NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE

DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

POSTGRADUATE STUDIES

MASTER THESIS

Discovering Spatial and Temporal Links Among RDF Data

Panayiotis Smeros

Supervisor: **Manolis Koubarakis**, Professor UoA

ATHENS

December 2014

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΕΣ ΣΠΟΥΔΕΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανακαλύπτοντας Χωρικούς και Χρονικούς Συνδέσμους Μεταξύ RDF Δεδομένων

Παναγιώτης Σμέρος

Επιβλέπων: **Μανόλης Κουμπαράκης**, Καθηγητής ΕΚΠΑ

ΑΘΗΝΑ

Δεκέμβριος 2014

MASTER THESIS

Discovering Spatial and Temporal Links Among RDF Data

Panayiotis Smeros

R.N.: M1227

SUPERVISOR:
Manolis Koubarakis, Professor UoA

EXAMINATION COMMITEE:
Isambo Karali, Assistant Professor UoA

ATHENS
December 2014

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανακαλύπτοντας Χωρικούς και Χρονικούς Συνδέσμους Μεταξύ RDF Δεδομένων

Παναγιώτης Σμέρος
Α.Μ.: Μ1227

ΕΠΙΒΛΕΠΩΝ:
Μανόλης Κουμπαράκης, Καθηγητής ΕΚΠΑ

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:
Ιζαμπώ Καράλη, Επίκουρη Καθηγήτρια ΕΚΠΑ

ΑΘΗΝΑ
Δεκέμβριος 2014

Abstract

Link Discovery is a new research area of the Semantic Web which studies the problem of finding semantically related entities lying in different knowledge bases. This area has become more crucial recently, as the volume of the available Linked Data on the web has been increasing considerably. Although many link discovery tools have been developed, most of them do not take into consideration the discovery of spatial and temporal relations, leaving datasets with such characteristics weakly interlinked and therefore not allowing the exploitation of the rich information they provide.

In this thesis, we propose new formalisms and accurate methods for Spatial and Temporal Link Discovery and provide the first implementation that covers this area. This implementation is based on the well-adapted framework Silk. Silk, enhanced with the new features, allows data publishers to discover a wide variety of spatial and temporal relations between their data and other Linked Open Data. We also experimentally evaluate Silk by using it in a real-world scenario, and showcase that it can generate 100% accurate links in a time efficient and scalable way.

SUBJECT AREA: Link Discovery
KEYWORDS: Spatial and Temporal Link Discovery, Semantic Web, Linked Data

Περίληψη

Η Ανακάλυψη Συνδέσμων είναι μια νέα ερευνητική περιοχή του Σημασιολογικού Ιστού που μελετά το πρόβλημα της εύρεσης σημασιολογικά συσχετιζόμενων οντοτήτων οι οποίες βρίσκονται σε διαφορετικές βάσεις γνώσης. Η περιοχή αυτή, τελευταία, γίνεται όλο και πιο χρήσιμη, καθώς ο όγκος των διαθέσιμων Διασυνδεδεμένων Δεδομένων στο διαδίκτυο αυξάνεται σημαντικά. Παρά το γεγονός ότι έχουν αναπτυχθεί πολλά εργαλεία για Ανακάλυψη Συνδέσμων, τα περισσότερα από αυτά δεν λαμβάνουν υπ' όψιν την εύρεση χωρικών και χρονικών σχέσεων, αφήνοντας δεδομένα με τέτοιου είδους χαρακτηριστικά ασθενώς συνδεδεμένα και μη επιτρέποντας την εκμετάλλευση της πλούσιας πληροφορίας που παρέχουν.

Σε αυτή την εργασία, προτείνουμε νέους φορμαλισμούς και ακριβείς μεθόδους για Χωρική και Χρονική Ανακάλυψη Συνδέσμων και παρέχουμε την πρώτη υλοποίηση που καλύπτει αυτή την περιοχή. Η υλοποίηση είναι βασισμένη στο γνωστό εργαλείο Silk. Το Silk, εμπλουτισμένο με τις νέες επεκτάσεις που αναπτύξαμε, επιτρέπει στους παρόχους δεδομένων να ανακαλύπτουν μεγάλη ποικιλία χωρικών και χρονικών σχέσεων μεταξύ των δεδομένων τους και άλλων Ανοιχτών Διασυνδεδεμένων Δεδομένων. Επίσης, αξιολογούμε πειραματικά το Silk, χρησιμοποιώντας το σε ένα σενάριο πραγματικού κόσμου, αποδεικνύοντας ότι μπορεί να ανακαλύψει 100% ακριβείς συνδέσμους με ένας χρονικά αποδοτικό και επεκτατό τρόπο.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Ανακάλυψη Συνδέσμων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Χωρική και Χρονική Ανακάλυψη Συνδέσμων, Σημασιολογικός Ιστός, Διασυνδεδεμένα Δεδομένα

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου και επιβλέποντα αυτής της διπλωματικής, Μανόλη Κουμπαράκη, για την καθοδήγησή του, καθώς και όλη την ερευνητική του ομάδα, της οποίας είχα την τιμή να είμαι μέλος, για την άριστη συνεργασία και την πολύτιμη στήριξη τους κατά τη διάρκεια εκπόνησης της εργασίας αυτής.

Επίσης θα ήθελα να ευχαριστήσω τους Hervé Caumont και Cesare Rossi από την Terradue (`http://www.terradue.com`) για την βοήθεια τους στην εκτέλεση των πειραμάτων σε περιβάλλον cloud.

Αυτή η εργασία χρηματοδοτήθηκε μερικώς από τα FP7 projects LEO (611141) (`http://linkedeodata.eu`) και MELODIES (603525) (`http://www.melodiesproject.eu`).

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Linked data is a research area which studies how one can make RDF data available on the Web, and interlink it with other data in order to increase its value for users [5]. The goal of Linked Data is to allow people to share structured data on the web as easily as they can do with documents today.

An important step for the evolution of the Web of documents to the Web of data is the transformation of the data from any form that they exist into a common format, the Resource Description Framework (RDF) so that it can be easily integrated with other data already transformed in this format. All the data that is compatible with the Linked Data Principles[1] composes the Linked Open Data (LOD) cloud (Figure 1.1).

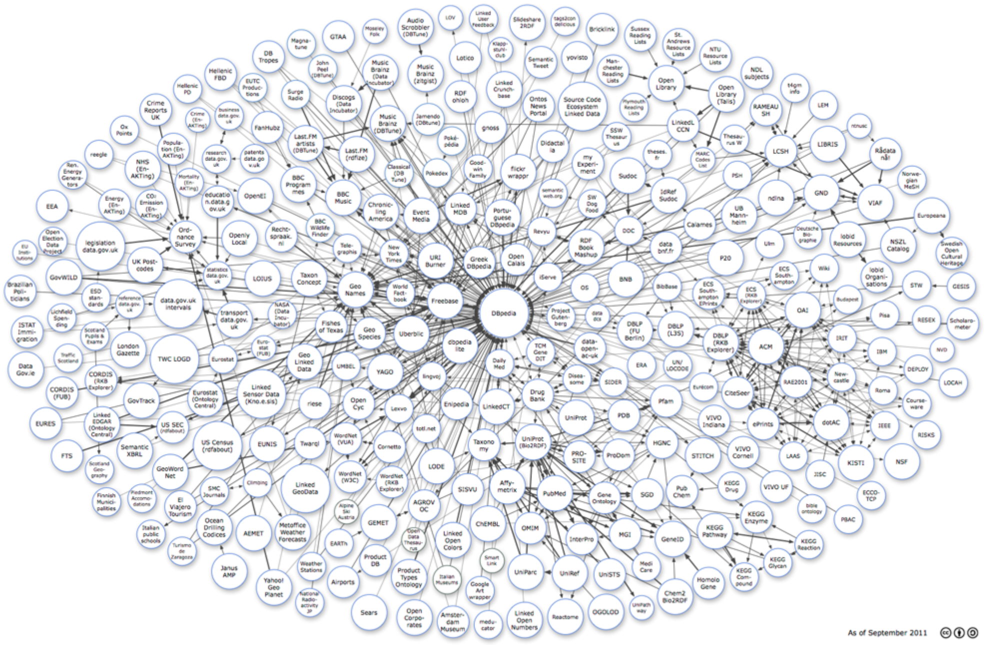
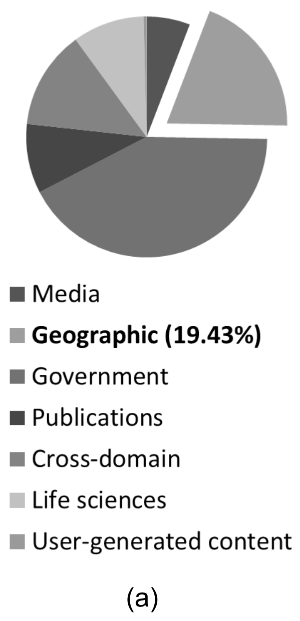


Figure 1.1: Recent state of the Linked Open Data cloud

Recently, spatial and temporal extensions to RDF have been proposed and implemented. GeoSPARQL [28] is a recent OGC standard that allows representing and querying geospatial data on the Semantic Web. Also, the data model stRDF accompanied by

[1]`http://www.w3.org/DesignIssues/LinkedData.html`



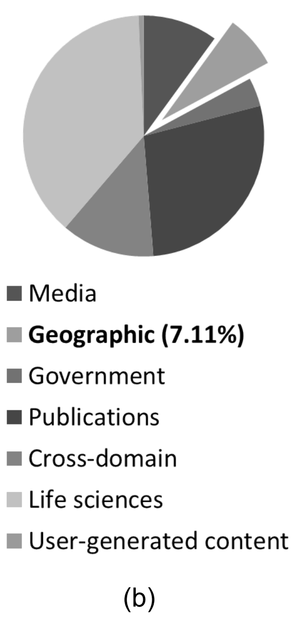


Figure 1.2: Distribution of (a) data by domain, (b) links by domain in the LOD Cloud

the query language stSPARQL [21] are extensions of the standard RDF and SPARQL for representing and querying geospatial data that changes over time. Both of the above extensions are implemented in the open source spatiotemporal RDF store Strabon [22].

Link Discovery is the fourth and the most important Linked Data Principle. Its main objective is to establish semantic links between entities in order to enhance and enrich the information that is known about them. Whilst the problem of Entity Resolution i.e., the problem of finding entities which are equivalent, has been studied a lot in areas such as Relational Databases and Information Retrieval, Link Discovery defines the more generic problem of finding semantically related entities lying in different knowledge bases [2].

Although a lot of effort has been given in the representation and the querying of geospatial and temporal RDF data, there are not many works in the respective area of Link Discovery. In the context of Spatial Link Discovery, state of the art techniques are focusing on finding only spatial equivalences between entities, leaving other kinds of relations e.g., topological relations, undiscovered and the rich geospatial information lying in many datasets unexploited (Figure 1.2). The situation regarding Temporal Link Discovery is even more premature and thus, to the best of our knowledge, there are almost no datasets with temporal information that are connected with each other with links that denote a temporal relation.

Another common use of Link Discovery is for detecting internal links within a single dataset. For example, by applying an Entity Resolution method on a dataset, we can discover all the similar entities i.e., all the duplicates of it. Hence, with Spatial and Temporal Link Discovery we can materialize all the spatial and temporal relations that hold between the entities of a dataset. This operation is very useful in the areas of Qualitative Spatial [32], Temporal [37, 25] and SpatioTemporal Reasoning [14, 31] where the large graphs that are created based on the qualitative relations are given as input to corresponding reasoners [12, 15] in order to extract useful information or verify the consistency of a dataset.

