AUTOMATING GEOSPATIAL RDF DATASET INTEGRATION AND ENRICHMENT

Der Fakultät für Mathematik und Informatik
der Universität Leipzig eingereichte

DISSERTATION

zur Erlangung des akademischen Grades

Doctor rerum naturalium
(Dr. rer. nat.)

im Fachgebiet Informatik vorgelegt von

M.Sc Mohamed Ahmed Mohamed Sherif

geboren am 05.12.1980 in Gharbya, Ägypten

Leipzig, den 09. Dezember 2016

Die Annahme der Dissertation wurde empfohlen von:

1. *Professor Dr. Klaus-Peter Fähnrich* (Leipzig)
2. *Professor Dr. Daniel P. Mirankar* (Austin, USA)

Die Verleihung des akademischen Grades erfolgte mit Bestehen
der Verteidigung am 05.12.2016 mit dem Gesamtprädikat *magna cum laude*

BIBLIOGRAPHIC DATA

TITLE:
Automating Geospatial RDF Dataset Integration and Enrichment

AUTHOR:
Mohamed Ahmed Mohamed Sherif

STATISTICAL INFORMATION:
13 chapters, 165 pages, 32 figures, 19 tables, 19 listings, 8 algorithms, 155 literature references

SUPERVISORS:
Prof. Dr. habil. Klaus-Peter Fähnrich
Prof. Dr. Jens Lehmann
Dr. Axel-Cyrille Ngonga Ngomo
Prof. Dr. Sören Auer

INSTITUTION:
Universität Leipzig, Fakultät für Mathematik und Informatik

TIME FRAME:
October 2012 - March 2016

ABSTRACT

Over the last years, the Linked Open Data (LOD) has evolved from a mere 12 to more than 10,000 knowledge bases. These knowledge bases come from diverse domains including (but not limited to) publications, life sciences, social networking, government, media, linguistics. Moreover, the LOD cloud also contains a large number of cross-domain knowledge bases such as *DBpedia* and *Yago2*. These knowledge bases are commonly managed in a decentralized fashion and contain partly overlapping information. This architectural choice has led to knowledge pertaining to the same domain being published by independent entities in the LOD cloud. For example, information on drugs can be found in *Diseasome* as well as *DBpedia* and *Drugbank*. Furthermore, certain knowledge bases such as *DBLP* have been published by several bodies, which in turn has lead to duplicated content in the LOD. In addition, large amounts of geo-spatial information have been made available with the growth of heterogeneous Web of Data.

The concurrent publication of knowledge bases containing related information promises to become a phenomenon of increasing importance with the growth of the number of independent data providers. Enabling the joint use of the knowledge bases published by these providers for tasks such as federated queries, cross-ontology question answering and data integration is most commonly tackled by creating links between the resources described within these knowledge bases. Within this thesis, we spur the transition from isolated knowledge bases to enriched Linked Data sets where information can be easily integrated and processed. To achieve this goal, we provide concepts, approaches and use cases that facilitate the integration and enrichment of information with other data types that are already present on the Linked Data Web with a focus on geo-spatial data.

The first challenge that motivates our work is the lack of *measures* that use the geographic data for linking geo-spatial knowledge bases. This is partly due to the geo-spatial resources being described by the means of vector geometry. In particular, discrepancies in granularity and error measurements across knowledge bases render the selection of appropriate distance measures for geo-spatial resources difficult. We address this challenge by evaluating existing literature for point-set measures that can be used to measure the similarity of vector geometries. Then, we present and evaluate the ten measures that we derived from the literature on samples of three real knowledge bases.

The second challenge we address in this thesis is the lack of automatic Link Discovery (LD) approaches capable of dealing with geo-spatial knowledge bases with *missing and erroneous data*. To this end,

we present Colibri, an unsupervised approach that allows discovering links between knowledge bases while improving the quality of the instance data in these knowledge bases. A Colibri iteration begins by generating links between knowledge bases. Then, the approach makes use of these links to detect resources with probably erroneous or missing information. This erroneous or missing information detected by the approach is finally corrected or added.

The third challenge we address is the lack of *scalable LD approaches* for tackling big geo-spatial knowledge bases. Thus, we present Deterministic Particle-Swarm Optimization (DPSO), a novel load balancing technique for LD on parallel hardware based on particle-swarm optimization. We combine this approach with the Orchid algorithm for geo-spatial linking and evaluate it on real and artificial data sets.

The lack of approaches for *automatic updating* of links of an evolving knowledge base is our fourth challenge. This challenge is addressed in this thesis by the Wombat algorithm. Wombat is a novel approach for the discovery of links between knowledge bases that relies exclusively on positive examples. Wombat is based on generalisation via an upward refinement operator to traverse the space of Link Specifications (LS). We study the theoretical characteristics of Wombat and evaluate it on different benchmark data sets.

The last challenge addressed herein is the lack of automatic approaches for geo-spatial knowledge base *enrichment*. Thus, we propose Deer, a supervised learning approach based on a refinement operator for enriching Resource Description Framework (RDF) data sets. We show how we can use exemplary descriptions of enriched resources to generate accurate enrichment pipelines. We evaluate our approach against manually defined enrichment pipelines and show that our approach can learn accurate pipelines even when provided with a small number of training examples.

Each of the proposed approaches is implemented and evaluated against state-of-the-art approaches on real and/or artificial data sets. Moreover, all approaches are peer-reviewed and published in a conference or a journal paper. Throughout this thesis, we detail the ideas, implementation and the evaluation of each of the approaches. Moreover, we discuss each approach and present lessons learned. Finally, we conclude this thesis by presenting a set of possible future extensions and use cases for each of the proposed approaches.

PUBLICATIONS

This thesis is based on the following publications and proceedings. References to the appropriate publications are included at the respective chapters and sections.

JOURNALS, PEER-REVIEWED

1. Sherif, M. A. and Ngonga Ngomo, A.-C. (2015b). Semantic quran: A multilingual resource for natural-language processing. *Semantic Web Journal*, 6:339–345

2. Zaveri, A., Lehmann, J., Auer, S., Hassan, M. M., Sherif, M. A., and Martin, M. (2013b). Publishing and interlinking the global health observatory dataset. *Semantic Web Journal*, Special Call for Linked Dataset descriptions(3):315–322

JOURNALS, SUBMITTED

3. Sherif, M. A. and Ngonga Ngomo, A.-C. (2015c). A systematic survey of point set distance measures for link discovery. *Semantic Web Journal*

CONFERENCES, PEER-REVIEWED

4. Sherif, M. A. and Ngonga Ngomo, A.-C. (2015a). An optimization approach for load balancing in parallel link discovery. In *SEMANTiCS 2015*

5. Sherif, M., Ngonga Ngomo, A.-C., and Lehmann, J. (2015). Automating RDF dataset transformation and enrichment. In *12th Extended Semantic Web Conference, Portoroz, Slovenia, 31st May - 4th June 2015*. Springer

6. Ngonga Ngomo, A.-C., Sherif, M. A., and Lyko, K. (2014). Unsupervised link discovery through knowledge base repair. In *Extended Semantic Web Conference (ESWC 2014)*

7. Sherif, M. A., Coelho, S., Usbeck, R., Hellmann, S., Lehmann, J., Brümmer, M., and Both, A. (2014). NIF4OGGD - NLP interchange format for open german governmental data. In *The 9th edition of the Language Resources and Evaluation Conference, 26-31 May, Reykjavik, Iceland*

8. Pokharel, S., Sherif, M. A., and Lehmann, J. (2014). Ontology based data access and integration for improving the effectiveness of farming in Nepal. In *Proc. of the International Conference on Web Intelligence*

9. Grange, J. J. L., Lehmann, J., Athanasiou, S., Rojas, A. G., Giannopoulos, G., Hladky, D., Isele, R., Ngonga Ngomo, A.-C., Sherif, M. A., Stadler, C., and Wauer, M. (2014). The geoknow generator: Managing geospatial data in the linked data web. In *Proceedings of the Linking Geospatial Data Workshop*

BOOK

10. Lehmann, J., Athanasiou, S., Both, A., Buehmann, L., Garcia-Rojas, A., Giannopoulos, G., Hladky, D., Hoeffner, K., Grange, J. J. L., Ngonga Ngomo, A., Pietzsch, R., Isele, R., Sherif, M. A., Stadler, C., Wauer, M., and Westphal, P. (2015). The geoknow handbook. Technical report

OTHER PUBLICATION

We present here a set of publications done during this PhD study period, but are not part of the presented thesis.

11. Stadler, C., Unbehauen, J., Westphal, P., Sherif, M. A., and Lehmann, J. (2015). Simplified RDB2RDF mapping. In *Proceedings of the 8th Workshop on Linked Data on the Web (LDOW2015), Florence, Italy*

12. Zaveri, A., Kontokostas, D., Sherif, M. A., Bühmann, L., Morsey, M., Auer, S., and Lehmann, J. (2013a). User-driven quality evaluation of DBpedia. In *To appear in Proceedings of 9th International Conference on Semantic Systems, I-SEMANTICS '13, Graz, Austria, September 4-6, 2013*, pages 97–104. ACM

For my beloved parents,
Ahmed & Nagat
who were the first teachers in my life...

And for my wife,
Ola
who is the best gift of my life...

And for my kids
Malak, Ahmed & Mariam,
who are my life...

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisors *Prof. Klaus-Peter Fähnrich* and *Prof. Sören Auer* for giving me the opportunity to pursue my PhD in the university of Leipzig.

I have been extremely fortunate to work with *Dr. Axel-C. Ngonga Ngomo* and *Prof. Jens Lehmann*, who were not only my mentors but also good friends. They encouraged me to explore on my own, and at the same time provide me with guidance in time of need. I would like deeply to appreciate all their valuable ideas and continuous support, which helped me improve my skills as a researcher and as a person as well. I am looking forward to the day that I would be as good mentor to my students as *Axel* and *Jens* have been to me.

Special thanks go to each of my colleagues at the AKSW research group, not only for their help and constructive comments, but also for their companionship that made AKSW my second home. I will not mention personal names here in order not to forget any one, although I admit that I learned much from each one of my colleagues.

I would like to thank the both the *"Ministry of Higher Education of the Arab Republic of Egypt"* (MoHE) and the *"Deutscher Akademischer Austauschdienst"* (DAAD) for awarding me the scholarship to fulfil my PhD in Germany.

I would like to express my gratefulness to all my former teachers and supervisors in Egypt who build my fundamental knowledge of science. Moreover, I would like to give special thanks to all my colleagues and advisors in the faculty of *Computer and Informatics, Suez Canal University, Egypt*.

I also would like to express my deepest gratitude to my parents *Ahmed Sherif* and *Nagat Bayomy* for their unconditional love and support. Also, I would like to thank my sister *Mona* and my brother *Mamdoh* for their love and emotional support.

And finally, I would like to thank my beloved wife *Ola*. *Ola*, a mother of three children with increasing responsibilities, specially the difficulties of living abroad, was able to provide me with her numerous support and encouragement, which paved the way throughout each stage of my PhD.

CONTENTS

I PRELIMINARIES 1

1 INTRODUCTION 1
- 1.1 Motivation . . . 1
- 1.2 Research Questions and Contributions . . . 4
- 1.3 Overview of the Thesis . . . 6

2 NOTATION 8
- 2.1 Link Discovery . . . 8
  - 2.1.1 Problem Definition . . . 8
  - 2.1.2 ORCHID . . . 9
- 2.2 Refinement Operators . . . 10

3 RELATED WORK 11
- 3.1 Point Set Distance Measures . . . 11
- 3.2 Supervised vs. Unsupervised LD . . . 12
- 3.3 Link Discovery for more than Two Datasets . . . 13
- 3.4 Load Balancing Approaches for Link Discovery . . . 14
- 3.5 Positive Only Machine Learning . . . 14
- 3.6 RDF Dataset Transformation and Enrichment . . . 16

II APPROACHES 17

4 A SYSTEMATIC EVALUATION OF POINT SET DISTANCE MEASURES FOR LINK DISCOVERY 18
- 4.1 Notation . . . 19
- 4.2 Systematic Survey Methodology . . . 20
  - 4.2.1 Research Question Formulation . . . 20
  - 4.2.2 Eligibility Criteria . . . 21
  - 4.2.3 Search Strategy . . . 21
  - 4.2.4 Search Methodology Phases . . . 22
- 4.3 Distance Measures for Point Sets . . . 22
  - 4.3.1 Mean Distance Function . . . 23
  - 4.3.2 Max Distance Function . . . 23
  - 4.3.3 Min Distance Function . . . 24
  - 4.3.4 Average Distance Function . . . 24
  - 4.3.5 Sum of Minimums Distance Function . . . 24
  - 4.3.6 Surjection Distance Function . . . 24
  - 4.3.7 Fair Surjection Distance Function . . . 25
  - 4.3.8 Link Distance Function . . . 25
  - 4.3.9 Hausdorff Distance Function . . . 26
  - 4.3.10 Fréchet Distance Function . . . 26
- 4.4 Evaluation . . . 27
  - 4.4.1 Experimental Setup . . . 27
  - 4.4.2 Point-to-Point Geographic Distance . . . 29

4.4.3 Scalability Evaluation . . . 29
4.4.4 Robustness Evaluation . . . 30
4.4.5 Scalability with ORCHID . . . 33
4.4.6 Experiment on Real Datasets . . . 34
5 COLIBRI– UNSUPERVISED LINK DISCOVERY THROUGH KNOWLEDGE BASE REPAIR 37
5.1 Notation . . . 38
5.2 The COLIBRI Approach . . . 41
5.2.1 Overview . . . 41
5.2.2 EUCLID . . . 42
5.2.3 Voting . . . 43
5.2.4 Instance Repair . . . 45
5.3 Evaluation . . . 46
5.3.1 Experimental Setup . . . 46
5.3.2 Experimental Results . . . 47
6 DPSO – AN OPTIMIZATION APPROACH FOR LOAD BALANCING IN PARALLEL LINK DISCOVERY 51
6.1 Notation . . . 52
6.2 Load Balancing Algorithms . . . 52
6.2.1 Naïve Load Balancer . . . 53
6.2.2 Greedy Load Balancer . . . 53
6.2.3 Pair-Based Load Balancer . . . 54
6.2.4 Particle Swarm Optimization . . . 55
6.2.5 Deterministic Particle Swarm Optimization Load Balancer . . . 56
6.3 Evaluation . . . 58
6.3.1 Experimental Setup . . . 59
6.3.2 Orchid vs. Parallel Orchid . . . 60
6.3.3 Parallel Load balancing Algorithms Evaluation 60
7 WOMBAT – A GENERALIZATION APPROACH FOR AUTOMATIC LINK DISCOVERY 63
7.1 Notation . . . 64
7.2 Constructing and Traversing Link Specifications . . . 65
7.2.1 Learning Atomic Specifications . . . 65
7.2.2 Combining Atomic Specifications . . . 66
7.3 WOMBAT Algorithm . . . 70
7.4 Evaluation . . . 72
8 DEER – AUTOMATING RDF DATASET TRANSFORMATION AND ENRICHMENT 79
8.1 Notation . . . 80
8.2 Knowledge Base Enrichment Refinement Operator . . . 81
8.3 Learning Algorithm . . . 83
8.3.1 Approach . . . 83
8.3.2 Most Promising Node Selection . . . 84
8.3.3 Termination Criteria . . . 85
8.4 Self-Configuration . . . 85

8.4.1 Dereferencing Enrichment Functions 86
8.4.2 Linking Enrichment Function 87
8.4.3 NLP Enrichment Function 87
8.4.4 Conformation Enrichment Functions 87
8.4.5 Filter Enrichment Function 89
8.5 Evaluation 89
8.5.1 Experimental Setup 89
8.5.2 Results 91

III APPLICATION SCENARIOS AND CONCLUSION 95

9 GHO – PUBLISHING AND INTERLINKING THE GLOBAL HEALTH OBSERVATORY DATASET 96
9.1 Dataset Conversion 98
9.2 Dataset Publishing and Linking 98
9.3 Use-Cases 101
9.3.1 Monitoring Health Care Scenarios 101
9.3.2 Disparity Analysis 103
9.3.3 Primary Source Providing Ground Truth 103
9.3.4 Human Development Data Warehouse 104
9.4 Related Initiatives 105
9.5 Summary and Outlook 106

10 SEMANTIC QURAN – A MULTILINGUAL RESOURCE FOR NATURAL-LANGUAGE PROCESSING 109
10.1 Data Sources 109
10.1.1 Tanzil Project 110
10.1.2 The Quranic Arabic Corpus Project 110
10.2 Ontology 111
10.3 Extraction Process 112
10.4 Linking 114
10.5 Use-Cases 115
10.5.1 Data Retrieval 115
10.5.2 Arabic Linguistics 115
10.5.3 Interoperability using NIF 116
10.5.4 Information Aggregation 117
10.6 Summary and Outlook 117

11 AGRINEPALDATA – ONTOLOGY BASED DATA ACCESS AND INTEGRATION FOR IMPROVING THE EFFECTIVENESS OF FARMING IN NEPAL 119
11.1 Methodology 120
11.2 Dataset Description 122
11.2.1 Data Sources 123
11.2.2 Extraction Process 124
11.3 Ontology 126
11.4 Linking 127
11.5 Quality Measurement 128
11.5.1 Link Verification 128

11.5.2 Dataset Verification . . . 129
11.6 Use-Cases . . . 130
11.6.1 Irrigation In Field . . . 130
11.6.2 Agriculture Planner, Policy Maker . . . 131
11.6.3 Agriculture Spatial Data Visualization . . . 132
11.7 Summary and Outlook . . . 132
12 NIF4OGGD 134
12.1 Open German Governmental Data . . . 134
12.2 Dataset . . . 136
12.2.1 LinkedGeoData . . . 136
12.2.2 Data Extraction . . . 136
12.3 Architecture . . . 137
12.3.1 Conversion of Documents to NIF . . . 137
12.3.2 Enrichment . . . 138
12.3.3 Visualization & Search . . . 138
12.4 Use-Cases . . . 139
12.4.1 Data Retrieval. . . . 140
12.4.2 Interoperability using NIF . . . 140
12.4.3 Information Aggregation . . . 140
12.5 Summary and Outlook . . . 140
13 CONCLUSION AND FUTURE WORK 142
13.1 Point Set Distance Measures for geospatial LD . . . 142
13.2 Unsupervised LD Through Knowledge Base Repair . . . 142
13.3 Load Balancing for LD . . . 143
13.4 A Generalization Approach for Automatic LD . . . 144
13.5 Automating RDF Dataset Enrichment and Transformation . . . 144

