Holistic Entity Clustering for Linked Data

Markus Nentwig*, Anika Groß[†] and Erhard Rahm[‡]
Database Group, Department of Computer Science
University of Leipzig
*nentwig@informatik.uni-leipzig.de, [†]gross@informatik.uni-leipzig.de, [‡]rahm@informatik.uni-leipzig.de

Abstract—**Pairwise link discovery approaches for the Web of Data do not scale to many sources thereby limiting the potential for data integration. We thus propose a holistic approach for linking many data sources based on a clustering of entities representing the same real-world object. Our clustering approach utilizes existing links and can deal with entities of different semantic types. The approach is able to identify errors in existing links and can find numerous additional links. An initial evaluation on real-world linked data shows the effectiveness of the proposed holistic entity matching.**

1. Introduction

Linking entities between sources has been a major effort in recent years to support data integration in the so-called Web of Data. A large number of tools for semi-automatic link discovery has been developed to facilitate the generation of new links (mostly of type `owl:sameAs`) [1]. Linked data sets can be accessed via platforms such as `datahub.io` or `sameas.org`, and repositories such as BioPortal [2] or LinkLion [3] where numerous links for many sources are collected to improve their availability and re-usability. The continuous availability of already determined links is important to avoid their repetitive determination for new applications and use cases.

Despite the advances made, there are significant limitations in the achieved linkage of data sources and in the current approaches for link discovery. First, the degree of inter-linking is still low and automatically generated links are wrong in many cases [4], [5]. Current approaches for link discovery only match two data sources at a time (pairwise linking) resulting in a poor scalability to many sources because the number of possible mappings increases quadratically with the number of sources. To be more precise, one needs up to $\frac{k\cdot(k-1)}{2}$ binary match mappings for k data sources. Hence, fully interlinking 200 sources in the Web of Data would require the determination and maintenance of almost 20.000 mappings.

Existing approaches to determine `owl:sameAs` links also focus on entities of the same type while many sources contain entities of different types (bibliographic datasets contain publication and author entities, geographical datasets contain numerous kinds of entities such as countries, cities, lakes etc.). Furthermore, existing links are hardly utilized when additional links need to be determined. A general problem for the Web of Data is that integrating the information about entities requires users to specify within SPARQL queries the respective data sources and the links to be traversed. This makes it difficult to fully reach and combine the available information about entities.

To address these shortcomings we propose a new clustering-based approach to holistically match entities between many sources. It combines matching entities from k sources in one cluster instead of maintaining a high number of binary links. The matching entities of a cluster with their different properties can be easily fused to derive a more comprehensive, integrated entity representation that can be centrally maintained and accessed, e.g., within a knowledge graph [6], [7]. As sketched in [8], the clustering-based approach also facilitates the integration of additional sources and entities since they only need to be matched with the set of already existing clusters rather than adopting a pairwise linking with numerous different sources. The proposed approach to determine initial entity clusters can optionally build on already existing links and it can deal with entities of different semantic types. When existing links are utilized, the approach is able to identify and eliminate wrong links and it will cluster many previously unconnected entities. The fact that we can utilize existing links shows that the proposed clustering-based approach complements the prevalent pairwise linking, e.g., to provide the fused entities and entity clusters within integrated data sources such as knowledge graphs.

We illustrate the approach by the example records in Table 1 about real geographical entities of different types from five sources. The black lines on the left indicate existing `owl:sameAs` links that we utilize as input to build the clusters. To achieve a good clustering we cannot solely rely on existing links but also have to check the similarity between records; for this purpose we consider the entity labels (names), their semantic type and the geographical coordinates. The table shows that the information about types and coordinates is typically incomplete making it difficult to achieve good match quality. We can also observe potential errors in the given links. For instance, *Lake Louise* is both a city (type 'settlement') and a lake (type 'body of water') so that entity (0) with an unknown type should only link to one of the two. The clustering will lead to many additional matches compared to the initial links. For exam-

id	label	source	type	latitude	longitude
0	Lake Louise (Canada)	NYTimes	-	51.42	-116.23
1	Lake Louise	GeoNames	BodyOfWater	51.41	-116.23
2	Lake Louise, Alberta	DBpedia	Settlement	-	-
3	lake louise alberta	FreeBase	Settlement	51.43	-116.16
4	Lake Louise (Alberta)	DBpedia	BodyOfWater	51.41	-116.23
5	lake louise	FreeBase	BodyOfWater	51.41	-116.23
6	Mystic (Conn)	NYTimes	-	41.35	-71.97
7	N17632379615920	FreeBase	-	41.35	-71.97
8	Mystic	GeoNames	Settlement	41.35	-71.97
9	Black Hill	GeoNames	Mountain	53.54	-1.89
10	Black Hill	DBpedia	Mountain	53.53	-1.88
11	Black Hill	LinkedGeoData	Mountain	53.96	-1.85
12	Black Hill	LinkedGeoData	Mountain	54.69	-2.15
13	katmandu (nepal)	NYTimes	-	27.72	85.32
14	Kathmandu	GeoNames	Settlement	27.7	85.32
15	Kathmandu	DBpedia	Settlement	27.7	85.33
16	Kathmandu	FreeBase	Settlement	27.7	85.37

Table 1: Sample entities from the geographical domain. Black lines on the left denote same-as links between entities.

ple, the cluster of the last four records about *Kathmandu* will implicitly represent six matches, i.e., three new ones (`(14)-(15)`, `(14)-(16)`, `(15)-(16)`).