The lack of research in the area Spatial and Temporal Link Discovery will be made more notable when more datasets with such characteristics will be made available in the LOD Cloud. Currently, a lot of initiatives are moving towards this direction by publishing open geospatial data and metadata coming out of open government directives[2] and open Earth Observation (EO) data and metadata that is currently made available by space and environment agencies (e.g., ESA, NASA and EEA)[3]. This data is usually measurements produced by observations with fundamental geospatial and temporal aspects. Making all this data available as linked data and interlinking them with semantic connections will allow the development of services with great environmental and commercial value.

Recently, EU projects such as LEO[4] and MELODIES[5], following the footsteps of TELEIOS[6], started exploiting this kind of datasets. In the context of these projects the whole life cycle of EO data is being studied. This life cycle includes stages such as Knowledge Extraction and Data Mining from EO data in raw form (e.g., satellite images), Transformation into RDF, Interlinking, etc. and has been specified in detail in [19].

In this thesis, we focus on the Interlinking stage of this life cycle. Specifically, we examine state of the art techniques and tools in the area of Link Discovery and propose new formalisms and accurate methods for Spatial and Temporal Link Discovery. Also, we provide the first implementation that covers this area which is based on the well-adapted framework Silk[7]. Silk, enhanced with the new features, allows data publishers to discover a wide variety of spatial and temporal relations between their data and other Linked Open Data. We also experimentally evaluate Silk by using it in a real-world scenario with datasets that contain very detailed and complex geometries, and showcase that it can generate 100% accurate links in a time efficient and scalable way.

[2]http://www.linkedopendata.gr/
[3]http://datahub.io/organization/teleios and http://datahub.io/organization/leo
[4]http://www.linkedeodata.eu/
[5]http://www.melodiesproject.eu/
[6]http://www.earthobservatory.eu/
[7]http://silk.wbsg.de/

The structure of the thesis is organized as follows. In Chapter 2 we provide the related work on the area of Link Discovery and in Chapter 3 some background knowledge on the representation of geospatial and temporal information. In Chapter 4 we propose new formalisms and accurate methods which cover the area of Spatial and Temporal Link Discovery. In Chapter 5 we desrcibe the implementation of the above methods on the Link Discovery Framework Silk and in Chapter 6 we present the experimental evaluation of them. Finally, in Chapter 7 we provide an example of using Silk in project LEO and in Chapter 8 we conclude the work by discussing future directions.

Chapter 2
Related Work

Up to now, little effort has been given in the research area of Spatial and Temporal Link Discovery. Most of the approaches on generic Link Discovery do not exploit the rich spatial and temporal information existing in some datasets, whereas domain specific approaches on Spatial Link Discovery are able to discover only spatial similarities. Hence, to the best of our knowledge, there are no approaches, either generic or domain specific, for discovering spatial or temporal relations other than equivalences among RDF datasets.

In the area of generic Link Discovery, the surveys [11, 2] perform a detailed review on state of the art algorithms and frameworks. Also, in [9, 10, 24, 6] the authors propose various methods that use Genetic Programming, Probabilistic, Logic Programming and Data Analytics techniques respectively. Hence, data integration platforms such as WOO and RDF-AI [3, 34], comprise components which also implement Link Discovery algorithms.

In the same area, the authors of [16] propose the declarative link specification language LinQL, which is translated to standard SQL by the framework LinQuer, for discovering semantic links over relational data.

Also, the LIMES framework [26] introduces a generic algorithm for Link Discovery which reduces the number of comparisons that are needed during the interlinking phase by utilizing the triangle inequality in metric spaces. For finding link specifications, LIMES implements supervised and unsupervised machine learning algorithms.

Similarly to LIMES, Silk [18] is also a generic framework for discovering relationships between data items within different Linked Data sources. Silk, which is the only open source generic Link Discovery framework, features a declarative link specification language for specifying which types of RDF links should be discovered between data sources as well as which conditions entities must fulfill in order to be interlinked. These linkage rules may combine various metrics and can take the graph around entities into account, which is addressed using an RDF path language. Silk accesses the data sources that should be interlinked via the SPARQL protocol and can thus be used against local as well as remote SPARQL endpoints.

In the area of Spatial Link Discovery there are some domain specific approaches which are able to discover only spatial equivalences among datasets. In order to achieve this, they combine the geographic distance of the geometries of the entities with other kinds of distances e.g., with the string distance of their labels. The supported geographic distances can be applied either between any kind of spatial objects (e.g., Hausdorff distance), or only between point objects (e.g., Orthodromic distance). Some of these approaches are presented in [38, 35, 33].

From the generic frameworks, LIMES also addresses the problem of Spatial Link Discovery in [27]. The computation of the distance between the spatial objects lies on a combination of Hausdorff and Orthodromic metrics. On the other hand, Silk supports the geographic distance only between point objects.

Chapter 3
Background

In this chapter we present in detail the background on which we were based in order to design the new methods for Spatial and Temporal Link Discovery and implement them on the Silk framework. Specifically, we present some preliminary knowledge on the representation of geospatial and temporal information and focus on how this representation is adopted in the RDF model.

3.1 Representation of Geospatial Information

The Open Geospatial Consortium[1] (OGC) has developed well-known standards that focus on solutions for geospatial data and services, GIS data processing and geospatial data sharing.

In OGC terminology, a *geographic feature* is an abstraction of a real world phenomenon and can have various attributes that describe its thematic and spatial characteristics. For example, a feature can represent a municipality. Thematic information about a municipality can include its name, its population, etc., while a spatial characteristic is its location on Earth. The spatial characteristics of a feature are represented using *geometries* such as points, lines, and polygons. Each geometry is associated with a *coordinate reference system* which describes the coordinate space in which the geometry is defined.

3.1.1 Coordinate Reference System

A *coordinate reference system* defines how to relate the coordinates of a geometric object to real locations on the surface of Earth. A *geographic coordinate reference system* is a three-dimensional coordinate reference system that utilizes latitude, longitude, and optionally altitude, to capture geographic locations on Earth [23]. In a geographic coordinate reference system, the earth's three-dimensional ellipsoid is mapped using a series of horizontal (longitude lines or parallels) and vertical (latitude lines or meridians) reference lines. The World Geodetic System (WGS) is the most well-known geographic coordinate

[1] `http://www.opengeospatial.org/`

reference system and its latest revision is WGS84[2]. WGS84 is the reference coordinate system used by the Global Positioning System (GPS).

Although a geographic coordinate system such as WGS84 is a comprehensive way to describe locations on Earth, some applications work on a projection of the Earth. In these cases a *projected geographic coordinate reference system* is used that transforms the 3-dimensional ellipsoid approximation of the Earth into a 2-dimensional surface. Individual countries or states have their own projected coordinate reference systems that are more precise for their geographic area (e.g., GGRS87[3] for Greece).

3.1.2 Well-Known Text

Well-Known Text (WKT) is a widely used OGC standard for representing geometries. More specifically, WKT can be used for the representation of vector geometry objects, on a map, spatial reference systems, and transformations between spatial reference systems. For example, *POINT(10 20)*, is the WKT representation of the point with longitude 10 and latitude 20. The complete syntax of the representation is presented in detail in [17]. The interpretation of the coordinates of a geometry depends on the coordinate reference system that is associated with the geometry, which, according to the WKT standard, is never embedded in the object's representation, but is given separately using appropriate notation.

3.1.3 Geography Markup Language

The Geography Markup Language (GML) [29] is the most common XML-based encoding standard for the representation of geospatial data. GML was developed by OGC and it is based on the OGC Abstract Specification. GML provides XML schemas for defining a variety of concepts that are of use in Geography: geographic features, geometry, coordinate reference systems, topology, time and units of measurement. Initially, the GML abstract model was based on RDF and RDFS, but later the consortium decided to use XML and XML Schema. The complete syntax of the GML representation of a geometry is presented in [29].

[2]`http://en.wikipedia.org/wiki/WGS84/`

[3]`http://spatialreference.org/ref/epsg/ggrs87-greek-grid/`

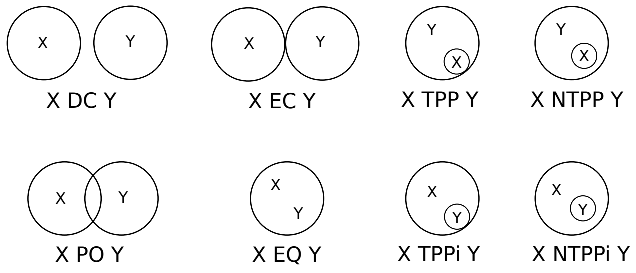


Figure 3.1: RCC-8 topological relations

3.1.4 Spatial Relations

Two well-know models for the representation of spatial relations are DE-9IM and RCC.

DE-9IM

The Dimensionally Extended 9-Intersection Model (DE-9IM) [7] is a well-known model for representing topological relations between geometries. More specifically, this model captures topological relations in $\mathbb{R}^2$, by considering the dimension of the intersections involving the interior (I), the boundary (B) and the exterior (E) of the two geometries. The intersection matrix of two geometries a and b is defined as follows:

$$\text{DE-9IM(a,b)} = \begin{bmatrix} dim(I(a) \cap I(b)) & dim(I(a) \cap B(b)) & dim(I(a) \cap E(b)) \\ dim(B(a) \cap I(b)) & dim(B(a) \cap B(b)) & dim(B(a) \cap E(b)) \\ dim(E(a) \cap I(b)) & dim(E(a) \cap B(b)) & dim(E(a) \cap E(b)) \end{bmatrix}$$

Given the above table, for any two spatial objects *a* and *b* that can be points, lines and/or polygonal areas, we can define relations derived from DE-9IM such as: *Intersects*, *Equals*, *Touches*, *Disjoint*, *Contains*, *Crosses*, *Covers*, *CoveredBy* and *Within*.

RCC

Another similar formalization that provides a sound and complete set of topological relations between two spatial regions is the Region Connection Calculus (RCC) [30]. RCC is a first order language which is used to represent spatial regions and reason about topological relations. Spatial regions are non empty subsets of some topological space, in our case $\mathbb{R}^2$, and may be associated by a topological relation, which is given as a constraint between them and is based on one primitive relation $C(a, b)$ which is true iff the closure of region a is connected to the closure of region b, i.e., they share a common point. Regions in general do not have to be internally connected, i.e., a region may consist of different disconnected parts. The domain of each spatial variable is the whole topological space ($\mathbb{R}^2$). Of particular importance are the sets that form a set of jointly exhaustive and pairwise disjoint relations (JEPD). These relations are called basic or atomic, because exactly one of them holds between any two regions.

Well known subsets of RCC are RCC-8 and RCC-5. RCC-8 (Figure 3.1.4), is based on the eight topological relations:

DC, EC, EQ, PO, TPP, NTPP, TPPi, NTPPi

where DC stands for DisConnected, EC for Externally Connected, TPP for Tangential Proper Part, NTPP, for Non Tangential Proper Part, and TPPi and NTPPi are the inverse relations from TPP and NTPP. RCC-5 is a looser subset which consists of five basic topological relations:

DR, PO, EQ, PP, PPi

The difference between RCC-5 and RCC-8 is that in RCC-5 there is no distinction between boundaries, and thus, DC and EC relations are merged into the relation DR and TPP and NTPP into PP respectively.

