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 \( \frac{k(k-1)}{2} \) 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-
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. Enties 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 $M_{i,j} = \{ (e_1, e_2, \text{sim}) | e_1 \in S_i, e_2 \in S_j, \text{sim} [0, 1], i \neq j \}$. Link discovery tools can assign a similarity value $\text{sim}$ 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 $M = \bigcup_{i,j=1}^{k} M_{i,j}$ as input in addition to the set of associated entities $E$ 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 $C_i = \{ c_{i,1}, \ldots, c_{i,n_i} \}$ 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 $\frac{z(z-1)}{2}$ 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

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

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

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.
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 \( \frac{k(k-1)}{2} \cdot l^2 \) 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 \( \frac{k(k-1)}{2} \cdot l \cdot \frac{m}{m} \). 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 \( n_i = \frac{k \cdot l}{2 \cdot m} \) with the average cluster size \( z_i \) (\( 1 \leq z_i \leq k \)). The total number of match comparisons is thus in the order of \( (k-1) \cdot l \cdot \frac{n_i}{z_i} = \frac{(k-1) \cdot k \cdot l^2}{2 \cdot m} \) 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 \( E \) from \( k \) sources, link set \( M \), domain knowledge \( D \), \( \text{simFunc} \) \( f_{sim} \), thresholds \( t_s, t_m \) |
| Output: Set of clusters \( C \) |

1. \( C \leftarrow \emptyset \)
2. \( E, M \leftarrow \text{preprocessing}(E, M, f_{sim}, D) \)  /* initial clustering */
3. \( C_{\text{init}} \leftarrow \text{computeConnectedComponents}(E, M) \)
4. \( M_{\text{init}} \leftarrow \text{computeLinkSim}(C_{\text{init}}, f_{sim}) \)
5. \( C_{\text{init}} \leftarrow \text{refineConnectedComponents}(C_{\text{init}}, M_{\text{init}}) \)  /* cluster decomposition */
6. foreach \( c \in C_{\text{init}} \) do
   7. \( C_{\text{split}} \leftarrow \text{groupByType}(c, M_{\text{init}}) \)
   8. \( C_{\text{split}} \leftarrow \text{simBasedRefinement}(C_{\text{split}}, M_{\text{init}}, t_s) \)
   9. \( C_{\text{split}} \leftarrow \text{createRepresentatives}(C_{\text{split}}) \)
   10. \( C \leftarrow C \cup C_{\text{split}} \)  /* create cluster mapping \( C_{\text{M}} \) */
11. \( C_{\text{M}} \leftarrow \text{computeClusterSim}(C, f_{sim}, t_m) \)
12. while \( C_{\text{M}} \neq \emptyset \) do
   13. \( (c_1, c_2) \leftarrow C_{\text{M}}.\text{getBestMatch()} \)  /* cluster merge */
   14. \( c_m \leftarrow \text{merge}(c_1, c_2) \)
   15. \( C \leftarrow C \setminus \{c_1, c_2\} \cup \{c_m\} \)
   16. \( C_{\text{M}} \leftarrow \text{adaptMapping}(C, C_{\text{M}}, c_m, c_1, c_2, f_{sim}, t_m) \)
17. return \( C \)

**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 clean 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 *city-town* 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
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 sources. The links (see Figure 1 b) happen that the connected components contain more than one source is linked to at most one entity of a second one-to-one cardinality restriction. In Table 1, this is not the total, despite the preprocessing ensuring that any entity of 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 \( C_{init} \) (see line 6-10). We first apply the two kinds of cluster decomposition (GroupBy-Type 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 (\( C_{split} \)) are successively added to the result set of clusters \( C \) (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 \( C_{split} \) 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

---

**Figure 1**: Application of holistic clustering to running example.
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 $C_{\text{split}}$ determined by groupByType and calls the actual refinement function simCRefine (Algorithm 2b).

**Algorithm 2a: simBasedRefinement**

```
Input: Set of clusters $C_{\text{input}}$ resulting from type-based grouping, link set $M_c$, split threshold $t_s$
Output: Set of clusters $C_{\text{result}}$

1. $C_{\text{result}} \leftarrow \emptyset$
2. foreach $c \in C_{\text{input}}$ do
3.   $C_{\text{result}} \leftarrow C_{\text{result}} \cup \text{simCRefine}(c, M_c, t_s)$
4. return $C_{\text{result}}$
```

**Algorithm 2b: simCRefine**

```
Input: Cluster $c$, link set $M_c$, split threshold $t_s$
Output: Set of clusters $C_{\text{result}}$

1. $c_{\text{asim}_{\text{min}}} \leftarrow \text{getMinAvgSimEntity}(c, M_c)$
2. if $\text{asim}_{\text{min}} < t_s$ then
3.   $c \leftarrow c \cup \{e\}$
4.   return createCluster$(e) \cup \text{simCRefine}(c, M_c, t_s)$
5. else
6.   return $c$
```

For each cluster, we determine the entity $e$ with lowest average similarity $\text{asim}_{\text{min}}$ (line 1 in Algorithm 2b) of its links to other cluster members using similarity values from the previously determined similarities in $M_c$. 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 1c, 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 $C_{\text{input}}$ 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 $f_{\text{sim}}$ 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 $C_M$ 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 $C_M$. In each iteration we select the pair of clusters $(c_1, c_2)$ with the highest similarity from $C_M$ 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 $C$ and the new cluster $c_m$ is added (line 15). We further need to adapt $C_M$ by removing all cluster pairs involving either $c_1$ or $c_2$. Furthermore, we extend $C_M$ 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 $(4, 5, r_5)$ into the new cluster $(0, 1, 4, 5, r_5)$ (see Figure 1c,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 1d) 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.
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
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.