We make the following contributions:

- We propose a holistic clustering-based approach for matching entities in the Web of Data that avoids the determination and maintenance of a huge number of pairwise mappings.
- We outline a first method for holistic clustering of real, possibly incompletely described entities of different semantic types. The approach utilizes existing links and can identify incorrect links as well as many additional links. After some preprocessing the approach determines initial clusters by computing connected components that are subsequently refined by cluster splits and merges.
- We provide an initial evaluation of the proposed approach for matching real entities from geographical data sources showing the effectiveness of the new approach.

In the next section we outline preliminaries on the data model and problem statement and sketch a base approach to incrementally determine entity clusters. In Section 3, we describe and illustrate the new holistic entity clustering that builds on existing same-as links to determine entity clusters. We present preliminary evaluation results in Section 4. Finally, we discuss related work (Section 5) and conclude.

2. Preliminaries

We first define the problem of holistic entity clustering for linked data. The main approach proposed in this paper (Section 3) is based on already existing links to determine the initial clusters. Alternatively, we may incrementally determine and extend the clusters. This approach is sketched in the second part of this section.

2.1. Problem statement

We consider a set of k data sources $S_1, \ldots, S_k$ containing entities of different semantic types $T_1, \ldots, T_m$. Each entity e is referenced by an URI and has a semantic type T_i, e.g., `dbo:Settlement`. Entities also have a label and domain-specific properties, e.g., geo-coordinates. For better readability we use short ids like `(1)` instead of URIs in Table 1 and in our examples. Two entities of different sources can be connected by a `owl:sameAs` link if they are found to represent the same real-world object. All same-as links between two sources S_i and S_j ($1 \leq i, j \leq k$) constitute a binary equivalence *mapping* $M_{i,j} = \{(e_1, e_2, sim) | e_1 \in S_i, e_2 \in S_j, sim[0,1], i \neq j\}$. Link discovery tools can assign a similarity value sim to indicate the strength of a connection with 1 denoting equality (highest similarity). Our main algorithm (Section 3) uses a set of existing (or pre-determined) mappings $\mathcal{M} = \bigcup_{i,j=1}^{k} M_{i,j}$ as input in addition to the set of associated entities $\mathcal{E}$ of the k data sources.

The goal of holistic entity clustering is to compute for each semantic entity type T_i ($1 \leq i \leq m$) a set of n_i clusters $\mathcal{C}_i = \{c_{i,1}, \ldots, c_{i,n_i}\}$ such that each cluster $c_{i,j}$ ($1 \leq i \leq m$, $1 \leq j \leq n_i$) only includes matching entities of type T_i (denoting the same real-world object) and that different clusters represent different entities. In this paper, we consider duplicate-free data sources, such that a cluster can contain at most k entities. Each cluster of $z \leq k$ entities represents $\frac{z \cdot (z-1)}{2}$ match pairs and is thus generally a much more compact representation than with the use of binary links. Clusters $c_{i,j}$ also have a *cluster representative* $r_{i,j}$ derived from the cluster entities to provide a unified entity representation. In our algorithm, we will use the representatives to compare entities with clusters in order to avoid a more expensive comparison with all cluster members.

2.2. Incremental clustering

Clustering entities from k sources can be performed in an incremental way like already depicted in [8] for schema and ontology models. One can bootstrap the clustering process with one of the sources, e.g., the largest one or a source with known high data quality, and use each of its entities as an initial cluster (assuming duplicate-free sources). Then one matches the entities of one source after another with the cluster representatives to decide on the best-matching cluster (of the respective entity type) or whether an entity should form a new cluster. This process can be continued until all sources are matched and clustered. Once the clusters have been established it is relatively easy to integrate additional entities of the existing sources or additional sources. Such additional entities have to be matched again to the existing clusters to be merged into existing clusters or to form new clusters.

In all these steps it is important to keep separate clusters per entity type and to match entities only with the clusters of the respective type in order to ensure that only entities of the same type are clustered and to reduce the number of necessary comparisons. For this purpose, it is necessary that all entities have comparable semantic entity types. Unfortunately, existing sources in the Web of Data use largely different entity types even for the same domain so that it becomes necessary to align these different types with each other. Furthermore, the entities should be assigned a semantic type from the consolidated set of types in a preprocessing step before matching and clustering. We will apply such an approach in the clustering scheme of Section 3.

