve a complexity of $O(n \log n)$ [31,19].

4.4. Average Distance Function

For computing the *average* point sets distance function, the orthodromic distances between all the source-target points pairs is cumulated and divided by the number of point source-target point pairs:

$$D_{avg}(s,t) = \frac{1}{nm} \sum_{s_i \in S, t_j \in T} \delta(s_i, t_j). \quad (5)$$

For our example, D_{avg} returns $22km$. A naive implementation of the *average* distance is $O(n^2)$, but it can be efficiently computed in $O(n \log n)$.

4.5. Sum of Minimums Distance Function

This distance function was first proposed by [23] and is computed as follows: First, the closest point t_j to each point s_i is to be detected, i.e., the point $t_j = \arg\min_{t_k \in t} \delta(s_i, t_k)$. The same operation is carried out with source and target reversed. Finally, the average of the two values is then the distance value. Formally, the *sum of minimums* distance $D_{som}(S,T)$ is defined as:

$$\frac{1}{2}\Big(\sum_{s_i \in S} \min_{t_j \in T} \delta(s_i, t_j) + \sum_{t_i \in T} \min_{s_j \in S} \delta(t_i, s_j)\Big). \quad (6)$$

Going back again to our example, the sum of minimum distances from each of DBpedia points describing Malta to the ones of Nuts is $37.27km$, and from Nuts to DBpedia is $178.58km$. Consequently, D_{som} returns $107.92km$ as the average of the two values. The *sum of minimum* has the same complexity as D_{min}.

4.6. Surjection Distance Function

The *surjection* distance function introduced by [25] defines the distance between two point sets as follows: The minimum distance between the sum of distances of the surjection of the larger set to the smaller one. Formally, the *Surjection* distance is defined as:

$$D_s(S,T) = \min_{\eta} \sum_{(e1,e2) \in \eta} \delta(e_1, e_2), \quad (7)$$

where η is the surjection from the larger of the point sets S and T to the smaller. In to our example, $\eta = (n_1, d_4)$, (n_2, d_1), (n_3, d_2), (n_4, d_3), (n_5, d_4), (n_6, d_1), (n_7, d_1), (n_8, d_1) and (n_9, d_1). Then, D_s returns $184.74km$ as the sum of the orthodromic distances between each of the point pairs included in η. A main drawback of the *surjection* is being biased toward some points ignoring some others in calculations. (i.e. putting more weight in some points more than the others) For instance in our example, η contains 5 different points surjected to d_1, while only one point surjected to d_2.

4.7. Fair Surjection Distance Function

In order to fix the biased drawback of *surjection*, [25] introduces an extension of the surjection distance which is dubbed *fair surjection*. The surjection between sets S and T is said to be *fair* if η' maps elements of S as evenly as possible to T. The *fair surjection* is defined formally as:

$$D_{fs}(S,T) = \min_{\eta'} \sum_{(e1,e2) \in \eta'} \delta(e_1, e_2), \quad (8)$$

where η' is the evenly mapped surjection from the larger of the sets S and T to the smaller. For our example, $\eta' = (n_1, d_1)$, (n_2, d_2), (n_3, d_3), (n_4, d_4), (n_5, d_1), (n_6, d_2), (n_7, d_3), (n_8, d_4) and (n_9, d_1). Then, D_{fs} returns $137.43km$ as the sum of the orthodromic distances between each of the point pairs included in η'.

4.8. Link Distance Function

The link distance introduced by [11] defines distance between two point sets S and T as a relation $R \subseteq S \times T$ satisfying

1. For all $s_i \in S$ there exists $t_j \in T$ such that $(s_i, t_j) \in R$
2. For all $t_j \in T$ there exists $s_i \in S$ such that $(s_i, t_j) \in R$

Formally, The *minimum link distance* $D_l(S,T)$ between two polygons S and T is defined by

$$D_l(S,T) = \min_{R} \sum_{(s_i,t_j)\in R} \delta(s_i, t_j), \tag{9}$$

where minimum is computed from all relations R, where R is a linking between S and T satisfying the previous two conditions. For our example, the small granularity of the Malta descriptions in the datasets at hand leads to D_l having the same results as D_{fs}. See [11] for complexity analysis for *surjection*, *fair surjection* and *link* distance functions.

