LIMES

Link Discovery Framework for Metric Spaces

User Manual

Version 0.6.RC4

October 7, 2014

Contents

1 Introduction

LIMES, the **Li**nk Discovery Framework for **Me**tric **S**paces, is a framework for discovering links between entities contained in Linked Data sources. LIMES is a hybrid framework that combines the mathematical characteristics of metric spaces as well prefix-, suffix- and position filtering to compute pessimistic approximations of the similarity of instances. These approximations are then used to filter out a large amount of those instance pairs that do not suffice the mapping conditions. By these means, LIMES can reduce the number of comparisons needed during the mapping process by several orders of magnitude and complexity without loosing a single link.

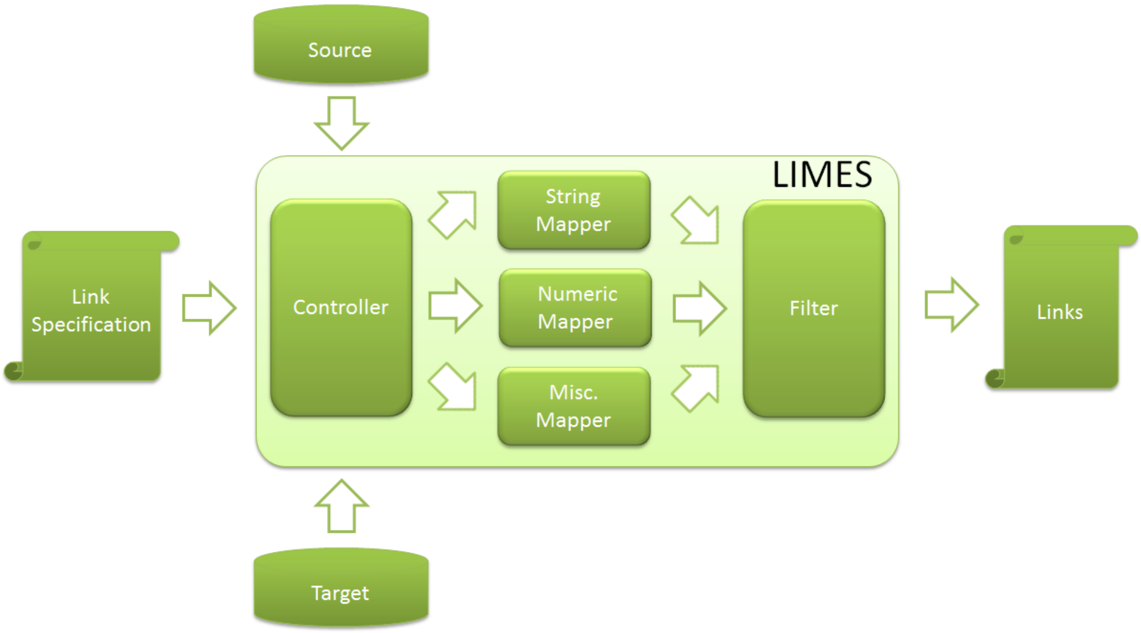

Figure 1: General Workflow of LIMES

The general workflow implemented by the LIMES framework is depicted in Figure 1. Given the source S, the target T and a link specification, LIMES first separates the different data types to merge. Strings are processed by using suffix-, prefix- and position filtering in the string mapper. Numeric values (and all values that can be mapped efficiently to a vector space) are mapped to a metric space and processed by the HYPPO algorithm. All other values are mapped by using the miscellaneous mapper. The results of all mappers processing are filtered and merged by using time-efficient set and filtering operations.

The advantages of LIMES' approach are manifold. First, it implements **highly time-optimized** mappers, making it a complexity class faster than other Link Discovery Frameworks. Thus, the larger the problem, the faster LIMES is w.r.t. other Link Discovery Frameworks. In addition, **LIMES is guaranteed to lead to exactly the same matching as a brute force approach while at the same time reducing significantly the number of comparisons**. In addition, LIMES supports a **large number of input and output formats** and can be extended very easily to fit new algorithms , new datatypes, new preprocessing functions and others thank to its modular architecture displayed in Figure 2.