IV APPENDIX 145
A CURRICULUM VITAE 146

BIBLIOGRAPHY 150

LIST OF FIGURES

Figure 1 Example of ORCHID space tiling. 10
Figure 2 Vector description of the country of *Malta*. . . 19
Figure 3 Fréchet vs other distance approaches 26
Figure 4 Scalability evaluation on the NUTS dataset. . . 30
Figure 5 Comparison of different point set distance measures against granularity discrepancies. 31
Figure 6 Comparison of point set measures against measurement discrepancies. 32
Figure 7 Comparison of point set measures against granularity and measurement discrepancies. 33
Figure 8 Scalability evaluation with ORCHID. 34
Figure 9 Example of four linked resources from four different knowledge bases. 40
Figure 10 Mappings between 3 sets of resources. 41
Figure 11 Results of the *Restaurants* data set. 49
Figure 12 Runtime and MSE of ORCHID vs. parallel implementations. 61
Figure 13 Runtime and MSE of parallel implementations. 62
Figure 14 Example of a complex LS. 65
Figure 15 Definition of the refinement operator ψ. 67
Figure 16 Precision, Recall and F-score of WOMBAT on benchmark data sets. 75
Figure 17 Runtime of WOMBAT on benchmark data sets. . 76
Figure 18 Best LS learned by WOMBAT for *DBLP-GS*. . . . 78
Figure 19 RDF graph of the running example. 81
Figure 20 Ibuprofen CBD before and after enrichment. . . 81
Figure 21 Ibuprofen CBD after final enrichment. 88
Figure 22 Graph representation of the learned pipeline. . 89
Figure 23 Screenshot of the *OntoWiki*. 100
Figure 24 Screenshot of *CubeViz*. 102
Figure 25 Class diagram of Semantic Quran ontology . . 112
Figure 26 *AgriNepalData* data management framework. . 121
Figure 27 Linked Data Lifecycle. 123
Figure 28 *AgriNepalData* ontology structure. 128
Figure 29 Visualization of the *Lumbini* rainfall station. . . 133
Figure 30 Architecture of the NIF4OGGD system. 138
Figure 31 *Lucene* index. 138
Figure 32 Searching for governmental documents. 139

LIST OF TABLES

Table 1 Retrieved articles in search methodology phases. 23
Table 2 Comparison of the orthodromic and great elliptic distances. 36
Table 3 Average F-measure of EUCLID and COLIBRI. . . 48
Table 4 Link Specification Syntax and Semantics 65
Table 5 10-fold cross validation F-Measure results. . . 73
Table 6 A comparison of WOMBAT F-Measure against 4 state-of-the-art approaches. 74
Table 7 The *pruning factor* of the benchmark data sets. 77
Table 8 WOMBAT comparison against [Kejriwal and Miranker, 2015]. 77
Table 9 Test of the effect of ω on the learning. 92
Table 10 Test of the effect of increasing number of positive examples in the learning process. 93
Table 11 Results of the 7 manual generated pipelines. . 94
Table 12 tatistical data sets available in the GHO. 97
Table 13 Technical details of the GHO. 99
Table 14 GHO links and precision. 101
Table 15 Technical details of the Quran RDF data set. . . 113
Table 16 *AgriNepalData* triples details. 126
Table 17 Technical details of the *AgriNepalData*. 126
Table 18 Links and its precision in *AgriNepalData*. . . . 128
Table 19 Data portals classification and features. 135

LISTINGS

Listing 1 RDF representation of the death value '127' using the RDF Data Cube Vocabulary 99
Listing 2 SPARQL for retrieving the number of deaths due to Measles in all countries. 102
Listing 3 SPARQL for retrieving the measles immunization coverage among 1-year-olds. 103
Listing 4 SPARQL for retrieving the number of deaths and number of trials for Tuberculosis and HIV/AIDS in all countries. 104
Listing 5 SPARQL for retrieving the public health expenditure. 105
Listing 6 Fragment of the link specification to the English Wiktionary. 114
Listing 7 Verses that contains moses in (i) Arabic (ii) English and (iii) German. 115
Listing 8 List all the Arabic prepositions with example statement for each. 116
Listing 9 List of different part of speech variations of one Arabic root of the word read *"ktb"*. 116
Listing 10 List of all occurrences of "Moses" using NIF . . 116
Listing 11 List of all senses of all English words of the first verse of the first chapter "qrn:quran1-1". . 117
Listing 12 RDF Conversion for Paddy produced in year 2011/12 in Taplejung district 125
Listing 13 Example of spatial and non-spatial RDF conversion of information for Gorkha district from an ESRI shapefile. 125
Listing 14 Fragment of the LS for linking districts of Nepal between AgriNepalData and DBpedia. 129
Listing 15 How much irrigation water is required for a wheat plant which was planted in November 1 through out the life time of plant (120 days)? . 131
Listing 16 Which districts are self dependent in their agri-products? 132
Listing 17 Select all streets of Berlin along with latitude and longitude. 137
Listing 18 Example NIF resources 139
Listing 19 List of all occurrences of *Baubeschluss* 140

LIST OF ALGORITHMS

Algorithm 1 The Colibri Approach. 42

Algorithm 2 Naïve Load Balancer 53
Algorithm 3 Greedy Load Balancer 54
Algorithm 4 Pair Based Load Balancer 54
Algorithm 5 Particle Swarm Optimization Load Balancer 57
Algorithm 6 DPSO Load Balancer 58

Algorithm 7 Wombat Learning Algorithm 71

Algorithm 8 Enrichment Pipeline Learner 86

ACRONYMS

BPSO	Binary Particle-Swarm Optimization
CBD	Concise Bounded Description
COG	Content Oriented Guidelines
CSV	Comma-Separated Values
DPSO	Deterministic Particle-Swarm Optimization
DALY	Disability Adjusted Life Year
ER	Entity Resolution
EM	Expectation Maximization
FAO	Food and Agriculture Organization of the United Nations
GHO	Global Health Observatory
HDI	Human Development Index
ICT	Information and communication technologies
LD	Link Discovery
LIMES	LInk discovery framework for MEtric Spaces
LS	Link Specifications
LDIF	Linked Data Integration Framework
LGD	LinkedGeoData
LOD	Linked Open Data
MSE	Mean Squared Error
NIF	Natural Language Processing Interchange Format
NIF4OGGD	NLP Interchange Format for Open German Governmental Data
NLP	Natural-Language Processing
OSM	OpenStreetMap
OWL	Web Ontology Language
PFM	Pseudo-F-Measures

PSO	Particle-Swarm Optimization
QA	Question Answering
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
SRL	Statistical Relational Learning
WHO	World Health Organization
WKT	Well-Known Text
W3C	World Wide Web Consortium
YPLL	Years of Potential Life Lost

Part I

PRELIMINARIES

In this part, we first introduce the thesis in Chapter 1, where we discuss the motivations and present our research questions. In Chapter 2, we present the basic notation that will be used throughout the rest of this thesis. Finally in Chapter 3, we give a general overview of the state-of-the-art techniques related to the proposed approaches in Part II.

1 INTRODUCTION

Over the last years, the Linked Open Data (LOD) has evolved from a mere 12 to more than 10,000 knowledge bases[1] [Auer et al., 2013]. These knowledge bases come from diverse domains including (but not limited to) publications, life sciences, social networking, government, media, linguistics. Moreover, the LOD cloud also contains a large number of cross-domain knowledge bases such as *DBpedia* [Lehmann et al., 2014] and *Yago2* [Hoffart et al., 2013]. These knowledge bases are commonly managed in a decentralized fashion and contain partly overlapping information. This architectural choice has led to knowledge pertaining to the same domain being published by independent entities in the LOD cloud [Saleem et al., 2013]. For example, information on drugs can be found in *Diseasome* as well as *DBpedia* and *Drugbank*. Furthermore, certain knowledge bases such as *DBLP* have been published by several bodies, which in turn has lead to duplicated content in the LOD. In addition, large amounts of geo-spatial information have been made available with the growth of heterogeneous Web of Data. For instance, *LinkedGeoData* with approximately 30 billion triples [Auer et al., 2009].

The concurrent publication of knowledge bases containing related information promises to become a phenomenon of increasing importance with the growth of the number of independent data providers. Enabling the joint use of the knowledge bases published by these providers for tasks such as federated queries, cross-ontology question answering and data integration is most commonly tackled by creating links between the resources described within these knowledge bases. Within this thesis, we spur the transition from isolated knowledge bases to enriched Linked Data sets where information can be easily integrated and processed. To achieve this goal, we provide concepts, approaches and use cases that facilitate the integration and enrichment of information with other data types that are already present on the LOD with a focus on geo-spatial data.

MOTIVATION

In the following, we outline the rationale and motivation underlying the research presented in this thesis:

1 http://lodstats.aksw.org

M1. Lack of measures that use the geographic data for linking geo-spatial knowledge bases.

While previous work has compared large number of measures with respect to how well they perform in the link discovery task [Cheatham and Hitzler, 2013], little attention have been paid to measures for linking geo-spatial resources. However, previous works have shown that domain-specific measures and algorithms are required to tackle the problem of geo-spatial link discovery [Ngonga Ngomo, 2013]. For example, 20,354 pairs of cities in *DBpedia*[2] share exactly the same label. For villages in *LinkedGeoData*[3], this number increases to 3,946,750. Consequently, finding links between geo-spatial resources requires devising means to distinguish them using their geo-spatial location. On the Web of Data, the geo-spatial location of resources is most commonly described using either points or more generally by means of vector geometry. Thus, discovering approaches for using geo-spatial information to improve LD requires providing means to measure distances between such vector geometry data.

M2. Lack of automatic LD approaches capable of dealing with knowledge bases with missing and erroneous data.

The basic architectural principles behind Web of Data are akin to those of the document Web and thus decentralized in nature[4]. This architectural choice has led to knowledge pertaining to the same domain being published by independent entities in the LOD cloud. With the growth of the number of independent data providers, the concurrent publication of datasets containing related information promises to become a phenomenon of increasing importance. Enabling the joint use of these datasets for tasks such as federated queries, cross-ontology question answering and data integration is most commonly tackled by creating links between the resources described in the datasets. Devising accurate Link Specifications (LS) to compute these links has been shown to be a difficult and time-consuming problem in previous works [Isele and Bizer, 2011; Isele et al., 2011a; Ngonga Ngomo et al., 2013b; Nikolov et al., 2012]. A recent avenue of research to address this problem is the unsupervised learning of LS [Nikolov et al., 2012; Ngonga Ngomo and Lyko, 2013]. Knowledge bases in the Web of Data with missing and erroneous data [Zaveri et al., 2015] represent big challenge for such unsupervised learning algorithms.

2 *DBpedia* version 3.7 available from http://wiki.dbpedia.org/Downloads
3 *LinkedGeoData* version 2010-07 available from http://downloads.linkedgeodata.org/
4 See http://www.w3.org/DesignIssues/LinkedData.html.

M3. Lack of scalable LD approaches for tackling big geo-spatial knowledge bases.

With the constant growth of geo-spatial knowledge bases over the last years comes the need to develop highly scalable algorithms for the discovery of links between data sources. While several architectures can be used to this end, previous works suggest that approaches based on local hardware resources suffer less from the data transfer bottleneck [Ngonga Ngomo et al., 2013a] and can thus achieve significantly better runtime than parallel approaches which rely on remote hardware (e.g., cloud-based approaches [Kolb and Rahm, 2013]). Moreover, previous works also suggest that load balancing (also called task assignment [Salman et al., 2002]) plays a key role in getting approaches for LD to scale. However, load balancing approaches for local parallel LD algorithms have been paid little attention to so far. In particular, mostly naïve implementations of parallel LD algorithms have been integrated into commonly used LD framework such as SILK [Isele et al., 2011b] and LIMES [Ngonga Ngomo, 2012].

M4. Lack of approaches for automatic updating of links of an evolving geo-spatial knowledge base.

The growth of the Data Web engenders an increasing need for automatic support when maintaining evolving datasets. One of the most crucial tasks when dealing with evolving datasets lies in updating the links from these data sets to other data sets. While supervised approaches have been devised to achieve this goal, they assume that they are provided with both positive and negative examples for links [Auer et al., 2013]. However, the links available on the Data Web only provide positive examples for relations and no negative examples[5]. The open-world assumption underlying the Web of Data suggests that given the non-existence of a link between two resources cannot be understood as stating these two resources are not related. Hence, it is impossible to construct negative examples based on existing positive examples for most relations. Consequently, state-of-the-art supervised learning approaches for link discovery can only be employed if the end users are willing to provide the algorithms with information that is generally not available on the LOD cloud, i.e., with negative examples.

5 3 678 RDF dataset dumps containing 714 714 370 triples analysed via LODStats (see `lodstats.aksw.org`) in March 2015 contained 10 116 041 `owl:sameAs` links and no `owl:differentFrom` links. Moreover, inferring `owl:differentFrom` links is often not possible due to missing schema integration and low expressiveness of knowledge bases.

M5. Lack of automatic approaches for geo-spatial knowledge base enrichment and transformation.

With the adoption of linked data cross academia and industry come novel challenges pertaining to the integration of these datasets for dedicated applications such as tourism, question answering, enhanced reality and many more. Providing consolidated and integrated datasets for these applications demands the specification of data enrichment pipelines, which describe how data from different sources is to be integrated and altered so as to abide by the precepts of the application developer or data user. Currently, most developers implement customized pipelines by compiling sequences of tools manually and connecting them via customized scripts. While this approach most commonly leads to the expected results, it is time-demanding and resource-intensive. Moreover, the results of this effort can most commonly only be reused for new versions of the input data but cannot be ported easily to other datasets. Over the last years, a few frameworks for RDF data enrichment such as *LDIF*[6] and *DEER*[7] have been developed. The frameworks provide enrichment methods such as entity recognition [Speck and Ngonga Ngomo, 2014], link discovery [Ngonga Ngomo, 2012] and schema enrichment [Buhmann and Lehmann, 2013]. However, devising appropriate configurations for these tools can prove to be a difficult endeavour, as the tools require (1) choosing the right sequence of enrichment functions and (2) configuring these functions adequately. Both the first and second task can be tedious.

RESEARCH QUESTIONS AND CONTRIBUTIONS

In this section, we outline the key research questions (RQ) that address the challenges in Section 1.1 along with our contributions towards each of them.

RQ1. What are the existing measures for linking geo-spatial resources?

To answer this research question, we carried out a systematic study of the literature on point sets distance measures according to the approach presented in [Kitchenham, 2004; Moher et al., 2009] (see Chapter 4 for details). By answering *RQ1*, we aim to create a holistic view on existing approaches and tools for geo-spatial link discovery. This is crucial for conceiving guidelines for developing more effective and intuitive link discovery frameworks for geo-spatial knowledge bases. We divide this general research question into the following more concrete sub-questions:

6 http://ldif.wbsg.de/
7 http://aksw.org/Projects/DEER.html

RQ1.1 Which of the existing measures is the most time-efficient?

RQ1.2 Which measures generate mappings with a high precision, recall or F-measure?

RQ1.3 How well do the measures perform when the datasets have different granularities?

RQ1.4 How sensitive are the measures to measurement discrepancies?

RQ1.5 How robust are the measures when both types of discrepancy occur?

RQ2. How can we exploit the intrinsic topology of the Web of Data not only for automating the data integration process but also repairing knowledge bases with missing and erroneous data?

To address this research question, we propose COLIBRI (see Chapter 5). The insight behind COLIBRI is to use the characteristics of transitive 1-to-1 and n-to-1 links (such relations relations occur in several domains such as geography (`locatedIn`) and biology (`descendantSpeciesOf`)) to *detect and correct errors* in the results of unsupervised LD algorithms and the underlying knowledge bases.

RQ3. What are the best load balancing approaches that can be used for linking big geo-spatial knowledge bases?

In Chapter 6, we address the research gap of load balancing for link discovering by first introducing the link discovery as well as the load balancing problems formally. We then introduce a set of heuristics for addressing this problem, including a novel heuristic dubbed DPSO. This novel heuristic employs the basic insights behind Particle-Swarm Optimization (PSO) to determine a load balancing for link discovery tasks in a deterministic manner. Our approach is generic and can be combined with any link discovery approach that can divide the LD problem into a set of tasks within only portions of the input datasets are compared, including methods based on blocking (e.g., *Multiblock* Isele et al. [2011b]) and on space tiling (e.g., [Ngonga Ngomo, 2013]).

RQ4. How can we learn accurate LS based only on the existing positive example in the Web of Data?

We address the drawback aforementioned in motivation (*M4*) of non-existing negative examples in the Web of data by proposing WOMBAT. WOMBAT is (to the best of out knowledge) the first approach for learning LS based on positive examples only. Our approach is inspired

by the concept of generalisation in quasi-ordered spaces. Given a set of positive examples and a grammar to construct LS, we aim to find a specification that covers a large number of positive examples (i.e., achieves a high recall on the positive examples) while still achieving a high precision. In Chapter 7 we give a formal detailed description of the WOMBAT algorithm together with its evaluations.

RQ5. How can we automate the process of geo-spatial knowledge base enrichment and transformation?

We address this challenge by proposing DEER, a supervised machine learning approach for the automatic detection of enrichment pipelines based on a refinement operator and self-configuration algorithms for enrichment functions (Chapter 8). Our approach takes pairs of Concise Bounded Descriptions (CBDs) of resources as input, where the second CBD is the enriched version of first one. Based on these pairs, our approach can learn sequences of atomic enrichment functions that aim to generate each enriched CBD out of the corresponding original one. The output of our approach is an enrichment pipeline that can be used on whole datasets to generate enriched versions.

OVERVIEW OF THE THESIS

In this section, we describe the structure of the thesis, in which we have 12 chapters divided into 3 parts. In Part I, we introduce a set of preliminaries that will be used through the rest of the thesis. In Chapter 1 we provide a general introduction to the thesis. In Chapter 2. we introduce the notation that will be used in the rest of the thesis. then in Chapter 3, We review the state of the art related to our proposed approaches.

Part II contains the main contribution of the thesis, as a set of approaches to deal with various challenges pertaining to automating geo-spatial data linking and enrichment. In Chapter 4, we describe the findings of a systematic literature review of point set distance functions and its usage in LD. Then, we propose COLIBRI in Chapter 5, an algorithm for unsupervised LD through knowledge base repair. Chapter 6 shows an optimization approach for load balancing in parallel LD. In Chapter 8 we introduce DEER, an approach for automating RDF dataset transformation and enrichment.

In the last part of this thesis, we introduce use cases and application scenarios of the algorithms in Part II. First in Chapter 9, we describe how we publish and integrate the Global Health Observatory (GHO) dataset. Then in Chapter 10, we introduce the multilingual dataset *Semantic Quran*. In Chapter 11, we show an ontology based data access and integration methodology for improving the effectiveness of farming in Nepal. Next in Chapter 12, we describe a process of integrating

a novel data set comprising several open datasets across Germany. Finally in Chapter 13, we conclude our thesis and proposes a set of future extensions for our approaches.

2 NOTATION

In this chapter we introduce the basic notation that will be used across the rest of the thesis. We begin by introducing LD problem (Section 2.1). Then, in Section 2.2 we give definitions of refinement operators and their properties.

