# Interlinking and Knowledge Fusion

Volha Bryl[1(✉)], Christian Bizer[1], Robert Isele[2], Mateja Verlic[3], Soon Gill Hong[4], Sammy Jang[4], Mun Yong Yi[4], and Key-Sun Choi[4]

[1] University of Mannheim, Mannheim, Germany
{volha,chris}@informatik.uni-mannheim.de
[2] Brox IT-Solutions GmbH, Hannover, Germany
mail@robertisele.com
[3] Zemanta d.o.o, Ljubljana, Slovenia
mateja.verlic@zemanta.com
[4] KAIST, Daejeon, Korea
{soonhong,sammy1221,munyi,kschoi}@kaist.ac.kr

**Abstract.** The central assumption of Linked Data is that data providers ease the integration of Web data by setting RDF links between data sources. In addition to linking entities, Web data integration also requires the alignment of the different vocabularies that are used to describe entities as well as the resolution of data conflicts between data sources. In this chapter, we present the methods and open source tools that have been developed in the LOD2 project for supporting data publishers to set RDF links between data sources. We also introduce the tools that have been developed for translating data between different vocabularies, for assessing the quality of Web data as well as for resolving data conflicts by fusing data from multiple data sources.

## 1 Introduction

The amount of Linked Open Data (LOD) already available on the Web of Data, or extracted using e.g. the methods presented in Chap. 3, is huge, as well as its potential for applications. However, the *quality* of the LOD sources varies greatly across domains and single datasets [1], making the efficient use of data problematic. An important quality-related problem is the lack of *data consistency*: same real world entities are described in different datasets using different vocabularies and data formats, and the descriptions often contain conflicting values.

According to the architecture of a Linked Data application illustrated in Fig. 1, four steps are necessary before the input coming from the Web of Data can be consumed by an application: *vocabulary mapping, identity resolution, data quality assessment* and *data fusion*.

This chapter presents methods and open source tools developed within the LOD2 project, which cover the above four steps of the process of integrating and cleansing the Linked Data from the Web.

*Vocabulary mapping*, or schema alignment step is inevitable as different LOD providers may use different vocabularies to represent the same type of information. E.g. *population* property of a country or city can come under different names


S. Auer et al. (Eds.): Linked Open Data, LNCS 8661, pp. 70–89, 2014.
DOI: 10.1007/978-3-319-09846-3_4

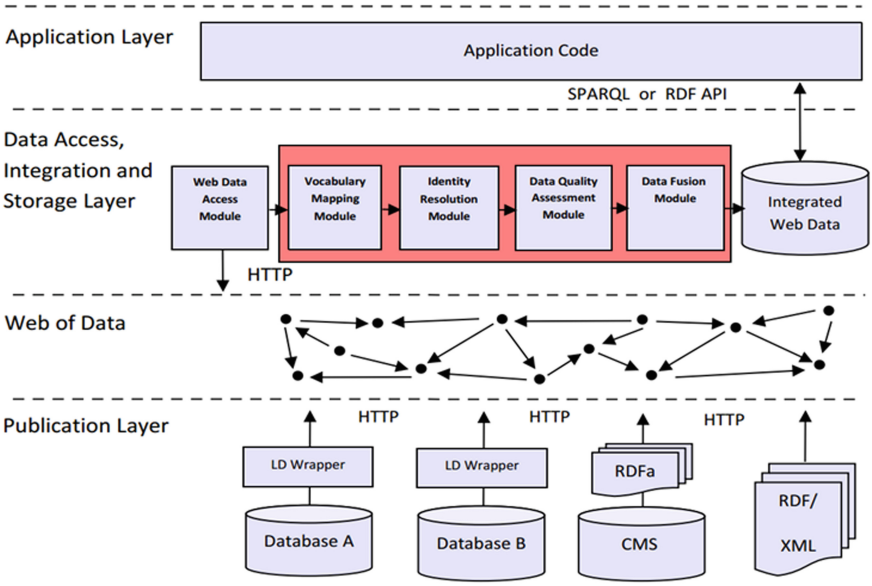


**Fig. 1.** Schematic architecture of a Linked Data application [7]

such as *population, populationTotal, numberOfInhabitants, hasPopulation*, etc. Therefore, tools that translate terms from different vocabularies into a single target schema are needed. Section 2 presents the R2R Framework, which enables Linked Data applications to discover and apply vocabulary mappings to translate the Web data to the application's target vocabulary.

*Identity resolution* aims at interlinking URIs that are used by different Linked Data sources to identify the same entity, for instance, a person or a place. Data sources may provide *owl:sameAs* links connecting data about the same real-world entity, but in many cases methods and tools for discovering these links are needed. In Sect. 3 we present the Silk Link Discovery Framework that supports identity resolution and data interlinking in the LOD context. Section 4 presents the LOD-enabled version of OpenRefine for data cleansing and reconciliation, which is also enhanced with crowdsourcing capabilities.

*Data quality assessment and data fusion* steps ensure the quality and consistency of data coming from the web. Depending on the application, different data quality aspects may become relevant: trustworthiness, precision, recency, etc. Section 5 presents Sieve – Linked Data Quality Assessment and Fusion tool, which allows filtering and then fusing the Web data according to user-defined data quality assessment and conflict resolution policies. One of the crowdsourcing use cases in Sect. 4 is related to improving the data quality via data enrichment.

In addition, Sect. 6 addresses the specific challenges of identity resolution and data fusion for some of the most wide-spread Asian languages: Korean, Chinese and Japanese.

## 2 Vocabulary Mapping

Linked Data sources use different vocabularies to describe the same type of objects. It is also common practice to mix terms from different widely used vocabularies[1] with proprietary terms. In contrast, Linked Data applications usually expect data to be represented using a consistent target vocabulary. Thus these applications need to translate Web data to their local schema before doing any sophisticated data processing.

To overcome these problems, we have developed the **R2R Framework**[2][3]. This open source framework consists of a *mapping language* for expressing term correspondences, *best practices* on how to publish mappings on the Web, and a *Java API* for transforming data to a given target vocabulary according to the mappings. We also provide the R2R Graphical User Interface, a web application that allows loading, editing and executing R2R mappings on data sources that are either located in a triple store or in RDF dumps.