In general, LIMES can be used to set links between two data sources, e.g., a novel data source created by a data publisher and existing data source such as DBpedia[1]. This functionality can also be used to detect duplicates within one data source for knowledge curation. The only

[1] `http://dbpedia.org`

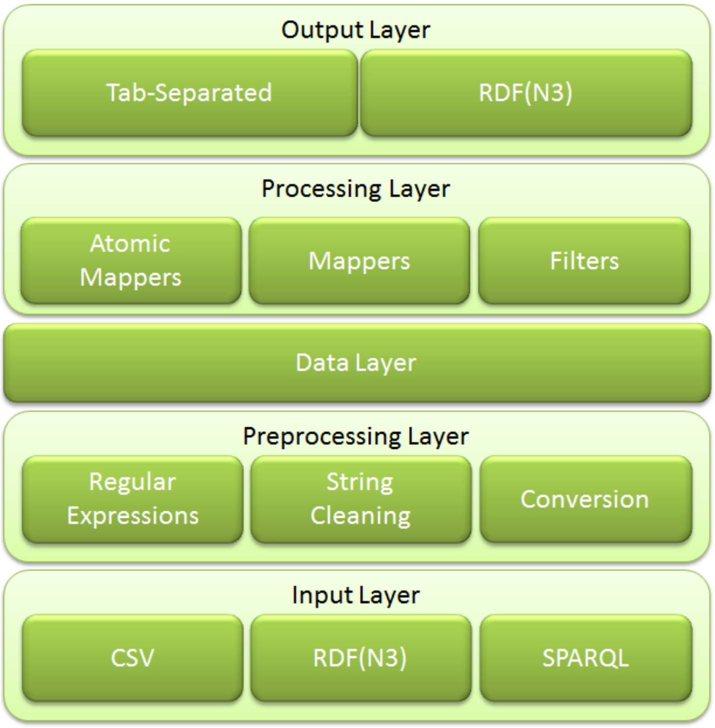

Figure 2: Architecture of LIMES

requirement to carry out these tasks is a simple XML-based configuration file. The purpose of this manual is to explicate the LIMES Configuration Language (LCL) that underlies these configuration files, so as allow users to generate their own configurations. An online version of LIMES is available online at `http://limes.aksw.org`.

2 Components of a LIMES Configuration File

A LIMES configuration file consists of ten parts, of which some are optional:

1. Metadata
2. Prefixes
3. Source data source
4. Target data source
5. Metric for similarity measurement
6. Acceptance condition
7. Review condition
8. Execution mode (optional)
9. Granularity (optional)
10. Output format

In the following, we will explicate these components by showing successively how LIMES can be configured to compute a mapping between diseases contained in Bio2RDF and LinkedCT.

2.1 Metadata

The metadata for a LIMES config file always consists of the following bits of XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LIMES SYSTEM "limes.dtd">
<LIMES>
```

2.2 Prefixes

Defining a prefix in a LIMES file demands setting two values: the namespace that will be addresses by the prefix and the prefix per se, as shown below.

```
<PREFIX>
    <NAMESPACE>http://www.w3.org/1999/02/22-rdf-syntax-ns#</NAMESPACE>
    <LABEL>rdf</LABEL>
</PREFIX>
```

Here, we set the prefix `rdf` to correspond to `http://www.w3.org/1999/02/22-rdf-syntax-ns#`. A LIMES link specification can contain as many prefixes as required.

2.3 Source Data Source

LIMES computes links between items contained in two Linked Data sources dubbed source and target. An example of a configuration for a source data source is shown below.