4.9. Hausdorff Distance Function

The *Hausdorff* distance is a measure of the maximum of the minimum distances between two sets of points. Hausdorff is one of the commonly used approach for determining the similarity between point sets [16]. Formally, the Hausdorff distance is defined as

$$D_h(S,T) = \max_{s_i\in S}\left\{\min_{t_j\in T}\left\{\delta(s_i,t_j)\right\}\right\}. \tag{10}$$

Back to our example, First, the algorithm finds the orthodromic distance between each of the points of DBpedia to the nearest point Nuts, which found to be the distances between the point pairs (d_1, n_5), (d_2, n_5), (d_3, n_4), and (d_4, n_4). Then, D_h is the maximum distance of them, which is between the point d_4 and n_4 equals $34.21km$. [22] introduces two efficient approaches for computing bound Hausdorff distance.

4.10. Fréchet Distance Function

Most of the distances presented before have a considerable common disadvantage. Consider the two curves shown in Figure 2, Any point on one of the curves has a nearby point on the other curve. Therefore, many of the measures presented so far (incl. Hausdorff, min, sum of mins) return a low distance. However, these curves are intuitively quite dissimilar: While they are close on a point-wise basis, they are not so close if we try to map the curves continuously to each other. A distance measure that captures this intuition is the *Fréchet* [14] distance.

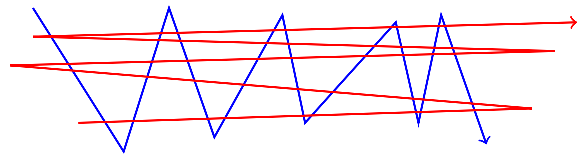


Fig. 2. Fréchet vs other distance approaches

The basic idea behind the Fréchet distance is encapsulated in the following example[13]: *Imagine two formula one racing cars. The first car, A, hurtles over a curve formulated by a first point set. The second car does the same over a curve formulated by the second point set. The first and second car will vary in velocity but they do not move backwards over their curves. Then the Fréchet distance between the point sets is the minimum length of a non-stretchable cable that would be attached to both cars and would not break during the race.*

In order to drive a formal definition of Fréchet distance, First we define *A curve* as a continuous mapping $f : [a,b] \rightarrow V$ with $a, b \in \mathbb{R}$, and $a < b$, where *V denote an arbitrary vector space*. A polygonal curve is $P : [0,n] \rightarrow V$ with $n \in \mathbb{N}$, such that for all $i \in \{0, 1, ..., n-1\}$ each $P[i, i+1]$ is *affine*, i.e. $P(i+\lambda) = (1-\lambda)P(i) + \lambda P(i+1)$ for all $\lambda \in [0,1]$. n is called the length of P. Then, Fréchet distance $D_f(S,T)$ is formally defined as:

$$\inf_{\substack{\alpha[0,1]\rightarrow[s_1,s_n]\\ \beta[0,1]\rightarrow[t_1,t_n]}}\left\{\sup_{t\in[0,1]}\left\{\delta(f(\alpha(t))-g(\beta(t)))\right\}\right\}, \tag{11}$$

where $f : [s_1, s_n] \rightarrow V$ and $g : [t_1, t_n] \rightarrow V$. α, β range over continuous and increasing functions with $\alpha(0) = s_1$, $\alpha(1) = s_n$, $\beta(0) = t_1$ and $\beta(1) = t_n$ only. Computing the Fréchet distance for our example returns $0.3km$. See [2] for a complexity analysis of the Fréchet distance.

Overall, the distance measures presented above return partly very different values ranging from $0.3km$

[13] Adapted from [1].

to $184.74km$ even on our small example. In the following, we evaluate how well these measures can be used for link discovery.

5. Evaluation

The goal of our evaluation was to answer the five questions mentioned in Section 1. To this end, we devised three series of experiments. First, we evaluated the scalability of the ten measures with growing dataset sizes. Then, we measured the robustness of these measures against measurement and granularity discrepancies as well as combinations of both. Finally, we measured the scalability of the measures when combined with the ORCHID algorithm.