As the *R2R mapping language* is designed for publishing mappings as Linked Data on the Web, mappings are represented as RDF and each mapping is assigned its own dereferenceable URI. The language defines the link type *r2r:hasMapping* to interlink mappings with RDFS or OWL term definitions and voiD dataset descriptions. The syntax of the R2R mapping language[3] is very similar to the SPARQL query language, which eases the learning curve.

The mapping language covers value transformation for use cases where RDF datasets use different units of measurement, and can handle one-to-many and many-to-one correspondences between vocabulary elements. R2R also offers modifiers to be used for assigning data types and language tags or converting a literal into a URI reference using a pattern. The language provides a set of common string functions, such as *concat* or *split*, arithmetic and list functions. See Listing 1 for a mapping example (prefix definition omitted), in which the *firstName* and *lastName* properties are concatenated into the *name* property.

```
p:manyToOnePropertyMapping
 a r2r:Mapping ;
 r2r:sourcePattern "?SUBJ foaf:firstName ?f . ?SUBJ foaf:lastName ?l" ;
 r2r:targetPattern "?SUBJ dbpedia:name ?n" ;
 r2r:transformation "?n = concat(?l,',', ?f)" ;
```

**Listing 1.** R2R mapping example

The R2R Mapping Engine applies a *mapping composition* method for selecting and chaining partial mappings from different sources based on a mapping quality assessment heuristic. The assumptions are that mappings provided by vocabulary maintainers and data publishers themselves are likely to be of a higher quality, and that the quality of data translations decreases with the length of the mapping chains.

[1] E.g FOAF for representing data about people – http://www.foaf-project.org/
[2] http://wifo5-03.informatik.uni-mannheim.de/bizer/r2r/
[3] Full specification at http://wifo5-03.informatik.uni-mannheim.de/bizer/r2r/spec/

We evaluated the R2R Mapping Language by formulating mappings between DBpedia and 11 data sources that are interlinked with DBpedia, see [3] for further details. The language proved to be expressive enough in this experiment to represent all mappings that were required. The experiment also showed that far more expressivity is required to properly translate data to a target schema than currently provided by standard terms such as *owl:equivalentClass, owl:equivalentProperty* or *rdfs:subClassOf, rdfs:subPropertyOf.*

## 3 The Silk Link Discovery Framework

A central problem of the Web of Linked Data as well as of data integration in general is to identify entities in different data sources that describe the same real-world object. While the amount of Linked Open Data has grown significantly over the last years, most data sources are still not sufficiently interlinked. Out of the over 31 billion RDF statements published as Linked Data less than 500 million represent RDF links between data sources; analysis confirms that the LOD cloud represents a weakly connected graph with most publishers only linking to one other data source [2].

This section presents the **Silk Link Discovery Framework**, which generates RDF links between data items based on user-provided or automatically learned *linkage rules*. Silk can be used by data providers to generate RDF links pointing at existing Web datasets, and then publish them together with the primary datasets. Furthermore, applications that consume Linked Data can use Silk as an identity resolution component to augment the data with additional RDF links that have not been discovered and/or published.

In Silk linkage rules are expressed using a declarative language, and define the conditions that data items must conform to in order to be interlinked. For instance, a linkage rule defines which properties should be compared (e.g. *movieTitle* in one dataset and *label* in another), which similarity measures should be used for comparison and how they are to be combined.

Writing good linkage rules by hand is a non-trivial problem, which requires considerable effort and expertise. To address this, Silk implements the *ActiveGenLink* algorithm which combines *genetic programming* and *active learning* techniques to generate high-quality expressive linkage rules interactively, minimizing the involvement of a human expert. In this section, we will briefly introduce the tool and the underlying algorithms. For further details readers are referred to [8,9].

### 3.1 Silk: Functionality and Main Concepts

The Silk Link Discovery Framework can be downloaded from its official homepage[4], which is also the source for the documentation, examples and updates.

[4] http://wifo5-03.informatik.uni-mannheim.de/bizer/silk/

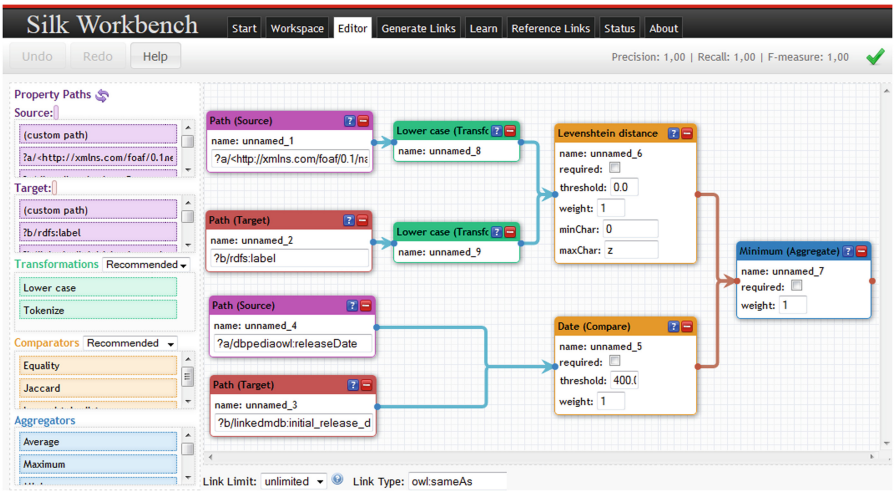


**Fig. 2.** Silk Workbench: linkage rule editor

It is an open source tool with the source code and the detailed developer documentation available online[5]. Silk can be used through the *Silk Workbench* graphical user interface or from the command line.

The Silk Workbench, developed in the course of the LOD2 project, aims at guiding the user through the process of interlinking different data sources. It is a web application offering the following functionality:

- Possibility to manage different data sources and linking tasks (with RDF dump files as well as local and remote SPARQL endpoints as input).
- Graphical editor to create and edit linkage rules (see Fig. 2).
- Possibility to evaluate links generated by the current linkage rule.
- User interface for learning linkage rules from existing reference links.
- Active learning interface, which learns a linkage rule by interactively asking the user to confirm or decline a number of candidate links.
- Possibility to create and edit a set of reference links used to evaluate the current link specification.