A detailed investigation of the incremental approach is beyond the scope of this paper. We rather focus on an alternate approach to determine entity clusters by building on existing same-as links. This scheme can then also apply the incremental approach to extend the determined sets of entity clusters by additional entities. While we leave the detailed evaluation of the sketched incremental approach for future work, we can roughly estimate its required number of match operations and compare it with the use of pairwise linking. Assuming l entities per source, pairwise linking of k sources requires $\frac{k\cdot(k-1)}{2}\cdot l^2$ match operations between entities. We can reduce the number of match operations by only matching entities of the same type with each other. For m entity types, this reduces the number of match operations to $\frac{k\cdot(k-1)}{2}\cdot\frac{l^2}{m}$. For incremental clustering, the number of match computations is restricted by the number of clusters of the respective entity type for any entity of any source but the first. We can approximate the number of clusters n_i for entity type T_i as $n_i = \frac{k\cdot l}{z_i\cdot m}$ with the average cluster size z_i ($1 \le z_i \le k$). The total number of match comparisons is thus in the order of $(k-1)\cdot l\cdot\frac{n_i}{2} = \frac{(k-1)\cdot k}{2\cdot z_i}\cdot\frac{l^2}{m}$ considering that on average an entity has only to be compared with the clusters from half the number of sources. In comparison, the number of match operations for the holistic approach is thus reduced by a factor of z_i compared to pairwise linking since we match only with the cluster representatives rather than with any cluster member. For an average cluster size $z_i = k/2$ this results in savings of at least one order of magnitude for $k = 20$ or more data sources. As discussed before, the cluster representation provides additional advantages as it provides physically integrated entity information avoiding the need to traverse different sources during query execution. Furthermore, clusters are more compact than many binary links and easier to maintain when new sources and new entities need to be integrated.

3. Holistic clustering based on existing links

In this section we outline our approach to determine entity clusters based on existing mappings between the data sources to integrate. A high-level description of the approach is given in Algorithm 1. Figure 1 illustrates the main workflow and its application to the sample records from Table 1. In addition to the existing set of links and the associated entities from different sources, the input of the algorithm includes domain knowledge about the semantic entity types, a similarity function f_{sim} and similarity thresholds t_s, t_m to determine the similarity of entities and clusters. While our algorithm is generic, it can be customized to specific domains by providing appropriate background knowledge, similarity functions and thresholds. For the considered geographical domain, the similarity function determines a combined similarity from the string (trigram) similarity on normalized labels, the semantic type similarity and the normalized geographical distance. The algorithm consists of four major steps that we will describe in the rest of this section in more detail: preprocessing, initial clustering (connected components), cluster decomposition, and cluster merge.

Algorithm 1: Holistic Clustering

```
Input: Set of entities $\mathcal{E}$ from $k$ sources, link set $\mathcal{M}$,
       domain knowledge $D$, simFunc $f_{sim}$, thresholds
       $t_s, t_m$
Output: Set of clusters $\mathcal{C}$
1  $\mathcal{C} \leftarrow \emptyset$
2  $\mathcal{E},\mathcal{M} \leftarrow$ preprocessing($\mathcal{E},\mathcal{M},f_{sim},D$)
                                   /* initial clustering */
3  $\mathcal{C}_{init} \leftarrow$ computeConnectedComponents($\mathcal{E}, \mathcal{M}$)
4  $\mathcal{M}_c \leftarrow$ computeLinkSim($\mathcal{C}_{init}, f_{sim}$)
5  $\mathcal{C}_{init} \leftarrow$ refineConnectedComponents($\mathcal{C}_{init},\mathcal{M}_c$)
                                /* cluster decomposition */
6  foreach $c \in \mathcal{C}_{init}$ do
7      $\mathcal{C}_{split} \leftarrow$ groupByType($c,\mathcal{M}_c$)
8      $\mathcal{C}_{split} \leftarrow$ simBasedRefinement($\mathcal{C}_{split},\mathcal{M}_c,t_s$)
9      $\mathcal{C}_{split} \leftarrow$ createRepresentatives($\mathcal{C}_{split}$)
10     $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}_{split}$
                        /* create cluster mapping $\mathcal{CM}$ */
11 $\mathcal{CM} \leftarrow$ computeClusterSim($\mathcal{C},f_{sim},t_m$)
12 while $\mathcal{CM} \neq \emptyset$ do
13     $(c_1, c_2) \leftarrow \mathcal{CM}.getBestMatch()$
                                       /* cluster merge */
14     $c_m \leftarrow merge(c_1, c_2)$
15     $\mathcal{C} \leftarrow \mathcal{C} \setminus \{c_1, c_2\} \cup \{c_m\}$
16     $\mathcal{CM} \leftarrow$ adaptMapping ($\mathcal{CM},\mathcal{C},c_m,c_1,c_2,f_{sim},t_m$)
17 return $\mathcal{C}$
```

Preprocessing

During preprocessing we normalize property values needed for the similarity computation. In our case, we transform the label property to lower case and remove words in parentheses and after delimiters. We further harmonize information about the semantic types of entities and check that the input mappings do not violate the assumption of duplicate-free data sources. As we will see, we identify and eliminate inconsistent links already during preprocessing to start the clustering process with cleaned input data.