3.1.5 GeoSPARQL

GeoSPARQL [28] is a recent OGC standard that allows representing and querying geospatial data on the Semantic Web. It defines a vocabulary for representing geospatial data in RDF, and an extension to the SPARQL query language for processing geospatial data. The GeoSPARQL standard follows a modular design which is typical of OGC standards and contains several components required for such a query language by providing vocabulary (classes, properties and functions) that can be used in RDF graphs and SPARQL queries to represent and query geospatial data.

GeoSPARQL uses literal values to encode geometries as a single unit, and introduces two RDF datatypes, the `geo:wktLiteral` and `geo:gmlLiteral` for these literals. The representation of the geometry is parameterized by the serialization standard of OGC to be used for encoding geometry literals (WKT or GML) and the version of the relevant standard. Literals of type `geo:wktLiteral` consist of an optional URI identifying the coordinate reference system followed by the WKT encoding of a geometry. An example of a geometry in GeoSPARQL is given below:

```
_:1 rdf:type geo:Geometry .
_:1 geo:hasGeometry
"<http://www.opengis.net/def/crs/EPSG/0/4326>
    POINT(10 20)"^^geo:wktLiteral .
```

3.1.6 stRDF: The Spatial Dimension

The data model stRDF [21] is an extension of the standard RDF for representing geospatial data. stRDF uses the well known OGC standards WKT and GML for the representation of geospatial data and introduces two new literal datatypes, the `stdf:WKT` and `strdf:GML` for the representation of geometries encoded in WKT and GML respectively. Moreover, the datatype `strdf:geometry` is also introduced, which is defined as the union of the datatypes `stdf:WKT` and `strdf:GML`. The syntax of a literal with datatype `stdf:WKT` consists of the WKT encoding of the geometry followed by an optional URI indicating the coordinate reference system. An example of a geometry in stRDF is shown below:

```
_:1 rdf:type strdf:Geometry .
_:1 strdf:hasGeometry
"POINT(10 20);
    <http://www.opengis.net/def/crs/EPSG/0/4326>"^^strdf:WKT .
```

3.1.7 W3C GEO

The W3C GEO representation is a basic RDF vocabulary for representing mapping/location data in RDF. It provides the basic terminology for describing points using a namespace for representing latitude, longitude and other information about spatially-located things, using WGS 84 as a reference system. The simplicity of this representation lies on the fact that it does not require expensive pre-coordination or changes to a centrally maintened schema. Below, we give a basic, standalone example of W3C GEO vocabulary:

```
 _:1 rdf:type wgs84geo:Point .
 _:1 wgs84geo:lat "10"^^xsd:double.
 _:1 wgs84geo:long "20"^^xsd:double.
```

Relation	Illustration
X **before** Y Y **after** X	X _____ Y _____
X **meets** Y Y **isMetBy** X	X _____ Y _____
X **overlaps** Y Y **isOverlappedBy** X	X _____ Y _____
X **starts** Y Y **isStartedBy** X	X _____ Y __________
X **during** Y Y **contains** X	X ___ Y __________
X **finishes** Y Y **isFinishedBy** X	X _____ Y __________
X **equals** Y	X _____ Y _____

Figure 3.2: Allen's temporal relations

3.2 Representation of Temporal Information

The introduction of time in data models and query languages has been the subject of extensive research in the field of relational databases [36, 8]. Three distinct kinds of time were introduced and studied: *user-defined* time which has no special semantics (e.g., January 1st, 1963 when John has his birthday), *valid* time which is the time a fact is true in the application domain (e.g., the time 2000-2012 when John is a professor) and *transaction* time which is the time when a fact is current in the database (e.g., the system time that gives the exact period when the tuple representing that John is a professor from 2000 to 2012 is current in the database).

3.2.1 Temporal Relations

A widely used algebra for temporal reasoning is Allen's interval calculus [1], which provides the definition of possible relations between time intervals. It is based on the thirteen distinct and exhaustive qualitative relations given in Figure 3.2.1. From these basic relations, one can build new by taking combinations of their disjunctions. Using this calculus, one can formalize given facts which can be used for automated reasoning.

3.2.2 stRDF: The Temporal Dimension

An approach for the representation of temporal information in RDF was introduced with the temporal dimension of stRDF [4]. This approach assumes a discrete time line and uses the value space of the datatype `xsd:dateTime` of XML-Schema to model time. Two kinds of time primitives are supported: time instants and time periods. Time instants are represented by literals of the `xsd:dateTime` datatype and time periods by literals of the datatype `strdf:period`. The values of the datatype `strdf:period` are used as objects of a triple to represent *user-defined time* and as *valid time* of *temporal triples*.

A *temporal triple* is an expression of the form `(s, p, o, t)` where `(s, p, o)` is an RDF triple and `t` is a time instant or a time period called the *valid time* of a triple. An *stRDF graph* is a set of triples or temporal triples. In other words, some triples in an stRDF graph might not be associated with a valid time.

3.3 Summary

In this chapter we gave an overview of vocabularies and data models that allow the representation of geospatial and temporal information in RDF. We presented some preliminary knowledge and terminology regarding how geometries and dates are represented, then we described the theoretical work that has been conducted on the spatial and temporal relations, and finally, the respective extensions on the RDF data model in order to allow these representations on which lies our design and implementation of the new methods for Spatial and Temporal Link Discovery.

Chapter 4
Formalisms and Methods for Spatial and Temporal Link Discovery

In this chapter we present the new formalisms and methods for Spatial and Temporal Link Discovery. Specifically, we provide a generalized definition of Link Discovery, which covers the area of Spatial and Temporal Link Discovery, we describe the new methods that we introduce and we prove theoretically their soundness and completeness.

4.1 Generalized Definition of Link Discovery

The definition of classic Link Discovery does not suffice for the area of Spatial and Temporal Link Discovery with respect to the spatial and temporal relations that we introduce. These relations, which are discussed in Sections 3.1.4 and 3.2.1, are in nature Boolean relations i.e., either they hold or they do not. Below we provide a generalized definition of Link Discovery which covers the case of Boolean relations.

Definition Let S and T be two sets of entities, r a relation, R_D and R_B the sets of distance-based and Boolean relations respectively, m_r a metric function for the relation r and θ_{m_r} a threshold for the metric function m_r.

Function m_r and threshold θ_{m_r} are defined as follows:

$$m_r(S \times T) = \begin{cases} [0,1] & \text{if } r \in R_D \\ \{true, false\} & \text{if } r \in R_B \end{cases}$$

$$\theta_{m_r} \in [0,1]$$

The domain of the function m_r is the Cartesian product of S and T. For the range, we distinguish two cases:

- If r is a distance-based relation then m_r returns the distance between a pair from the

Cartesian product normalized to the interval $[0,1]$.

- If r is a Boolean relation then m_r returns $true$ or $false$ depending on whether r holds between a pair from the Cartesian product or not.

The domain of θ_{m_r} is also normalized to the interval $[0,1]$.

Based on the above definitions, the set of discovered links DL can be defined as follows:

$$DL = \begin{array}{l} \{(s,r,t) \mid r \in R_D \ \wedge \ s \in S \ \wedge \ t \in T \ \wedge \ m_r(s,t) < \theta_{m_r}\} \cup \\ \{(s,r,t) \mid r \in R_B \ \wedge \ s \in S \ \wedge \ t \in T \ \wedge \ m_r(s,t)\} \end{array}$$

DL contains triples which have as `subject` an entity from dataset S, as `object` an entity from dataset T and as `predicate` the relation r. A triple belongs to DL if one of the following holds:

- r is a distance-based relation and the metric function m_r applied to the entities of the datasets S and T returns a distance that does not exceed the threshold θ_{m_r}.
- r is a Boolean relation and the metric function m_r applied to the entities of the datasets S and T returns true i.e., the relation r between these entities holds.

4.2 Methods for Spatial and Temporal Link Discovery

In this section we describe the new methods that we introduce for Spatial and Temporal Link Discovery. These methods are based on the background knowledge that we discussed in detail in Sections 3.1 and 3.2.

4.2.1 Spatial Relations

Spatial information lying in datasets is exploited by state of the art Link Discovery methods only for finding equivalences between entities. For example, if we want to interlink a city with its DBpedia entry, we first compute the distance of their coordinates, then the distance of their labels and then, if these distances do not exceed the appropriate thresholds, we consider them as the same.

In this thesis we introduce new methods for Spatial Link Discovery which are based on the spatial relations described in Section 3.1.4. Since we adopt two well-known models for the representation of spatial relations, DE-9IM and RCC, we can interlink datasets that contain points, lines and/or polygonal areas with relations such as: *Intersects*, *Contains*, *Crosses*, etc.

Spatial relations R_s belong to the wider family of Boolean relations R_B ($R_s \subset R_B$) as we have defined them above. A metric function m_r for a spatial relation $r \in R_s$ decides whether r holds between two geometries or not.

4.2.2 Temporal Relations

Similarly to spatial relations, in this thesis we also introduce relations for Temporal Link Discovery. These relations are based on the thirteen distinct and exhaustive qualitative relations of Allen's interval calculus as described in section 3.2.1. Hence, one can discover temporal relations between datasets such as: *During*, *Meets*, *Starts*, etc.

Temporal relations R_t belong to the family of Boolean relations R_B as well ($R_t \subset R_B$). A metric function m_r for a temporal relation $r \in R_t$ decides whether r holds between two time intervals or not.

4.2.3 Spatial and Temporal Distances

For the spatial distance between two geometries we follow the well-adapted approach of the orthodromic distance[1] between the closest points of these geometries. Similarly, for the temporal distance between two time intervals we consider the difference between these intervals expressed in a well-known time unit e.g., seconds.

4.2.4 Spatial and Temporal Transformations

As different datasets usually use different data formats, a common preprocessing technique in classic Link Discovery is to apply transformations to normalize the values prior to comparison. For example, for string values with different case, a `lowercase` transformation can be applied in order to avoid false non-existing links that could occur after the comparison.

Similarly to string transformations, in this thesis we also introduce a set of spatial and temporal transformers that can be applied in the respective attributes of the entities.

[1] http://en.wikipedia.org/wiki/Great-circle_distance

CRS Transformer. As discussed in Section 3.1.1, geometries can be expressed in different Coordinate Reference Systems. This transformer converts the CRS of a geometry to WGS 84.

Serialization Transformer. In Sections 3.1.2 and 3.1.3 we describe the two main serializations of geometries in RDF: WKT and GML. This transformer converts geometries of any serialization to WKT.

Geometry Literal Transformer. As we mention in Sections 3.1.5, 3.1.6 and 3.1.7, geometries can be expressed in different vocabularies. This transformer converts geometry literals from GeoSPARQL, stRDF or W3C GEO vocabulary to pure WKT literal.

Simplification Transformer. Some datasets have very complex geometries, which makes the computation of spatial relations inefficient. This transformer simplifies a geometry according to a given distance tolerance, ensuring that the result is a valid geometry having the same dimension and number of components as the input.

Envelope Transformer. This transformer computes the envelope (minimum bounding rectangle) of a geometry and it is useful in cases that we want to compute approximately spatial relations between two datasets.

Buffer Transformer. This transformer computes the buffer function of a geometry according to a given distance.

Area Transformer. In some cases it is enough to compare just the areas of two geometries to infer whether they are the same or not. This transformer computes the area of a given geometry in square metres.

Points-To-Centroid Transformer. In crowdsourcing datasets like OpenStreetMap[2], multiple users can define the position of the same placemark. As a better approximation of the real position of this placemark we can compute the centroid of these positions. This transformer computes the centroid of a cluster of points.

Time Zone Transformer. This transformer converts the time zone of a given time interval to Coordinated Universal Time (UTC).