Additionally, Silk provides 3 command line applications: *Silk Single Machine* generates RDF links on a single machine, with input datasets either residing on the same machine or accessed via the SPARQL protocol; *Silk MapReduce* generate RDFs links between datasets using a cluster of multiple machines, is based on Hadoop and thus enables Silk to scale out to very big datasets. Finally, *Silk Server* [10] can be used as an identity resolution component within applications that consume Linked Data from the Web.

The basic concept in Silk is that of a *linkage rule*, which specifies the conditions under which two entities are considered to be referring to the same

[5] https://www.assembla.com/spaces/silk/wiki

real-world entity. A linkage rule assigns a similarity value to a pair of entities. We represent a linkage rule as a tree built from 4 types of operators.

*Property Operator:* Retrieves all values of a specific property of each entity, such as of values of the *label* property.

*Transformation Operator:* Transforms the values according to a specific data transformation function, e.g. case normalization, tokenization, concatenation. Multiple transformation operators can be nested in order to apply a sequence of transformations.

*Comparison Operator:* Evaluates the similarity between the values of two input operators according to a specific distance measure, such as Levenshtein, Jaccard, or geographic distance. A user-specified threshold specifies the maximum distance, and is used to normalize the measure.

*Aggregation Operator:* As often the similarity of two entities cannot be determined by evaluating a single comparison, an aggregation operator combines the scores from multiple comparison or aggregation operators according to an aggregation function, e.g. weighted average or minimum.

The resulting linkage rule forms a tree where the terminal nodes are represented by property operators and the internal nodes are represented by transformation, comparison and aggregation operators, see Fig. 2 for an example.

### 3.2 The GenLink Algorithm

Creating a good linkage rule by hand, that is choosing and combining appropriate operators and thresholds, is non-trivial and time-consuming. One way to reduce this effort is to use *supervised learning* to generate links from existing reference links, which contain pairs of entities labeled as matches or non-matches. Creating such reference links is much easier than writing linkage rules as it requires no previous knowledge about similarity measures or the specific linkage rule format used by the system. Usually, reference links are created by domain experts who confirm or reject the equivalence of a number of entity pairs from the input datasets.

In [8] we have presented the **GenLink algorithm** for automatically learning linkage rules from a set of reference links. The algorithm is based on *genetic programming* and generates linkage rules that can be understood and further improved by humans.

Genetic programming starts with a randomly created population of individuals, where each individual is represented by a tree which is a potential solution to the given problem. In Silk, in order to reduce the search space, before generating the initial population we build, given a set of positive reference links, a list of property pairs which hold similar values, and then use this set to build a random linkage rule.

Starting with the initial population, the genetic algorithm breeds a new population by evolving selected linkage rules using the reproduction, crossover and mutation genetic operators. A *fitness function* is used to assign a value to each

linkage rule which indicates how close the rule is to the desired solution. We use the fitness measure based on *Matthews correlation coefficient*, and penalize linkage rules based on their number of operators. Based on the fitness of each linkage rule, the *selection method*, tournament selection in our case, selects the rules to be evolved. The algorithm iteratively evolves the population until either a linkage rule has been found which covers all reference links or a configured maximum number of 50 iterations is reached.

The experimental evaluation [8] shows that the GenLink produces better results than the state-of-art genetic programming approaches for identity resolution.

### 3.3 The ActiveGenLink Algorithm

For most real-world datasets it is not feasible, however, to manually create reference links covering a big enough subset of pairs of entities. Moreover, in order for the supervised learning algorithms to perform well on unknown data, reference links need to include all relevant *corner cases*. For example, while for most pairs of movie descriptions comparing titles is enough to establish that the two refer to the same movie, there might exist variations in titles of the same movie, or different movies having the same title but different release dates. The user has to label a very large number of *randomly selected* pairs in order to include these rare corner cases reliably.

To reduce the number of candidates to be labeled, we employ *active learning*, which in our case means iteratively selecting the most informative entity pair to be labeled by the user as matching or non-matching. In [9] we introduced the **ActiveGenLink algorithm**, which evolves a population of candidate solutions iteratively while building a set of reference links. The algorithm starts with a random population of linkage rules and an initially empty set of reference links. In each iteration, it selects a link candidate for which the current population of linkage rules is uncertain, from a pool of unlabeled links using a so called *query strategy*. After the link has been labeled by a human expert, the algorithm evolves the population of linkage rules using the GenLink algorithm and the extended set of reference links.

The query strategy Silk implements is based on a *query-by-committee* strategy: the selected link candidate is determined from the voting of all members of a committee, which consists of the current linkage rules in the population. We take as a baseline the widely used *query-by-vote-entropy*, which selects the candidate for which the members in the committee disagree the most, and introduce an improved strategy as follows. Firstly, as the unlabeled links are not distributed uniformly across the similarity space, we aim at distributing the links onto different clusters by selecting links that, based on the Jensen-Shannon divergence, are different from already labeled links. Secondly, the voting committee, i.e. the evolved population of linkage rules, may contain suboptimal linkage rules that do not cover all reference links. We implement the *restricted committee* voting, in which only linkage rules which fulfill a specific reference link are allowed

to vote. Our experiments show that the improved query strategy outperforms the query-by-vote-entropy baseline.

The performance of the ActiveGenLink algorithm was evaluated on the same datasets as we used to evaluate the supervised GenLink algorithm [8]. The results show that by labeling a small number of links, ActiveGenLink achieves a comparable performance to GenLink on the complete reference link set. One of the datasets on which the evaluation was performed is *SiderDrugBank* from the *Ontology Alignment Evaluation Initiative* (OAEI) 2010 data interlinking track[6]. Our evaluation showed that with ActiveGenLink only about 30 links had to be labeled until a linkage rule could be learned which achieves an F-measure similar to the one GenLink gets using all 1718 reference links.

Two other experiments were done on the datasets that have been used frequently to evaluate the performance of different record linkage approaches: *Cora* and *Restaurant* datasets[7]. The results show that labeling a small number of links is enough to reach high performance. In addition, we successfully evaluated how the learned linkage rules compare to rules manually created by a human expert for the same dataset, and studied the scalability of the proposed approach. For the details of all the evaluation experiments the reader is referred to [9].