5.1. Experimental Setup

5.1.1. Datasets

We used three publicly available datasets for our experiments. The first dataset, *Nuts*[14] was used as core dataset for our scalability experiments. We chose this dataset because it contains fine-granular descriptions of 1,461 geo-spatial resources located in Europe. For example, Norway is described by 1981 points. The second dataset, *DBpedia*[15], contains all the 731,922 entries from DBpedia that possess geometry entries. We chose DBpedia because it is commonly used in the Semantic Web community. Finally, the third dataset, *LGD*, contains all 3,836,119 geo-spatial objects from `http://linkgeodata.org` that are instances of the class `Way`.[16]. Further details to the datasets can be found in [22].

5.1.2. Benchmark

To the best of our knowledge, there is no gold standard benchmark geographic dataset that can be used to evaluate the robustness of geo-spatial distance measures. We thus adapted the benchmark generation approach proposed by [13] and to geo-spatial distance measures. We implemented two modifiers, which both take a polygon s and a threshold as input and return a polygon. The *granularity modifier* M_g regards the threshold $\gamma \in [0, 1]$ as the probability that a point of s will be in the output polygon. To ensure that an empty polygon is never generated, the modifier always includes the first point of s into its output. For all other points s_i, a random number r between 0 and 1 is generated. If $r \leq \gamma$, then s_i is added to the output of the modifier. Else, s_i is discarded. The *measurement error modifier* M_e emulates measurement errors across datasets. To this end, it alters the latitude and longitude of each the points of s by at most the threshold μ. Consequently, the new coordinates of a point s_i are located within a square of size 2μ with s_i at the center. We used a sample of 200 points from each dataset for our discrepancy experiments.

To measure how well each of the distances performed w.r.t. to the modifiers, we first created a reference mapping $M = \{(s, s) \in S\}$ when given a set of input resources S. Then, we applied the modifier to all the elements of S to generate a target dataset T. We then measured the distance between each of the point sets in the set T and the resources in S. For each element of S we stored the closest point $t \in T$ in a mapping M'. We now computed the precision, recall and F-measure achieved within the experiment by comparing the pairs in M' with those in M.

5.1.3. Hardware

All experiments were carried out on a 64-core server running *OpenJDK* 64-Bit Server 1.6.0_27 on *Ubuntu* 12.04.2 LTS. The processors were 12 Hexa-core *AMD Opteron* 6128 clocked at 2.0 GHz. Unless stated otherwise, each experiment was assigned 8 GB of memory and was ran 5 times.

5.2. Scalability Evaluation

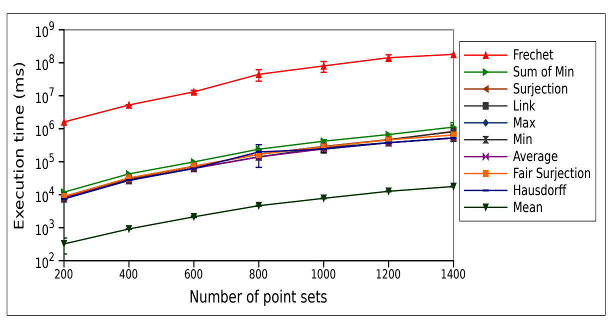


Fig. 3. Scalability evaluation on the Nuts dataset.

To quantify how well the measures scale, we measured the runtime of the measures on fragments of

[14]Version 0.91 available at `http://nuts.geovocab.org/data/` is used in this work

[15]We used version 3.8 as available at `http://dbpedia.org/Datasets`.

[16]We used the RelevantWays dataset (version of April 26th, 2011) of *LinkedGeoData* as available at `http://linkedgeodata.org/Datasets`.

growing size of each of the input datasets. This experiment emulates a naive deduplication on datasets of various sizes. The results achieved on Nuts are shown in Figure 3. We chose to show Nuts because it is the smallest and most fine-granular of our datasets. Thus, the measures achieved here represent an upper bound for the runtime behaviour of the different approach. D_{mean} is clearly the most time-efficient approach. This was to be expected as its algorithmic complexity is linear. While most of the other measures are similar in their efficiency, the Fréchet distance sticks out as the slowest to run. Overall, it is at least two orders of magnitude slower than the other measures. These results give a clear answer to question Q_1, which pertains to the time-efficiency of the measures at hand: D_{mean} is clearly the fastest.