```
<SOURCE>
  <ID>mesh</ID>
  <ENDPOINT>http://mesh.bio2rdf.org/sparql</ENDPOINT>
  <VAR>?y</VAR>
  <PAGESIZE>5000</PAGESIZE>
  <RESTRICTION>?y rdf:type meshr:Concept</RESTRICTION>
  <PROPERTY>dc:title</PROPERTY>
  <TYPE>sparql</TYPE>
</SOURCE>
```

Six properties need to be set.

1. Each data source must be given an ID via the tag `ID`.

2. The SPARQL endpoint of the data source needs to be explicated via the `ENDPOINT` tag. In case local files (CSV, N3, TURTLE, etc.) are to be linked, `ENDPOINT` should be set to the absolute path of the file containing the data to link.

3. The variable associated with this endpoint must be specified. This is done by setting the `VAR` tag. This variable is used later when specifying the metric used to compare the entities retrieved from the source and target endpoints.

4. The fourth property is set via the `PAGESIZE` tag. This property must be set to the maximal number of triples returned by the SPARQL endpoint to address. For example, the DBpedia endpoint at `http://dbpedia.org/sparql` returns a maximum of 1000 triples for each query. LIMES' SPARQL module can still retrieve all relevant instances for the mapping if given this value. If the SPARQL endpoint does not limit the number of triple it returns or if the input is a file, the value of `PAGESIZE` should be set to -1.

5. The restrictions of the data to retrieved can be set via the `RESTRICTION` tag. This tag allows to limit the entries that are retrieved the LIMES' query module. In this particular example, we only instances of MESH concepts.

6. The `PROPERTY` tag allows to specify the properties that will be used during the linking. It is important to note that the property tagcan also be used to specify the preprocessing on the input data. For example, setting `rdfs:label AS nolang`, one can ensure that the language tags get removed from each `rdfs:label` before it is written in the cache. Preprocessing functions can be piped into one another by using `->`. For example, `rdfs:label AS nolang->lowercase` will compute `lowercase(nolang(rdfs:label))`.

The pre-processing functions include:

- `nolang` for removing language tags,
- `lowercase` for converting the input string into lower case,
- `uppercase` for converting the input string into upper case,
- `number` for ensuring that only the numeric characters, “.” and “,” are contained in the input string,
- `replace(String a,String b)` for replacing each occurrence of `a` with `b`,
- `cleaniri` for removing all the prefixes from IRIs,
- `celsius` for converting Fahrenheit to Celsius,
- `fahrenheit` for converting Celsius to Fahrenheit.

Sometimes, generating the right link specification might either require merging property values (for example, the `dc:title` and `foaf:name` of MESH concepts) or splitting property values (for example, comparing the label and `foaf:homepage` of source instances and the `foaf:homepage` of target instances as well as `foaf:homepage AS cleaniri` of the target instances with the `rdfs:label` of target instances. To enable this goal, LIMES provides the `RENAME` operator which simply store either the values of a property or the results of a preprocessing into a different property field. For example, `foaf:homepage AS cleaniri RENAME label` would stored the homepage of a object without all the prefixes in the name property. The user could then access this value during the specification of the similarity measure for comparing sources and target instances. Note that the same property value can be used several times. Thus, the following specification fragment is valid and leads to the the `dc:title` and `foaf:name` of individuals) of MESH concepts being first cast down to the lowercase and then merged to a single property.

```
<SOURCE>
  <ID>mesh</ID>
  <ENDPOINT>http://mesh.bio2rdf.org/sparql</ENDPOINT>
  <VAR>?y</VAR>
  <PAGESIZE>5000</PAGESIZE>
  <RESTRICTION>?y rdf:type meshr:Concept</RESTRICTION>
  <PROPERTY>dc:title AS lowercase RENAME name</PROPERTY>
  <PROPERTY>foaf:name AS lowercase RENAME name</PROPERTY>
  <TYPE>sparql</TYPE>
</SOURCE>
```

In addition, the following allows splitting the values of `foaf:homepage` into the property values `name` and `homepage`.