In order to further improve the linking precision we have developed the Silk Free Text Preprocessor [12], an extension of Silk for producing a structured representation for linking the data that contains or is derived from free text.

## 4 Data Cleansing and Reconciliation with LODRefine

Data cleansing and linking are very important processes in the life cycle of linked data, especially when creating new linked data. Nowadays data comes from different sources and it is published in many formats, e.g. XML, CSV, HTML, as dumps from relational databases or different services.

Unfortunately, cleansing and linking are rarely trivial, and can be very tedious, especially with the lack of good tools. A good cleansing tool should be able to assist users in detecting non-consistent data, removing duplicates, quickly performing transformations on a relatively large amount of data at once, and exporting cleansed data into different formats. It should be relatively simple to use and available on different operating systems. Fortunately, there is a open source (BSD licensed) solution available meeting all the above criteria. *OpenRefine*, previously Google Refine, was specifically created for dealing with messy data, is extendable, works on all three major operating systems and provides functionalities to reconcile data against Freebase. For needs of the LOD2 project we developed a LOD-enabled version of *OpenRefine* – **LODRefine**[8].

[6] http://oaei.ontologymatching.org/2010/im/index.html

[7] XML version: http://www.hpi.uni-potsdam.de/naumann/projekte/dude_duplicate_detection.html

[8] Available for download at http://sourceforge.net/projects/lodrefine/ or as a source code at https://github.com/sparkica/LODRefine

Even with tools available, data cleansing and linking in the LOD cycle still require a lot of manual work, and the current algorithms and fully automatated tools cannot completely replace human intuition and knowledge: it can be too complicated or costly to encode all our knowledge and experience into rules and procedures computers can understand. Although we already have good cleansing and linking tools at our disposal requiring only minimal human intervention, with huge amounts of data even simple tasks add up and time resources become an issue.

However, such tasks are often simple and include evaluation of automatically obtained results, finding missing information or, in rare cases, demand special domain knowledge. *Crowdsourcing* seems to be a promising solution for such situations and offers hiring affordable working power for certain repetitive but relatively simple tasks, especially when automated processing does not give good enough results, e.g. when classifying spam, categorizing images, and disambiguating data. To bring crowdsourcing closer to the needs of the LOD2 project we added support for using *CrowdFlower*[9], a popular and versatile crowdsourcing platform, directly from the LODRefine environment. In the rest of this section we introduce LODRefine and shortly describe three use cases of using crowdsourcing for different tasks, namely, data cleansing and disambiguation of reconciliation results.

### 4.1 LODRefine

LODRefine, a powerful tool for cleansing and automatically reconciling data with external databases, includes all core features of *OpenRefine* and extends them with LOD-specific ones. *Core features* include:

- Importing data from various formats.
- Cleansing data: finding duplicates, removing them, finding similar values.
- Filtering data using faceted browsing.
- Filtering data with regular expressions.
- Google Refine Expression language (GREL): a powerful language for transforming data.
- Reconciling with Freebase: the ability of linking your data to Freebase.
- Extending data from Freebase: the ability of adding data from Freebase to your reconciled data.

Figure 3 features faceted browsing, using regular expressions and GREL.
*LOD-enabling features* added support for:

- Reconciliation and extending data with DBpedia.
- Named-entity recognition: recognizing and extracting named entities from text using different services.
- Using crowdsourcing: creating crowdsourcing jobs and uploading data to crowdsourcing platforms.

[9] http://www.crowdflower.com/

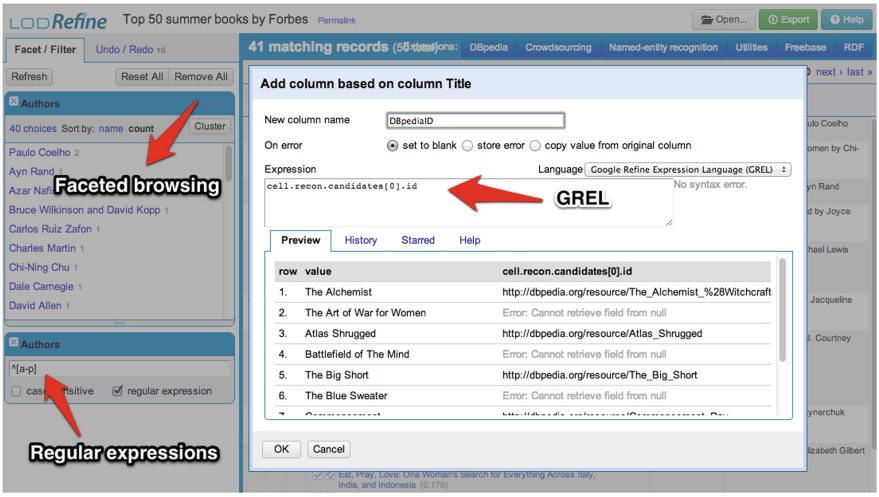


**Fig. 3.** LODRefine: faceted browsing, support for regular expressions and using GREL

## 4.2 Use Cases

The quality of reconciliation may vary with data and manual evaluation of those results is needed. In case we do not have available human resources crowdsourcing might be considered as a viable solution.

In the following we describe three use cases of using crowdsourcing from and within LODRefine. For further details readers are referred to the corresponding project deliverable [13].

***Evaluating reconciliation results.*** Quality of linking (reconciliation) in the context of Linked Data can be evaluated by using rather sophisticated algorithms or manually with human evaluators. In the last case crowdsourcing can significantly speed up the process, especially when LODRefine is used to create a job from reconciliation evaluation template.

In this use case crowdsourcing was used to evaluate the quality of reconciled dataset of National Football League players[10]. Data contains names of players and links to their official profiles on NFL webpage. Freebase was used for reconciliation, and manual evaluation was done first by a group of in-house trained evaluators and then by workers at CrowdFlower. Because we already had verified evaluation results, we were able to asses the quality of results obtained by crowdsourcing.