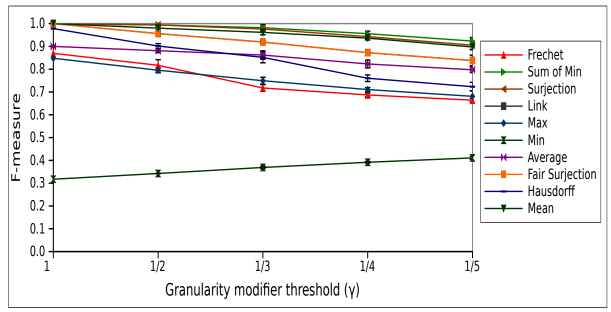
5.3. *Robustness Evaluation*

We carried out three types of evaluations to measure the robustness of the measures at hand. First, we measured their robustness against discrepancies in granularity. Then, we measured their robustness against measurement discrepancies. Finally, we combined discrepancies in measurement and granularity and evaluated all our measures against these. We chose to show only a portion of our results for the sake of space. All results can be found at `http://limes.sf.net`.

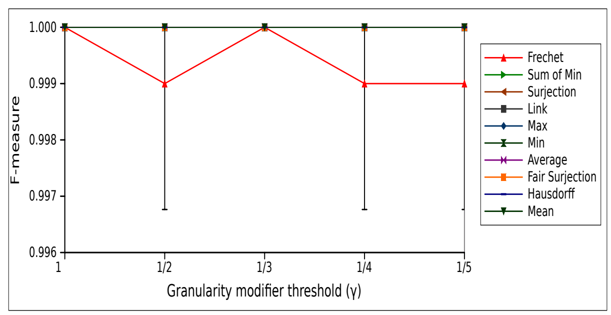
5.3.1. *Robustness against Discrepancies in Granularity*

We measured the effect of changes in granularity on the measures at hand by using the five granularity thresholds 1, $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$ and $\frac{1}{5}$. Note that the threshold of 1 means that the dataset no change was actually carried out. This setting allows us to answer Q_2, which pertains to the measures that are most adequate for deduplication. On Nuts (see Figure 4(a)), our results suggest that D_{min} is the least robust of the measures w.r.t. the F-measure. In addition to being the least time-efficient measure, Fréchet is also not robust against changes in granularity. The best performing measure w.r.t. to its F-measure is the *sum of minimums*, followed closely by the surjection and mean measures. On the DBpedia and LGD datasets, all measures apart from the Fréchet distance perform in a similar fashion (see Figure 4(b)). This is yet simply due the sample of the dataset containing point sets that were located far apart from each other. Thus, the answer to question Q_3 on the effect of discrepancies in granularity is that while the *sum of mins* is the least sensitive to changes in granularity. However, note that sum of mins is closely followed by the mean measure.

The answer to Q_2 can be derived from the evaluation with the granularity threshold set to 1. Here, mean, fair surjection, surjection, sum of mins and link perform best. Thus, mean should be used because it is more time-efficient.



(a) Nuts

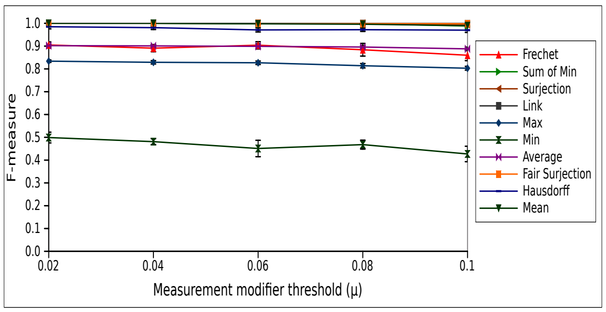


(b) LinkedGeoData

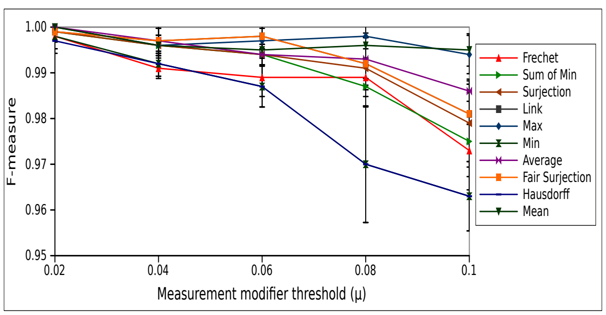
Fig. 4. Comparison of different point set distance measures against granularity discrepancies.