[2]http://www.openstreetmap.org

4.3 Blocking Technique

Since the size of the datasets with spatial and temporal information has increased significantly, approaches that perform exhaustive checks between datasets are considered inefficient. Thus, there is a need for scalable, yet sound and complete techniques for decreasing the number of checks by dismissing definitive non-matches prior to the actual check. The most well-known technique to achieve this is known as *Blocking*.

The *Blocking* technique partitions ``close" entities into clusters by reducing the checks only between entities of the same cluster. The partitioning is based on one or more attributes of the entities. For example, suppose that we want to detect duplicates among DBpedia cities. We can partition them by location, by label or by a combination of them. The more attributes participating in the partitioning, the more dimensions a block has. The dimensionality of a block is 2 when blocking by location (locations are defined by their latitude and longitude coordinates), 1 when blocking by label and 3 when blocking by the combination of them. The size of each dimension of a block is selected based on the relevant threshold. In the same example, if we consider as duplicates cities with distance lower than 0.5km, cities having at 90% the same label or cities with both of these characteristics, then each block will have size 0.5×0.5, 0.9 and $0.5 \times 0.5 \times 0.9$ respectively.

This technique has a trade-off between accuracy and time complexity. If we want to guarantee that no false dismissals and thus no loss of recall will occur, we can create accordingly overlapping blocks. If we are more aggressive, we can achieve orders of magnitude decrease in the number of checks and consequently in the time complexity of our algorithm, with the danger of not discovering a set of links.

The *Blocking* technique is more straightforward in distance-based relations (R_D) where we can compute the exact size of each block based on the relevant thresholds. In this thesis we introduce a *Blocking* technique adapted for spatial (R_s) and temporal (R_t) relations.

4.3.1 Blocking technique for Spatial Relations

The coordinate reference system WGS 84 considers the surface of earth as spheroidal with longitude range $[-180°, 180°]$ and latitude range $[-90°, 90°]$. We choose this reference system for our *Blocking* technique is because we prefer having highly accurate calculations rather than fast ones. If we consider the Earth as a plane, we can use Cartesian mathematics to do common calculations such as areas, distances, lengths, intersections,

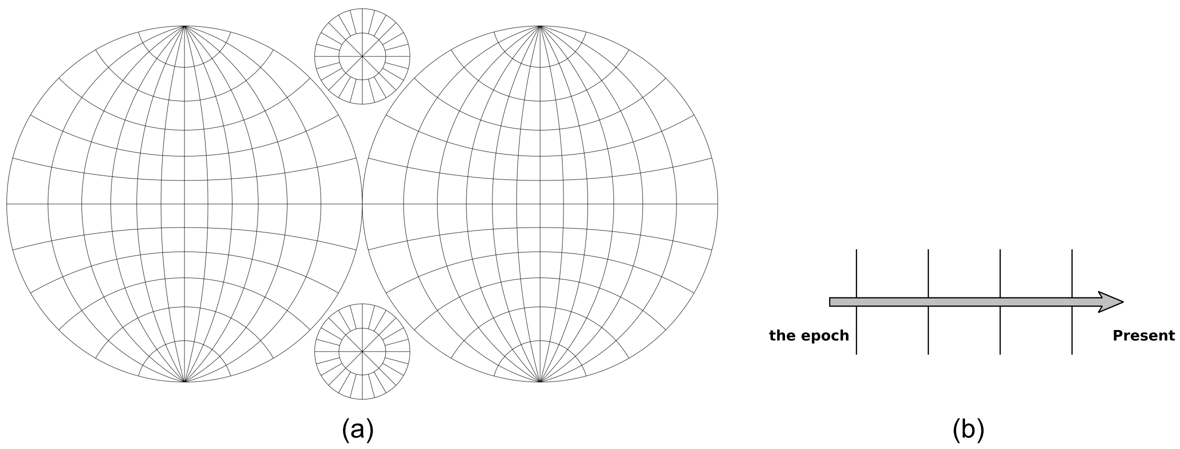


Figure 4.1: Blocking technique for (a) Spatial and (b) Temporal Relations

etc. For example, in this case, the shortest path between two points on the plane is just a straight line. If we consider the Earth as a sphere, the above calculations require more complicated mathematics. In the same example, the shortest path between two points on the sphere is a circle arc. However, in the second case the calculations are more accurate than the first one, because we consider a better approximation of the surface of the earth.

With the proposed *Blocking* technique we build blocks that divide the earth into curved rectangles as depicted in Figure 4.1(a). The area of the blocks is measured in square degrees and can be adjusted with a blocking factor bf. The formula for the computation of the area is the following:

$$blockArea = \frac{1}{bf^2}^{\circ^2}$$

The bigger the value of bf gets, the more and smaller blocks will be created. For example, if we assign to bf the value 10, our *Blocking* technique will create $6,480,000$ non-overlapping blocks with area 0.01°^2} that cover the whole surface of the earth.

After the division of the space into blocks, we compute for each geometry the respective set of blocks that it must be inserted into. In order to achieve this, we first compute the minimum bounding box (*MBB*) that contains each geometry, and then we find, in both dimensions, the blocks that this *MBB* intersects with. Thus, in each block, we insert all the geometries whose *MBB* intersects with it.

The described technique is used for the spatial relations $R_s \setminus \{Disjoint\}$. In *Disjoint* relation we don't divide the space in blocks and thus we exhaustively check all the possible pairs of entities from the datasets.

4.3.2 Blocking technique for Temporal Relations

In the *Blocking* technique for temporal relations we follow a similar approach to the one for spatial relations. Time is one-dimensional and thus the blocks are 1-d as well. Let us suppose that all the time intervals of our data are included in the period from *the epoch*[3] until the *Present* (Figure 4.1(b)). Following the same strategy as before, we divide the time in blocks whose length can be adjusted with bf. The formula for the computation of the length of the blocks is the following:

$$blockLength = \frac{1}{bf} \text{ time units}$$

After the division of the time into blocks, we insert, in each block, all the time periods or instances that temporally intersect with it.

The above technique is used for the temporal relations $R_t \setminus \{Before, After\}$. In *Before* and *After* relations, we exhaustively check all the possible pairs of entities from the datasets.

4.3.3 Blocking technique for Spatial and Temporal Distances

In the *Blocking* technique for spatial and temporal distances we follow the same approach as with the respective relations. The only difference is that in both spatial and temporal distances, we enlarge the geometries and the time intervals respectively, in order to ensure that each instance will be placed in the appropriate set of blocks. More specifically, in both cases we extend each MBB and each time interval by half of the given distance. Hence, two instances with distance less or equal to the given distance will be placed in the same block.

4.3.4 Parallelization

With the described *Blocking* techniques, all the built blocks are completely independent with each other. Thus, the check of a relation or the computation of a distance among the geometries of a block can be performed in parallel with respect to the blocks.

[3]The epoch has been set to January 1, 1970, 00:00:00 GMT.

4.4 Soundness and Completeness

In this section we discuss the soundness and completeness of the proposed methods for Spatial and Temporal Link Discovery. We first prove that the methods are sound and complete when performing an exhaustive check of all the possible pairs of entities (Cartesian product) and then that the *Blocking* technique that we propose does not affect the accuracy of the discovered links.

Cartesian Product Technique In the methods for Spatial and Temporal Link Discovery we are based on relations that are proven sound in [7], [30] and [1]. Also, by definition, Cartesian product denotes that we perform an exhaustive (complete) check of all the pairs of entities from the datasets. Hence, our methods are proved to be sound and complete.

Blocking Technique We prove that the application of the *Blocking* technique does not affect the completeness of our methods for Spatial Link Discovery with reduction to absurdity. As we have mentioned before, the *Blocking* technique cannot be used for the *Disjoint* relation and thus we exclude it from the proof.

Proof. Let two geometries lying in different blocks with a spatial relation $r \in R_s \setminus \{Disjoint\}$ holding between them.

If r holds between two geometries then they intersect at least at one point. (from the definition of the spatial relations)

If two geometries intersect at one point, so do their *MBBs*. (from the definition of *MBB*)

If the *MBBs* of two geometries intersect, then there is at least one block in which they will be both inserted. (as described in Section 4.3)

This results in a contradiction because we assumed that the two geometries are lying in different blocks.

Therefore the initial assumption must be false. $\square$

The above proves that if a relation other than *Disjoint* holds between two geometries, then they will be placed in at least one common block and consequently the relation between them will be discovered.

With a similar proof for temporal relations and spatial and temporal distances we can state that our methods for Spatial and Temporal Link Discovery remain complete, even after the application of the *Blocking* technique. Hence, common metrics for Link Discovery such as precision, recall and F-measure are also proved to be equal to 100%. The latter is a very useful theoretical outcome for the proposed methods because it guarantees absolute accuracy of the discovered links.

4.5 Summary

In this chapter we provided a generalized definition of Link Discovery, which covers the areas of Spatial and Temporal Link Discovery, we describe the new methods that we introduce and we prove theoretically their soundness and completeness. In the following chapter we describe the implementation of the proposed methods in Silk Link Discovery Framework.

Chapter 5
The Link Discovery framework Silk and its Spatial and Temporal Extensions

All the proposed methods for Spatial and Temporal Link Discovery have been implemented as extensions on Silk framework[1]. As mentioned in Chapter 2 Silk is the only, to the best of our knowledge, open-source generic framework for discovering relationships between data items within different Linked Data sources.

5.1 The Link Discovery Engine of Silk

The Link Discovery Engine of Silk builds the core of the Silk Framework. It is responsible for loading the entities from the data sources as well as generating the links based on the user-provided link specifications. The workflow of the Engine can be separated in 5 discrete phases (Figure 5.1). For our implementation we extended only the *Blocking* and *Link Generation* components of the respective phases[2].

5.1.1 Blocking

Silk employs a blocking technique which maps entities to a multidimensional index. After the mapping, the entities are divided into multidimensional and optionally overlapping blocks. Blocking works on arbitrary link specifications and no separate configuration is required. For our implementation we adapted the blocking technique that we introduced in Section 4.3 to the one that Silk uses by utilizing a 2-dimensional index for the spatial relations and a 1-dimensional for the temporal ones.

[1]The source code of the spatial and temporal extensions of Silk is publicly available (`https://github.com/psmeros/stSilk`). We plan to collaborate with the main developers of Silk, in order to include these extensions in the next release of the default version of Silk (`https://github.com/silk-framework/silk`).

[2]A full documentation of Silk can be found here: `https://www.assembla.com/spaces/silk/wiki`.

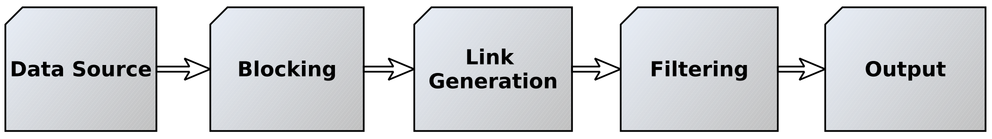


Figure 5.1: Workflow of Silk Engine

5.1.2 Link Generation

In this phase, Silk reads the incoming entities and computes a distance or checks a relation for each pair. As different datasets usually use different data formats, a transformation operator can be applied to normalize the values. Then a comparison or check operator evaluates two inputs and computes a distance or checks a relation between them. For our implementation, we implemented as components for this phase the spatial and temporal check, distance and transformation operators that we introduced in Sections 4.2.1-4.2.4 respectively.

5.2 Silk Variants

Silk is provided in two different variants: *Silk Single Machine* and *Silk MapReduce*.

5.2.1 Silk Single Machine

This variant is used to generate RDF links on a single machine. The datasets that should be interlinked can either reside on the same machine or on remote machines which are accessed via the SPARQL protocol. Silk Single Machine provides multithreading and caching. In addition, the performance can be further enhanced using the optional blocking feature.