Validation using crowdsourcing was split in two batches. For the first batch we collected three judgments per unit and for the second batch we lowered the overall costs by collecting only two judgments per unit. Although the quality of judgments dropped slightly for the second batch, the ratio costs versus quality of results was satisfiable.

[10] Official NFL webpage: http://www.nfl.com/

***Validating named entities extracted from blogs.*** Integrating new entities into recommendation system is crucial for suggesting relevant contents to bloggers. New entities can be discovered by extracting links from blog posts. New links are considered as potential new named entities and links' anchors as entity aliases. In this use case we use crowdsourcing to verify extracted links from blog posts and mark them as entities if appropriate. If link was considered an entity, contributors also provided the type of entity.

We published this job on all available channels, which was reflected in the high number of overall units per hour, while in other two use cases we only used 2–4 labor channels (Amazon MTurk was always selected) and thus obtained much lower overall units per hour.

***Data enrichment – finding missing information about festivals.*** Finding missing bits of information and enriching data is a frequently appearing type of assignment on crowdsourcing platforms.

In this use case we used crowdsourcing to enrich a dataset about festivals, which was extracted from blog posts mentioning festival and conference-like events either with their short names or using the full titles. In some cases blog posts mentioned words "festival" or "fest", but in a different context, and were wrongly extracted as a festival. We wanted to identify such cases and enrich data about actual festivals.

Data enrichment took much longer than other two use cases. Searching for data about festivals was more time consuming and questions were slightly more difficult. The price set was also relatively low, which was the other factor impacting time needed to collect responses.

### 4.3 Quality Evaluation of Crowdsourcing Results

All results obtained by crowdsourcing have been evaluated by comparing them to results provided by in-house trained evaluators. A lot depends on how instructions and questions are formulated, how much quality control is involved and which labor channels tasks are published on. In our case we got the best results in the first use case, in which contributors had to choose one of the provided suggestions or find a link in Freebase and thus there was not much room for subjectivity. Second best use case was data enrichment, where contributors had to check, whether the link contains information about a certain type of event – a festival – and provide a full name, a short name and a homepage for it. Again, the instructions and questions did not allow for too much subjectivity. The least good results were obtained in the second use case involving the evaluation of named entities. There are many possible causes for this: it might be not easy to grasp the notion of a named entity for an average contributor, contributors might not read instructions carefully enough, or instructions might have been too complicated.

Crowdsourcing is a relatively new business model and still requires some time before it will be fully mature and properly supported by legislation, but it can be regarded as a useful and feasible approach at least for some types of

LOD-related problems and tasks that were described and demonstrated above. There are several considerations that need to be taken into account while using crowdsourcing: quality assurance measures and constraints have to be applied and ethical issues related to fair price and reward for all contributors have to be considered. Although we would not use it for sensitive data or for gigabytes of data at once, crowdsourcing can serve as a starting point for developing automated solutions. It can provide means to collect enough data to train algorithms and to evaluate results obtained by those algorithms.

## 5 Data Quality Assessment and Fusion

The vocabulary alignment and interlinking steps presented in Sects. 2 and 3, result in interlinked entity descriptions originating from a number of heterogeneous data sources. The *quality* of data from these sources is very diverse [1], as values may be out of date, incomplete or incorrect, either because of data extraction errors or due to the errors by human data editors. Situations in which *conflicting values* for a property of a real-world object are provided often occur. In order for Linked Data applications to efficiently consume data, the latter should be assessed and integrated based on their quality.

Quality is a subjective matter, often defined as a "fitness for use", meaning that the interpretation of the quality of a data item depends on who will use it and for what task. Data quality has many dimensions such as accuracy, timeliness, completeness, relevancy, objectivity, believability, understandability, consistency, conciseness, availability, verifiability, etc. These dimensions are not independent of each other and typically only a subset of them is relevant in a specific situation. With the objective of supporting user applications in dealing with data quality and conflict resolution issues, we created **Sieve – Linked Data Quality Assessment and Fusion framework**[11] [11], which we summarize in this section.

Sieve consists of two components: Data Quality Assessment and Data Fusion, and takes as input data to be fused and an XML file containing both quality assessment and data fusion configurations. The input XML-based specification language allows a user to manually define conflict resolution strategies and quality assessment metrics to use for each data property.

Sieve takes as input two or more RDF data sources, along with the data provenance information. It is assumed that schema and object identifiers have been normalized, namely, if two descriptions refer to the same real-world object then they have the same identifier (URI), and if two properties refer to the same real-world attribute then there should be two values for the same property URI for a given subject URI. Each property value in the input is expressed by a quad *(subject,property,object,graph)* where the *graph* is a named graph, which is used to attach provenance information to a fact or a set of facts. For an example see Listing 2, where the input data for the population of Amsterdam coming from three different DBpedia editions is given, along with the last edit

[11] http://sieve.wbsg.de

date information. Note that the 4th quad component, the provenance graph for *lastedit* property, is the same for all three triples and is omitted due to space reasons.

```
dbp:Amsterdam dbpedia-owl:population"820654" en:Amsterdam:population
dbp:Amsterdam dbpedia-owl:population "758198" ca:Amsterdam:population
dbp:Amsterdam dbpedia-owl:population "820654" it:Amsterdam:population
en:Amsterdam:population dpb-meta:lastedit "2013-01-13T14:52:13Z"
ca:Amsterdam:population dpb-meta:lastedit "2009-06-14T10:36:30Z"
it:Amsterdam:population dpb-meta:lastedit "2013-03-24T17:04:16Z"
```

**Listing 2.** Data Fusion with Sieve: input data

### 5.1 Quality Assessment Metrics

Three main concepts Sieve uses in the quality assessment configuration are *quality indicators*, *scoring functions* and *assessment metrics*. A *data quality indicator* is an aspect of a data item or dataset that may indicate the suitability of the data for some intended use. The types of information used as quality indicators may stem from the metadata about the circumstances in which information was created, to information about the information provider, to data source ratings. A *scoring function* produces a numerical representation of the suitability of the data, based on some quality indicator. Each indicator may be associated with several scoring functions, e.g. max or average functions can be used with the data source rating indicator. *Assessment metrics* are procedures for measuring information quality based on a set of quality indicators and scoring functions. Additionally, assessment metrics can be *aggregated* through the average, sum, max, min or threshold functions.