5.3.2. *Robustness against Measurement Discrepancies*

The evaluation of the robustness of the measures at hand against discrepancies in measurement are shown in Figure 5. Interestingly, the results differ across the different datasets. On the Nuts data, where the regions are described with high granularity, five of the measures (mean, fair surjection, link, sum of mins and surjection) perform well. On LGD, the number of points pro resources is considerably smaller. Moreover, the resources are partly far from each other. Here, the Hausdorff distance is the poorest while max and mean perform comparably well. Finally, on the DBpedia dataset, all measures apart from Fréchet are comparable. Our results thus suggest that the answer to Q_4 is as follows: The mean distance is the distance of choice when computing links between geo-spatial datasets which contain measurement errors, especially if the resources described have a high geographical density or the difference in granularity is significant.



(a) Nuts



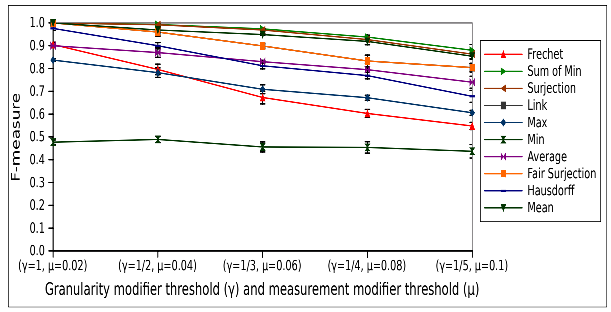
(b) LinkedGeoData

Fig. 5. Comparison of different point set distance measures against measurement discrepancies.

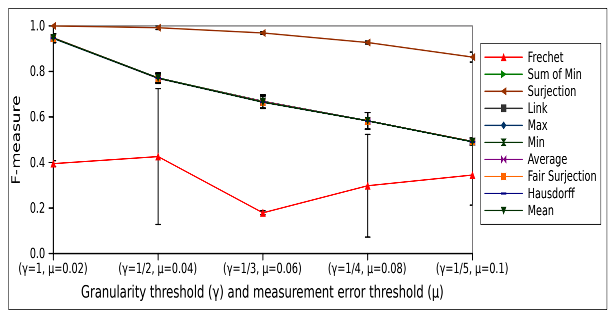
5.3.3. *Overall Robustness*

We emulated the differences across various real geographic datasets by combining the granularity and the measurement modifiers. Given a dataset S, we generated a modified dataset S' using the granularity modifier. The modified dataset was used as input for a measurement modifier, which generated our final dataset T. The results of our experiments are shown in Figure 6. Again, the results vary across the different datasets. While mean performs well on Nuts 6(a) and LGD, it is surjection that outperforms all the other measures on DBpedia 6(b). This surprising result is due to the measurement errors having only a small effect on our DBpedia sample. Thus, after applying the granularity modifier, the surjection value is rarely affected.

Overall, our results suggest that the following answer to Q_5: In most cases, using the mean distance leads to high F-measures. Moreover, mean present the advantage of being an order of magnitude faster than the other approaches. Still, the surjection measure should also be considered when comparing different datasets as it can significantly outperform the mean measure



(a) Nuts



(b) DBpedia

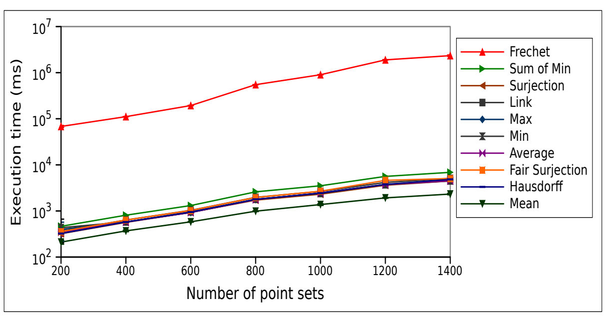
Fig. 6. Comparison of different point set distance measures against granularity and measurement discrepancies.