5.2.2 Silk MapReduce

This variant is used to generate RDF links between data sets using a cluster of multiple machines. Silk MapReduce is based on Hadoop[3] and it can scale out to very big datasets by distributing the link generation to multiple machines.

[3] `http://hadoop.apache.org/`

5.3 Parallelization

In both of the above Silk variants our *Blocking* technique divides the source datasets into blocks and then the transformation and the check operators run in parallel with respect to the block. In *Silk Single Machine* we have multi-thread parallelization and in *Silk MapReduce* multi-machine parallelization.

5.4 Summary

In this chapter we presented the Link Discovery Framework Silk with its new spatial and temporal extensions that we developed. More details about Silk can be found in the appendix of this thesis.

Chapter 6
Experimental Evaluation

In this chapter we experimentally evaluate the spatial extensions of Silk by using it in a real-world scenario. The respective experiments for the temporal extensions are omitted due to space limitations. The datasets and other useful information for reproducing the experiments are publicly available[1].

As we discussed in Section 2, to the best of our knowledge, there is no related framework with which we can discover spatial and temporal relations other than equivalences among RDF datasets. Hence, in the experiments that we conducted, we compared only against variants of Silk and the state of the art, according to the relevant benchmark [13], spatiotemporal RDF store Strabon [22]. Strabon is not considered as a Link Discovery framework but since it supports the *GeoSPARQL* standard, *NAMED GRAPHS* and *CONSTRUCT* queries it can be used for discovering spatial relations e.g., `intersects`, with a query like the following:

```
CONSTRUCT {?s geo:intersects ?t .}
WHERE{
GRAPH ex:source{?s geo:hasGeometry/geo:asWKT ?sg.}
GRAPH ex:target{?t geo:hasGeometry/geo:asWKT ?tg.}

FILTER(geof:sfIntersects(?sg, ?tg))}
```

The only restriction that we face with Strabon is that both the source and the target datasets must be stored locally, in different named graphs. On the other hand, with Silk, we can interlink a local dataset with a remote one, that is published by another data publisher. The only access that we need to it, is via a SPARQL endpoint.

6.1 Environment of Experiments

We conducted our experiments both in a single machine and a distributed environment. For the single machine environment, we used a machine which is equipped with two Intel

[1]`http://silk.linkeodata.eu/experiments`

Dataset	Instances	Points of Geometries
GAG	325	979,929
CLCG	4,868	8,004,058
HG	37,048	148,192

Table 6.1: Characteristics of the Datasets

Xeon E5620 processors with 12MB L3 cache running at 2.4 GHz, 32 GB of RAM and a RAID-5 disk array that consists of four disks. Each disk has 32 MB of cache and its rotational speed is 7200 rpm. For the distributed environment we used a cluster provided by the European Public Cloud Provider Interoute[2]. In this cluster we reserved 1 Master Node with 2 CPUs, 4GB RAM and 10GB disk and 20 Slave Nodes with 2 CPUs, 4GB RAM and 10GB disk.

We run our experiments using the latest version of Silk with the spatial and temporal enhancements and the latest version of Strabon (v3.2.10) with accordingly tuned PostgreSQL (v9.1.13) and PostGIS (v2.0) as proposed by the developers.

6.2 Scenario

In [20] the authors presented a real-time wildfire monitoring service that exploits satellite images and linked geospatial data to detect and monitor the evolution of fire fronts. This service is now operational at the National Observatory of Athens and is being used during the summer season by emergency managers monitoring wildfires in Greece[3].

A part of the processing chain of the service is to improve the thematic accuracy of the detected fires (hotspots) by correlating them with auxiliary geospatial data. More specifically, the service finds the land cover of the area that a hotspot threatens in order to avoid false alarms from fires detected in big agricultural plains, which are typically started by farmers as part of their agricultural practices and they do not constitute an emergency situation. Also, it finds the municipalities that a hotspot threatens and thus competent authorities such as the Civil Protection Agency and the Fire Brigade are made aware about the existence of a fire in their area of responsibility.

Below, we provide a sort description of the datasets of the scenario, whilst in Table 6.1 we present some quantitative characteristics of them. These datasets also constitute a subset of the datasets used in the state of the art benchmark for Geospatial RDF Stores, Geographica [13].

[2]`http://www.interoute.com/`
[3]`http://bit.ly/FiresInGreece`

Hotspots of Greece (HG) The HG dataset contains the location of detected fires as produced by the National Observatory of Athens after processing appropriate satellite images. In our experiments we used a subset of the dataset that contains all the fires detected during the fire season of 2007.

CORINE Land Cover of Greece (CLCG) The Corine Land Cover project is an activity of the European Environment Agency that provides data regarding the land cover of European countries[4]. The CLCG is a subset of the whole dataset that contains all the available information about Greece.

Greek Administrative Geography (GAG) The GAG dataset contains an ontology that describes the administrative divisions of Greece (prefectures, municipalities, districts, etc.) which has been populated with relevant data that are publicly available in the Greek open government data portal[5].

6.3 Using Silk in the Scenario

With Silk, the above scenario can be translated into two interlinking tasks between the HG and the CLCG and the GAG datasets respectively. In these tasks, Silk will discover `intersects` relations between the geometries of the datasets. Hence, for example, a hotspot that threatens a municipality will be interlinked with it with the property `geo:sfIntersects`.

6.4 Experiment 1: Adjusting the Blocking Factor

In the first experiment we analyze the performance of the single machine implementation of Silk with respect to different blocking factors (*bf*s). As we have discussed in Section 4.3, *bf* adjusts the area of the blocks in which we divide the surface of the earth. The bigger the value of *bf* gets, the more and smaller blocks are created. For this experiment we used the full datasets of the scenario described above and we measured the computation times for performing an interlinking task between HG-GAG and HG-CLCG respectively.

The graph of Figure 6.3 summarizes the results of this experiment. When *bf* takes values close to 0, the blocks are spanning into big surfaces of the earth. Hence, most of

[4]`http://www.eea.europa.eu/publications/COR0-landcover`
[5]`http://geodata.gov.gr`

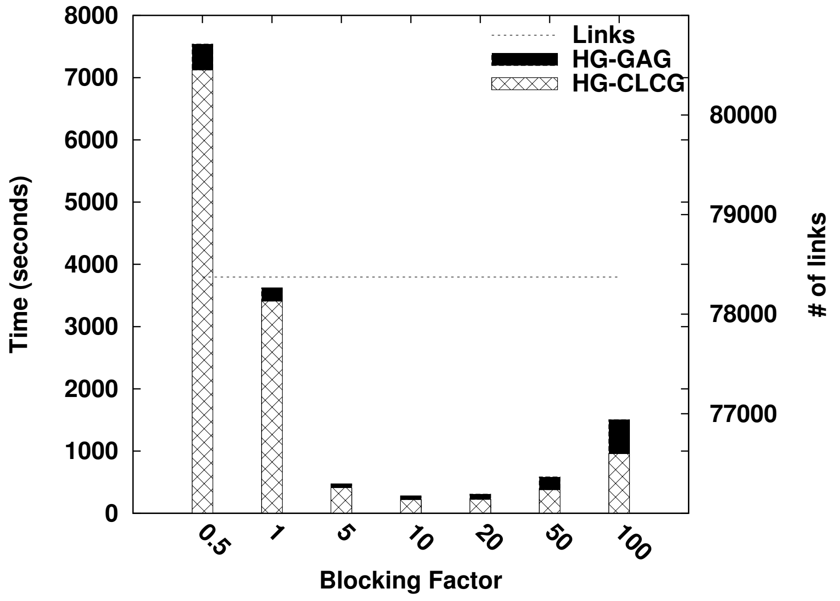


Figure 6.1: Experiment 1

the geometries of the datasets are inserted in the same block, making the total number of checks of a relation slightly smaller than the number of the pairs of the Cartesian product.

As *bf* gets bigger, Silk seems to perform better. However, this improvement continues until a certain value (value 20) and then, as the *bf* increases, the computation time deteriorates. This is due to the fact that a big value for *bf* causes the division of space into very small blocks and thus each geometry is inserted into a big number of them. If two geometries are inserted into multiple blocks, then the check of the spatial relation is performed independently in each block that they appear. Hence, in this case we have redundant checks of the same relation that decrease the performance of Silk.

The optimal value of *bf* depends on the distribution and the size of the geometries which are not known in advance in the case of our scenario. The value that gave the best performance was 10 and thus it will be used in the second experiment.

Another useful outcome from this experiment is the comparison of the time consumed for the link discovery task between HG-GAG and HG-CLCG. Figure 6.3 shows that the HG-CLCG interlinking takes orders of magnitude more than the respective HG-GAG. Notice in Table 6.1 that the CLCG dataset has more geometries than GAG, whereas GAG has more complex ones. Hence, we can claim that in the cases of relations like `intersects`, the bottleneck is the number and not the complexity of the geometries.

One final observation from this experiment is the number of the discovered links. This number remains the same independently from the value of *bf*. This is expected, since we

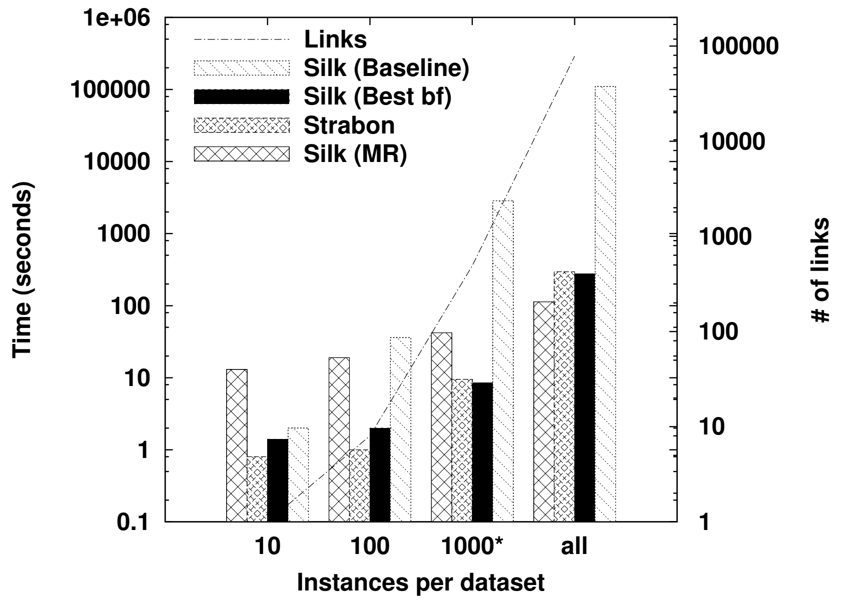


Figure 6.2: Experiment 2

have already proven in Section 4.4 that the *Blocking* technique that we propose does not affect the accuracy of the discovered links.

6.5 Experiment 2: Adjusting the Instances per Dataset

In the second experiment we analyze the performance of three variants of Silk and Strabon with respect to different number of instances per dataset. The datasets that we used in this experiment are subsets of the datasets of the scenario and the measured time is the total computation time for performing an interlinking task between HG-GAG and HG-CLCG.

The first variant (Silk (Baseline)), which we consider as baseline, computes the full Cartesian product of the entities in order to check if a spatial relation between them holds. For example, for the first measurement of the experiment, it makes 10×10 checks of the relation `intersects` between HG-GAG and HG-CLCG respectively. The second variant (Silk (Best bf)), utilizes the *Blocking* technique with the best *bf*, as the latter occurred from the previous experiment. The third (Silk (MR)), is the distributed variant of Silk, which also utilizes the *Blocking* technique with the best *bf*. In the case of Strabon, the datasets are stored locally and a `CONSTRUCT` query like the one we described above is performed.