LINK DISCOVERY

The formal specification of LD adopted herein is akin to that proposed in [Ngonga Ngomo, 2012]. In the following, we will use *link discovery* as an umbrella term for *deduplication*, *record linkage*, *entity resolution* and similar terms used across literature.

Problem Definition

Given two sets S respectively T of source respectively target resources as well as a relation R, the goal of Link Discovery (LD) is is to find the set $M \subseteq S \times T$ of pairs $(s,t) \in S \times T$ such that $R(s,t)$. Note that, S and T are two not necessarily distinct sets of instances. One way to automate this discovery is to compare the $s \in S$ and $t \in T$ based on their properties using a (in general complex) similarity metric. Two entities are then considered to be linked via R if their similarity is superior to a threshold θ. If R is `owl:sameAs`, then we are faced with a *deduplication task*. We are aware that several categories of approaches can be envisaged for discovering links between instances, for example using formal inferences or semantic similarity functions. Throughout this thesis, we will consider LD via properties. This is the most common definition of instance-based LD [Ngonga Ngomo and Auer, 2011; Volz et al., 2009a], which translates into the following formal definition:

Definition 1 (Link Discovery)**.** *Given two sets S (source) and T (target) of instances, a (complex) similarity measure σ over the properties of $s \in S$ and $t \in T$ and a similarity threshold $\theta \in [0,1]$, the goal of LD is to compute the set of pairs of instances $(s,t) \in S \times T$ such that $\sigma(s,t) \geqslant \theta$.*

This problem can be expressed equivalently as follows:

Definition 2 (Link Discovery on Distances)**.** *Given two sets S and T of instances, a (complex) distance measure δ over the properties of $s \in S$ and $t \in T$ and a distance threshold $\theta \in [0,\infty[$, the goal of LD is to compute the set of pairs of instances $(s,t) \in S \times T$ such that $\delta(s,t) \leqslant \tau$.*

Note that a distance function δ can always be transformed into a normed similarity function σ by setting $\sigma(x,y) = (1+\delta(x,y))^{-1}$.

Hence, the distance threshold τ can be transformed into a similarity threshold θ by means of the equation $\theta = (1+\tau)^{-1}$. Consequently, distance and similarities are used interchangeably within this thesis.

Although it is sometimes sufficient to define atomic similarity functions (i.e., similarity functions that operate on exactly one property pair) for LD, many LD problems demand the specification of complex similarity functions to return accurate links. For example, while the name of bands can be used for detecting duplicate bands across different knowledge bases, linking cities from different knowledge bases requires taking more properties into consideration (e.g., the different names of the cities as well as their latitude and longitude) to compute links accurately. The same holds for movies, where similarity functions based on properties such as the label and length of the movie as well as the name of its director are necessary to achieve high-accuracy link discovery. Consequently, linking on the Data Web demands frameworks that support complex link specifications.

ORCHID

Given that the explicit computation of M is usually a very complex endeavor, M is usually approximated by a set $\tilde{M} = \{(s,t,\delta(s,t)) \in S \times T \times \mathbb{R}^+ : \delta(s,t) \leqslant \theta\}$, where δ is a distance function and $\theta \geqslant 0$ is a distance threshold. For geographic data, the resources s and t are described by using single points or (ordered) sets of points, which we regard as polygons. Given that we can regard points as polygons with one node, we will speak of resources being described as polygons throughout this thesis. We will use a subscript notation to label the nodes that make up resources. For example, if s had three nodes, we would denote them s_1, s_2, and s_3. For convenience's sake, we will write $s = \{s_1, s_2, s_3\}$ and $s_i \in s$.

Most algorithms for LD achieve scalability by first dividing S respectively T into non-empty subsets $S_1 \ldots S_k$ resp.$T_1 \ldots T_l$ such that $\bigcup_{i=1}^{k} S_i = S$ and $\bigcup_{j=1}^{l} T_j = T$. Note that the different subsets of S respectively T can overlap. In a second step, most time-efficient algorithms determine pairs of subsets (S_i, T_j) whose elements are to be compared. All elements (s,t) of all Cartesian products $S_i \times T_j$ are finally compared by means of the measure δ and only those with $\delta(s,t) \leqslant \theta$ are written into $\tilde{M}$.

One of the first space tiling algorithms for dealing with LD problem based on a geo-spatial data is ORCHID [Ngonga Ngomo, 2013]. The idea behind ORCHID is to reduce the number of comparisons needed for computing $\tilde{M}$ while remaining complete and being reduction-ratio-optimal. To achieve this goal, ORCHID uses a space discretization approach and only compares polygons $t \in T$ which lie within a certain range of $s \in S$. An example of the discretization gen-

erated by ORCHID is shown in Figure 1. Instead of comparing Oslo with all other elements of the dataset, ORCHID would only compare it with the geo-spatial objects shown in the gray cells.

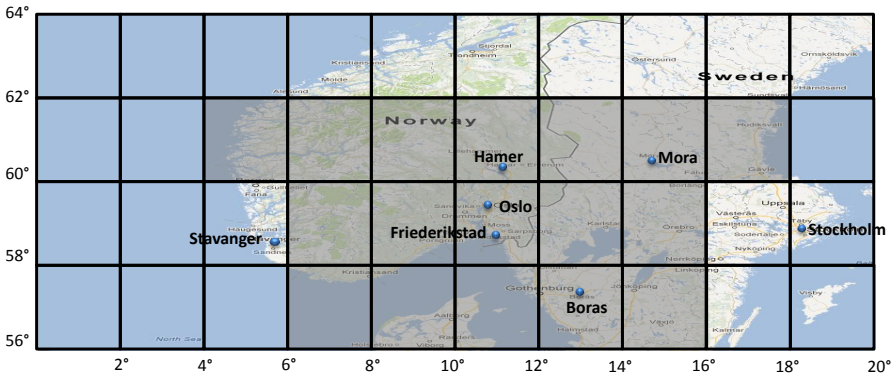


Figure 1: Example of tiling for $\alpha = 1$ and $\theta = 222.6\text{km}$ (i.e., $\Delta_R = 2°$). Here, the resource to link is Oslo. The gray cells are the elements of A(Oslo). [Ngonga Ngomo, 2013]

REFINEMENT OPERATORS

Refinement operators have traditionally been used, e.g. in [Lehmann and Hitzler, 2010], to traverse search spaces in structured machine learning problems. Their theoretical properties give an indication of how suitable they are within a learning algorithm in terms of accuracy and efficiency.

Definition 3 (Refinement Operator). *Given a quasi-ordered space* $(S, \preccurlyeq)$ *an upward refinement operator* r *is a mapping from* S *to* 2^S *such that* $\forall s \in S : s' \in r(s) \Rightarrow s \preccurlyeq s'$. s' *is then called a generalization of* s.

Definition 4 (Refinement chain). *A set* $M_2 \in \mathcal{M}$ *belongs to the refinement chain of* $M_1 \in \mathcal{M}$ *iff* $\exists k \in \mathbb{N} : M_2 \in r^k(M_1)$, *where* $r^0(M) = M$ *and* $r^k(M) = r(r^{k-1}(M))$.

A refinement operator r over the quasi-ordered space $(S, \preccurlyeq)$ can abide by the following criteria.

Definition 5 (Finiteness). r *is finite iff* $r(s)$ *is finite for all* $s \in S$.

Definition 6 (Properness). r *is proper if* $\forall s \in S, s' \in r(s) \Rightarrow s \neq s'$.

Definition 7 (Completeness). r *is said to be complete if for all* s *and* s', $s' \preccurlyeq s$ *implies that there is a refinement chain between* s *and* s'.

Definition 8 (Redundancy). *A refinement operator* r *over the space* $(S, \preccurlyeq)$ *is redundant if two different refinement chains can exist between* $s \in S$ *and* $s' \in S$.

3

RELATED WORK

In this chapter, we introduce a set of state-of-the-art related to the approaches proposed in Part II.

POINT SET DISTANCE MEASURES

Several reviews on distances for point sets have been published. For example, Eiter and Mannila [1997] reviewed some of the distance functions proposed in the literature and presented efficient algorithms for the computation of these measures. Also, Atallah et al. [1991] presented parallel implementation of some distance functions between convex and non-convex (possibly intersecting) polygons.

Ramon and Bruynooghe [2001] introduced a metric computable in polynomial time for dealing with the point set similarity problem. Also, Tănase et al. [2005] presented an approach to compute the similarity between multiple polylines and a polygon using dynamic programming. Barequet et al. [1997] showed how to compute the respective nearest- and furthest-site Voronoi diagrams of point sites in the plane. In another work done by Barequet et al. [2001], he provided near-optimal deterministic time algorithms to compute the corresponding nearest- and furthest-site Voronoi diagrams of point sites.

Hausdorff distances are commonly used in fields such as object modeling, computer vision and object tracking. Atallah [1983] focused on the Hausdorff distance and presents an approach for its efficient computation between convex polygons. While the approach is quasi-linear in the number of nodes of the polygons, it cannot deal with non-convex polygons as commonly found in geographic data. A similar approach presented by Tang et al. [2009] allows approximating Hausdorff distances within a certain error bound, while Bartoň et al. [2010] presents an exact approach. Nutanong et al. [2011] proposes an approach to compute Hausdorff distances between trajectories using R-trees within an L_2-space.

Fréchet distance is basically used in piecewise curve similarity detection like in case of hand writing recognition. For example, Alt and Godau [1995] introduced an algorithm for computing Fréchet distance between two polygonal curves, while Chambers et al. [2010] presented a polynomial-time algorithm to compute the homotopic Fréchet distance between two given polygonal curves in the plane avoiding a given set of polygonal obstacles. Driemel et al. [2012] proposed an approximation of Fréchet distance for realistic curves in

near linear time. Cook IV et al. [2011] presented three different methods to adapt the original Fréche distance in non-flat surfaces.

There are number of techniques presented in literature that *-if applied in combination with the distance approaches-* can achieve better performance. In order to limit the number of polygons to be compared in deduplication problems, Joshi et al. [2009] proposed a dissimilarity function for clustering geospatial polygons. A kinematics-based method proposed in [Saykol et al., 2002] approximates large polygon using less number of points is proposed, thus requires less execution time for distance measurement. Yet, another algorithm presented by [Quinlan, 1994] models non-convex polygons as the union of a set of convex components, Guthe et al. [2005] showed an approach for the comparison of 3D models represented as triangular meshes. The approach is based on a subdivision sampling algorithm that makes used of octrees to approximate distances. ORCHID [Ngonga Ngomo, 2013] was designed especially for the Hausdorff distance but can be extended to deal with other measures.

SUPERVISED VS. UNSUPERVISED LINK DISCOVERY APPROACHES

Most LD approaches for learning link specifications developed so far abide by the paradigm of supervised machine learning. One of the first approaches to target this goal was presented in [Isele and Bizer, 2011]. While this approach achieves high F-measures, it also requires large amounts of training data. However, creating training data for link discovery is a very expensive process, especially given the size of current knowledge bases. Supervised LD approaches which try to reduce the amount training data required are most commonly based on active learning (see, e.g., [Isele et al., 2012; Ngonga Ngomo et al., 2013b]). Still, these approaches are not guaranteed to require a small amount of training data to converge. In newer works, unsupervised techniques for learning LD specifications were developed [Ngonga Ngomo and Lyko, 2013; Nikolov et al., 2012]. The main advantage of unsupervised learning techniques is that they do not require any training data to discover mappings. Moreover, the classifiers they generate can be used as initial classifiers for supervised LD approaches. In general, unsupervised approaches assume some knowledge about the type of links that are to be discovered. For example, unsupervised approaches for ontology alignment such as PARIS [Suchanek et al., 2011] aim to discover exclusively `owl:sameAs` links. To this end, PARIS relies on a probabilistic model and maps instances, properties and ontology elements. Similarly, the approach presented in [Nikolov et al., 2012] assumes that a 1-to-1 mapping is to be discovered. Here, the mappings are discovered by using a genetic programming approach whose fitness function is set to a *Pseudo-F-measure*. The main drawback of this approach is that it is not deterministic. Thus, it pro-

vides no guarantee of finding a good specification. This problem was addressed by EUCLID [Ngonga Ngomo and Lyko, 2013] which is deterministic.

LINK DISCOVERY FOR MORE THAN TWO DATASETS

While ontology-matching approaches that rely on more than 2 ontologies have existed for almost a decade [Doan et al., 2003; Euzenat, 2008; Madhavan and Halevy, 2003], LD approaches that aim to discover between n datasets have only started to emerge in newer literature. For instance, the approach proposed by Hartung et al. [2013] suggests a composition method for link discovery between n datasets. The approach is based on strategies for combining and filtering mappings between resources to generate links between knowledge bases. The framework introduced by Jiang et al. [2012] aims to predict links in multi-relational graph. To this end, it models the relations of the knowledge bases using set of description matrices and combines them using an additive model. Consequently, it tries to achieve efficient learning using an alternating least squares approach exploiting sparse matrix algebra and low-rank approximations. The *Multi-Core Assignment Algorithm* presented by Böhm et al. [2012] automated the creation of `owl:sameAs` links across multiple knowledge bases in a globally consistent manner. A drawback of this approach is that it requires a large amount of processing power.

In contrast to many other approaches which model data as independent and identically distributed, Statistical Relational Learning (SRL) approaches model assume that the input data points have properties. Examples of SRL approaches that can be used for predicate detection include CP and Tucker [Kolda and Bader, 2009] as well as RESCAL [Nickel et al., 2012], which all rely on tensor factorization. In general, approaches which rely on tensor factorization have a higher complexity than EUCLID [Ngonga Ngomo and Lyko, 2013]. For example, CP's complexity is quadratic in the number of predicates. Related approaches that have been employed on Semantic Web data and ontologies include approaches related to Bayesian networks, inductive learning and kernel learning [Bloehdorn and Sure, 2007; d'Amato et al., 2008; Nickel et al., 2012; Pérez-Solà and Herrera-Joancomartí, 2013; Sutskever et al., 2009]. Due to the complexity of the models they rely on, most of these approaches are likely not to scale to very large datasets. The LInk discovery framework for MEtric Spaces (LIMES) (in which EUCLID is implemented) has yet been shown to scale well on large datasets [Ngonga Ngomo, 2012]. More details on SRL can be found in [Getoor and Taskar, 2007].

LOAD BALANCING APPROACHES FOR LINK DISCOVERY

Load balancing techniques have been applied in a large number of disciplines that deal with big data. For handling massive graphs such as the ones generated by social networks, Yan et al. [2015] introduces two message reduction techniques for distributed graph computation load balancing. For dealing with federated queries, Ali et al. [2014] proposes an RDF query routing index that permits a better load balancing for distributed query processing. Kolb et al. [2012] proposes two approaches for load balancing for the complex problem of Entity Resolution (ER), which utilize a preprocessing *MapReduce* job to analyze the data distribution. Kolb and Rahm [2013] demonstrates a tool called *Dedoop* for *MapReduce*-based ER of large datasets that implements similar load balancing techniques. A comparative study for different load balancing algorithms for *MapReduce* environment is presented in [Hefny et al., 2014].

Finding an optimal load balancing is known to be NP-complete. Thus, Ludwig and Moallem [2011] provides two heuristics for distributed grid load balancing, one is based on *ant-colony optimization* and the other is based on *particle-swarm optimization*. Yet, another heuristic proposed in [Jin et al., 2004], Binary Particle-Swarm Optimization (BPSO), is used for network reconfiguration load balancing. Pan et al. [2015] proposes an artificial bee-colony-based load balancing algorithm for cloud computing.

The study [Akl, 2004] introduces *superlinear* performance analyses of real-time parallel computation. The study shows that parallel computers with n processors can solve a computational problem more than n times faster than a sequential one. In another work [Alba, 2002], the *superlinear* performance concluded to be also possible for parallel evolutionary algorithms both theoretically and in practice.

POSITIVE ONLY MACHINE LEARNING

There is a significant body of related work on *positive only learning*, which we can only briefly cover here. For instance, the work presented by Muggleton [1997] showed that logic programs are *learnable* with arbitrarily low expected error from positive examples only. Nigam et al. [2000] proposed an algorithm for learning from labeled and unlabeled documents based on the combination of Expectation Maximization (EM) and a naive Bayes classifier. Denis et al. [2005] provides an algorithm for learning from positive and unlabeled examples for statistical queries. The pLSA algorithm [Zhou et al., 2010] extends the original probabilistic latent semantic analysis, which is a purely unsupervised framework, by injecting a small amount of supervision information from the user.

For learning with *refinement operators*, significant previous work exists in the area of Inductive Logic Programming and more generally concept learning. A milestone was the Model Inference System in [Shapiro, 1991]. Shapiro describes how refinement operators can be used to adapt a hypothesis to a sequence of examples. Afterwards, refinement operators became widely used as a learning method. In [van der Laag and Nienhuys-Cheng, 1994] some general results regarding refinement operators in quasi-ordered spaces were published. Nonexistence conditions for ideal refinement operators relating to infinite ascending and descending refinement chains and covers have been developed. This has been used to show that ideal refinement operators for clauses ordered by θ-subsumption do not exist. Unfortunately, we could not make use of these results directly, because proving properties of covers in description logics without using a specific language is likely to be harder than directly proving the results. Nienhuys-Cheng et al. [1993] discussed refinement for different versions of subsumption, in particular weakenings of logical implication. A few years later, it was shown in [Nienhuys-Cheng et al., 1999] how to extend refinement operators to learn general prenex conjunctive normal form. Perfect operators, i.e. operators which are weakly complete, locally finite, non-redundant, and minimal, were discussed in [Badea and Stanciu, 1999]. Because such operators do not exist for clauses ordered by θ-subsumption, as previously shown in [van der Laag and Nienhuys-Cheng, 1994], weaker versions of subsumption were considered. This was later extended to theories, i.e. sets of clauses [Fanizzi et al., 2003]. A less widely used property of refinement operators, called flexibility, was discussed in [Badea, 2000]. Flexibility essentially means that previous refinements of an operator can influence the choice of the next refinement. The article discusses how flexibility interacts with other properties and how it influences the search process in a learning algorithm.

For *description logics*, a significant body of work has been devoted to the study of refinement operators. In [Esposito et al., 2004] and later [Iannone et al., 2007], algorithms for learning in description logics (in particular for the language $\mathcal{ALC}$) were created which also make use of refinement operators. Badea and Nienhuys-Cheng [2000] presents a refinement operator for $\mathcal{ALER}$. From the author's work, studies of refinement operators include [Lehmann and Hitzler, 2007] which analysed properties of $\mathcal{ALC}$ refinement operators and was later in [Lehmann and Hitzler, 2010] extended to more expressive description logics. A constructive existence proof for ideal (complete, proper and finite) operators in the lightweight $\mathcal{EL}$ description logics has been shown in [Lehmann and Haase, 2009].