For an example see Listing 3, where *recency* assessment metric uses the *last update* timestamp of a dataset or a single fact, a quality indicator which is transformed by *TimeCloseness* scoring function into a numeric score using a range parameter (in days) to normalize the scores. Other scoring functions available in Sieve include normalizing the value of a quality indicator, or calculating the score based on whether the indicator value belongs to some interval or exceeds a given threshold. The complete list of supported scoring functions is available at the Sieve webpage; users can define their own scoring functions using Scala and the guidelines provided at the webpage.

The output of the quality assessment module is a set of quads, where the calculated scores are associated with each graph. A graph can contain the whole dataset (e.g. Dutch DBpedia) or a subset of it (all properties of Berlin in Freebase) or a single fact. The scores represent the user-configured interpretation of quality and are then used by the Data Fusion module.

```
<QualityAssessment>
  <AssessmentMetric id="sieve:recency">
    <ScoringFunction class="TimeCloseness">
      <Param name="timeSpan" value="500"/>
      <Input path="?GRAPH/dpb-meta:lastedit"/>
    </ScoringFunction>
  </AssessmentMetric>
</QualityAssessment>
```

```
<Fusion>
  <Class name="dbpedia-owl:PopulatedPlace">
    <Property name="dbpedia-owl:population">
      <FusionFunction class="KeepFirst" metric="sieve:recency"/>
    </Property>
  </Class>
</Fusion>
```

**Listing 3.** Data Fusion with Sieve: specification

### 5.2 Fusion Functions

In the context of data integration, Data Fusion is defined as the "process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation" [5]. Data Fusion is commonly seen as a third step following schema mapping and identity resolution, as a way to deal with conflicts that either already existed in the original sources or were generated by integrating them.

The Sieve Data Fusion module is inspired by [4], a framework for data fusion in the context of relational databases that includes three major categories of conflict handling strategies:

- Conflict-ignoring strategies, which defer conflict resolution to the user. For instance, *PassItOn* strategy simply relays conflicts to the user or applications consuming integrated data.
- Conflict-avoiding strategies, which apply a unique decision to all data. For instance, strategy *TrustYourFriends* strategy prefers data from specific data sources.
- Conflict-resolution strategies, which decide between existing data (e.g. *KeepUpToDate*, which takes the most recent value), or mediate the creation of a new value from the existing ones (e.g. *Average*).

In Sieve, fusion functions are of two types. *Filter functions* remove some or all values from the input, according to some quality metric, for example *keep the value with the highest score* for a given metric (e.g. recency or trust) or *vote* to select the most frequent value. *Transform functions* operate over each value in the input, generating a new list of values built from the initially provided ones, e.g. computing the *average* of the numeric values. The complete list of supported fusion functions is available at the Sieve webpage, and users have the possibility to implement their own functions.

The example of the specification in Listing 3 illustrates how a fusion function for the *population* property of a populated place is configured to use *KeepFirst* fusion function (i.e. keep the highest score) applied to *recency* quality assessment metric.

The output of the data fusion module is a set of quads, each representing a fused value of a subject-property pair, with the 4th component of the quad identifying the named graph from which a value has been taken. An extension of Sieve for automatically learning an optimal conflict resolution policy is presented in [6].

## 6 Data Interlinking and Fusion for Asian Languages

As the proportion of non-Western data that went to open is relatively small, recently many projects have been initiated to extend the boundary of Linked Open Data to non-European language resources, especially in such writing systems as Korean, Chinese and Japanese. Interlinking and integrating multilingual resources across languages and writing systems will allow exploiting the full potential of Linked Data.

This section presents the tools we developed to support interlinking and integrating Asian resources: **Asian Resource Linking Assistant** and **Asian Data Fusion Assistant**. The Asian Resource Linking Assistant extends Silk with Korean Phonemic Distance metric, Korean Transliteration Distance metric, and Han Edit Distance metric. The Asian Data Fusion Assistant extends Sieve by providing functions for translating phrases among Korean, Chinese, Japanese and English.

### 6.1 Interlinking Korean Resources in the Korean Alphabet: Korean Phoneme Distance

Using string distance metrics such as Levenshtein, or edit distance, is a popular way to measure distance between two strings in Western languages. When we apply Levenshtein distance to Korean strings, the output is based not on the number of letters but on the number of combinations of letters. This is because the unit of comparison is different between a single-byte code set and a multi-byte code set. The unit of comparison for single-byte code is a single letter. In Korean, however, a combination of two or three (occasionally four but rarely more) letters, which constitutes a syllable, is assigned to one code point. So when the edit distance between Korean strings is 1, it could mean 1 (as in case of English strings), but also 2, 3 or occasionally more letters. Therefore, we have developed a new Korean distance metric that reflects the nature of a multi-byte code set.

Our approach is based on the idea that the more the phonemes are distributed across the syllables, the less is the possibility the two string have the same meaning. For example, if two target strings have two different phonemes, then one target string with one syllable containing two different phonemes is closer to a source string than the other target string with two syllables each containing one different phoneme. For example, the distance between "바람" ("wind" in English) and "보름" ("15 days" in English) with a syllable-based metric is $2\beta$ whereas the distance between them is $1\beta{+}1\alpha$ ($\alpha$: weight for syllable, $\beta$: weight for phoneme) with our metric.

The Korean Phoneme Distance metric is defined in Eq. (1), where sD is the syllable distance, and pDn is a list of phoneme distances of each syllable.

$$\text{if } (sD_0 > 0) \min_{0 \le i \le I}[(sD_i - 1) * \beta + \min_{0 \le n \le N}(spD_{in}) * \alpha], \text{ else } 0 \tag{1}$$

This new metric can control the range of the target string more precisely, especially for those string pairs that have only one or two phonemes different

from only one syllable. For example, if a threshold of two is applied, then a search for "호랑이" ("tiger" in English) would find its dialect "호래이" as a candidate (only the first syllable is different). But a syllable-based metric would find many of its variations such as "오라이" ("OK" in English) as well (the first and second syllables are different). This algorithm is especially useful for finding matching words in dialects and words with long histories, which have many variations across the country, and the variations have some similar patterns. This metric can fine-tune the distribution of different phonemes, and precision can be improved by reducing irrelevant information to be retrieved. Therefore, this metric is especially effective in enhancing precision by reducing the number of irrelevant records compared with Levenshtein.