Information about the semantic type of entities differs substantially between sources or may be missing. For instance, DBpedia uses *City* and *Town* whereas Freebase has a type *citytown* and other related types. To overcome such differences, we use background knowledge about the equivalence and comparability of entity types of different sources to harmonize the type information. For this study, we manually determined this type mapping for our geographical

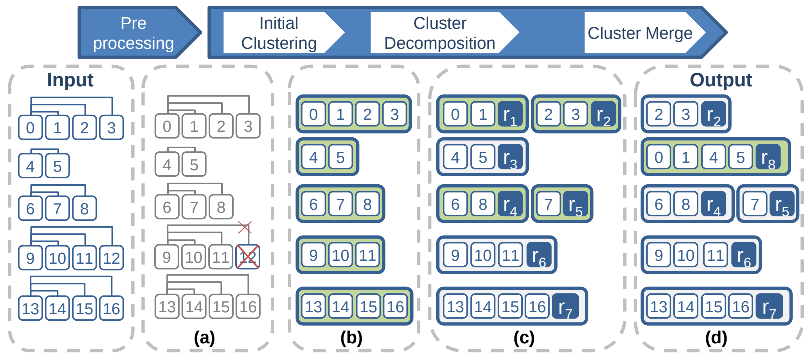

Figure 1: Application of holistic clustering to running example.

sources. Alternatively, the mapping could be constructed with the help of schema or ontology matching approaches based on linguistic and structural matching techniques [9], [10]. Based on the type mapping we simplified numerous types to more general ones, e.g., the types *city*, *village* or *suburb* are treated as type *Settlement*. After harmonizing the type information, we remove already all links where the linked entities have incompatible types. Note that we do not exclude links to entities with missing type information, e.g., entity (0).

For duplicate-free data sources, each entity should have at most one equivalent entity in any other data source. We therefore check whether the input mappings observe this one-to-one cardinality restriction. In Table 1, this is not the case for the Geonames entity (9) about *Black Hill* that links to two LinkedGeoData entities `(11,12)`. In such cases, we only keep the best link for an entity (based on the similarity function f_{sim}) to obey the one-to-one criterion. In our example, we discard the link `(9)-(12)` (see Figure 1 a) since `(9)` is geographically closer to `(11)` than to `(12)` according to the coordinates in Table 1.

Initial Clustering

Using the preprocessed entities and mappings we first identify a set of initial clusters ($\mathcal{C}_{\text{init}}$) by computing all connected components as the transitive closure from the given links (see Algorithm 1 line 3). Each resulting connected component builds an initial cluster c covering all entities that are directly or indirectly connected via a same-as link in $\mathcal{M}$. In our running example, we create five different clusters covering 2-4 entities (see Figure 1 b).

If the data sources are not really duplicate-free or if there are errors in the predetermined same-as links, it may happen that the connected components contain more than one entity per data source and more than k entities in total, despite the preprocessing ensuring that any entity of one source is linked to at most one entity of a second source. For example, assume entities a_1 and a_2 from the same source and entities b_1 and c_1 from two different sources. The links $(a_1 - b_1), (b_1 - c_1), (c_1 - a_2)$ obey the 1:1 restriction but result in a connected component of all four entities of three sources. We thus determine within procedure `refineConnectedComponents` (line 5 of Algorithm 1) connected components with more than one entity per data source and eliminate entities to ensure the restriction to enforce the assumption of duplicate-free sources. The entities to be removed are selected based on the similarity to other entities in the cluster. For this purpose we calculate the similarity between any two entities of a cluster using similarity function f_{sim} and keep the result in link set $\mathcal{M}_c$ (line 4 of Algorithm 1). Based on these similarities, we keep from each source with multiple entities only the entity with the maximal similarity to the other entities.

Cluster Decomposition

The initially created clusters can contain entities that should actually be separated, e.g., due to wrong input links or because of an insufficiently high transitive similarity between entities. For decomposition we use two main approaches. First, we split clusters with elements of different or incompatible semantic types. Second, we split clusters containing entities with insufficient similarity to other cluster members. In Algorithm 1, we apply the decomposition approach to each cluster in $\mathcal{C}_{\text{init}}$ (see line 6-10). We first apply the two kinds of cluster decomposition (*GroupByType* and *Similarity-based Refinement*). Finally, we compute a cluster representative for each of the resulting clusters (line 9). The resulting clusters from the decomposition phase ($\mathcal{C}_{\text{split}}$) are successively added to the result set of clusters $\mathcal{C}$ (line 10).