5.4. *Scalability with* ORCHID

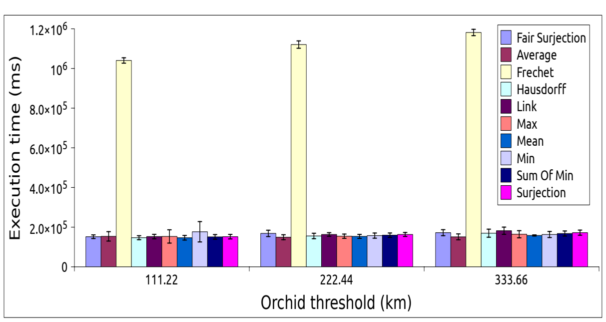
We aimed to know how far the runtime of measures such as mean, surjection and sum of mins can be reduced so as to ensure that these measures can be used on large datasets. We thus combined these measures with the ORCHID approach presented in [22]. The idea behind ORCHID is to improve the runtime of algorithms for measuring geo-spatial distances by adapting an approach akin to divide-and-conquer. ORCHID assumes that it is given a distance measure (not necessarily a metric) m that abides by $m(s,t) \leq \theta \rightarrow \forall s_i \in s\ \exists t_j \in t : \delta(s_i,t_j) \leq \theta$. This condition is obviously not satisfied by all measures considered herein, including min and mean. However, dedicated extensions of ORCHID can be developed for these measures. Overall, ORCHID begins by partitioning the surface of the planet. The points in a given partition are then only compared with points in partitions that abide by the distance threshold underlying the computation.

We used the default settings of the implementation provided in the LIMES framework and the distance threshold of $0.02°$ (2.2km). Figure 7(a) shows the runtime results achieved on the same datasets as Figure 3. Clearly, the runtimes of the approaches can be decreased by up to an order of magnitude. Therewith, ORCHID allows most measures (i.e., all apart from Fréchet) to scale in a manner comparable to that of the mean measure. Therewith, the measures can now be used on the whole of the datasets at hand. For example, all distance measures apart from the Fréchet distance require less than five minutes to run on the whole of the DBpedia dataset (see 7(b)).

Overall, we can conclude that all measures apart from the Fréchet distance are amenable to being used for link discovery. While *mean performs best overall*, *surjection-based and minimum-based measures are good candidates* to use if mean returns unsatisfactory results. The Fréchet distance on the other hand seems inadequate for link discovery. This can yet be due to the point set approach chosen in this paper. An analysis of the Fréchet distance on the description of resources as polygons remains future work.



(a) Nuts



(b) DBpedia

Fig. 7. Scalability evaluation with ORCHID.

5.5. *Experiment on Real Datasets*

We were interested in knowing whether the mean function performs well on real data. Validating link discovery results on geo-spatial data is difficult due to the lack of reference datasets. We thus measured the increase in precision and recall achieved by using geo-spatial information by sampling 100 links from the results of real link discovery tasks and evaluating these links manually. The links were evaluated by the authors who reached an agreement of 100%.

In the first experiment, we computed links between cities in DBpedia and LinkedGeoData by comparing solely their labels by means of an exact match string similarity. No geo-spatial similarity metric was used, leading to cities being linked if they have exactly the same name. Overall only 74% of the links in our sample were correct. The remaining 26% differed in country or even continent. We can assume that a recall of 1 would be achieved by using this approach as a particular city will most probably have the same name across different geo-spatial datasets. Thus, in the best case, linking geo-spatial resources in DBpedia to LinkedGeoData would only lead to an F-measure of 0.85.

In our second experiment, we extended the specification described above by linking two cities if their names were exact matches (which was used in the first experiment) and the mean distance function between their geometry representation returned a value under 100km. In our sample, we achieved a perfect accuracy and thus an F-measure of 1. While this experiment is small, it clearly demonstrates the importance of using geo-spatial information for linking geo-spatial resources. Moreoveor, it suggest that the mean distance is indeed reliable on real data. More experiments yet need to be carried out to ensure that the empirical results we got in this experiment are not just a mere artifact in the data. We will achieve this goal by creating a benchmark for geo-spatial link discovery in future work.