### 6.2 Interlinking Korean Resources in Korean and English: Korean Transliteration Distance

During translation some terms may be transliterated from one language to another in case translators cannot find proper corresponding terminology in that local language. Transliteration is a way of converting letters from one writing system into another without concern for representing the original phonemics. For example, a Korean popular food called "칼국수" (translated as knife-cut Korean noodles) can be transliterated as *kalguksu* in English.

The best approach to measure distance between transliterated strings would be to apply Korean Phoneme Distance to the strings transliterated back. This approach, however, is complicated because a transliterated Korean string could be back transliterated into several possible original Korean strings in case the transliterated string has no explicit notation for identifying syllables. Unfortunately, existing transliteration systems do not consider back-transliteration, so no explicit borders of syllables, which is important to restore the original Korean words, exist. Although many efforts have focused on the back transliteration to help people identify the original string better, many existing Korean-English transliteration systems lack this mechanism.

Due to this difficulty, we decided to take a simpler but more practical approach for measuring distance between transliterated Korean strings, namely, Korean Transliteration Distance, which chooses one random letter for each consonant group from the International Phonetic Alphabet (IPA) chart. For example, as 'b' and 'p' belong to the bilabial plosive group, Korean Transliteration Distance replaces every 'b' with 'p'. Similarly, it replaces 'g' with 'k', 'd' with 't', and 'l' with 'r' (although 'l' and 'r' do not belong to the same group, they are used interchangeably in Korea). The main difference between Soundex and Korean Transliteration Distance is that Korean Transliteration Distance does not eliminate vowels or other consonants, does not remove duplicates, and does not limit the number of letters for comparison. There are three reasons for this. First, the Korean alphabet has 10 vowels and 14 consonants compared with 5 vowels and 21 consonants in English, so Korean vowels play a more important role in matching words than English vowels do. Second, the Korean alphabet has letters with fortis, which is expressed with two identical consonants in succession,

so the meaning of the string will be lost if duplicate consonants are removed. Third, keeping all the letters is a more conservative and safer way to measure distance. This metric is especially useful for enhancing recall while keeping precision almost intact compared with Levenshtein and for enhancing precision compared with Soundex, thereby contributing to obtaining a higher number of correct links when used in Silk.

### 6.3 Interlinking Asian Resources in Chinese Alphabet: Han Edit Distance

China, Japan and Korea (CJK) are geographically close and have influenced each other language systems for thousand years. CJK share Han Chinese even though Japan and Korea have their own writing systems, and many currently used words in the three countries were derived from ancient Han Chinese. That means language resources existing in the three countries can be better matched and interlinked when the lineage is properly utilized. Therefore, a new linguistic similarity metric was developed to measure distances between commonly used Chinese letters among the three languages.

Han Edit Distance (HED) is a new similarity metric for Chinese, Japanese and Korean based on the Unihan database. The Unihan database covers more than 45,000 codes and contains mapping data to allow conversion to and from other coded character sets and additional information to help implement support for the various languages that use the Han ideographic script. As the Unihan database provides a variety of information associated with Han Chinese, HED measures similarities between two words by using this information.

As Han Chinese has been used in many Asian countries for a long period of time, Han characters are pronounced differently, and some of the shapes have changed over time in different regions. *Reading* category in the Unihan database shows the pronunciation of the same unified ideographs in Mandarin, Cantonese, Tang-dynasty Chinese, Japanese, Sino-Japanese, Korean and Vietnamese. The *Variants* category includes a variety of relationships with Han Chinese that can be used for interlinking. In Han Edit Distance, each piece of information about Han Chinese characters was classified into Reading and Semantic categories. That is, kMandarin, kJapaneseOn, kJapaneseKun, kKorean and kHangul are classified into the Reading category, and kSemanticVariant, kCompatibilityVariant, kSimplifiedVariant, kRSUnicode and kTotalStroke are classified into the Semantic category.

Figure 4 shows how HED is measured: it calculates Reading and Semantic distances using each category, sums the total distance, and normalizes the distance. The number of matching properties from the Reading category is the distance between two characters. Therefore, the maximum reading distance is 5 because Reading category has 5 properties. Semantic distance is calculated by comparing 3 semantic properties (semantic, compatibility, simplified variant). If any of the three matches, the two characters are believed to have a semantic relationship. If no match exists, then a semantic distance is calculated by counting radical strokes. That is, the number of total strokes of two characters when the

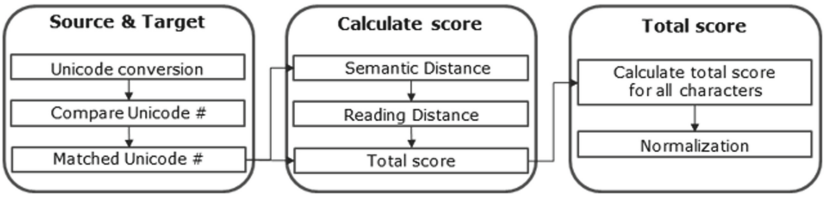


**Fig. 4.** Detailed process of Han Edit Distance algorithm

family root is the same becomes the distance. We defined 30 to be the maximum number of total strokes, even though the total number of strokes is larger than 30, but the number of Chinese characters that have more than 30 strokes is rare (about 0.23 %) in the Unihan database.

As Han Chinese words can be composed of one or more characters, we performed two types of experiments to compare with Levenshtein distance by using commonly used Han Chinese characters (1,936 pairs) and by using Han Chinese words (618 pairs). The F-measure scores of both experiments show better performance, specially as high as 23 % for Han Chinese words. From the experiment, the HED method shows performance improvements in comparison with Levenshtein distance for Han characters commonly used in Chinese, Japanese and Korean.