Type-based Grouping. While we eliminate links with incompatible semantic types during preprocessing, there are entities without type information that can lead to clusters with entities of different types during the initial clustering. For our running example, this is the case for the cluster `(0,1,2,3)` (see Figure 1 b). Our method *groupByType* splits such clusters into several smaller sub-clusters $\mathcal{C}_{\text{split}}$ with entities of the same type. Entities without semantic types are then added to the sub-cluster of their most similar neighbor using the previously computed link similarities in $\mathcal{M}_c$. For the considered cluster of our example, we first build sub-clusters `(2,3)` for type *Settlement* and the singleton cluster `(1)` of type *BodyOfWater*. The untyped entity `(0)` is assigned to the cluster of the best matching (geographically closer) entity `(1)` resulting in sub-cluster `(0,1)`.

Similarity-based Refinement. We further split clusters based on the computed intra-cluster similarity between entities (line 8 in Algorithm 1). Algorithm 2a and 2b show the computation of the similarity-based refinement in more detail. Algorithm 2a just covers the iteration over the set of clusters $\mathcal{C}_{\text{split}}$ determined by `groupByType` and calls the actual refinement function `simCRefine` (Algorithm 2b).

Algorithm 2a: simBasedRefinement

```
Input: Set of clusters $\mathcal{C}_{\text{input}}$ resulting from type-based
       grouping, link set $\mathcal{M}_c$, split threshold $t_s$
Output: Set of clusters $\mathcal{C}_{\text{result}}$
1 $\mathcal{C}_{\text{result}} \leftarrow \emptyset$
2 foreach $c \in \mathcal{C}_{input}$ do
3     $\mathcal{C}_{\text{result}} \leftarrow \mathcal{C}_{\text{result}} \cup$ simCRefine($c$,$\mathcal{M}_c$,$t_s$)
4 return $\mathcal{C}_{result}$
```

Algorithm 2b: simCRefine

```
Input: Cluster $c$, link set $\mathcal{M}_c$, split threshold $t_s$
Output: Set of clusters $\mathcal{C}_{\text{result}}$
1 $e$,$asim_{min} \leftarrow$ getMinAvgSimEntity($c$,$\mathcal{M}_c$)
2 if $asim_{min} < t_s$ then
3     $c \leftarrow c \setminus \{e\}$
4     return createCluster($e$) $\cup$
        simCRefine($c$,$\mathcal{M}_c$,$t_s$)
5 else
6     return $c$
```

For each cluster, we determine the entity e with lowest average similarity $asim_{min}$ (line 1 in Algorithm 2b) of its links to other cluster members using similarity values from the previously determined similarities in $\mathcal{M}_c$. If the average similarity is not below a certain threshold t_s, we do not split the cluster but leave the cluster unchanged (line 6 in Algorithm 2b). Otherwise (lines 2-4) we separate entity e and recursively call the similarity-based refinement for the reduced cluster to possibly identify further entities to separate. In the merge phase, such separated entities may be added to other more similar clusters. In our running example, the cluster `(6,7,8)` is decomposed into `(6,8)` and `(7)` as shown in Figure 1 c, since entity `(7)` has a low similarity with `(6)` and `(8)` due to its completely different label (see Table 1).

Cluster Representative. For each cluster in the output $\mathcal{C}_{\text{input}}$ of the previous steps we create a cluster representative (line 9 in Algorithm 1) to facilitate the computation of inter-cluster similarities in the merge step. The representatives of the final clusters can also be used to efficiently match new entities, e.g., from additional data sources. We create the representative by combining the properties from all entities in a cluster and select a preferred value for each property with multiple values, e.g., based on a majority consensus, the maximal length of string values or pre-determined source priorities. In the representative, we also keep track of the data sources represented in the cluster (this helps to avoid considering merges with entities of already covered data sources). For our use case, we use the longest string value for the preferred label value and prefer geo-coordinates from GeoNames and DBpedia. For the example cluster `(2,3)` the representative r_2 has label *lake louise alberta*, type *Settlement*, coordinates (51.43, -116.16) and sources (*DBpedia*, *Freebase*).

Cluster Merge

The last step of our holistic clustering approach is the possible merging of clusters below the maximally possible cluster size k. For this purpose, we first apply method *computeClusterSim* (line 11) to determine the similarity between clusters by applying similarity function f_{sim} on the cluster representatives. This operation is likely expensive as it incurs a quadratic complexity w.r.t. the number of clusters. We can reduce the number of comparisons by only considering clusters with fewer than k elements. We further do not consider all cluster pairs that differ in their entity type or that overlap in their sets of covered data sources. The cluster mapping $\mathcal{CM}$ computed for the remaining cluster pairs is restricted to the most similar pairs of clusters with a similarity exceeding the merge similarity threshold t_m.