```
<SOURCE>
  <ID>mesh</ID>
  <ENDPOINT>http://mesh.bio2rdf.org/sparql</ENDPOINT>
  <VAR>?y</VAR>
  <PAGESIZE>5000</PAGESIZE>
  <RESTRICTION>?y rdf:type meshr:Concept</RESTRICTION>
  <PROPERTY>foaf:homepage AS lowercase RENAME homepage</PROPERTY>
  <PROPERTY>foaf:homepage AS cleaniri->lowercase RENAME name</PROPERTY>
  <TYPE>sparql</TYPE>
</SOURCE>
```

In addition, a source type can be set via TYPE. The default type is set to SPARQL (for a SPARQL endpoint) but LIMES also supports reading files directly from the harddrive. The supported data formats are

- CSV: Character-separated file can be loaded directly into LIMES. Note that the separation character is set to TAB as a default. The user can alter this setting programmatically.
- N3 (which also reads NT files) reads files in the N3 language.
- N-TRIPLE reads files in W3C's core N-Triples format.[2]
- TURTLE allows reading files in the Turtle syntax.[3]

Consequently, if you want to download data from a SPARQL endpoint, there is no need to set the <TYPE> tag. If instead you want to read the source (or target) data from a file, the <ENDPOINT> tag should contain the path to the file to read, e.g. <ENDPOINT>C:/Files/dbpedia.nt</ENDPOINT> In addition, the <TYPE> tag then needs to be set, for example by writing <TYPE>NT</TYPE>.

2.4 Target Data Source

Configuring the target data source is very similar to configuring the source data source. The only difference lies in the beginning tag, i.e., TARGET instead of SOURCE. In the example shown below, we retrieve the condition_name of a condition from LinkedCT. We do no set the type of the source. Thus, LIMES supposes it is a SPARQL endpoint.

```
<TARGET>
  <ID>linkedct</ID>
  <ENDPOINT>http://data.linkedct.org/sparql</ENDPOINT>
  <VAR>?x</VAR>
  <PAGESIZE>5000</PAGESIZE>
  <RESTRICTION>?x rdf:type linkedct:condition</RESTRICTION>
  <PROPERTY>linkedct:condition_name</PROPERTY>
</TARGET>
```

2.5 Metric Expression for Similarity Measurement

One of the core improvements of the newest LIMES kernels is the provision of a highly flexible language for the specification of complex metrics for linking (set by using the METRIC tag as exemplified below).

```
<METRIC>
trigrams(y.dc:title, x.linkedct:condition_name)
</METRIC>
```

In this example, we use the Trigrams metric to compare the dc:title of the instances retrieved from the source data source, with which the variable y is associated, with the linkedct:condition_name of the instances retrieved from the target data source, with which the variable x is associated. While such simple metrics can be used in many cases, complex metrics are necessary in complex linking cases. LIMES includes a formal grammar for specifying complex

[2] http://www.w3.org/TR/rdf-testcases/#ntriples

[3] http://www.w3.org/TR/turtle/

configurations of arbitrary complexity. For this purpose, two categories of binary operations are supported: Metric operations and boolean operations.

Metric operations allow to combine metric values. They include the operators MIN, MAX, ADD and MULT, e.g. as follows:

```
MAX(trigrams(x.rdfs:label,y.dc:title),euclidean(x.lat|long, y.latitude|longitude)).
```

This specification computes the maximum of (1) the trigram similarity of x's rdfs:label and y's dc:title and (2) the 2-dimension euclidean distance of x's lat and long mit y's latitude and longitude, i.e.,
$\sqrt{(x.lat - y.latitude)^2 + (x.long - y.longitude)^2}$.

Note that euclidean supports arbitrarily many dimensions. In addition, note that ADD allows to define weighted sums as follows:

```
ADD(0.3*trigrams(x.rdfs:label,y.dc:title),
    0.7*euclidean(x.lat|x.long, y.latitude|y.longitude)).
```

Boolean operations allow to combine and filter the results of metric operations and include AND, OR, DIFF, e.g. as follows:

```
AND(trigrams(x.rdfs:label,y.dc:title)|0.9,
    euclidean(x.lat|x.long, y.latitude|y.longitude)|0.7).
```

This specification returns all links such that (1) the trigram similarity of x's rdfs:label and y's dc:title is greater or equal to 0.9 and (2) the 2-dimension euclidean distance of x's lat and long mit y's latitude and longitude is greater or equal to 0.7.

The current version of LIMES supports the string metrics

- Trigrams,
- Cosine,
- Jaccard,
- Levenshtein,
- Jaro and
- Jaro-Winkler

Overlap as well as Monge-Elkan are currently being added. In addition it supports comparing numeric vectors by using the

- Euclidean metric as well as
- the Orthodromic distance.

While the Euclidean measure can deal with n-dimensional data, the orthodromic distance assumes that it is given a WKT POINT as input. If the input is a polygon, it uses the Hausdorff distance. The similarity between polygons can be measured by using the

- Hausdorff distance,
- Sum of minimums distance,
- Frchet distance,
- Fair surjection,
- Surjection and

- `SymmetricHausdorff` distance.

Currently, these distances can deal with POLYGON and LINESTRING. More complex distance measures are being added.

2.6 Acceptance Condition

Setting the acceptance condition basically consists of setting the value for the threshold above which links are considered to be valid and not to required further curation. This can be carried out as exemplified below.

```
<ACCEPTANCE>
  <THRESHOLD>0.98</THRESHOLD>
  <FILE>accepted.nt</FILE>
  <RELATION>owl:sameAs</RELATION>
</ACCEPTANCE>
```

By using the `THRESHOLD` tag, the user can set the value for the metric value above which two instances are considered to be linked via the relation specified by using the tag `RELATION`, i.e., `owl:sameAs` in our example. Setting the tag `FILE` allows to specify where the links should be written. Currently, LIMES produces output files in the N3 format.

Future versions of LIMES will allow to write the output to other streams and in other data formats.

2.7 Review Condition

Setting the condition upon which links must be reviewed manually is very similar to setting the acceptance condition as shown below.

```
<REVIEW>
  <THRESHOLD>0.95</THRESHOLD>
  <FILE>reviewme.nt</FILE>
  <RELATION>owl:sameAs</RELATION>
</REVIEW>
```

All instances that have a similarity between the threshold set in `REVIEW` (0.95 in our example) and the threshold set in `ACCEPTANCE` (0.98 in our example) will be written in the review file and linked via the relation set in `REVIEW`.

The LIMES configuration file should be concluded with `</LIMES>`

2.8 Execution Mode (optional)

The user can choose between the executions modes `SIMPLE` and `FILTER` to tune LIMES' runtime.

```
<EXECUTION>SIMPLE</EXECUTION>.
```

Moreover, the user can select how the mappings returned by LIMES are to be postprocessed. `OneToN` leads to LIMES returning only the best matching t to any given s in the mapping $M = \{(s,t) \in S \times T\}$.`OneToOne` leads to LIMES aiming to find the best one-to-one mapping out of the output in a way similar to that above.

```
<EXECUTION>SIMPLE</EXECUTION>.
```

2.9 Granularity (optional)

The user can choose positive integers to set the granularity of HYPPO, HR3 or ORCHID by setting

```
<GRANULARITY>2</GRANULARITY>.
```

2.10 Output Format

The user can choose between TAB and N3 as output format by setting

```
<OUTPUT>N3</OUTPUT>
```

3 Example of a Configuration File

The following shows the whole configuration file for LIMES explicated in the sections above.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LIMES SYSTEM "limes.dtd">
<LIMES>
  <PREFIX>
    <NAMESPACE>http://www.w3.org/1999/02/22-rdf-syntax-ns#</NAMESPACE>
    <LABEL>rdf</LABEL></PREFIX>
  <PREFIX>
    <NAMESPACE>http://www.w3.org/2000/01/rdf-schema#</NAMESPACE>
    <LABEL>rdfs</LABEL></PREFIX>
  <PREFIX>
    <NAMESPACE>http://www.w3.org/2002/07/owl#</NAMESPACE>
    <LABEL>owl</LABEL></PREFIX>
  <PREFIX>
    <NAMESPACE>http://data.linkedct.org/resource/linkedct/</NAMESPACE>
    <LABEL>linkedct</LABEL></PREFIX>
  <PREFIX>
    <NAMESPACE>http://purl.org/dc/elements/1.1/</NAMESPACE>
    <LABEL>dc</LABEL></PREFIX>
  <PREFIX>
    <NAMESPACE>http://bio2rdf.org/ns/mesh#</NAMESPACE>
    <LABEL>meshr</LABEL></PREFIX>

  <SOURCE>
    <ID>mesh</ID>
    <ENDPOINT>http://mesh.bio2rdf.org/sparql</ENDPOINT>
    <VAR>?y</VAR>
    <PAGESIZE>5000</PAGESIZE>
    <RESTRICTION>?y rdf:type meshr:Concept</RESTRICTION>
    <PROPERTY>dc:title</PROPERTY>
  </SOURCE>

  <TARGET>
    <ID>linkedct</ID>
    <ENDPOINT>http://data.linkedct.org/sparql</ENDPOINT>
    <VAR>?x</VAR>
    <PAGESIZE>5000</PAGESIZE>
    <RESTRICTION>?x rdf:type linkedct:condition</RESTRICTION>
    <PROPERTY>linkedct:condition_name</PROPERTY>
  </TARGET>

  <METRIC>
   MAX(trigrams(y.dc:title, x.linkedct:condition_name),
       cosine(y.dc:title, x.linkedct:name))
  </METRIC>

  <ACCEPTANCE>
    <THRESHOLD>0.98</THRESHOLD>
    <FILE>accepted.txt</FILE>
    <RELATION>owl:sameAs</RELATION>
  </ACCEPTANCE>
  <REVIEW>
    <THRESHOLD>0.95</THRESHOLD>
    <FILE>reviewme.txt</FILE>
    <RELATION>owl:sameAs</RELATION>
  </REVIEW>
</LIMES>
```