RDF DATASET TRANSFORMATION AND ENRICHMENT

Linked Data enrichment is an important topic for all applications that rely on a large number of knowledge bases and necessitate a unified view on this data, e.g., Question Answering (QA) frameworks [Lopez et al., 2013], Linked Education [Dietze et al., 2013] and all forms of semantic mashups [Hoang et al., 2014]. In recent work, several challenges and requirements to Linked Data consumption and integration have been pointed out [Millard et al., 2010]. Several approaches and frameworks have been developed with the aim of addressing many of these challenges. For example, the R2R framework [Bizer and Schultz, 2010] addresses those by enabling the publish of mappings across knowledge bases that allow to map classes and defined the transformation of property values. While this framework supports a large number of transformations, it does not allow the automatic discovery of possible transformations. The Linked Data Integration Framework (LDIF) [Schwarte et al., 2011], whose goal is to support the integration of RDF data, builds upon R2R mappings and technologies such as SILK [Isele and Bizer, 2011] and LDSpider[1]. The concept behind the framework is to enable users to create periodic integration jobs via simple XML configurations. Still these configurations have to be created manually. The same drawback holds for the Semantic Web Pipes[2] [Phuoc et al., 2009], which follows the idea of Yahoo Pipes[3] to enable the integration of data in formats such as RDF and XML. By using Semantic Web Pipes, users can efficiently create semantic mashups by using a number of operators (such as getRDF, getXML, etc.) and connect these manually within a simple interface. KnoFuss [Nikolov et al., 2009] addresses data integration from the point of view of link discovery. It begins by detecting URIs that stand for the same real-world entity and either merging them together or linking them via `owl:sameAs`. In addition, it allows to monitor the interaction between instance and dataset matching (which is similar to ontology matching [Euzenat and Shvaiko, 2007]). Fluid Operations' Information Workbench[4] allows to search through, manipulate and integrate datasets for purposes such as business intelligence.

With the advent of social networking data, Choudhury et al. [2009] describes a framework for semantic enrichment, ranking and integration of web videos, and Abel et al. [2011] presents semantic enrichment framework of *Twitter* posts. Finally, Hasan et al. [2011] tackles the linked data enrichment problem for sensor data via an approach that sees enrichment as a process driven by situations of interest.

1 http://code.google.com/p/ldspider/
2 http://pipes.deri.org/
3 http://pipes.yahoo.com/pipes/
4 http://www.fluidops.com/information-workbench/

Part II

APPROACHES

In this part of the thesis, we propose a set of approaches for automating RDF data sets integration and enrichment. In Chapter 4, we evaluate various point set distance functions for LD of geo-spatial resources. Then, in Chapter 5, we propose COLIBRI, an unsupervised LD approach through knowledge base repair. We introduce DPSO in Chapter 6, a novel load balancing approach for LD. The WOMBAT algorithm for supervised data sets linking is presented in Chapter 7. Finally in Chapter 8, we demonstrate DEER, an algorithm for automating data sets transformation and enrichment.

4

A SYSTEMATIC EVALUATION OF POINT SET DISTANCE MEASURES FOR LINK DISCOVERY

In this chapter, we present a systematic evaluation of point sets measures for geo-spatial LD. A paper about the work is submitted to the Semantic Web Journal [Sherif and Ngonga Ngomo, 2015c]. The author analysed the behaviour of different measures through a survey, implemented the resulted point sets measures, carried out the evaluations and also co-wrote the paper.

While previous works have compared a large number of measures with respect to how well they perform in the LD task [Cheatham and Hitzler, 2013], measures for linking geo-spatial resources have been paid little attention to. Previous works have yet shown that domain-specific measures and algorithms are required to tackle the problem of geo-spatial LD [Ngonga Ngomo, 2013]. For example, 20,354 pairs of cities in *DBpedia 2014* share exactly the same label. For villages in *LinkedGeoData 2014*, this number grows to 3,946,750. Consequently, finding links between geo-spatial resources requires devising means to distinguish them using their geo-spatial location. On the Web of Data, the geo-spatial location of resources is most commonly described using either points or more generally by means of vector geometry. Thus, devising means for using geo-spatial information to improve LD requires providing means to measure distances between such vector geometry data.

Examples of vector geometry descriptions for the country of *Malta* are shown in Figure 2. As displayed in the examples, two types of discrepancies occur when one compares the vector descriptions of the same real-world entity (e.g., Malta) in different data sets: First, the different vector descriptions of a given real-world entity often comprise different points across different data sets. For example, Malta's vector description in DBpedia contains the point with latitude 14.46 and longitude 35.89. In LinkedGeoData, the same country is described by the point of latitude 14.5 and longitude 35.9. We dub the discrepancy in latitude and longitude for points in the vector description *measurement discrepancy*. A second type of discrepancy that occurs in the vector description of geo-spatial resources across different data sets are discrepancies in *granularity*. For example, Malta is described by one polygon in DBpedia, two polygons in NUTS and a single point in LinkedGeoData.

Analysing the behaviour of different measures with respect to these two types of discrepancies is of central importance to detect the measures that should be used for geo-spatial LD. In this chapter, we address this research gap by first surveying existing measures that can be used for comparing point sets. We then compare these measures in series of experiments on samples extracted from three real data sets with the aim of answering the questions introduced in Section 4.2.

Note that throughout this chapter, we model complex representations of geo-spatial objects as point sets. While more complex rep-

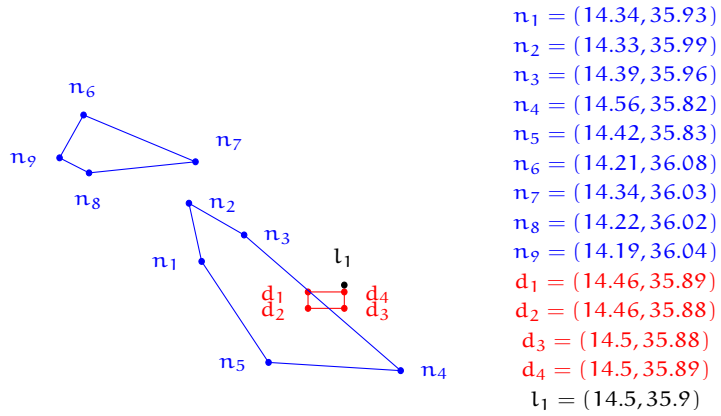


Figure 2: Vector description of the country of *Malta*. The blue polygons shows the vector geometry for *Malta* in the *NUTS* dataset, the red polygon shows the same for the *DBpedia*, while the black point shows the location of the same real-world entity according to *LinkedGeoData*.

resentations can be chosen, comparing all corresponding measures would go beyond the scope of this work. In addition, we are only concerned with atomic measures and do not consider combinations of measures. Approaches that allow combining measures can be found in [Nentwig et al., 2015].

The remainder of this chapter is structured as follows: Section 4.1 introduces some basic assumption and notations that will be used all over the rest of the chapter. Section 4.2 introduces our systematic survey methodology. Then, in Section 4.3 we give a detailed description of each of point set distance functions, as well as their mathematical formulation and different implementations. Thereafter, in Section 4.4 we introduce evaluation of our work for both scalability and robustness. All measures and algorithms presented herein were integrated into the LIMES framework.[1]

NOTATION

Here, we extend the formal specification of LD first introduced in Section 2.1. In addition to bearing properties similar to those bared by other types of resources (label, country, etc.), geo-spatial resources are commonly described by means of vector geometry.[2] Each vector description can be modelled as a set of points. We will write $s = (s_1, \ldots, s_n)$ to denote that the vector description of the resource s comprises the points $s_1, \ldots, s_n$. A point s_i on the surface of the planet is fully described by two values: its latitude $\mathrm{lat}(s_i) = \varphi_i$ and its

1 http://limes.sf.net

2 Most commonly encoded in the Well-Known Text (WKT) format, see http://www.opengeospatial.org/standards/sfa.

longitude $lon(s_i) = \lambda_i$. We will denote points s_i as pairs (φ_i, λ_i). Then, the distance between two points s_1 and s_2 can be computed by using the *orthodromic distance*

$$\delta(s_1, s_2) = \mathbf{R} \cos^{-1}\left(\sin(\varphi_1)\sin(\varphi_2) + \cos(\varphi_1)\cos(\varphi_2)\cos(\lambda_2 - \lambda_1)\right),$$

where $\mathbf{R} = 6371\text{km}$ is the planet's radius.[3]

Alternatively, the distance between two points s_1 and s_2 can be computed based on the *great elliptic curve distance* [Bowring, 1984]. Note that this distance is recommended in previous works (e.g., [Chrisman and Girres, 2013]) as it is more accurate than the *orthodromic distance*. However, given that our evaluations (see Table 2) showed that the distance error of *orthodromic distance* did not affect the LD results and that the *orthodromic distance* has a lower time complexity that the great elliptic curve distance, we rely on the orthodromic distance throughout the explanations in this chapter.

Computing the distance between sets of points is yet a more difficult endeavor. Over the last years, several measures have been developed to achieve this task. Most of these approaches regard vector descriptions as ordered set of points. In the following sections, we present such measures and evaluate their robustness against different types of discrepancies.

SYSTEMATIC SURVEY METHODOLOGY

We carried out a systematic study of the literature on distance measures for point sets according to the approach presented in [Kitchenham, 2004; Moher et al., 2009]. In the following, we present our survey approach in more detail.

Research Question Formulation

We began by defining research questions that guided our search for measures. These questions were as follows:

- Q_1: Which of the existing measures is the most time-efficient measure?
- Q_2: Which measure generates mappings with a high precision, recall, or F-measure?
- Q_3: How well do the measures perform when the data sets have different granularities?
- Q_4: How sensitive are the measures to measurement discrepancies?
- Q_5: How robust are the measures when both types of discrepancy occur?

3 Here, we assume the planet to be a perfect sphere.

Eligibility Criteria

To direct our search process towards answering our research questions, we created two lists of inclusion/exclusion criteria for papers. Papers had to abide by all inclusion criteria and by none of the exclusion criteria to be part of our survey:

- Inclusion Criteria
  - Work published in English between 2003 and 2013.
  - Studies on geographic terms based LD.
  - Algorithms for finding distance between point sets.
  - Techniques for improving performance of some well-known point sets distance Algorithms.
- Exclusion Criteria
  - Work that were not peer-reviewed or published.
  - Work that were published as a poster abstract.
  - Distance functions that focused on finding distances only between convex point sets.

Search Strategy

Based on the research question and the eligibility criteria, we defined a set of most related keywords. There were as follows: *Linked Data, LD, record linkage, polygon, point set, distance, metric, geographic, spatial, non-convex*. We used those keywords as follows:

- *Linked Data* AND (*Link discovery* OR *record linkage*) AND (*geographic* OR *spatial*)
- *Non-convex* AND (*polygon* OR *point set*) AND (*distance* OR *metric*)

A keyword search was applied in the following list of search engines, digital libraries, journals, conferences and their respective workshops:

- Search Engines and digital libraries:
  - Google Scholar[4]
  - ACM Digital Library[5]
  - Springer Link[6]

4 http://scholar.google.com/
5 http://dl.acm.org/
6 http://link.springer.com/

  - Science Direct[7]
  - ISI Web of Science[8]

- Journals:
  - Semantic Web Journal (SWJ)[9]
  - Journal of Web Semantics (JWS)[10]
  - Journal of Data and Knowledge Engineering (JDWE)[11]

Search Methodology Phases

In order to conduct our systematic literature review, we applied a six-phase search methodology:

1. Apply keywords to the search engine using the time frame from 2003–2013.
2. Scan article titles based on inclusion/exclusion criteria.
3. Import output from phase 2 to a reference manager software to remove duplicates. Here, we used *Mendeley*[12] as it is free and has functionality for deduplication.
4. Review abstracts according to include/exclude criteria.
5. Read through the papers, looking for some approaches that fits the inclusion criteria and exclude papers that fits the exclusion criteria. Also, retrieve and analyze related papers from references.
6. Implement point sets distance functions found in phase 5.

Table 1 provides details about the number of retrieved articles through each of the first five search phases. Note that in the sixth phase we only implemented distance functions found in the articles resulted from phase 5.

DISTANCE MEASURES FOR POINT SETS

In the following, we present each of the distance measures derived from our systematic survey and exemplify it by using the DBpedia and NUTS descriptions of Malta presented in Figure 2. The input for the distance measures consists of two point sets $s = (s_1, \ldots, s_n)$ and $t = (t_1, \ldots, t_m)$, where n resp. m stands for the number of distinct points in the description of s resp. t. W.l.o.g, we assume $n \geqslant m$.

7 http://www.sciencedirect.com/
8 http://portal.isiknowledge.com/
9 http://www.semantic-web-journal.net/
10 http://www.websemanticsjournal.org/
11 http://www.journals.elsevier.com/data-and-knowledge-engineering/
12 http://www.mendeley.com/

Table 1: Number of retrieved articles during each of the search methodology phases.

Search Engines	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
Google Scholar	9,860	21	19	10	4
ACM Digital Library	3,677	16	16	5	3
Springer Link	5,101	22	21	11	8
Science Direct	1055	21	18	10	4
ISI Web of Science	176	15	14	4	2
SWJ	0	0	0	0	0
JWS	0	0	0	0	0
JDWE	0	0	0	0	0

Mean Distance Function

The mean distance is one of the most efficient distance measures for point sets [Duda et al., 2001]. First, a mean point is computed for each point set. Then, the distance between the two means is computed by using the orthodromic distance. Formally:

$$D_{mean}(s,t) = \delta\left(\frac{\sum_{s_i \in s} s_i}{n}, \frac{\sum_{t_j \in t} t_j}{m}\right). \tag{1}$$

D_{mean} can be computed in $O(n)$. For our example, the mean of the DBpedia description of Malta is the point (14.48, 35.89). The mean for the NUTS description are (14.33, 35.97). Thus, D_{mean} returns 18.46km as the distance between the two means points.

Max Distance Function

The idea behind this measure is to compute the overall maximal distance between points $s_i \in s$ and $t_j \in t$. Formally, the maximum distance is defined as:

$$D_{max}(s,t) = \max_{s_i \in s, t_j \in t} \delta(s_i, t_j). \tag{2}$$

For our example, D_{max} returns 38.59km as the distance between the points d_3 and n_6. Due to its construction, this distance is particularly sensitive to outliers. While the naive implementation of *Max* is in $O(n^2)$, Bhattacharya and Toussaint [1983] introduced an efficient implementation that achieves a complexity of $O(n \log n)$.

Min Distance Function

The main idea of the *Min* is akin to that of *Max* and is formally defined as

$$D_{min}(s,t) = \min_{s_i \in s, t_j \in t} \delta(s_i, t_j). \tag{3}$$

Going back to our example, D_{min} returns 7.82km as the distance between the points d_2 and n_5. Like D_{max}, D_{min} can be implemented to achieve a complexity of $O(n \log n)$ [Toussaint and Bhattacharya, 1981; McKenna and Toussaint, 1985].

Average Distance Function

For computing the *average* point sets distance function, the orthodromic distance measures between all the source-target points pairs is cumulated and divided by the number of point source-target point pairs:

$$D_{avg}(s,t) = \frac{1}{nm} \sum_{s_i \in s, t_j \in t} \delta(s_i, t_j). \tag{4}$$

For our example, D_{avg} returns 22km. A naive implementation of the *average* distance is $O(n^2)$,

Sum of Minimums Distance Function

This distance function was first proposed by [Niiniluoto, 1987] and is computed as follows: First, the closest point t_j to each point s_i is to be detected, i.e., the point $t_j = \arg\min_{t_k \in t} \delta(s_i, t_k)$. The same operation is carried out with source and target reversed. Finally, the average of the two values is then the distance value. Formally, the *sum of minimums* distance is defined as:

$$D_{som}(s,t) = \frac{1}{2}\Big(\sum_{s_i \in s} \min_{t_j \in t} \delta(s_i, t_j) + \sum_{t_i \in t} \min_{s_j \in s} \delta(t_i, s_j) \Big). \tag{5}$$

Going back again to our example, the sum of minimum distances from each of DBpedia points describing Malta to the ones of NUTS is 37.27km, and from NUTS to DBpedia is 178.58km. Consequently, D_{som} returns 107.92km as the average of the two values. The *sum of minimum* has the same complexity as D_{min}.

Surjection Distance Function

The *surjection* distance function introduced by Oddie [1978] defines the distance between two point sets as follows: The minimum dis-

tance between the sum of distances of the surjection of the larger set to the smaller one. Formally, the *Surjection* distance is defined as:

$$D_s(s,t) = \min_{\eta} \sum_{(e1,e2)\in\eta} \delta(e_1,e_2), \tag{6}$$

where η is the surjection from the larger of the point sets s and t to the smaller. In to our example, $\eta = (n_1,d_4)$, (n_2,d_1), (n_3,d_2), (n_4,d_3), (n_5,d_4), (n_6,d_1), (n_7,d_1), (n_8,d_1) and (n_9,d_1). Then, D_s returns 184.74km as the sum of the orthodromic distances between each of the point pairs included in η. A main drawback of the *surjection* is being biased toward some points ignoring some others in calculations. (i.e. putting more weight in some points more than the others) For instance in our example, η contains 5 different points surjected to d_1, while only one point surjected to d_2.

Fair Surjection Distance Function

In order to fix the bias of the *surjection* distance function, Oddie [1978] introduces an extension of the *surjection* function which is dubbed *fair surjection*. The surjection between sets S and t is said to be *fair* if η' maps elements of s as evenly as possible to t. The *fair surjection* is defined formally as:

$$D_{fs}(s,t) = \min_{\eta'} \sum_{(e1,e2)\in\eta'} \delta(e_1,e_2), \tag{7}$$

where η' is the evenly mapped surjection from the larger of the sets s and t to the smaller. For our example, $\eta' = (n_1,d_1)$, (n_2,d_2), (n_3,d_3), (n_4,d_4), (n_5,d_1), (n_6,d_2), (n_7,d_3), (n_8,d_4) and (n_9,d_1). Then, D_{fs} returns 137.42km as the sum of the orthodromic distances between each of the point pairs included in η'.

Link Distance Function

The link distance introduced by Eiter and Mannila [1997] defines distance between two point sets s and t as a relation $R \subseteq s \times t$ satisfying

1. For all $s_i \in s$ there exists $t_j \in t$ such that $(s_i,t_j) \in R$
2. For all $t_j \in t$ there exists $s_i \in s$ such that $(s_i,t_j) \in R$

Formally, The *minimum link distance* between two point sets s and t is defined by

$$D_l(s,t) = \min_{R} \sum_{(s_i,t_j)\in R} \delta(s_i,t_j), \tag{8}$$

where minimum is computed from all relations R, where R is a linking between s and t satisfying the previous two conditions. For our

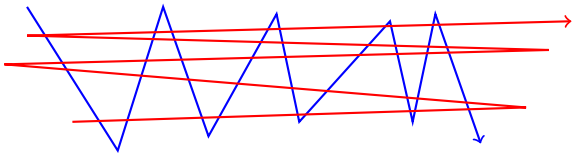


Figure 3: Fréchet vs other distance approaches

example, the small granularity of the Malta descriptions in the data sets at hand leads to D_l having the same results as D_{fs}. See [Eiter and Mannila, 1997] for complexity analysis for *surjection*, *fair surjection* and *link* distance functions.