Cluster merging is an iterative process (lines 12-16 of Algorithm 1) that continues as long as there are merge candidates in $\mathcal{CM}$. In each iteration we select the pair of clusters (c_1, c_2) with the highest similarity from $\mathcal{CM}$ and merge it into a new cluster c_m (line 13-14). This merging also includes the computation of a new representative for c_m. The "old" clusters c_1 and c_2 are removed from $\mathcal{C}$ and the new cluster c_m is added (line 15). We further need to adapt $\mathcal{CM}$ by removing all cluster pairs involving either c_1 or c_2. Furthermore, we extend $\mathcal{CM}$ by similar cluster pairs for the new cluster c_m if c_m has fewer than k elements. For this purpose we determine the similarity of c_m with all other clusters of the same type and with different sources. The termination of the loop and the merge step is guaranteed since we reduce the number of clusters in each iteration. The number of potential merge candidates is further reduced for increasing cluster sizes. Applying the approach to our example leads to the merging of $(0,1,r_1)$ and $(4,5,r_3)$ into the new cluster $(0,1,4,5,r_8)$ (see Figure 1 c,d) due to a high label, type and geo-coordinate similarity as apparent from Table 1.

For our running example, the proposed approach could holistically cluster matching entities from five data sources thereby finding previously unknown links and eliminating wrong existing links for improved data quality. The six clusters in the result set (Figure 1 d) implicitly represent 17 pairwise entity links compared to 12 initially given links (Table 1) from which 3 turned out to be incorrect. In particular, we could now identify matches between previously unconnected sources such as GeoNames and Freebase.

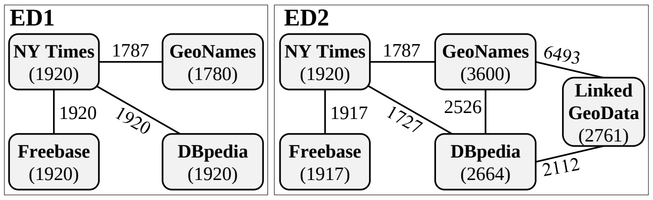

Figure 2: Data set structures for ED1 and ED2 with number of entities and links.

4. Evaluation

We evaluate our holistic clustering approach using two datasets. Evaluation dataset 1 (ED1) corresponds to the DI location subtask of the OAEI 2011 Instance Matching benchmark [1] with links of presumed high quality. Evaluation dataset 2 (ED2) has been downloaded from a LOD link repository (LinkLion [2]). Figure 2 shows the number of links between the four (ED1) resp. five (ED2) geographical data sources and the number of entities that are interconnected by these links. We retrieved additional entity properties via SPARQL endpoints or REST APIs in the respective sources in 2015. Still, in ED1 geo-coordinates were missing for 1009 entities (13.4%) and the type information even for 2525 entities (33.5%). Similarly, in ED2 957 entities (7.4%) have no geo-coordinates and 2722 (21.2%) do not cover type information. For instance, NY Times entities do not provide any type information. We use the similarity function described in Section 3. For the similarity thresholds t_s, t_m we tested different settings and use a default of 0.7 that showed to produce good results w.r.t. cluster sizes and accuracy.

We first evaluate the resulting cluster sizes for the different phases of our holistic clustering approach applied to these datasets (Figure 3). During the preprocessing (not shown in the Figure), we already removed seven wrong NYT-GeoNames links based on the one-to-one cardinality restriction for ED1. In ED2, the situation is more complex due to many one-to-one violations (e.g., several entities can be linked among each other between two data sources). We therefore removed 6921 links, including over 5500 LinkedGeoData - GeoNames links, to hold the one-to-one cardinality for each entity. For instance, there is a group of 94 entities interlinked by 367 links. Each of the 94 entities represents a 'Black Hill' whereof 3 entities occur in DBpedia, 45 in LinkedGeoData and 48 in GeoNames. Within each source the entities do not represent duplicates but are actually different 'Black Hills'. In particular, there are 364 links between the 45 LinkedGeoData and 48 Geo-Names entities, i.e., many entities are interlinked with several entities in the respective other source. During preprocessing we therefore keep only the best link for each entity.

As shown in Figure 3, the initial clustering leads to clusters of sizes 3 and 4 for ED1 whereas ED2 provides clusters of size 2–5. This satisfies the condition that a cluster should not cover more entities than considered data sources. Applying the type-based grouping and similarity-based refinement results in a significant number of cluster splits and clusters of size 1 and 2 due to incompatible entity types and partially low intra-cluster similarity. In particular, the similarity-based decomposition points out that several clusters contain dissimilar entities. During the merge phase some of the smaller clusters can be merged into larger ones leading to more clusters of sizes 3, 4 (and 5). In particular, 15 (23) singleton clusters could be merged into clusters of size 2, 3 (and 4) for ED1 (ED2). Overall, the resulting clusters in ED1 represent 10423 links with 4803 new links compared to the input link set. In particular, we could cluster many entities from the previously unconnected sources GeoNames, DBpedia and Freebase. For ED2, we had to remove a high number of links that violate the one-to-one cardinality restriction (preprocessing) leading to 9641 input links for the initial clustering. Similar to ED1, we then identify 4411 new links for ED2 in the subsequent workflow phases.