6. Related Work

This paper is related to distance measures for point sets and link discovery. Several reviews on distances for point sets have been published. For example, [12] reviews some of the distance functions proposed in the literature presents efficient algorithms for the computation of these measures. [29] presents an approach to compute the similarity between multiple polylines and a polygon using dynamic programming. [3] focuses on the Hausdorff distance and presents an approach for its efficient computation between convex polygons. While the approach is quasi-linear in the number of nodes of the polygons, it cannot deal with non-convex polygons as commonly found in geographic data. [30] present a similar approach that allows approximating Hausdorff distances within a certain error bound, while [6] presents an exact approach. [24] present an approach to compute Hausdorff distances between trajectories using R-trees within an L_2-space. Fréchet distance is basically used in piecewise curve similarity detection like in case of hand writing recognition. For example, [2] introduces an algorithm for computing Fréchet distance between two polygonal curves, while [8] presents a polynomial-time algorithm to compute the homotopic Fréchet distance between two given polygonal curves in the plane avoiding a given set of polygonal obstacles. [10] provides an approximation of Fréchet distance for realistic curves in near linear time.

There are number of techniques presented in literature that -if applied in combination with the presented distance approaches- can achieve better performance. In order to limit the number of polygons to be compared in deduplication problems, [17] proposed a dissimilarity function for clustering geospatial polygons. A kinematics-based method proposed in [28] approximates large polygon using less number of points is proposed, thus requires less execution time for distance measurement. Yet, another algorithm presented by [26] models non-convex polygons as the union of a set of convex components, the algorithm construct a hierarchical bounding representation based on spheres. [15] shows an approach for the comparison of 3D models represented as triangular meshes. The approach is based on a subdivision sampling algorithm that makes used of octrees to approximate distances. ORCHID [22] was designed especially for the Hausdorff distance but can be extended to deal with other measures.

7. Conclusion and Future Work

In this paper, we presented an evaluation of point set distance measures for link discovery on geo-spatial resources. We evaluated these distances on sample from three different datasets. Our results suggest that while different measures perform best on the data sets we used, the *mean distance measure* is the most time-efficient and overall best measure to use for link discovery. We also showed that all measures apart from the Fréchet distance can scale even on large datasets when combine with an approach such as ORCHID. While working on this paper, we realized the need for a full-fledged benchmark for geo-spatial link discovery. In future work, we will devise such a benchmark and make it available to the community. All the measures presented in this paper were integrated in the LIMES framework available at `http://limes.sf.net`. In future work, we will extend this framework with dedicated versions of ORCHID for the different measures presented herein. Moreover, we will aim to devise means to detect the best measure for any given geo-spatial dataset.

References

[1] Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995.

[2] Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

[3] Mikhail J. Atallah. A linear time algorithm for the hausdorff distance between convex polygons. Technical report, Purdue University, Department of Computer Science, 1983.

[4] Sören Auer, Jens Lehmann, and Sebastian Hellmann. LinkedGeoData - adding a spatial dimension to the web of data. In *Proc. of 8th International Semantic Web Conference (ISWC)*, 2009.

[5] Sören Auer, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. Introduction to linked data and its lifecycle on the web. In *Reasoning Web*, pages 1–75, 2011.

[6] Michael Bartoň, Iddo Hanniel, Gershon Elber, and Myung-Soo Kim. Precise hausdorff distance computation between polygonal meshes. *Comput. Aided Geom. Des.*, 27(8):580–591, November 2010.

[7] Binay K Bhattacharya and Godfried T Toussaint. Efficient algorithms for computing the maximum distance between two finite planar sets. *Journal of Algorithms*, 4(2):121 – 136, 1983.

[8] Erin Wolf Chambers, Éric Colin de Verdière, Jeff Erickson, Sylvain Lazard, Francis Lazarus, and Shripad Thite. Homotopic fréchet distance between curves *or, walking your dog in the woods in polynomial time*. *Computational Geometry*, 43(3):295–311, 2010.