We have experimented with several subsets of the datasets of the scenario and measured how this affects the performance. For the third measurement we have an exception because GAG dataset contains less than 1000 instances (Table 6.1).

The results of this experiment can be seen in Figure 6.4. As we can see from the graph, Strabon seems to be faster for small number of instances per dataset whereas Silk (Best bf) is faster when interlinking the full datasets. This happens because Silk fully utilizes the cores of the running machine by assigning the workload of each block it creates into a new thread. For big datasets, where the total workload is big enough, the *Blocking* approach of Silk seems to be the most efficient.

This is more remarkable with the distributed variant of Silk (Silk (MR)). With Silk (MR) the total workload is divided into different machines and in each machine it is divided into different cores. We should mention at this point that the computation time of *Hadoop* for small datasets is negligible with respect to the initialization time and the time consumed to copy the data from the local file system to the distributed one and vice versa. Hence, Silk (MR) outperforms the other Silk variants and Strabon only for the measurement with the full datasets. Also, it seems to have the best scaling factor.

6.6 Summary

In this chapter we evaluated experimentally the spatial and temporal extensions of the Link Discovery Framework Silk. In the next chapter we present a potential use of Silk in the context of the project LEO.

Chapter 7
Using Silk in Project LEO

In this chapter we present an example of a potential use of Silk in the context of project LEO. Specifically, we demonstrate the procedure that we follow in order to interlink an open ecological dataset named Natura 2000[1] with a dataset containing precision farming information for field structures. The latter is a product of the talking**fields** project[2], the closest project to LEO. More details on the characteristics of the datasets that we interlink are provided below.

7.1 Datasets

Natura 2000 The Natura 2000 dataset is an ecological network of protected areas in the territory of the European Union. It comprises Special Areas of Conservation (SACs), Special Protection Areas (SPAs), Marine Protected Areas (MPAs) and some special forms that are defined on a national basis. The dataset is subject to a regular validation and updating process. In the context of LEO, we use the subset of Natura 2000 that contains the protected areas of Bavaria. This dataset comprises 5266 entities. An example data item of Natura 2000 can been seen below:

```
natura:id/1
        rdf:type natura:Natura2000Area;
        natura:has_SITENAME "KUSTENGEBIETE"^^xsd:string;
        geo:hasGeometry natura:Geometry/1 .

natura:Geometry/1
        geo:asWKT "MULTIPOLYGON((...))"^^geo:wktLiteral .
```

Talking Fields (TF) The Talking Fields dataset contains products for precision farming developed within the project talking**fields**. This dataset contains information about the fields of a farm, such as the crop type, the variety, the main crop/catch crop, the seeding

[1]http://natura2000.eea.europa.eu
[2]http://www.talkingfields.de

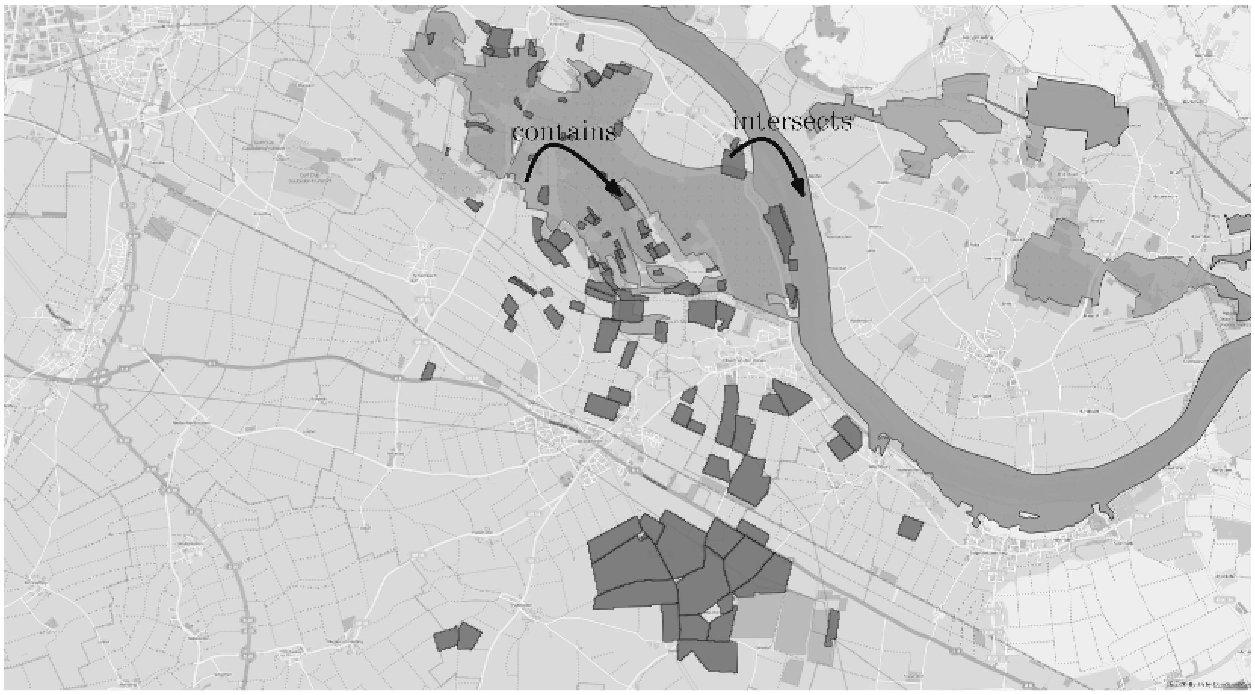


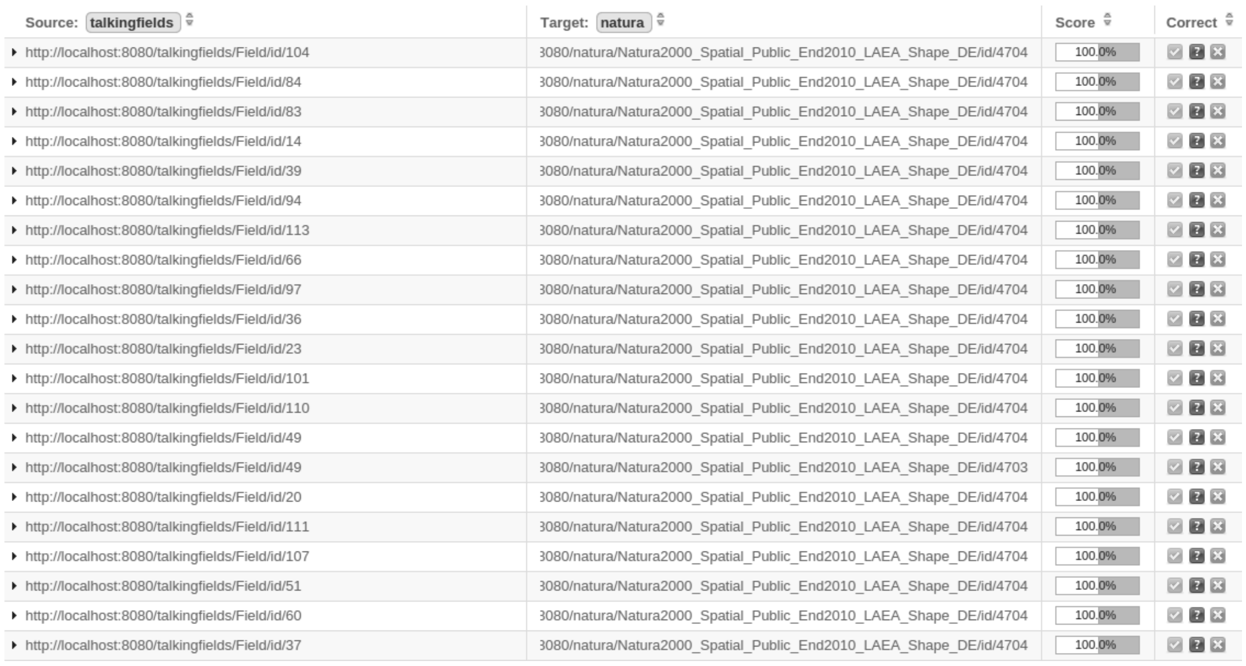
Figure 7.1: Topological relations between Natura 2000 and Talking Fields datasets

date, etc. The sample dataset used comprises 114 entities. Below, we provide an example data item of the Talking Fields dataset:

```
talkingFields:Field/id/1
     rdf:type talkingFields:Field;
     talkingFields:hasFieldName "9 Mitterweg re.b."^^xsd:string;
     talkingFields:hasRasterCell talkingFields:RasterCell/3041 .
     geo:hasGeometry talkingFields:Geometry/1 .

talkingFields:Geometry/1
     geo:asWKT  "MULTIPOLYGON((...))"^^geo:wktLiteral .
```

Given the above two datasets, it would be useful for the precision farming application, developed in LEO, to combine them and discover links, such as the fields that are contained in a Natura area or the fields that intersect with Natura areas, as it is depicted in Figure 7.1.



Source: talkingfields	Target: natura	Score	Correct
http://localhost:8080/talkingfields/Field/id/104	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/84	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/83	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/14	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/39	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/94	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/113	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/66	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/97	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/36	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/23	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/101	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/110	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/49	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/49	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4703	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/20	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/111	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/107	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/51	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/60	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕
http://localhost:8080/talkingfields/Field/id/37	3080/natura/Natura2000_Spatial_Public_End2010_LAEA_Shape_DE/id/4704	100.0%	✓ ? ✕

Figure 7.2: The discovered links as produced by Silk

7.2 Link Discovery Procedure

The procedure that we follow in Silk in order to find such topological relations between two datasets is described in detail in Chapter 5. An overview of the steps of this procedure is the following:

- First we select the data sources and retrieve specific entities from them e.g., entities with type Field.
- Then we transform the geometric attribute of these entities in a common format. Specifically, we transform the geometries in a common vocabulary and a common coordinate reference system and then we keep only the well known text representation.
- Afterwards, we check if the given topological relation (in this case the `Intersects` relation) holds between the geometries.
- Finally, we export the generated links in the given output file.

The link specification that was given as input in Silk in order to execute the described procedure is presented in the Appendix of this thesis.

Figure 7.2 shows the resulting pairs of entities, whose geometries intersect, as they are produced by Silk. As we have discussed in Chapter 4, `intersects` is a boolean relation and thus the score of all the generated links is 100% i.e., the relation between the entities holds.

The overall number of links produced by Silk is 65. It is remarkable that even in a small sample dataset of fields, the 57% of them are intersecting with a protected area. This can be a very useful input for the presicion farming application, developed in LEO, which provides management decision support for plant protection and fertilization activities regarding spatial and legal restrictions by taking into account the surrounding of the fields.

7.3 Summary

In this chapter we presented an example use case from project LEO that demonstrates the usability of Silk. By this, we can combine datasets with geospatial information and therefore increase their value.

Chapter 8
Conclusions and Future Work

In this thesis, we proposed new formalisms and accurate methods for Spatial and Temporal Link Discovery and provided the first implementation that covers this area. This implementation is based on the well-adapted framework Silk. Silk, enhanced with the new features, allows data publishers to discover a wide variety of spatial and temporal relations between their data and other Linked Open Data. We also experimentally evaluated Silk by using it in a real-world scenario with datasets that contain very detailed and complex geometries, and showcase that it can generate 100% accurate links in a time efficient and scalable way.