### 6.4 Asian Data Fusion Assistant

Integrating multilingual resources to derive new or unified values has not shown the full potential in the context of Linked Data partly because of language barriers. Asian Fusion Assistant, an extension of Sieve, aims at facilitating the fusion process by adding translation functionality from one Asian language to another. While machine translation systems pursue full automatic translation without human intervention by using a large bilingual corpora, building a machine translation system for each pairs of languages is hard to achieve. Therefore, we adopted a *translation memory* approach for two reasons. First, existing human translations can be fully utilized. Second, not every part of RDF triples ought to be translated, but only plain literals that have language tags.

A translation memory system provides similar translation pairs upon translator's requests and stores new translation pairs produced by human translators. Wikipedia (and hence, DBpedia) texts with inter-language links for many languages is a valuable source of translation memories. Therefore, parallel text pairs were collected from Korean, English, Chinese and Japanese DBpedia and stored separately. Although RDF triple translation follows the architecture of translation memory systems, one major difference is that human translators are substituted with (free) internet translation services. The advantages of using the Internet translation API services (e.g. Microsoft Bing) are that they usually support many language pairs and because the concerns about translation quality are reduced as texts to be translated are not sentences but nouns or noun phrases.

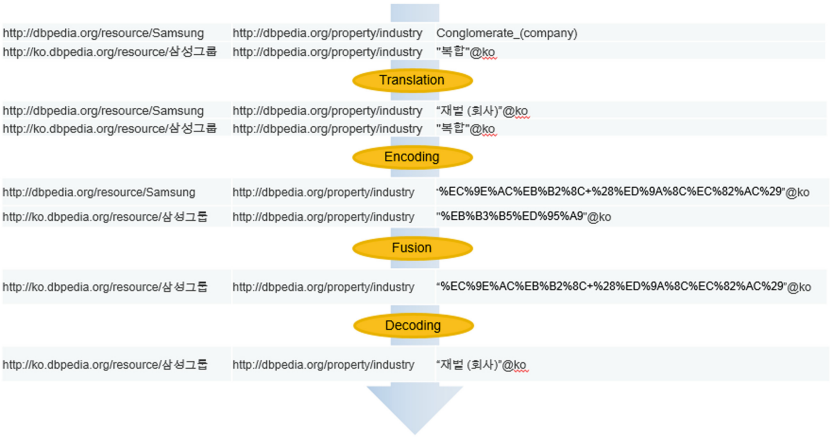


**Fig. 5.** Asian fusion process

Asian resource fusion process consists of 4 steps: translation, encoding, quality assessment/conflict resolution and decoding as shown at Fig. 5. Translation is only invoked when input triples contain plain literals with language tags. Encoder encodes multi-byte letters (e.g. Korean) into a stream of single-byte letters, and then Sieve performs quality assessment and conflict resolution to produce an integrated result. Finally, Decoder decodes all encoded strings into the original language again. We expect that translation memory systems can be globally interconnected and can boost the multilingual data fusion.

## 7 Conclusion

In this chapter we have presented the tools for vocabulary mapping, data interlinking, quality assessment and fusion, developed in the context of the LOD2 project. Specifically, R2R supports vocabulary mappings, Silk and LODRefine facilitate the process of creating and evaluating the quality of links among datasest, Sieve assists its users in assessing the data quality and resolving value conflicts. Additionally, Silk and Sieve has been extended to address interlinking and fusion issues specific to CJK (Chinese, Japanese and Korean) languages.

The presented tools are open source and make part of the Linked Data stack (see Chap. 6). The tools have been extensively evaluated, for the details the reader is referred to the respective sections, cited articles and tools' webpages. These tools have been applied within LOD2 project, e.g. in a media publishing, enterprise and public procurement use cases, for the details see Chaps. 7, 8 and 10 of the present book, respectively.

## References


1. Acosta, M., Zaveri, A., Simperl, E., Kontokostas, D., Auer, S., Lehmann, J.: Crowdsourcing linked data quality assessment. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, Ch., Janowicz, K. (eds.) ISWC 2013, Part II. LNCS, vol. 8219, pp. 260–276. Springer, Heidelberg (2013)
2. Bizer, C., Jentzsch, A., Cyganiak, R.: State of the LOD cloud. Technical report, Freie Universität Berlin (2011). http://lod-cloud.net/state/
3. Bizer, C., Schultz, A.: The R2R framework: publishing and discovering mappings on the web. In: Proceedings of the 1st International Workshop on Consuming Linked Data (COLD) (2010)
4. Bleiholder, J., Naumann, F.: Declarative data fusion – syntax, semantics, and implementation. In: Eder, J., Haav, H.-M., Kalja, A., Penjam, J. (eds.) ADBIS 2005. LNCS, vol. 3631, pp. 58–73. Springer, Heidelberg (2005)
5. Bleiholder, J., Naumann, F.: Data fusion. ACM Comput. Surv. **41**(1), 1:1–1:41 (2009)
6. Bryl, V., Bizer, C.: Learning conflict resolution strategies for cross-language Wikipedia data fusion. In: 4th Workshop on Web Quality Workshop (WebQuality) at WWW 2014 (2014)
7. Heath, T., Bizer, C.: Linked data: evolving the web into a global data space. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, San Rafael (2011)
8. Isele, R., Bizer, C.: Learning expressive linkage rules using genetic programming. Proc. VLDB Endowment **5**(11), 1638–1649 (2012)
9. Isele, R., Bizer, C.: Active learning of expressive linkage rules using genetic programming. J. Web Semant. **23**, 2–15 (2013)
10. Isele, R., Jentzsch, A., Bizer, C.: Silk server - adding missing links while consuming linked data. In: Proceedings of the 1st International Workshop on Consuming Linked Data (COLD 2010), pp. 85–97 (2010)
11. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: linked data quality assessment and fusion. In: EDBT/ICDT Workshops, pp. 116–123 (2012)
12. Petrovski, P., Bryl, V., Bizer, C.: Integrating product data from websites offering Microdata markup. In: 4th Workshop on Data Extraction and Object Search (DEOS2014) at WWW 2014 (2014)
13. Verlic, M.: Release of documentation and software infrastructure for using Google Refine along with Amazon Mechanical Turk, 2013. LOD2 project delivarable D4.6.2. http://static.lod2.eu/Deliverables/D4.6.2.pdf