4 The LIMES Distribution

4.1 Content

The LIMES distribution in its current version 0.5.RC1 contains the files

- `LIMES.jar`, which implements our framework,
- `limes.dtd`, the data type definition for LIMES configuration files and
- `user_manual.pdf`, this file.

In addition, it contains the folders

- `lib`, which contains all the libraries necessary to run our framework and
- `examples`, which contains examples of configuration files.

4.2 Running the Framework

Once the configuration file (dubbed `config.xml` in this manual) has been written, the last step consists of actually running the LIMES framework. For this purpose, simply run

```
java -jar LIMES.jar config.xml.
```

In case your system runs out of memory, please use the `-Xmx` option to allocate more memory to the Java Virtual Machine. Please ensure that the Data Type Definition file for LIMES, `limes.dtd`, is in the same folder as the `LIMES.jar` and everything should run just fine. Enjoy.

5 Support Information

For support, please contact:

Axel-Cyrille Ngonga Ngomo
Johanisgasse 26
Room 5-22
04103 Leipzig
`ngonga@informatik.uni-leipzig.de`

6 License and Warranty Information

LIMES is free to use for non-commercial purposes. For any kind of commercial use, please contact us. Also note that LIMES is distributed without any warranty of any type.

7 Known Issues

None.

8 Change log

8.1 Version 0.6RC4

- Added support for several geo-spatial similarity functions (geomean, surjection, fairsurjection, geosumofmin, frechet, link)

- Added support for temporal geo-spatial similarity functions (daysim, datesim, yearsim)
- Added parallel implementation for ORCHID
- Added support for Jaro and Jaro-Winkler

8.2 Version 0.6RC3

- Added support for geo-spatial similarity function based on Hausdorff distance
- Added support for geo-spatial similarity function based on symmetric Hausdorff distance
- Added support for orthodromic distance
- Implemented ORCHID for time-efficient linking of geo-spatial resources
- Added support for exact matches

8.3 Version 0.6RC2

- Time-efficient self-configuration (genetic, linear, boolean)
- Can now read use most RDF serialization formats (RDF/XML, N3, NT, TTL) as input

8.4 Version 0.6RC1

- Kernel update
- HR3 algorithm for vector space. Default granularity is now 4.
- Update for data readers and writers.
- Genetic Learning

8.5 Version 0.5RC1

- Kernel change, more than 4 orders of magnitude faster
- HYPPO algorithm for vector spaces
- Fast prefix, suffix and position filtering for strings
- Support for more metrics

8.6 Version 0.4.1

- Added support for data source type (tab-separated vectors)
- Added factory for query modules

8.7 Version 0.4

- Added support for data source type (Sparql, CSV)
- Added hybrid cache
- Implemented CSV reader
- Faster organizer