Future work concentrates on extending Silk with more spatial and temporal relations (e.g., directional relations). These relations will be based on algebras and calculi that appear frequently in the relevant bibliography and are useful for specific use cases.

Appendix

Silk Link Specification Language

The Silk framework provides a declarative language for specifying which types of RDF links should be discovered between data sources as well as which conditions data items must fulfill in order to be interlinked. This section describes the language constructs of the Silk Link Specification Language (Silk-LSL).

The Silk-LSL is expressed in XML as specified by a corresponding Silk XML Schema. The root tag name is `<Silk>`. A valid document may contain the following types of top-level statements beneath the root element:

```
<?xml version="1.0" encoding="utf-8" ?>
<Silk>
    <Prefixes ... />
      ...
    <DataSources ... />
      ...
    [<Blocking />]
      ...
    <Interlinks ... />
      ...
</Silk>
```

The prefix and data source definitions and the link specifications are mandatory statements, while the blocking is optional. Let us now present each of these statements in more detail.

Prefix Definitions

Prefix definitions are top-level statements that allow the binding of a prefix to a namespace:

```
<Prefixes>
  <Prefix id="prefix id" namespace="namespace URI" />
</Prefixes>
```

Data Source Definitions

Data source definitions are top-level statements that allow the specification of access parameters to local and remote SPARQL endpoints or RDF files. The defined data sources may later be referred to and used by their ID within link specification statements.

```
<DataSources>
  <DataSource id="data source ID" type="dataSource type">
    <Param name="parameter name" value="parameter value" />
    ...
  </DataSource>
</DataSources>
```

For **SPARQL endpoint** data sources the following parameters exist:

Parameter	Description
endpointURI	The URI of the SPARQL endpoint.
login	Login required for authentication.
password	Password required for authentication.
instanceList	A list of instances to be retrieved. If not given, all instances will be retrieved. Multiple instances can be separated by a space.
pageSize	Limits each SPARQL query to a fixed amount of results. The SPARQL data source implements a paging mechanism which translates the pagesize parameter into SPARQL LIMIT and OFFSET clauses.
graph	Only retrieve instances from a specific graph.
pauseTime	To allow rate-limiting of queries to public SPARQL severs, the pauseTime statement specifies the number of milliseconds to wait between subsequent queries.
retryCount	To recover from intermittent SPARQL endpoint connection failures, the retryCount parameter specifies the number of times to retry connecting.
retryPause	Specifies how long to wait between retries.
queryParameters	Additional parameters to be appended to every request.
parallel	If multiple queries should be executed in parallel for faster retrieval.

Note that all parameters except the endpoint URI are optional and can be left out.

For **RDF files** data sources the following parameters exist:

Parameter	Description
file	The location of the RDF file.
format	The format of the RDF file. Allowed values: "RDF/XML", "N-TRIPLE", "TURTLE", "TTL", "N3".

Blocking Data Items

`<Blocking />` statement enables the blocking phase as described in Section 5.1. Additional configuration is not required as Silk will automatically generate a blocking function from the link specification. If no `<Blocking />` statement is supplied in a link specification, the comparison will loop over all resource pairs (Cartesian Product).

Link Specifications

Link specification statements state that a link of a given type should be established between two data items if a specified condition is satisfied. This condition may contain different metrics, aggregation and transformation functions, thresholds and weights.

A Silk linking configuration may contain several link specifications if different types of links should be generated. Link specifications are structured as follows:

```
<Interlinks>
  <Interlink id="interlink id">
    <LinkType>link type URI</LinkType>
    <SourceDataset dataSource="dataSource id" var="name">
        [<RestrictTo>SPARQL restriction</RestrictTo>]
    </SourceDataset>
    <TargetDataset dataSource="dataSource id" var="name">
        [<RestrictTo>SPARQL restriction</RestrictTo>]
    </TargetDataset>
    <LinkageRule>
        <Aggregate type="average|max|min|...">
            <Compare metric="metric">
                <Input path="RDF path" />
                <TransformInput function="name">
                  <Input path="RDF path" />
                </TransformInput>
                <Param name="name" value="value" />
            </Compare>
            <Compare ...>
            </Compare>
            ...
        </Aggregate>
    </LinkageRule>

    <Filter limit="limit" />

    <Outputs>
      <Output type="type" minConfidence="lower threshold"
                          maxConfidence="upper threshold">
        <Param name="name" value="value" />
        ...
      </Ouput>
    </Outputs>
  </Interlink>
  <Interlink id="...">
    ...
  </Interlink>
  ...
</Interlinks>
```

Let us now describe each of these statements in more detail.

LinkType

The LinkType directive defines the type of the generated links e.g., owl:sameAs links.

SourceDataset and TargetDataset

The SourceDataset and TargetDataset directives define the set of data items which are to be compared. The entitites which are to be interlinked are selected by providing a restriction for each data source. In its simplest form a restriction just selects all entities of a specific type inside the data source. For instance, in order to interlink cities in DBpedia, a valid restriction may select all entities with the type dbpedia:City. For more complex restrictions, arbitrary SPARQL triple patterns are allowed to be specified.

Linkage Rule

A linkage rule specifies how two data items are compared, in order to decide if a relation between them holds. A linkage rule consists of four basic components:

Path Input. An input retrieves all values which are connected to the entities by a specific path. Every path statement begins with a variable (as defined in the datasets), which may be followed by a series of path elements. If a path cannot be resolved due to a missing property or a too restrictive filter, an empty result set is returned. The following operators can be used to traverse the graph:

Operator	Name	Use	Description
/	forward operator	<path segment>/<property>	Moves forward from a subject resource (set) through a property to its object resource (set).
\	reverse operator	<path segment>\<property>	Moves backward from an object resource (set) through a property to its subject resource (set).
[ ]	filter operator	<path_segment> [<property> <comp_operator> <value>] or <path_segment> [@lang <comp_operator> <value>]	Reduces the currently selected set of resources to the ones matching the filter expression. comp_operator may be one of >, <, >=, <=, =, !=.

Transformation. As different datasets usually use different data formats, a transformation can be used to normalize the values prior to comparison. Some of the transformation functions that are available by default are the following:

Function and parameters	Description
upperCase	Convert a string to upper case.
regexReplace(string regex, string replace)	Replace all occurrences of a regex "regex" with "replace" in a string.
concat	Concatenates strings from two inputs.

The new spatial transformation functions that were added in Silk and are described in detail in Section 4 are the following:

Function and parameters	Description
AreaTransformer	Returns the Area of the input geometry.
BufferTransformer(double distance)	Returns the buffered geometry of the input geometry.
EnvelopeTransformer	Returns the Envelope (Minimum Bounding Rectangle) of the input geometry.
GeometryTransformer	Trasforms a geometry expressed in GeoSPARQL, stSPARQL or W3C Geo vocabulary from any serialization (WKT or GML) and any Coordinate Reference System (CRS) to WKT and WGS 84 (latitude-longitude).
PointsToCentroidCTransformer	Transforms a cluster of points expressed in W3C Geo vocabulary to their centroid expressed in WKT and WGS 84 (latitude-longitude).
SimplifyTransformer(double distanceTolerance, boolean preserveTopology)	Simplifies a geometry according to a given distance tolerance.

Comparison. A comparison operator evaluates two inputs and computes a distance or checks a relation between them based on a user-defined measure and threshold. The parameters of a comparison operator are the following:

Parameter	Description
required (optional)	If required is true, the parent aggregation only yields a confidence value if the given inputs have values for both instances.
weight (optional)	Weight of this comparison. The weight is used by some aggregation functions such as the weighted average aggregation.
threshold	The maximum distance. For boolean measures e.g., spatial relations, this parameter is optional.
distanceMeasure	The used measure.
Inputs	The 2 inputs for the comparison.

Silk supports a wide variety of distance measures. Some of these that are available by default are the following:

Measure and parameters	Description	Type
levenshteinDistance	Levenshtein distance. The minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character.	String Measure
num	Computes the numeric difference between two numbers.	Numeric Measure
date	Computes the distance between two dates ("YYYY-MM-DD" format). Returns the difference in days.	Date Measure

The new spatial measures and relations that were added in Silk and are described in detail in Chapter 4 are the following:

Measure and parameters	Description	Type
CentroidDistanceMetric	Computes the distance between the centroids of two geometries in meters.	Spatial Distance
MinDistanceMetric	Computes the minimum distance between two geometries in meters.	Spatial Distance
SContainsCheck	Checks the relation "contains" between two geometries.	Spatial Relation
CrossesCheck	Checks the relation "crosses" between two geometries.	Spatial Relation
DisjointCheck	Checks the relation "disjoint" between two geometries.	Spatial Relation
SEqualsCheck	Checks the relation "equals" between two geometries.	Spatial Relation
IntersectsCheck	Checks the relation "intersects" between two geometries.	Spatial Relation
SOverlapsCheck	Checks the relation "overlaps" between two geometries.	Spatial Relation
TouchesCheck	Checks the relation "touches" between two geometries.	Spatial Relation
WithinCheck	Checks the relation "within" between two geometries.	Spatial Relation
RelateCheck(string relation)	Checks every relation from DE-9IM between two geometries.	Spatial Relation

The respective temporal measures and relations also described in Chapter 4 are the following:

Measure and parameters	Description	Type
MillisecsDistanceMetric	Computes the distance in milliseconds between two time periods or instants.	Temporal Distance
SecsDistanceMetric	Computes the distance in seconds between two time periods or instants.	Temporal Distance

MinsDistanceMetric	Computes the distance in minutes between two time periods or instants.	Temporal Distance
HoursDistanceMetric	Computes the distance in hours between two time periods or instants.	Temporal Distance
DaysDistanceMetric	Computes the distance in days between two time periods or instants.	Temporal Distance
MonthsDistanceMetric	Computes the distance in months between two time periods or instants.	Temporal Distance
YearsDistanceMetric	Computes the distance in years between two time periods or instants.	Temporal Distance
AfterCheck	Checks the relation "after" between two time periods or instants.	Temporal Relation
BeforeCheck	Checks the relation "before" between two time periods or instants.	Temporal Relation
TContainsCheck	Checks the relation "contains" between two time periods or instants.	Temporal Relation
DuringCheck	Checks the relation "during" between two time periods or instants.	Temporal Relation
FinishesCheck	Checks the relation "finishes" between two time periods or instants.	Temporal Relation
IsFinishedByCheck	Checks the relation "isFinishedBy" between two time periods or instants.	Temporal Relation
MeetsCheck	Checks the relation "meets" between two time periods or instants.	Temporal Relation
IsMetByCheck	Checks the relation "isMetBy" between two time periods or instants.	Temporal Relation
TOverlapsCheck	Checks the relation "overlaps" between two time periods or instants.	Temporal Relation
IsOverlappedByCheck	Checks the relation "isOverlappedBy" between two time periods or instants.	Temporal Relation
StartsCheck	Checks the relation "starts" between two time periods or instants.	Temporal Relation

IsStartedByCheck	Checks the relation "isStartedBy" between two time periods or instants.	Temporal Relation
TEqualsCheck	Checks the relation "equals" between two time periods or instants.	Temporal Relation

Aggregation. An aggregation combines multiple confidence values into a single value. In order to determine if two entities are duplicates it is usually not sufficient to compare a single property. For instance, when comparing geographic entities, we may aggregate the similarities between the names of the entities and the distance between the locations of the entities. The parameters of a aggregation operator are the following:

Parameter	Description
required (optional)	The required attribute can be set if the aggregation only should generate a result if a specific sub-operator returns a value.
weight (optional)	Some comparison operators might be more relevant for the correct establishment of a link between two resources than others. For example, depending on data formats/quality, matching labels might be considered less important than matching coordinates when linking cities. If this modifier is not supplied, a default weight of 1 will be assumed. The weight is only considered in the aggregation functions average, quadraticMean and geometricMean.