Evaluating the quality of the resulting clusters and thus the linking quality is challenging as it requires a perfect clustering for comparison. Determining such a perfect clustering is inherently difficult even for humans and very time-consuming. We therefore evaluate the quality for a sample of the result clusters in this initial evaluation (see Figure 4). We randomly selected 4–5% of the clusters according to the distribution of the cluster sizes in the datasets. This leads to 82 clusters for ED1 and 140 clusters for ED2. We manually check the accuracy of the created clusters with all implicitly contained links. The *cluster accuracy* denotes the percentage of correct clusters (clusters that only contain entities with same-as semantics) in all sampled clusters, and the *link accuracy* is the proportion of correctly created links. For ED1, all determined links in clusters of size 2 are correct such that the cluster and link accuracy is 100%. For cluster size 3 there is one wrong cluster with two wrong links and one wrong cluster of size 4 with 4 wrong links. The cluster of size 4 should have been actually separated into two clusters of size 2. The evaluated clusters in ED2 showed to be very accurate. Only one cluster of size 3 contained two incorrect links.

Overall, we achieve very high cluster accuracy of 97.5% (99.3%) and a link accuracy of 98.6% (99.4%) for ED1 (ED2). This meets our requirements of creating highly accurate clusters that can later be iteratively expanded by adding further entities. So far, we still have clusters of the size 1 and need to check whether those entities build own clusters or need to be added to other clusters. Based on these results we will extend our dataset samples and do more extensive evaluations w.r.t. the cluster quality in the future.

5. Related Work

Link discovery has been studied intensively and a large number of approaches and prototypes has been developed as surveyed in [1]. Virtually all approaches determine links

1. OAEI 2011 IM: http://oaei.ontologymatching.org/2011/instance/
2. LinkLion: http://www.linklion.org/

ED1

cluster size	initial clustering	decomposition type-based	decomposition sim-based	cluster merge
1	-	50	174	159
2	-	115	153	154
3	140	180	228	229
4	1780	1680	1594	1597

ED2

cluster size	initial clustering	decomposition type-based	decomposition sim-based	cluster merge
1	-	46	362	339
2	711	794	836	831
3	759	773	817	807
4	1067	1011	1063	1075
5	586	580	432	435

Figure 3: Cluster sizes in workflow phases for ED1 and ED2.

ED1

cluster size	|cluster|	|sampled clusters|	cluster accuracy	link accuracy
2	154	6	100.0	100.0
3	229	9	88.89	92.59
4	1597	64	98.44	98.96
	1980	**82**	**97.47**	**98.56**

ED2

cluster size	|cluster|	|sampled clusters|	cluster accuracy	link accuracy
2	831	37	100.0	100.0
3	807	36	100.0	100.0
4	1075	48	97.91	98.61
5	435	19	100.0	100.0
	3148	**140**	**99.29**	**99.36**

Figure 4: Cluster accuracy for a sample of result clusters in ED1 and ED2.

between only two sources and for entities of one semantic type. A few approaches such as [11], [12] try to utilize existing mappings for deriving additional mappings, e.g., by their transitive composition. Similarly, [13] uses data from multiple linked data sources to detect weak or not existing relations based on the transitivity of correct links. In [14] the authors aim at improving the quality of joins on Linked Open Data by determining highly connected entity groups in a set of given links using metrics such as edge betweenness.

Even these approaches reusing existing links focus on deriving new or correcting existing pairwise links. By contrast our goal is to holistically cluster entities from many data sources and centrally maintain such clusters for easy usability and extensibility. Our approach can deal with entities of different semantic types and is able to find many additional links and to identify and eliminate wrong links for improved match quality.

A holistic matching of concepts in LOD sources has been proposed in [15]. The authors first apply a topical grouping of concepts and then perform pairwise matching of concepts within groups (based on keywords from the concept labels and descriptions) to finally determine clusters of matching concepts. The approach is interesting as it tries to holistically combine conceptual knowledge across data sources, e.g., as useful for the construction of knowledge graphs. However, the approach suffered from scalability and coverage limitations and does not address clustering of entities as in our scheme.

Clustering-based approaches have also been studied for entity resolution [16] outside the Web of Data, however, mainly for only one or two data sources. For two sources, proposed clustering approaches have similarities to our scheme in that they derive the clusters from a binary mapping consisting of pairs of matching entities (correspondences) [17]. Some approaches construct a similarity graph from the match correspondences and determine subgraph clusters of connected and highly similar entities [18], [19]. These approaches are not only limited to two data sources but also consider only one type of entities. They also do not consider the data quality issues we had to deal with regarding wrong links and missing property values.

6. Conclusion

We proposed a new holistic approach for clustering-based link discovery for many data sources. The approach utilizes existing links and can match entities of different semantic types. The determined entity clusters facilitate the integration of more data sources without having to individually link them to each other data source. An initial evaluation for linked data from the geographical domain confirmed that the new approach holds great promise as it can identify wrong links and many additional links even between previously unconnected sources.

In the future, we will evaluate the scalability and quality of our approach on larger datasets and more sources from different domains based on a parallel Hadoop-based implementation that is currently under development. We also plan a detailed comparison with the incremental construction of entity clusters sketched in Section 2.