Hausdorff Distance Function

The *Hausdorff* distance is a measure of the maximum of the minimum distances between two sets of points. Hausdorff is one of the commonly used approach for determining the similarity between point sets [Huttenlocher et al., 1992]. Formally, the Hausdorff distance is defined as

$$D_h(s,t) = \max_{s_i \in s} \left\{ \min_{t_j \in t} \left\{ \delta(s_i, t_j) \right\} \right\}. \tag{9}$$

Back to our example, First, the algorithm finds the orthodromic distance between each of the points of DBpedia to the nearest point NUTS, which found to be the distances between the point pairs (d_1, n_5), (d_2, n_5), (d_3, n_4), and (d_4, n_4). Then, D_h is the maximum distance of them, which is between the point d_4 and n_4 equals 34.21km. Ngonga Ngomo [2013] introduces two efficient approaches for computing bound Hausdorff distance.

Fréchet Distance Function

Most of the distance measures presented before have a considerable common disadvantage. Consider the two curves shown in Figure 3, Any point on one of the curves has a nearby point on the other curve. Therefore, many of the measures presented so far (incl. Hausdorff, min, sum of mins) return a low distance. However, these curves are intuitively quite dissimilar: While they are close on a point-wise basis, they are not so close if we try to map the curves continuously to each other. A distance measure that captures this intuition is the *Fréchet* [Fréchet, 1906] distance.

The basic idea behind the Fréchet distance is encapsulated in the following example[13]: *Imagine two formula one racing cars. The first car, A, hurtles over a curve formulated by a first point set. The second car does the same over a curve formulated by the second point set. The first and second car will vary in velocity but they do not move backwards over their curves.*

13 Adapted from [Alt and Godau, 1995].

Then the Fréchet distance between the point sets is the minimum length of a non-stretchable cable that would be attached to both cars and would not break during the race.

In order to drive a formal definition of Fréchet distance, First we define *A curve* as a continuous mapping $f : [a, b] \to V$ with $a, b \in \mathbb{R}$, and $a < b$, where V *denote an arbitrary vector space*. A polygonal curve is $\mathbf{P} : [0, n] \to V$ with $n \in \mathbb{N}$, such that for all $i \in \{0, 1, ..., n-1\}$ each $P[i, i+1]$ is *affine*, i.e. $\mathbf{P}(i+\kappa) = (1-\kappa)\mathbf{P}(i) + \kappa\mathbf{P}(i+1)$ for all $\kappa \in [0, 1]$. n is called the length of $\mathbf{P}$. Then, Fréchet distance is formally defined as:

$$D_f(s,t) = \inf_{\substack{\alpha[0,1]\to[s_1,s_n] \\ \beta[0,1]\to[t_1,t_m]}} \left\{ \sup_{\tau\in[0,1]} \left\{ \delta(f(\alpha(\tau)) - g(\beta(\tau))) \right\} \right\}, \qquad (10)$$

where $f : [s_1, s_n] \to V$ and $g : [t_1, t_m] \to V$. α, β range over continuous and increasing functions with $\alpha(0) = s_1$, $\alpha(1) = s_n$, $\beta(0) = t_1$ and $\beta(1) = t_m$ only. Computing the Fréchet distance for our example returns 0.3km. See [Alt and Godau, 1995] for a complexity analysis of the Fréchet distance.

Overall, the distance measures presented above return partly very different values ranging from 0.3km to 184.74km even on our small example. In the following, we evaluate how well these measures can be used for LD.

EVALUATION

The goal of our evaluation was to answer the five questions mentioned in Section 4.2.1. To this end, we devised four series of experiments. First, we evaluated the use of different point-to-point geographical distance formulas together with the point set distance introduced in Section 4.3. Next, we evaluated the scalability of the ten measures with growing data set sizes. Then, we measured the robustness of these measures against measurement and granularity discrepancies as well as combinations of both. Finally, we measured the scalability of the measures when combined with the ORCHID algorithm (See Section 2.1.2).

Experimental Setup

In this section, we describe the experimental setup used throughout our experiments.

Datasets

We used three publicly available data sets for our experiments. The first data set, *NUTS*[14] was used as core data set for our scalability ex-

14 Version 0.91 available at http://nuts.geovocab.org/data/ is used in this work

periments. We chose this data set because it contains fine-granular descriptions of 1,461 geo-spatial resources located in Europe. For example, Norway is described by 1,981 points. The second data set, *DBpedia*[15], contains all the 731,922 entries from DBpedia that possess geometry entries. We chose DBpedia because it is commonly used in the Semantic Web community. Finally, the third data set, *LinkedGeoData*, contains all 3,836,119 geo-spatial objects from `http://linkgeodata.org` that are instances of the class `Way`.[16]. Further details to the data sets can be found in [Ngonga Ngomo, 2013].

Benchmark

To the best of our knowledge, there is no gold standard benchmark geographic data set that can be used to evaluate the robustness of geo-spatial distance measures. We thus adapted the benchmark generation approach proposed by Ferrara et al. [2011] to geo-spatial distance measures. In order to generate our benchmark data sets, we implemented two modifiers dubbed as *granularity* and *measurement error*. The implemented geo-spatial modifiers are analogous with the data sets generation algorithms from the field cartographic generalisation [Mackaness et al., 2011]. The granularity modifier implements the most commonly used *simplification* operator [McMaster, 1987], while the measurement error modifier is akin with the *displacement* operator [Nickerson and Freeman, 1986].

Both modifiers take a point set s and a threshold as input and return a point set s'. The *granularity modifier* M_g regards the threshold $\gamma \in [0, 1]$ as the probability that a point of s will be in the output point set s'. To ensure that an empty point set is never generated, the modifier always includes the first point of s into s'. For all other points $s_i \in s$, a random number r between 0 and 1 is generated. If $r \leqslant \gamma$, then s_i is added to s'. Else, s_i is discarded.

The *measurement error modifier* M_e emulates measurement errors across data sets. To this end, it alters the latitude and longitude of each points $s_i \in s$ by at most the threshold μ. Consequently, the new coordinates of a point s'_i are located within a square of size 2μ with s_i at the center. We used a sample of 200 points from each data set for our discrepancy experiments. To measure how well each of the distance measures performed w.r.t. to the modifiers, we first created a reference mapping $M = \{(s, s) \in S\}$ when given a set of input resources S. Then, we applied the modifier to all the elements of S to generate a target data set T. We then measured the distance between each of the point sets in the set T and the resources in S. For each element of S we stored the closest point $t \in T$ in a mapping M'. We

15 We used version 3.8 as available at `http://dbpedia.org/Datasets`.

16 We used the RelevantWays data set (version of April 26th, 2011) of *LinkedGeoData* as available at `http://linkedgeodata.org/Datasets`.

now computed the precision, recall and F-measure achieved within the experiment by comparing the pairs in M' with those in M.

Hardware

All experiments were carried out on a server running *OpenJDK* 64-Bit Server 1.6.0_27 on *Ubuntu* 14.04.2 LTS. The processors were 64-core *AuthenticAMD* clocked at 2.3 GHz. Unless stated otherwise, each experiment was assigned 8 GB of memory and was ran 5 times.

Point-to-Point Geographic Distance Evaluation

To evaluate the effect of the basic point-to-point geographic distance $\delta(s_i, t_j)$ in the point sets distance functions from Section 4.3, we carried out two sets of experiments. In the first set of experiments, we used the *orthodromic distance* (see Equation 1) as the basic point-to-point distance function $\delta(s_i, t_j)$, while in the second set of experiments we used the *great elliptic curve distance* [Bowring, 1984] to compute $\delta(s_i, t_j)$. As input we used a sample of 200 randomly picked resources from the three data sets of NUTS, DBpedia, and LinkedGeoData. We did not apply any modifiers in these two sets of experiments as we aimed to evaluate how the measures perform on real data.In each of the two sets of the experiments, we measured the precision, recall, F-measure and run time for each of the 10 point sets distance function.

The results (see Table 2) show that both the orthodromic and elliptic curve distances achieved the same precision, recall and F-measure when applied to the same resources. Moreover, the elliptic distance (in average) was 3.9 times slower than the orthodromic distance. Given that the great elliptic curve distance is known to be more accurate than the orthodromic distance [Chrisman and Girres, 2013], these observations emphasise that (1) the distance error of the orthodromic distance did not affect the LD results and that (2) the orthodromic distance has a lower time complexity than the great elliptic distance. Therefore, we rely on the orthodromic distance throughout the rest of experiments in this chapter.

Scalability Evaluation

To quantify how well the measures scale, we measured the runtime of the measures on fragments of growing size of each of the input data sets. This experiment emulates a naive deduplication on data sets of various sizes. The results achieved on NUTS are shown in Figure 4. We chose to show NUTS because it is the smallest and most fine-granular of our data sets. Thus, the measures achieved here represent an upper bound for the runtime behaviour of the different

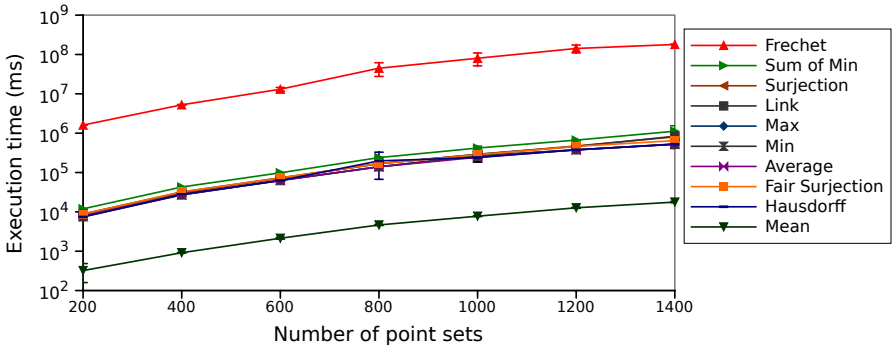


Figure 4: Scalability evaluation on the NUTS dataset.

approaches. D_{mean} is clearly the most time-efficient approach. This was to be expected as its algorithmic complexity is linear. While most of the other measures are similar in their efficiency, the Fréchet distance sticks out as the slowest to run. Overall, it is at least two orders of magnitude slower than the other measures. These results give a clear answer to question Q_1, which pertains to the time-efficiency of the measures at hand: D_{mean} is clearly the fastest.

Robustness Evaluation

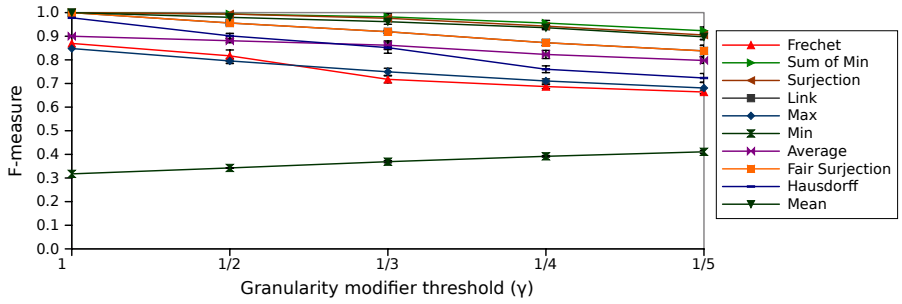
We carried out three types of evaluations to measure the robustness of the measures at hand. First, we measured their robustness against discrepancies in granularity. Then, we measured their robustness against measurement discrepancies. Finally, we combined discrepancies in measurement and granularity and evaluated all our measures against these. We chose to show only a portion of our results for the sake of space. All results can be found at `http://limes.sf.net`.

Robustness against Discrepancies in Granularity

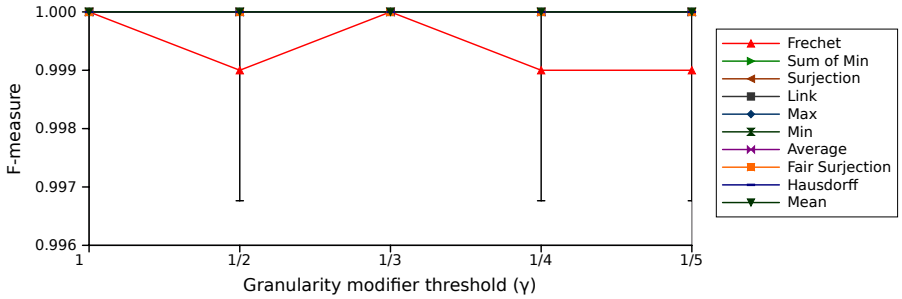
We measured the effect of changes in granularity on the measures at hand by using the five granularity thresholds 1, $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$ and $\frac{1}{5}$. Note that the threshold of 1 means that the data set was not altered. This setting allows us to answer Q_2, which pertains to the measures that are most adequate for deduplication. On NUTS (see Figure 5a), our results suggest that D_{min} is the least robust of the measures w.r.t. the F-measure. In addition to being the least time-efficient measure, Fréchet is also not robust against changes in granularity. The best performing measure w.r.t. to its F-measure is the *sum of minimums*, followed closely by the surjection and mean measures. On the DBpedia and LinkedGeoData data sets, all measures apart from the Fréchet distance perform in a similar fashion (see Figure 5b). This is yet simply due the sample of the data set containing point sets that were located far apart from each other. Thus, the answer to question Q_3

on the effect of discrepancies in granularity is that while the *sum of mins* is the least sensitive to changes in granularity. However, note that sum of mins is closely followed by the mean measure.

The answer to Q_2 can be derived from the evaluation with the granularity threshold set to 1. Here, mean, fair surjection, surjection, sum of mins and link perform best. Thus, mean should be used because it is more time-efficient.



(a) NUTS

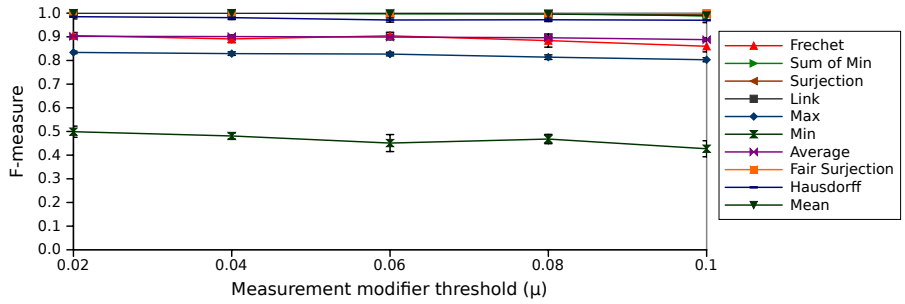


(b) LinkedGeoData

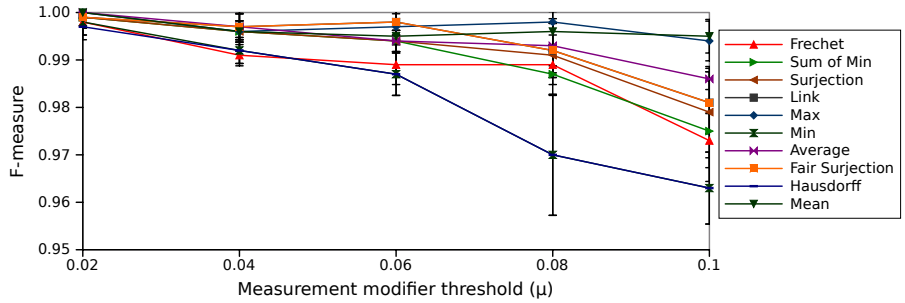
Figure 5: Comparison of different point set distance measures against granularity discrepancies.

Robustness against Measurement Discrepancies

The evaluation of the robustness of the measures at hand against discrepancies in measurement are shown in Figure 6. Interestingly, the results differ across the different data sets. On the NUTS data, where the regions are described with high granularity, five of the measures (mean, fair surjection, link, sum of mins and surjection) perform well. On LinkedGeoData, the number of points pro resources is considerably smaller. Moreover, the resources are partly far from each other. Here, the Hausdorff distance is the poorest while max and mean perform comparably well. Finally, on the DBpedia data set, all measures apart from Fréchet are comparable. Our results thus suggest that the answer to Q_4 is as follows: The mean distance is the distance of choice when computing links between geo-spatial data sets which contain measurement errors, especially if the resources described have a high geographical density or the difference in granularity is significant.



(a) NUTS



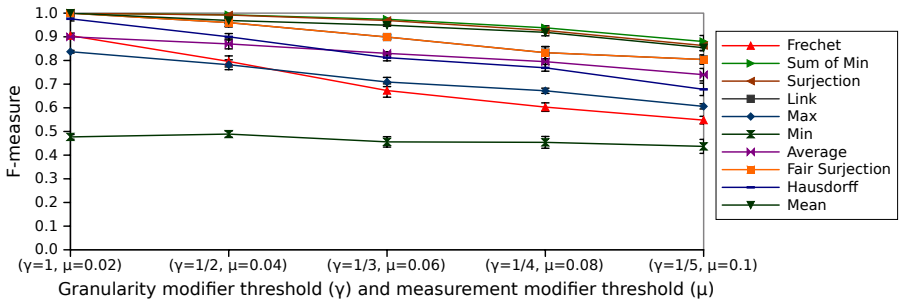
(b) LinkedGeoData

Figure 6: Comparison of different point set distance measures against measurement discrepancies.

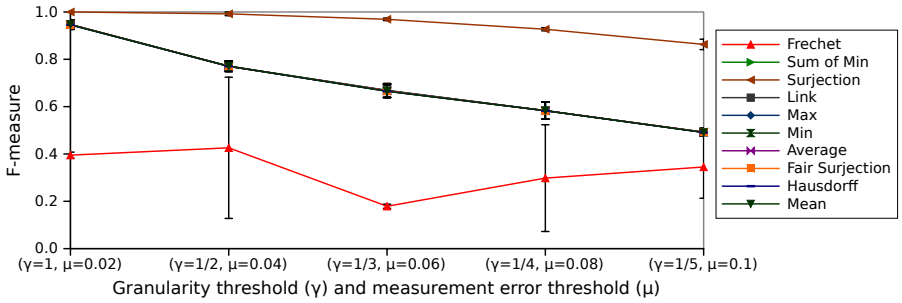
Overall Robustness

We emulated the differences across various real geographic data sets by combining the granularity and the measurement modifiers. Given a data set S, we generated a modified data set S′ using the granularity modifier. The modified data set was used as input for a measurement modifier, which generated our final data set T. The results of our experiments are shown in Figure 7. Again, the results vary across the different data sets. While mean performs well on NUTS Figure 7a and LinkedGeoData, it is surjection that outperforms all the other measures on DBpedia Figure 7b. This surprising result is due to the measurement errors having only a small effect on our DBpedia sample. Thus, after applying the granularity modifier, the surjection value is rarely affected.