The aggregate functions that Silk supports are the following:

Function	Description
AverageAggregator	Evaluate the (weighted) average of confidence values.
MaximumAggregator	Evaluate the highest confidence in the group.
MinimumAggregator	Evaluate the lowest confidence in the group.
QuadraticMeanAggregator	Apply Euclidean distance aggregation.
GeometricMeanAggregator	Compute the (weighted) geometric mean of a group of confidence values.

Link Filter

The Link Filter allows for filtering the generated links. It only has the parameter `limit` that defines the number of links originating from a single data item. Only the n highest-rated links per source data item will remain after the filtering. If no limit is provided, all links will be returned.

Running Silk

In this section we describe how we run the Single Machine and the MapReduce variant of Silk.

Silk Single Machine

Silk Single Machine tool is distributed as a single `jar` file. In order to be able to run it, users need to:

- Have installed Java Runtime Environment 7[1].
- Have SPARQL access to the datasets that should be interlinked.
- Have written a link specification as explained in the Appendix.

Then they have to execute the following command:

```
$ java -DconfigFile=<Silk-LSL configuration file> -jar silk.jar
```

After the execution users can open the output file that has been defined in the `configFile` and review the generated links.

Silk MapReduce

Silk MapReduce tool is distributed as a single jar file as well and it has the same prerequisites as Silk Single Machine tool with the addition of the installation of the Hadoop framework.

The linking workflow in this tool is divided into 2 phases. For the **Load phase**, users have to execute the following command:

```
$ hadoop jar silkmr.jar load configFile ouputDir
```

The `outputDir` is the directory, where the instance cache will be written to. This will be the `inputDir` of the **Link Generation phase**. For the **Link Generation phase**, users have to execute the following command:

[1]`http://java.com/en/`

```
$ hadoop jar silkmr.jar match inputDir ouputDir
```

The `outputDir` in this execution is the directory, where the generated links will be written to.

Link Specification Example

```
<?xml version="1.0" encoding="utf-8" ?>
<Silk>
  <Prefixes>
    <Prefix namespace="http://www.w3.org/1999/02/22-rdf-syntax-ns#" id="rdf"></Prefix>
    <Prefix namespace="http://dbpedia.org/resource/" id="dbpedia"></Prefix>
    <Prefix namespace="http://www.w3.org/2002/07/owl#" id="owl"></Prefix>
    <Prefix namespace="http://schema.org/" id="schema"></Prefix>
    <Prefix namespace="http://www.w3.org/2000/01/rdf-schema#" id="rdfs"></Prefix>
    <Prefix namespace="http://dbpedia.org/ontology/" id="dbpediaowl"></Prefix>
    <Prefix namespace="http://localhost:8080/natura/ontology#" id="natura"></Prefix>
    <Prefix namespace="http://localhost:8080/tf/ontology#" id="talkingfields"></Prefix>
    <Prefix namespace="http://www.opengis.net/ont/geosparql#" id="geo"></Prefix>
  </Prefixes>

  <DataSources>
    <DataSource type="sparqlEndpoint" id="talkingfields">
      <Param name="pageSize" value="1000"></Param>
      <Param name="pauseTime" value="0"></Param>
      <Param name="retryCount" value="3"></Param>
      <Param name="retryPause" value="1000"></Param>
      <Param name="endpointURI" value="http://localhost:8080/tf/Query"></Param>
    </DataSource>
  </DataSources>
  <Blocking />
  <Interlinks>
    <Interlink id="talkingfields-natura">

      <LinkType>geo:sfIntersects</LinkType>

      <SourceDataset dataSource="talkingfields" var="s">
        <RestrictTo> ?s rdf:type talkingfields:Field </RestrictTo>
      </SourceDataset>

      <TargetDataset dataSource="talkingfields" var="t">
        <RestrictTo> ?t rdf:type natura:Natura2000Area </RestrictTo>
      </TargetDataset>

      <LinkageRule>
        <Compare metric="IntersectsCheck" indexing="true">
          <TransformInput function="GeometryTransformer">
            <Input path="?s/geo:hasGeometry/geo:asWKT" />
          </TransformInput>

          <TransformInput function="GeometryTransformer">
            <Input path="?t/geo:hasGeometry/geo:asWKT" />
          </TransformInput>
        </Compare>
      </LinkageRule>
      <Filter />
      <Outputs>
        <Output type="file">
          <Param name="file" value="TalkingfieldsNaturaLinks.nt" />
          <Param name="format" value="ntriples" />
        </Output>
      </Outputs>
    </Interlink>
  </Interlinks>
</Silk>
```

Bibliography

[1] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832--843, November 1983.

[2] Sören Auer, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, and Amrapali Zaveri. Introduction to Linked Data and Its Lifecycle on the Web. In Sebastian Rudolph, Georg Gottlob, Ian Horrocks, and Frank van Harmelen, editors, *Reasoning Web*, volume 8067 of *Lecture Notes in Computer Science*, pages 1--90. Springer, 2013.

[3] Kedar Bellare, Carlo Curino, Ashwin Machanavajihala, Peter Mika, Mandar Rahurkar, and Aamod Sane. Woo: a scalable and multi-tenant platform for continuous knowledge base synthesis. *Proceedings of the VLDB Endowment*, 6(11):1114--1125, 2013.

[4] Konstantina Bereta, Panayiotis Smeros, and Manolis Koubarakis. Representation and querying of valid time of triples in linked geospatial data. In Philipp Cimiano, Oscar Corcho, Valentina Presutti, Laura Hollink, and Sebastian Rudolph, editors, *The Semantic Web: Semantics and Big Data*, volume 7882 of *Lecture Notes in Computer Science*, pages 259--274. Springer Berlin Heidelberg, 2013.

[5] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1--22, 2009.

[6] Hans Chalupsky et al. Unsupervised link discovery in multi-relational data via rarity analysis. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 171--178. IEEE, 2003.

[7] Eliseo Clementini, Paolino Di Felice, and Peter van Oosterom. A small set of formal topological relationships suitable for end-user interaction. In David Abel and Beng Chin Ooi, editors, *Advances in Spatial Databases*, volume 692 of *Lecture Notes in Computer Science*, pages 277--295. Springer Berlin Heidelberg, 1993.

[8] Chris J. Date, Hugh Darwen, and Nikos A. Lorentzos. *Temporal data and the relational model*. Elsevier, 2002.

[9] Moisés G de Carvalho, Alberto HF Laender, Marcos André Gonçalves, and Altigran Soares da Silva. A genetic programming approach to record deduplication. *Knowledge and Data Engineering, IEEE Transactions on*, 24(3):399--412, 2012.

[10] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. Problog: A probabilistic prolog and its application in link discovery. In *IJCAI*, volume 7, pages 2462--2467, 2007.

[11] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection: A survey. *IEEE Trans. on Knowl. and Data Eng.*, 19(1):1--16, January 2007.

[12] Zeno Gantner, Matthias Westphal, and Stefan Woelfl. GQR - A Fast Reasoner for Binary Qualitative Constraint Calculi. In *AAAI Workshop on Spatial and Temporal Reasoning*, 2008.

[13] George Garbis, Kostis Kyzirakos, and Manolis Koubarakis. Geographica: A benchmark for geospatial rdf stores (long version). In *The Semantic Web--ISWC 2013*, pages 343--359. Springer, 2013.

[14] Alfonso Gerevini and Bernhard Nebel. Qualitative Spatio-Temporal Reasoning with RCC-8 and Allen's Interval Calculus: Computational Complexity. In Frank van Harmelen, editor, *ECAI*, pages 312--316. IOS Press, 2002.

[15] Stella Giannakopoulou, Charalampos Nikolaou, and Manolis Koubarakis. A reasoner for the RCC-5 and RCC-8 calculi extended with constants. In Carla E. Brodley and Peter Stone, editors, *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 2659--2665. AAAI Press, 2014.

[16] Oktie Hassanzadeh, Anastasios Kementsietsidis, Lipyeow Lim, Renée J Miller, and Min Wang. A framework for semantic link discovery over relational data. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1027--1036. ACM, 2009.

[17] John R. Herring, editor. *OpenGIS Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture*. Open Geospatial Consortium Inc., 2011.

[18] Robert Isele and Christian Bizer. Active learning of expressive linkage rules using genetic programming. *Web Semantics: Science, Services and Agents on the World Wide Web*, 23:2--15, 2013.

[19] Manolis Koubarakis. Linked Open Earth Observation Data: The LEO Project, 2014.

[20] Manolis Koubarakis, Charalambos Kontoes, and Stefan Manegold. Real-time wildfire monitoring using scientific database and linked data technologies. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 649--660. ACM, 2013.

[21] Manolis Koubarakis and Kostis Kyzirakos. Modeling and querying metadata in the semantic sensor web: The model stRDF and the query language stSPARQL. In *ESWC*, 2010.

[22] Kostis Kyzirakos, Manos Karpathiotakis, and Manolis Koubarakis. Strabon: a semantic geospatial dbms. In *The Semantic Web--ISWC 2012*, pages 295--311. Springer, 2012.

[23] Paul Longley. *Geographic information systems and science*. John Wiley & Sons, 2005.

[24] Raymond J Mooney, Prem Melville, Lappoon R Tang, Jude Shavlik, Ines de Castro Dutro, David Page, and Vitor S Costa. Relational data mining with inductive logic programming for link discovery. Technical report, DTIC Document, 2002.

[25] Bernhard Nebel. Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ORD-Horn class. *Constraints*, 1(3):175--190, 1997.

[26] Axel-Cyrille Ngonga Ngomo and Sören Auer. Limes: A time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 2312--2317. AAAI Press, 2011.

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

[28] OGC. GeoSPARQL - A geographic query language for RDF data, November 2010.

[29] Clemens Portele. OpenGIS Geography Markup Language (GML) Encoding Standard (OGC 07-036). OpenGIS Standard, August 2007.

[30] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In *KR*, pages 165--176, 1992.

[31] Jochen Renz. Qualitative Spatial and Temporal Reasoning: Efficient Algorithms for Everyone. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 526--531, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[32] Jochen Renz and Bernhard Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the Region Connection Calculus. *Artificial Intelligence*, 108(1-2):69 -- 123, 1999.

[33] J Salas and Andreas Harth. Finding spatial equivalences accross multiple RDF datasets. In *Proceedings of the Terra Cognita Workshop on Foundations, Technologies and Applications of the Geospatial Web*, pages 114--126. Citeseer, 2011.

[34] François Scharffe, Yanbin Liu, and Chunguang Zhou. Rdf-ai: an architecture for rdf datasets matching, fusion and interlink. In *Proc. IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US)*, 2009.

[35] Vivek Sehgal, Lise Getoor, and Peter D Viechnicki. Entity resolution in geospatial data integration. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, pages 83--90. ACM, 2006.

[36] Richard T. Snodgrass, editor. *The TSQL2 Temporal Query Language*. Kluwer, 1995.

[37] Peter van Beek. Reasoning about qualitative temporal information. *Artificial Intelligence*, 58(1-3):297 -- 326, 1992.

[38] Luis M Vilches-Blázquez, Víctor Saquicela, and Oscar Corcho. Interlinking geospatial information in the web of data. In *Bridging the Geographic Information Sciences*, pages 119--139. Springer, 2012.