Acknowledgments

The research was supported by the Deutsche Forschungsgemeinschaft (DFG) grant number RA 497/19-2.

References

[1] M. Nentwig, Hartung, A.-C. Ngonga Ngomo, and E. Rahm, “A Survey of Current Link Discovery Frameworks,” *Semantic Web Journal*, 2016.

[2] N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M. D. Storey, C. G. Chute, and M. A. Musen, “BioPortal: ontologies and integrated data resources at the click of a mouse,” *Nucleic Acids Research*, vol. 37, no. Web-Server-Issue, pp. 170–173, 2009. [Online]. Available: http://dx.doi.org/10.1093/nar/gkp440

[3] M. Nentwig, T. Soru, A.-C. N. Ngomo, and E. Rahm, “LinkLion: A Link Repository for the Web of Data,” in *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, 2014, pp. 439–443. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11955-7_63

[4] A.-C. Ngonga Ngomo and S. Auer, “LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data,” in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three*, ser. IJCAI’11. AAAI Press, 2011, pp. 2312–2317. [Online]. Available: http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-385

[5] D. Faria, E. Jiménez-Ruiz, C. Pesquita, E. Santos, and F. M. Couto, “Towards Annotating Potential Incoherences in BioPortal Mappings,” in *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, 2014, pp. 17–32. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11915-1_2

[6] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, “Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion,” in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14, New York, NY, USA - August 24 - 27, 2014*, 2014, pp. 601–610. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623623

[7] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, “A Review of Relational Machine Learning for Knowledge Graphs,” *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2016. [Online]. Available: http://dx.doi.org/10.1109/JPROC.2015.2483592

[8] E. Rahm, *The Case for Holistic Data Integration*. Cham: Springer International Publishing, 2016, pp. 11–27. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-44039-2_2

[9] ——, “Towards Large-Scale Schema and Ontology Matching,” in *Schema Matching and Mapping*, ser. Data-Centric Systems and Applications, Z. Bellahsene, A. Bonifati, and E. Rahm, Eds. Springer Berlin Heidelberg, 2011, pp. 3–27. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-16518-4_1

[10] P. Shvaiko and J. Euzenat, “Ontology Matching: State of the Art and Future Challenges,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 158–176, 2013. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2011.253

[11] C. Böhm, G. de Melo, F. Naumann, and G. Weikum, “LINDA: Distributed Web-of-Data-Scale Entity Matching,” in *21st ACM International Conference on Information and Knowledge Management, CIKM’12, Maui, HI, USA, October 29 - November 02, 2012*, 2012, pp. 2104–2108. [Online]. Available: http://doi.acm.org/10.1145/2396761.2398582

[12] M. Hartung, A. Groß, and E. Rahm, “Composition Methods for Link Discovery,” in *Datenbanksysteme für Business, Technologie und Web (BTW), 15. Fachtagung des GI-Fachbereichs ”Datenbanken und Informationssysteme” (DBIS), 11.-15.3.2013 in Magdeburg, Germany. Proceedings*, 2013, pp. 261–277.

[13] A. N. Ngomo, M. A. Sherif, and K. Lyko, “Unsupervised Link Discovery through Knowledge Base Repair,” in *The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings*, 2014, pp. 380–394. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-07443-6_26

[14] J. Kalo, S. Homoceanu, J. Rose, and W. Balke, “Avoiding Chinese Whispers: Controlling End-to-End Join Quality in Linked Open Data Stores,” in *Proceedings of the ACM Web Science Conference, WebSci 2015, Oxford, United Kingdom, June 28 - July 1, 2015*, 2015, pp. 5:1–5:10. [Online]. Available: http://doi.acm.org/10.1145/2786451.2786466

[15] T. Grütze, C. Böhm, and F. Naumann, “Holistic and Scalable Ontology Alignment for Linked Open Data,” in *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012*, 2012. [Online]. Available: http://ceur-ws.org/Vol-937/ldow2012-paper-08.pdf

[16] P. Christen, *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*, ser. Data-Centric Systems and Applications. Springer, 2012. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31164-2

[17] O. Hassanzadeh, F. Chiang, R. J. Miller, and H. C. Lee, “Framework for Evaluating Clustering Algorithms in Duplicate Detection,” *PVLDB*, vol. 2, no. 1, pp. 1282–1293, 2009. [Online]. Available: http://www.vldb.org/pvldb/2/vldb09-1025.pdf

[18] A. Gruenheid, X. L. Dong, and D. Srivastava, “Incremental Record Linkage,” *PVLDB*, vol. 7, no. 9, pp. 697–708, 2014. [Online]. Available: http://dx.doi.org/10.14778/2732939.2732943

[19] M. Pershina, M. Yakout, and K. Chakrabarti, “Holistic entity matching across knowledge graphs,” in *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015*, 2015, pp. 1585–1590. [Online]. Available: http://dx.doi.org/10.1109/BigData.2015.7363924