Overall, our results suggest that the following answer to Q_5: In most cases, using the mean distance leads to high F-measures. Moreover, mean present the advantage of being an order of magnitude faster than the other approaches. Still, the surjection measure should also be considered when comparing different data sets as it can significantly outperform the mean measure



(a) NUTS



(b) DBpedia

Figure 7: Comparison of different point set distance measures against granularity and measurement discrepancies.

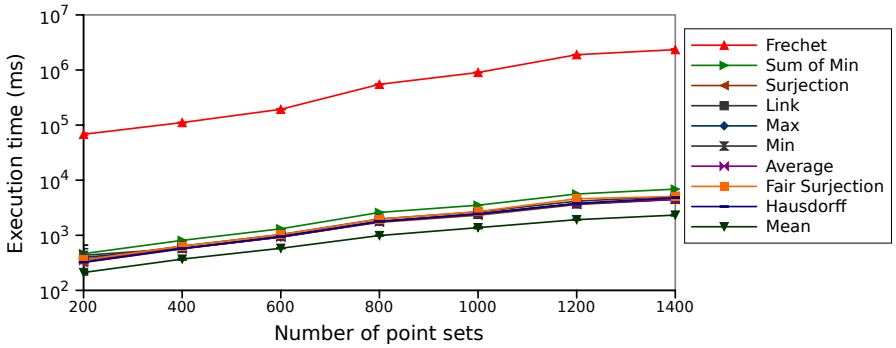
Scalability with ORCHID

We aimed to know how far the runtime of measures such as mean, surjection and sum of mins can be reduced so as to ensure that these measures can be used on large data sets. We thus combined these measures with the ORCHID approach introduced in Section 2.1.2. The idea behind ORCHID is to improve the runtime of algorithms for measuring geo-spatial distance measures by adapting an approach akin to divide-and-conquer. ORCHID assumes that it is given a distance measure (not necessarily a metric) m that abides by $m(s,t) \leqslant \theta \rightarrow \forall s_i \in s\ \exists t_j \in t : \delta(s_i,t_j) \leqslant \theta$. This condition is obviously not satisfied by all measures considered herein, including min and mean. However, dedicated extensions of ORCHID can be developed for these measures. Overall, ORCHID begins by partitioning the surface of the planet. The points in a given partition are then only compared with points in partitions that abide by the distance threshold underlying the computation.

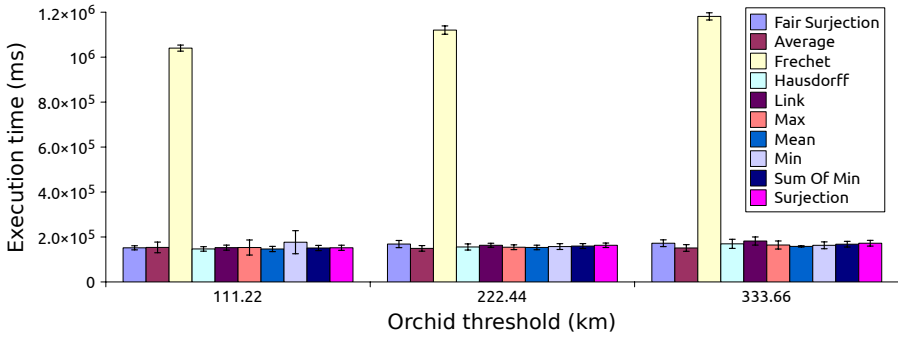
We used the default settings of the implementation provided in the LIMES framework and the distance threshold of $0.02°$ (2.2km). Figure 8a shows the runtime results achieved on the same data sets as Figure 4. Clearly, the runtimes of the approaches can be decreased by up to an order of magnitude. Therewith, ORCHID allows most measures (i.e., all apart from Fréchet) to scale in a manner comparable to that of the mean measure. Therewith, the measures can now be used

on the whole of the data sets at hand. For example, all distance measures apart from the Fréchet distance require less than five minutes to run on the whole of the DBpedia data set (see Figure 8b).

Overall, we can conclude that all measures apart from the Fréchet distance are amenable to being used for LD. While *mean performs best overall*, *surjection-based and minimum-based measures are good candidates* to use if mean returns unsatisfactory results. The Fréchet distance on the other hand seems inadequate for LD. This can yet be due to the point set approach chosen in this chapter. An analysis of the Fréchet distance on the description of resources as polygons remains future work. Note that the high Fréchet distances computed when minor discrepancies between representations of geo-spatial objects occurred can be of importance when carrying out other tasks such as analyzing the quality of RDF datasets.



(a) NUTS



(b) DBpedia

Figure 8: Scalability evaluation with ORCHID.

Experiment on Real Datasets

We were interested in knowing whether the mean function performs well on real data. Validating LD results on geo-spatial data is difficult due to the lack of reference data sets. We thus measured the increase in precision and recall achieved by using geo-spatial information by sampling 100 links from the results of real LD tasks and evaluating

these links manually. The links were evaluated by the authors who reached an agreement of 100%.

In the first experiment, we computed links between cities in DBpedia and LinkedGeoData by comparing solely their labels by means of an exact match string similarity. No geo-spatial similarity metric was used, leading to cities being linked if they have exactly the same name. Overall only 74% of the links in our sample were correct. The remaining 26% differed in country or even continent. We can assume that a recall of 1 would be achieved by using this approach as a particular city will most probably have the same name across different geo-spatial data sets. Thus, in the best case, linking geo-spatial resources in DBpedia to LinkedGeoData would only lead to an F-measure of 0.85.

In our second experiment, we extended the specification described above by linking two cities if their names were exact matches (which was used in the first experiment) and the mean distance function between their geometry representation returned a value under 100km. In our sample, we achieved a perfect accuracy and thus an F-measure of 1. While this experiment is small, it clearly demonstrates the importance of using geo-spatial information for linking geo-spatial resources. Moreoveor, it suggest that the mean distance is indeed reliable on real data. More experiments yet need to be carried out to ensure that the empirical results we got in this experiment are not just a mere artifact in the data. We will achieve this goal by creating a benchmark for geo-spatial LD in future work.

Table 2: Comparison of the orthodromic and great elliptic distances using 200 randomly selected resources from each data set, where precision (P), recall (R), F-measure (F) and run time (T) are presented. Note that all run times are in milliseconds.

Dataset	Measure	Orthodromic Distance				Elliptic Distance			
		P	R	F	T	P	R	F	T
NUTS	Min	0.19	1.00	0.32	1806	0.19	1.00	0.32	7506
	Max	0.85	0.85	0.85	1696	0.85	0.85	0.85	7448
	Average	0.90	0.90	0.90	1676	0.90	0.90	0.90	7468
	Sum of Min	1.00	1.00	1.00	3421	1.00	1.00	1.00	15035
	Link	1.00	1.00	1.00	2357	1.00	1.00	1.00	8878
	Surjection	1.00	1.00	1.00	2066	1.00	1.00	1.00	8666
	Fair Surjection	1.00	1.00	1.00	2253	1.00	1.00	1.00	8879
	Hausdorff	0.96	1.00	0.98	1719	0.96	1.00	0.98	7524
	Mean	1.00	1.00	1.00	185	1.00	1.00	1.00	250
	Frechet	1.00	1.00	1.00	1311	1.00	1.00	1.00	3652
DBpedia	Min	1.00	1.00	1.00	122	1.00	1.00	1.00	108
	Max	1.00	1.00	1.00	64	1.00	1.00	1.00	102
	Average	1.00	1.00	1.00	46	1.00	1.00	1.00	100
	Sum of Min	1.00	1.00	1.00	46	1.00	1.00	1.00	159
	Link	1.00	1.00	1.00	146	1.00	1.00	1.00	140
	Surjection	1.00	1.00	1.00	124	1.00	1.00	1.00	246
	Fair Surjection	1.00	1.00	1.00	107	1.00	1.00	1.00	153
	Hausdorff	1.00	1.00	1.00	40	1.00	1.00	1.00	87
	Mean	1.00	1.00	1.00	84	1.00	1.00	1.00	77
	Frechet	1.00	1.00	1.00	110	1.00	1.00	1.00	286
LinkedGeoData	Min	1.00	1.00	1.00	1175	1.00	1.00	1.00	4554
	Max	1.00	1.00	1.00	1113	1.00	1.00	1.00	4483
	Average	1.00	1.00	1.00	1079	1.00	1.00	1.00	4480
	Sum of Min	1.00	1.00	1.00	2180	1.00	1.00	1.00	8999
	Link	1.00	1.00	1.00	1552	1.00	1.00	1.00	5603
	Surjection	1.00	1.00	1.00	1397	1.00	1.00	1.00	5406
	Fair Surjection	1.00	1.00	1.00	1472	1.00	1.00	1.00	5491
	Hausdorff	1.00	1.00	1.00	1107	1.00	1.00	1.00	4510
	Mean	1.00	1.00	1.00	101	1.00	1.00	1.00	244
	Frechet	1.00	1.00	1.00	1201	1.00	1.00	1.00	4493

5

COLIBRI– UNSUPERVISED LINK DISCOVERY THROUGH KNOWLEDGE BASE REPAIR

This chapter we present Colibri, *an iterative unsupervised approach for* LD *in knowledge bases with erroneous or missing data. A paper about the approach is published in ESWC'14 [Ngonga Ngomo et al., 2014]. The whole evaluation of* Colibri *was carried out by author, who also co-implemented the algorithm and co-wrote the paper.*

In the previous chapter, we considered geospatial distance functions for Link Discovery (LD). In this Chapter, we analyse LD across more than two knowledge bases. For example, imagine being given three knowledge bases K_1 that contains cities, K_2 that contains provinces and K_3 that contains countries as well as the `dbo:locatedIn` predicate[1] as relation. The specification that links K_1 to K_2 might compare province labels while the specifications that link K_1 and K_2 to K_3 might compare country labels. Imagine the city `Leipzig` in K_1 were linked to `Saxony` in K_2 and to `Germany` in K_3. In addition, imagine that `Saxony` were erroneously linked to `Prussia`. If we assume the first Linked Data principle (i.e., *"Use URIs as names for things"*)[2], then the following holds: By virtue of the transitivity of `dbo:locatedIn` and of knowing that it is a many-to-1 relation,[3] we can deduce that one of the links in this constellation must be wrong. Note that this inference would hold both under open- and closed-world assumptions. Thus, if we knew the links between `Leipzig` and `Germany` as well as `Leipzig` and `Saxony` to be right, we could then repair the value of the properties of `Saxony` that led it to be linked to `Prussia` instead of `Germany` and therewith ensure that is linked correctly in subsequent LD processes.

We implement this intuition by presenting Colibri, a novel iterative and unsupervised approach for LD. Colibri uses LD results for *transitive many-to-1 relations* (e.g., `locatedIn` and `descendantSpeciesOf`) and *transitive 1-to-1 relations* (e.g., `owl:sameAs`) between instances in knowledge bases for the sake fo attempting to repair the instance knowledge in these knowledge bases and improve the overall quality of the links. In contrast to most of the current unsupervised LD approaches, Colibri takes an n-set[4] of set of resources $K_1, \ldots, K_n$ with $n \geqslant 2$ as input. In a *first step*, our approach applies an *unsupervised machine-learning* approach to each pair (K_i, K_j) of sets of resources (with $i \neq j$). By these means, Colibri generates $n(n-1)$ mappings. Current unsupervised approaches for LD would terminate after this step and would not make use of the information contained in some mappings to improve other mappings. The intuition behind Colibri

1 The prefix `dbo:` stands for `http://dbpedia.org/ontology/`.

2 http://www.w3.org/DesignIssues/LinkedData.html

3 From this characteristic, we can infer that (1) a city cannot be located in two different provinces, (2) a city cannot be located in two different countries and (3) a province cannot be located in two different countries.

4 An n-set is a set of magnitude n.

is that using such information can help improve the overall accuracy of a LD process if the links are *many-to-1 and transitive* or *1-to-1 and transitive*. To implement this insight, all mappings resulting from the first step are forwarded to a *voting approach* in a *second step*. The goal of the voting approach is to detect possible errors within the mappings that were computed in the previous step (e.g., missing links). This information is subsequently used in the *third step* of COLIBRI, which is the *repair step*. Here, COLIBRI first detects the sources of errors in the mappings. These sources of errors can be wrong or missing property values of the instances. Once these sources of errors have been eliminated, a new iteration is started. COLIBRI iterates until a termination condition (e.g., a fixpoint of its objective function) is met.

Overall, the main contributions of this work are as follows:

- We present the (to the best of our knowledge) the first unsupervised LD approach that attempts to repair instance data for improving the LD process.
- Our approach is the first unsupervised LD approach that can be applied to $n \geqslant 2$ knowledge bases and which makes use of the intrinsic topology of the Web of Data.
- We evaluate our approach on six data sets. Our evaluation shows that we can improve the results of state-of-the-art approaches w.r.t. the F-measure while reliably detecting and correcting errors in instance data.

We rely on EUCLID [Ngonga Ngomo and Lyko, 2013] as machine-learning approach and thus provide a fully deterministic approach. We chose EUCLID because it performs as well as non-deterministic approaches on the data sets used in our evaluation [Ngonga Ngomo and Lyko, 2013] while presenting the obvious advantage of always returning the same result for a given input and a given setting. Moreover, it is not tuned towards discovery exclusively `owl:sameAs` links [Suchanek et al., 2011]. Still, COLIBRI is independent of EUCLID and can be combined with any link specification learning approach. The approaches presented herein were implemented in LIMES.[5]

NOTATION

In this section, we present some of the notation and concepts necessary to understand the rest of the chapter. We use Figure 9 to exemplify our notation. The formalization of LD provided below is an extension of the formalization for two input knowledge bases first introduced in Section 2.1. Given n knowledge bases $K_1 \dots K_n$, LD aims to discover pairs $(s_i, s_j) \in K_i \times K_j$ that are such that a given relation R holds between s_i and s_j. The direct computation of the pairs for

5 http://limes.sf.net

which R holds is commonly very tedious if at all possible. Thus, most frameworks for LD resort to approximating the set of pairs for which R holds by using Link Specifications (LS). A LS can be regarded as a classifier C_{ij} that maps each element of the Cartesian product $K_i \times K_j$ to one of the classes of $Y = \{+1, -1\}$, where K_i is called the *set of source instances* while K_j is the *set of target instances*. $(s,t) \in K_i \times K_j$ is considered by C_{ij} to be a correct link when $C_{ij}(s,t) = +1$. Otherwise, (s,t) is considered not to be a potential link. In our example, C_{12} returns $+1$ for $s =$ `ex1:JohnDoe` and $t =$ `ex2:JD`.

We will assume that the classifier C_{ij} relies on comparing the value of complex similarity function $\sigma_{ij} : K_i \times K_j \to [0,1]$ with a threshold θ_{ij}. If $\sigma_{ij}(s,t) \geqslant \theta_{ij}$, then the classifier returns $+1$ for the pair (s,t). In all other cases, it returns -1. The complex similarity function σ_{ij} consists of a combination of atomic similarity measures $\pi^l_{ij} : K_i \times K_j \to [0,1]$. These atomic measures compare the value of a particular property of $s \in K_i$ (for example its `rdfs:label`) with the value of a particular property of $t \in K_j$ (for example its `:name`) and return a similarity score between 0 and 1. In our example, σ_{12} relies on the single atomic similarity function $trigrams($`:ssn`, `:ssn`$)$, which compares the social security number attributed to resources of K_1 and K_2.

We call the set of all pairs $(s,t) \in K_i \times K_j$ that are considered to be valid links by C_{ij} a *mapping*. We will assume that the resources in each of the knowledge bases $K_1, \ldots, K_n$ can be ordered (e.g., by using the lexical ordering of their URI) and thus assigned an index. Then, a mapping between the knowledge bases K_i and K_j can be represented as a matrix M_{ij} of dimensions $|K_i| \times |K_j|$, where the entry in the x^{th} row and y^{th} column is denoted $M_{ij}(x,y)$. If the classifier maps (s,t) to -1, then $M_{ij}(x,y) = 0$ (where x is the index of s and y is the index of t). In all other cases, $M_{ij}(x,y) = \sigma(s,t)$. For the sake of understandability, we will sometimes write $M_{ij}(s_x, t_y)$ to signify $M_{ij}(x,y)$. In our example, C_{34} is a linear classifier, $\sigma_{34} = trigrams($`:id`, `:id`$)$ and $\theta_{34} = 1$. Thus, (`ex3:J36`, `ex4:Cat40_1`) is considered a link.

Supervised approaches to the computation of link specifications use labelled training data $L \subseteq K_i \times K_j \times Y$ to minimize the error rate of C_{ij}. COLIBRI relies on an unsupervised approach. The idea behind *unsupervised approaches* to learning link specifications is to refrain from using any training data (i.e., $L = \emptyset$). Instead, unsupervised approaches aim to optimize an *objective function*. The objective functions we consider herein approximate the value of the F-measure achieved by a specification and are thus Pseudo-F-Measures (PFM) [Nikolov et al., 2012].

In this work, we extend the PFM definition presented in [Ngonga Ngomo and Lyko, 2013]. Like in [Nikolov et al., 2012; Suchanek et al., 2011; Hassanzadeh et al., 2013], the basic assumption behind this PFM is that one-to-one links exist between the resources in S and T. We chose to extend this measure to ensure that it is symmetrical w.r.t.