[9] Michelle Cheatham and Pascal Hitzler. String similarity metrics for ontology alignment. In *International Semantic Web Conference (2)*, pages 294–309, 2013.

[10] Anne Driemel, Sariel Har-Peled, and Carola Wenk. Approximating the fréchet distance for realistic curves in near linear time. *Discrete & Computational Geometry*, 48(1):94–127, 2012.

[11] Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34:103–133, 1997.

[12] Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.

[13] A. Ferrara, S. Montanelli, J. Noessner, and H. Stuckenschmidt. Benchmarking Matching Applications on the Semantic Web. In *The Semantic Web: Research and Applications*, 2011.

[14] M.Maurice Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo*, 22(1):1–72, 1906.

[15] Michael Guthe, Pavel Borodin, and Reinhard Klein. Fast and accurate hausdorff distance calculation between meshes. *J. of WSCG*, 13:41–48, 2005.

[16] Daniel P. Huttenlocher, Klara Kedem, and Jon M. Kleinberg. On dynamic voronoi diagrams and the minimum hausdorff distance for point sets under euclidean motion in the plane. In *Proceedings of the Eighth Annual Symposium on Computational Geometry*, SCG '92, pages 110–119, New York, NY, USA, 1992. ACM.

[17] Deepti Joshi, Ashok Samal, and Leen-Kiat Soh. A dissimilarity function for clustering geospatial polygons. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 384–387. ACM, 2009.

[18] B. Kitchenham. Procedures for performing systematic reviews. Technical report, Joint Technical Report Keele University Technical Report TR/SE-0401 and NICTA Technical Report 0400011T.1, 2004.

[19] Michael McKenna and Godfried T Toussaint. Finding the minimum vertex distance between two disjoint convex polygons in linear time. *Computers & Mathematics with Applications*, 11(12):1227–1242, 1985.

[20] David Moher, Alessandro Liberati, Jennifer Tetzlaff, Douglas G Altman, and PRISMA Group. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *PLoS medicine*, 6(7), 2009.

[21] Axel-Cyrille Ngonga Ngomo. On link discovery using a hybrid approach. *J. Data Semantics*, 1(4):203–217, 2012.

[22] Axel-Cyrille Ngonga Ngomo. Orchid - reduction-ratio-optimal computation of geo-spatial distances for link discovery. In *Proceedings of ISWC 2013*, 2013.

[23] I. Niiniluoto. *Truthlikeness*. Synthese Library. Springer, 1987.

[24] Sarana Nutanong, Edwin H. Jacox, and Hanan Samet. An incremental hausdorff distance calculation algorithm. *Proc. VLDB Endow.*, 4(8):506–517, May 2011.

[25] G Oddie. Verisimilitude and distance in logical space. *The logic and epistemology of scientific change, Acta Philosophica Fennica*, 30(2-4):227–42, 1978.

[26] Sean Quinlan. Efficient distance computation between non-convex objects. In *In Proceedings of International Conference on Robotics and Automation*, pages 3324–3329, 1994.

[27] Muhammad Saleem, Shanmukha Sampath Padmanabhuni, Axel-Cyrille Ngonga Ngomo, Jonas S. Almeida, Stefan Decker, and Helena F. Deus. Linked cancer genome atlas database. In *Proceedings of I-Semantics2013*, 2013.

[28] Ediz Saykol, Gürcan Gülesir, Ugur Güdükbay, and Özgür Ulusoy. Kimpa: A kinematics-based method for polygon approximation. In *Advances in Information Systems*, pages 186–194. Springer, 2002.

[29] Mirela Tănase, Remco C Veltkamp, and Herman Haverkort. Multiple polyline to polygon matching. In *Algorithms and Computation*, pages 60–70. Springer, 2005.

[30] Min Tang, Minkyoung Lee, and Young J. Kim. Interactive hausdorff distance computation for general polygonal models. *ACM Trans. Graph.*, 28(3):74:1–74:9, July 2009.

[31] Godfried T. Toussaint and Binay K. Bhattacharya. Optimal algorithms for computing the minimum distance between two finite planar sets. In *Pattern Recognition Letters*, pages 79–82, 1981.