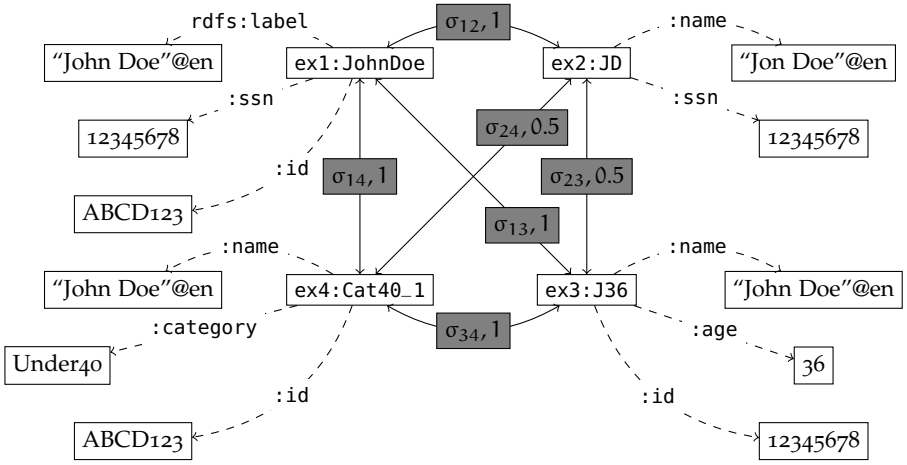


Figure 9: Example of four linked resources from four different knowledge bases. The white nodes are resources or literals. Properties are represented by dashed labeled arrows. Links are represented by plain arrows. The gray boxes on the links show the names of the similarity measures used to link the resources they connect as well as the similarity value for each of these resource pairs. $\sigma_{12} = \text{trigrams}(\texttt{:ssn}, \texttt{:ssn})$, $\sigma_{13} = \sigma_{14} = \text{trigrams}(\texttt{:id}, \texttt{:id})$, $\sigma_{23} = \sigma_{24} = \sigma_{34} = \text{dice}(\texttt{:name}, \texttt{:name})$, $\sigma_{ij} = \sigma_{ji}$.

to the source and target data sets, i.e., PFM(S, T) = PFM(T, S). Our pseudo-precision $\mathcal{P}$ computes the fraction of links that stand for one-to-one links and is equivalent to the strength function presented in [Hassanzadeh et al., 2013]. Let $\text{links}(K_i, M_{ij})$ be the subset of K_i whose elements are linked to at least one element of K_j. Then,

$$\mathcal{P}(M_{ij}) = \frac{|\text{links}(K_i, M_{ij})| + |\text{links}(K_j, M_{ij})|}{2|M_{ij}|}. \tag{11}$$

The pseudo-recall R computed the fraction of the total number of resources (i.e., $|K_i| + |K_j|$) from that are involved in at least one link:

$$R(M_{ij}) = \frac{|\text{links}(K_i, M_{ij})| + |\text{links}(K_j, M_{ij})|}{|K_i| + |K_j|}. \tag{12}$$

Finally, the PFM $\mathcal{F}_\beta$, is defined as

$$\mathcal{F}_\beta = (1 + \beta^2)\frac{\mathcal{P}R}{\beta^2\mathcal{P} + R}. \tag{13}$$

For the example in Figure 10, $\mathcal{P}(M_{12}) = 1$, $R(M_{12}) = \frac{2}{3}$ and $\mathcal{F}_1 = \frac{4}{5}$. Our PFM works best if S and T are of comparable size and one-to-one links are to be detected. For example, Euclid achieves 99.7% F-measure on the OAEI Persons1 data set.[6] It even reaches 97.7% F-measure on the DBLP-ACM data set, therewith outperforming the best supervised approach (FEBRL) reported in [Köpcke et al.,

6 http://oaei.ontologymatching.org/

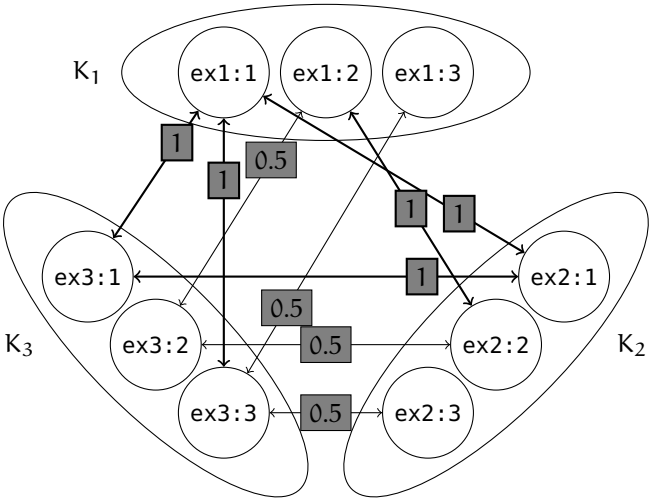


Figure 10: Example of mappings between 3 sets of resources. K_1 has the namespace ex1, K_2 the namespace ex2 and K_3 the namespace ex3. Thick lines stand for links with the similarity value 1 while thin lines stand for links with the similarity value 0.5.

2010]. Yet, EUCLID achieves worse results compared to FEBRL on the Amazon-Google Products data set with an F-measure of 43% against 53.8%, where $|T| \approx 3|S|$.

THE COLIBRI APPROACH

In this section, we present the COLIBRI approach and its components in detail. We begin by giving an overview of the approach. Then, for the sake of completeness, we briefly present EUCLID, the unsupervised LD approach currently underlying COLIBRI. For more information about EUCLID, please see [Ngonga Ngomo and Lyko, 2013]. Note that COLIBRI can be combined with any unsupervised LD approach. After the overview of EUCLID, we present the voting approach with which COLIBRI attempts to detect erroneous or missing links. In a final step, we present how COLIBRI attempts to repair these sources of error.

Overview

Most of the state-of-the-art approaches to LD assume scenarios where two sets of resources are to be linked. COLIBRI assumes that it is given n sets of resources $K_1, \ldots, K_n$. The approach begins by computing mappings M_{ij} between resources of pairs of sets of resources (K_i, K_j). To achieve this goal, it employs the EUCLID algorithm [Ngonga Ngomo and Lyko, 2013] described in the subsequent section. The approach then makes use of the transitivity of R by computing voting matrices V_{ij} that allow detecting erroneous as well as missing links. This infor-

mation is finally used to detect resources that should be repaired. An overview of COLIBRI is given in Algorithm 1. In the following sections, we explain each step of the approach.

Algorithm 1: The COLIBRI Approach.

Input: $\mathcal{M}$: the set of all M_{ij}; $\tilde{\mathcal{V}}$: the set of all $\tilde{V}_{ij}$; maxIterations to ensures that the approach terminates;

```
1  F_new ← 0, F_old ← 0, iterations ← 0;
2  while F_new − F_old > 0 and iterations < maxIterations do
3      F_old ← F_new;
4      F_new ← 0;
5      for i ∈ {1, ..., n} do
6          for j ∈ {1, ..., n}, j ≠ i do
7              M_ij ← EUCLID (K_i, K_j);
8              F_new ← F_new+PSEUDOF(M_ij);
9              F_new ← F_new/(n(n − 1));
10     if F_new − F_old > 0 then
11         for i ∈ {1, ..., n} do
12             for j ∈ {1, ..., n}, j ≠ i do
13                 V_ij ←COMPUTEVOTING(M_ij, M);
14                 Ṽ_ij ←POSTPROCESS(V_ij);
15         for (a, b) ∈ GETWORSTLINKS(Ṽ) do
16             (r_a, r_b) ←GETREASON(a, b);
17             REPAIR(r_a, r_b)
18     iterations ← iterations + 1;
```

EUCLID

Over the last years, non-deterministic approaches have been commonly used to detect highly accurate link specifications (e.g., [Ngonga Ngomo et al., 2013b; Nikolov et al., 2012]). EUCLID (Line 7 of Algorithm 1) is a deterministic unsupervised approach for learning link specifications. The core idea underlying the approach is that link specifications of a given type (linear, conjunctive, disjunctive) can be regarded as points in a link specification space. Finding an accurate link specification is thus equivalent to searching through portions of this specification space. In the following, we will assume that EUCLID tries to learn a conjunctive classifier, i.e., a classifier which returns $+1$ for a pair $(s,t) \in K_i \times K_j$ when $\bigwedge_{l=1}^{m} (\pi_{ij}^l(s,t) \geqslant \theta_{ij}^l)$ holds. The same approach can be used to detect disjunctive and linear classifiers. EUCLID assumes that it is given a set of m atomic similarity functions π_{ij}^l with which it can compare $(s,t) \in K_i \times K_j$. The atomic functions π_{ij}^l

build the basis of an m-dimensional space where each of the dimensions corresponds to exactly one of the π^l_{ij}. In this space, the specification $\bigwedge_{l=1}^{m}(\pi^l_{ij}(s,t) \geqslant \theta^l_{ij})$ has the coordinates $(\theta^1_{ij}, \ldots, \theta^m_{ij})$. The core of EUCLID consists of a hierarchical grid search approach that aims to detect a link specification within a hypercube (short: cube) which maximizes the value of a given objective function $\mathcal{F}$. The hypercubes considered by EUCLID are such that their sides are all orthogonal to the axes of the space. Note that such a hypercube can be described entirely by two points $b = (b_1, \ldots, b_m)$ and $B = (B_1, \ldots, B_m)$ with $\forall i \in \{1, \ldots, m\}(b_i \leqslant B_i)$.

EUCLID begins by searching through the cube defined by $b = \underbrace{(0, \ldots, 0)}_{m \text{ times}}$ and $B = \underbrace{(1, \ldots, 1)}_{m \text{ times}}$ (i.e., the whole of the similarity space). A point w with coordinates $(w_1, \ldots, w_m)$ corresponds to the classifier with the specific function $\bigwedge_{l=1}^{m}(\pi^l_{ij}(s_i, s_j) \geqslant w_l)$. Let $\alpha \in \mathbb{N}, \alpha \geqslant 2$ be the granularity parameter of EUCLID. The search is carried out by generating a grid of $(\alpha+1)^m$ points g whose coordinates $g_i = \left(b_i + k_i \frac{(B_i - b_i)}{\alpha}\right)$, where $k_i \in \{0, \ldots, \alpha\}$. We call $\Delta_i = \frac{(B_i - b_i)}{\alpha}$ the *width* of the grid in the ith dimension. EUCLID now computes the pseudo-F-measure $\mathcal{F}$ of the specification corresponding to each point on the grid. Let g^{max} be a point that maximizes $\mathcal{F}$. Then, EUCLID updates the search cube by updating the coordinates of the points b and B as follows: $b_i = \left(\max\left\{0, g^{max}_i - \Delta_i\right\}\right)$ and $B_i = \left(\min\left\{1, g^{max}_i + \Delta_i\right\}\right)$. Therewith, EUCLID defines a new and smaller search cube. The search is iterated until a stopping condition such as a given number of iterations is met.

Voting

The result of EUCLID is a set of $n(n-1)$ mappings M_{ij} which link the resource set K_i with the resource set K_j. The goal of the second step of a COLIBRI iteration is to determine the set of resources that might contain incomplete or erroneous information based on these mappings. The basic intuition behind the approach is to exploit the transitivity of the relation R is as follows: If the link $(s,t) \in K_i \times K_j$ is correct, then for all k with $1 \leqslant k \leqslant n$ with $k \neq i, j$, there should exist pairs of links (s,z) and (z,t) with $M_{ik}(s,z) > 0$ and $M_{kj}(z,t) > 0$. Should such pairs not exist or be weakly connected, then we can assume that some form of error was discovered.

Formally, we go about implementing this intuition as follows: We first define the voting matrices V_{ij} as

$$V_{ij} = \frac{1}{n}\left(M_{ij} + \sum_{k=0, k\neq i,j}^{n} M_{ik}M_{kj}\right) \text{ (Line 13 of Algorithm 1).}$$

In the example shown in Figure 10, the mappings are

$$M_{12} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, M_{13} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix} \text{ and } M_{23} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}.$$

The corresponding voting matrices are thus

$$V_{12} = \begin{pmatrix} 1 & 0 & 0.25 \\ 0 & 0.625 & 0 \\ 0 & 0 & 0.125 \end{pmatrix}, V_{13} = \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.25 \end{pmatrix} \text{ and } V_{23} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.25 \end{pmatrix}.$$

Each voting matrix V_{ij} encompasses the cumulative results of the linking between all pairs of resource sets with respect to the resources in (K_i, K_j). Computing V_{ij} as given above can lead to an explosion in the number of resources associated to s_i. In our example, the erroneous link between `ex1:1` and `ex3:3` leads to `ex1:1` being linked not only to `ex2:1` but also to `ex2:3` in V_{12}. We thus post-process each V_{ij} by only considering the best match for each $s \in K_i$ within V_{ij}, i.e., by removing each non-maximal entry from each row of V_{ij} (Line 14 of Algorithm 1). We label the resulting matrix $\tilde{V}_{ij}$. For our example, we get the following matrices:

$$\tilde{V}_{12} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.625 & 0 \\ 0 & 0 & 0.125 \end{pmatrix}, \tilde{V}_{13} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.25 \end{pmatrix} \text{ and } \tilde{V}_{23} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.25 \end{pmatrix}.$$

Colibri now assumes that the links encoded in $\tilde{V}_{ij}$ are most probably correct. All entries of $\tilde{V}_{ij}$ being 1 are thus interpreted as all matrices agreeing on how to link the resources in (K_i, K_j). In the example in Figure 10, this is the case for $\tilde{V}_{12}$(`ex1:1`, `ex2:1`). Should this not be the case, then the disagreement between the matrices can result from the following reasons:

1. *Missing links*: This is the case in our example for the link (`ex1:3`, `ex2:3`) which is not contained in M_{12}. For this reason, $\tilde{V}_{12}$(`ex1:3`, `ex2:3`) is minimal.

2. *Weak links*: This is the case for the second-lowest entry in $\tilde{V}_{12}$, where the entry for (`ex1:2`, `ex2:2`) is due to M_{13}(`ex1:2`, `ex3:2`) and M_{32}(`ex3:2`, `ex2:2`) being 0.5.

Colibri now makes use of such disagreements to repair the entries in the knowledge bases with the aim of achieving a better linking. To this end, it selects a predetermined number of links (a, b) over all $\tilde{V}_{ij}$ whose weight is minimal and smaller than 1 (getWorstLinks in Algorithm 1). These links are forwarded to the instance repair.

Instance Repair

For each of the links (a, b) selected by the voting approach, the instance repair routine of COLIBRI begins by computing why $\tilde{V}_{ij}(a, b) < 1$. To achieve this goal, COLIBRI computes the *reason*(r_a, r_b) as:

$$\text{reason}(r_a, r_b) \in \left(K_i \times \bigcup_{k=1, k\neq i}^{n} K_k \right) \cup \left(\bigcup_{k=1, k\neq j}^{n} K_k \times K_j \right)$$

by detecting the smallest entry that went into computing $\tilde{V}_{ij}(a, b)$. Three possibilities occur:

1. $(r_a, r_b) \in K_i \times K_j$: In this case, the weak or missing link is due to the initial mapping M_{ij}.
2. $(r_a, r_b) \in K_i \times K_k$ with $k \neq i \wedge k \neq j$: In this case, the weak or missing link is due to the in-between mapping M_{ik}.
3. $(r_a, r_b) \in K_k \times K_j$ with $k \neq i \wedge k \neq j$: Similarly to the second case, the weak or missing link is due to the in-between mapping M_{kj}.

In all three cases, the repair approach now aims to improve the link by repairing the resource rs or rt that most probably contains erroneous or missing information. To achieve this goal, it makes use of the similarity measure σ used to generate (r_a, r_b). The value of this measure being low suggests that the property values p^l and q^l used across the similarity measures π^l are dissimilar. The idea of the repair is then to overwrite exclusively the values of $p^l(rs)$ with those of $q^l(rt)$ or vice-versa. The intuition behind deciding upon whether to update rs or rt is based on the *average similarity* $\bar{\sigma}(rs)$ resp. $\bar{\sigma}(rt)$ of the resources rs and rt to other resources. For a resource $s \in K_i$, this value is given by

$$\bar{\sigma}(s) = \frac{1}{n-1} \left(\sum_{k=1, k\neq i}^{n} \max_{t \in K_k} \sigma_{ik}(s, t) \right). \tag{14}$$

Here, the assumption is that the higher the value of $\bar{\sigma}$ for a given resource, the higher the probability that it does not contain erroneous information.

Let us consider a new the example given in Figure 10 and assume that the link that is to be repaired is $(\texttt{ex1:2}, \texttt{ex2:2})$. One reason for this link would be $rs = \texttt{ex1:2}$ and $rt = \texttt{ex3:2}$. Now $\bar{\sigma}(\texttt{ex1:2}) = 0.75$ while $\bar{\sigma}(\texttt{ex3:2}) = 0.5$. COLIBRI would thus choose to overwrite the values of `ex3:2` with those of `ex1:2`.

The overwriting in itself is carried out by overwriting the values of $q^l(rt)$ with those of $p^l(rs)$ if $\bar{\sigma}(rs) \geqslant \bar{\sigma}(rt)$ and vice-versa. This

step terminates an iteration of COLIBRI, which iterates until a termination condition is reached, such as the average value of $\mathcal{F}$ for the mappings generated by EUCLID declining or a maximal number of iterations. The overall complexity of each iteration of COLIBRI is $O(n^2 \times E)$, where E is the complexity of the unsupervised learning algorithm employed to generate the mappings. Thank to the algorithms implemented in LIMES which have a complexity close to $O(m)$ where $m = \max\{|S|, |T|\}$ for each predicate, EUCLID has a complexity of $O(pm)$, where p is the number of predicates used to compare entities. Consequently, the overall complexity of each iteration of COLIBRI is $O(pmn^2)$ when it relies on EUCLID. While we observed a quick converge of the approach on real and synthetic data sets within our evaluation (maximally 10 iterations), the convergence speed of the approach may vary on the data sets used.

EVALUATION

The aim of our evaluation was to measure whether COLIBRI can improve the F-measure of mappings generated by unsupervised LD approaches. To this end, we measured the increase in F-measure achieved by COLIBRI w.r.t to the number of iterations it carried out on a synthetic data set generated out of both synthetic and real data. To the best of our knowledge, no benchmark data set is currently available for LD across $n > 2$ knowledge bases. We thus followed the benchmark generation approach for instance matching presented in [Ferrara et al., 2011] to generate the evaluation data for COLIBRI.

Experimental Setup

We performed controlled experiments on data generated automatically from two synthetic and three real data sets. The synthetic data sets consisted of the *Persons1* and *Restaurant* data sets from the benchmark data sets of *OAEI2010*.[7] The real data sets consisted of the *ACM-DBLP*, *Amazon-Google* and *Abt-Buy* data sets.[8] We ran all experiments in this section on the source data set of each of these benchmark data sets (e.g., *ACM* for *ACM-DBLP*). We omitted *OAEI2010*'s *Person2* because its source data set is similar to *Person1*'s. Given the lack of benchmark data for LD over several sources, we generated a synthetic benchmark as follows: Given the initial source data set K_1, we first generated $n-1$ copies of K_1. Each copy was altered by using a subset of the operators suggested in [Ferrara et al., 2011]. The alteration strategy consisted of randomly choosing a property of a randomly chosen resource and altering it. We implemented three syntactic operators to

7 Available online at http://oaei.ontologymatching.org/2010/.
8 Available online at http://dbs.uni-leipzig.de/en/research/projects/object_matching/fever/benchmark_datasets_for_entity_resolution.

alter property values, i.e., misspellings, abbreviations and word permutations. The syntactic operator used for altering a resource was chosen randomly. We call the probability of a resource being chosen for alteration the *alteration probability* (ap). The goal of this series of experiments was to quantify (1) the gain in F-measure achieved by COLIBRI over EUCLID and (2) the influence of ap and of the number n of knowledge bases on COLIBRI's F-measure.

The F-measure of EUCLID and COLIBRI was the average F-measure they achieved over all pair (K_i, K_j) with $i \neq j$. To quantify the amount of resources that were altered by COLIBRI in the knowledge bases $K_1, \ldots, K_n$, we computed the average *error rate* in the knowledge bases after each iteration as follows:

$$errorrate = 1 - \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} \frac{2|K_i \cap K_j|}{|K_i| + |K_j|}. \tag{15}$$

The maximal number of COLIBRI iterations was set to 10. We present the average results but omit the standard deviations for the sake of legibility. For precision, the standard deviation was maximally 4%. The recall's standard deviation never exceeded 1% while it reached 2% for the F-measure.

Experimental Results

We varied the number of knowledge bases between 3 and 5. Moreover, we varied the alteration probability between 10% and 50% with 10% increments. We then measured the precision, recall, F-measure, runtime and number of repairs achieved by the batch version of COLIBRI over several iterations. We present portions of the results we obtained in Figure 11 and Table 3.[9] Table 3 shows an overview of the results we obtained across the different data sets. Our results show clearly that COLIBRI can improve the results of EUCLID significantly on all data sets. On the Restaurant data set for example, COLIBRI is 6% better than EUCLID on average. On ACM, the average value lies by 4.8%. In the best case, COLIBRI improves the results of EUCLID from 0.85 to 0.99 (*Amazon*, ap = 50%, KBs = 4). Moreover, COLIBRI never worsens the results of EUCLID. This result is of central importance as it suggests that our approach can be used across the Linked Data Web for any combination of number of knowledge and error rates within the knowledge bases.

The results achieved on the Restaurant data set are presented in more detail in Figure 11. Our results on this data set (which were corroborated by the results we achieved on the other data sets) show that the results achieved by EUCLID alone depend directly on the probability of errors being introduced into the data sets. For example, EUCLID

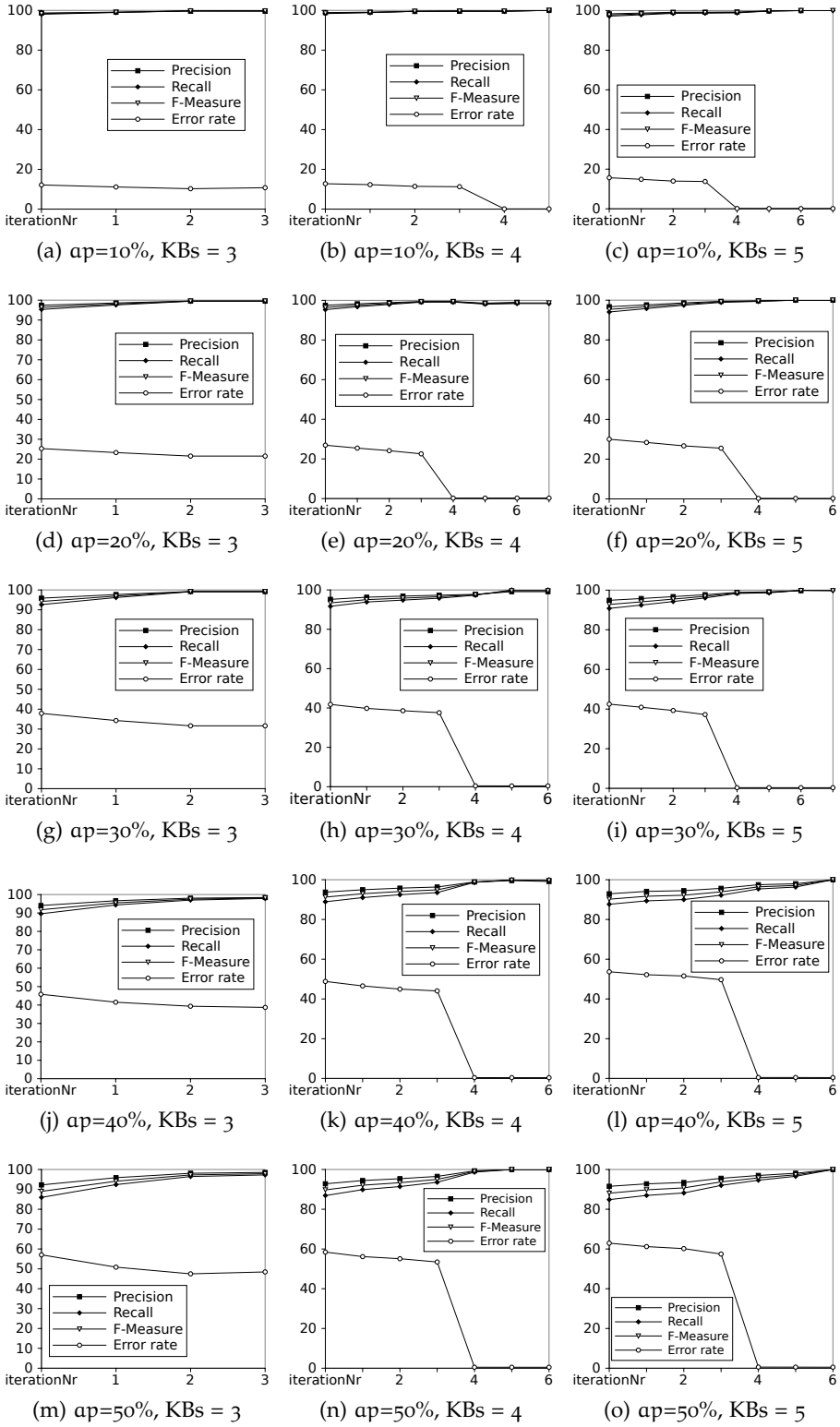
9 See http://limes.sf.net for more results.

ap	10%				30%				50%			
Measures	F_E	F_C	R	L	F_E	F_C	R	L	F_E	F_C	R	L
KBs	Restaurant											
3	0.98	1.00	0.6	4	0.94	0.99	0.5	17	0.89	0.98	0.4	43
4	0.99	1.00	1.2	8	0.93	1.00	1.0	33	0.90	1.00	0.9	35
5	0.98	1.00	1.8	20	0.93	1.00	1.5	30	0.88	1.00	1.3	34
KBs	Persons1											
3	0.99	1.00	225.6	11	0.96	1.00	206.2	38	0.94	1.00	190.4	57
4	0.98	1.00	494.3	23	0.96	1.00	422.1	47	0.93	1.00	349.9	77
5	0.98	1.00	819.4	20	0.95	1.00	747.6	75	0.93	1.00	656.2	110
KBs	ACM											
3	0.95	0.96	85.7	220	0.89	0.96	69.3	301	0.84	0.95	66.5	484
4	0.94	0.94	168	12	0.88	0.88	140.4	36	0.83	0.96	131.1	261
5	0.94	0.94	271.7	30	0.87	0.94	240.9	821	0.82	0.84	202.8	348
KBs	DBLP											
3	0.94	0.98	135	220	0.85	0.97	117.2	828	0.77	0.82	111	2686
4	0.93	0.98	268.8	312	0.83	0.90	234.7	306	0.76	0.81	201.1	350
5	0.93	0.98	334.9	517	0.82	0.84	395.9	182	0.76	0.77	338.1	156
KBs	Amazon											
3	0.97	0.99	90.4	60	0.92	0.99	85.2	177	0.86	0.98	81.8	300
4	0.97	0.99	187.5	98	0.91	0.98	172.6	185	0.85	0.99	160.4	150
5	0.96	0.99	301.8	131	0.90	0.99	278.7	369	0.84	0.88	246.8	60

Table 3: Average F-measure of EUCLID (F_E) and COLIBRI (F_C) after 10 iterations, runtime (R, in seconds) and number of repaired links L achieved across all experiments. KBs stands for the number of knowledge bases used in our experiments.

is able to achieve an F-measure of 0.94 when provided with data sets with an error rate of 30%. Yet, this F-measure sinks to 0.88 when the error rate is set to 50%. These results do suggest that EUCLID is robust against errors. This is due to the approach being able to give properties that contain a small error percentage a higher weight. Still, the COLIBRI results show clearly that COLIBRI can accurately repair the knowledge bases and thus achieve even better F-measures. On this particular data, the approach achieves an F-measure very close to 1 in most cases. Note that the number of iterations required to achieve this score depends directly on the number of knowledge bases and on the error probability.

One interesting observation is that the average F-measure achieved by EUCLID decreases with the number of knowledge bases used for linking. This is simply due to the overall larger number of errors generated by our evaluation framework when the number of knowledge bases is increased. While larger number also make the detection of errors more tedious, COLIBRI achieves significant increase of F-measure



(a) ap=10%, KBs = 3 (b) ap=10%, KBs = 4 (c) ap=10%, KBs = 5

(d) ap=20%, KBs = 3 (e) ap=20%, KBs = 4 (f) ap=20%, KBs = 5

(g) ap=30%, KBs = 3 (h) ap=30%, KBs = 4 (i) ap=30%, KBs = 5

(j) ap=40%, KBs = 3 (k) ap=40%, KBs = 4 (l) ap=40%, KBs = 5

(m) ap=50%, KBs = 3 (n) ap=50%, KBs = 4 (o) ap=50%, KBs = 5

Figure 11: Overview of the results on the *Restaurants* data set.

in this setting. In particular, the F-measure of EUCLID is improved upon by up to 12% absolute on the Restaurant data set ($ap = 50\%$) as well as 7% absolute on Persons1 ($ap = 50\%$).

As expected, the runtime of our approach grows quadratically with the number of knowledge bases. This is simply due to EUCLID being run for each pair of knowledge bases.The runtimes achieved suggest that COLIBRI can be used in practical settings and on large data sets as long as the number of dimensions in EUCLID's search space remains small. In particular, one iteration of the approach on the DBLP data sets required less than 2 minutes per iteration for 3 knowledge bases, which corresponds to 3 EUCLID runs of which each checked 3125 link specifications. The worst runtimes were achieved on the *Persons1* data set, where COLIBRI required up to 11 min/iteration. This was due to the large number of properties associated with each resource in the data set, which forced EUCLID to evaluate more than 78,000 specifications per iteration.

6

DPSO – AN OPTIMIZATION APPROACH FOR LOAD BALANCING IN PARALLEL LINK DISCOVERY

In this chapter, we present a novel load balancing approach for LD in parallel hardware dubbed as DPSO. A paper about this work is published at SEMANTiCS'15 [Sherif and Ngonga Ngomo, 2015a]. The author developed the ideas behind DPSO, implemented it together with all other load balancing algorithms presented the chapter, carried out the evaluations and co-wrote the paper.

In Chapter 4, we tackled the challenge of finding geospatial distance functions for Link Discovery (LD), while in Chapter 5 we investigated the LD across more than two knowledge bases. In this Chapter, we address the need to develop highly scalable algorithms for the discovery of links between knowledge bases. While several architectures can be used to this end, previous works suggest that approaches based on local hardware resources suffer less from the data transfer bottleneck [Ngonga Ngomo et al., 2013a] and can thus achieve significantly better runtime than parallel approaches which rely on remote hardware (e.g., cloud-based approaches [Kolb and Rahm, 2013]). Moreover, previous works also suggest that load balancing (also called task assignment [Salman et al., 2002]) plays a key role in getting approaches for LD to scale. However, load balancing approaches for local parallel LD algorithms have been paid little attention to so far. In particular, mostly naïve implementations of parallel LD algorithms have been integrated into commonly used LD framework such as SILK [Isele et al., 2011b] and LIMES [Ngonga Ngomo, 2012].

The load balancing problem, which is know to be NP-complete [Salman et al., 2002], can be regarded as follows: Given n tasks $\tau_1, ..., \tau_n$ of known computational complexity (also called cost) $c(\tau_1), ..., c(\tau_n)$ as well as m processors, distribute the tasks τ_i across the m processors as evenly as possible, i.e., in such a way that there is no other distribution which would lead to a smaller discrepancy from a perfectly even distribution of tasks. Consider for example 3 tasks τ_1, τ_2 respectively τ_3 with computation complexities 3, 4 resp. 6. An optimal distribution of these tasks amongst two processors would consist of assigning τ_1 and τ_2 to the one of the processor (total costs: 7) and task τ_3 to the other processor (total costs: 6). No other task distribution leads to a more balanced load of tasks.

In this chapter, we address the research gap of load balancing for link discovering by first introducing the link discovery as well as the load balancing problems formally. We then introduce a set of heuristics for addressing this problem, including a novel heuristic dubbed DPSO. This novel heuristic employs the basic insights behind Particle-Swarm Optimization (PSO) to determine a load balancing for link discovery tasks in a deterministic manner. Our approach is generic and can be combined with any link discovery approach that can divide the LD problem into a set of tasks within only portions of the input datasets are compared, including methods based on blocking

(e.g., *Multiblock* Isele et al. [2011b]) and on space tiling (e.g., [Ngonga Ngomo, 2013]). We evaluate our approach on both synthetic and real data.

NOTATION

In this section, we present some of the notation and concepts necessary to understand the rest of the chapter. The formal specification of LD adopted herein is akin to that introduced in Section 2.1

The idea behind load balancing for LD is to distribute the computation of the distance function δ over the Cartesian products $S_i \times T_j$ over n processors. We call running δ through a Cartesian product $S_i \times T_j$ a *task*. The set of all tasks assigned to a single processor is called a *block*. The cost $c(\tau)$ of the task τ is given by $c(S_i \times T_j) = |S_i| \cdot |T_j|$ while the cost of a block B is the sum of the cost of all its elements, i.e., $c(B) = \sum_{t \in B} c(\tau)$. Finding an optimal load balancing is known to be NP-hard. Hence, we refrain from trying to find a perfect solution in this chapter. Rather, we aim to provide a heuristic that (1) achieves a good assignment of tasks to processors while (2) remaining computationally cheap. We measure the quality of an assignment by measuring the Mean Squared Error (MSE) to a potentially existing perfect solution. Let $B_1, ..., B_m$ be the blocks assigned to our m processors. Then, the MSE is given by

$$\sum_{i=1}^{m} \left| c(B_i) - \sum_{j=1}^{m} \frac{c(B_j)}{m} \right|^2 . \tag{16}$$

It is obvious that there might not be a solution with an MSE of 0. For example, the best possible MSE when distributing the 3 tasks τ_1, τ_2 respectively τ_3 with computation complexities 3, 4 respectively 6 over 2 processors is 0.5.

LOAD BALANCING ALGORITHMS

The main idea behind load balancing techniques is to utilize parallel processing to distribute the tasks necessary to generate the solution to a problem across several processing units. Throughput maximization, response time minimization and resources overloading avoidance are the main purposes of any load balancing technique. We devised, implemented and evaluated five different load balancing approaches for linking geo-spatial datasets.

In each of the following algorithms, as input, we assume having a set $\mathcal{T}$ of n tasks and a set of m processors. Through each of the algorithms, we try to achieve load balancing among the m processors by creating a list of balanced task blocks $\mathcal{B} = \{B_1, ..., B_m\}$ with size m, where each processor p_i will be assigned its respective block B_i.

Algorithm 2: Naïve Load Balancer

input : $\mathcal{T} \leftarrow \{\tau_1, ..., \tau_n\}$: set of tasks of size n
m : number of processors
output: $\mathcal{B} \leftarrow \{B_1, ..., B_m\}$: a partition of $\mathcal{T}$ to a list of m blocks of tasks

```
1 i ← 1;
2 foreach task τ in 𝒯 do
3 |   addTaskToBlock(τ, B_i);
4 |   i ← (i mod m) + 1;
5 return ℬ;
```

In order to ease the explanation of the following load balancing algorithms, we introduce a simple running example where we assume having a set of four tasks $\{\tau^7, \tau^1, \tau^8, \tau^3\}$ where the superscript of the task stands for its computational cost. Moreover, we assume having two processing units p_1 and p_2. The goal of our running example is to find two balanced tasks blocks B_1 and B_2 to be assigned to p_1 respectively p_2. In the following, we present different approaches for load balancing.

Naïve Load Balancer

The idea behind the naïve load balancer is to divide all tasks between all processors based on their index and regardless of their complexity. Each task with the index i is assigned to the processor with index $((i+1) \mod m) + 1$. Hence, each of the m processors is assigned at most $\lceil \frac{n}{m} \rceil$ tasks. Algorithm 2 shows the pseudo-code of our implementation of a naïve load balancing approach in which tasks are assigned to processors in the order of the input set. Applying the naïve load balancer to our running example we get $B_1 = \{\tau^7, \tau^8\}$, $B_2 = \{\tau^1, \tau^3\}$ and MSE = 30.25.

Greedy Load Balancer

The main idea behind the greedy load balancing [Caragiannis et al., 2011] technique is to sort the input tasks in descending order based on their complexity. Then, starting from the most complex task, the greedy load balancer assigns tasks to processors in order. This approach is basically a heuristic that aims at achieving an even distribution of the total task complexity over all processors. The pseudo code of the greedy load balancer technique in presented in Algorithm 3. Back to our running example, the greedy load balancer first sorts the example tasks (line 2) to be $\{\tau^8, \tau^7, \tau^3, \tau^1\}$. Then, in order, the tasks are

Algorithm 3: Greedy Load Balancer

input : $\mathcal{T} \leftarrow \{\tau_1, ..., \tau_n\}$: set of tasks of size n
m : number of processors
output: $\mathcal{B} \leftarrow \{B_1, ..., B_m\}$: a partition of $\mathcal{T}$ to a list of m blocks of balanced tasks

```
1 T ← descendingSortTasksByComplexity(T);
2 i ← 1;
3 foreach task τ in T do
4 |   addTaskToBlock(τ, B_i);
5 |   i ← (i mod m) + 1;
6 return B;
```

Algorithm 4: Pair Based Load Balancer

input : $\mathcal{T} \leftarrow \{\tau_1, ..., \tau_n\}$: set of tasks of size n
m : number of processors
output: $\mathcal{B} \leftarrow \{B_1, ..., B_m\}$: a partition of $\mathcal{T}$ to a list of m blocks of balanced tasks

```
1 T ← sortTasksByComplexity(T);
2 i ← 1;
3 for i ⩽ ⌈n/2⌉ do
4 |   addTaskToBlock(τ_i, B_i);
5 |   addTaskToBlock(τ_{n−i+1}, B_i);
6 |   i ← i + 1;
7 return B;
```

assigned to the task blocks (line 4) to have $B_1 = \{\tau^8, \tau^3\}$, $B_2 = \{\tau^7, \tau^1\}$ with MSE = 2.25.

Pair-Based Load Balancer

The pair-based load balancing [Kolb et al., 2012] is reminiscent of a two-way breadth-first-search. The approach assigns processors tasks in pairs of the form *(most complex, least complex)*. In order to get most homogeneous pairs, the algorithm first sorts all input tasks according to tasks' complexities. Afterwards, from the sorted list of tasks, the pair based algorithm generates the $\lceil \frac{n}{2} \rceil$ pairs of tasks where the pair i is computed by selecting i^{th} and $(n - i + 1)^{th}$ tasks from the sorted list of tasks. The pseudo-code of the pair based technique is shown in Algorithm 4.

The pair-based load balancer starts dealing with our running example tasks by sorting them to be $\{\tau^1, \tau^3, \tau^7, \tau^8\}$ (line 1). Afterwards, the algorithm generates tasks pairs as (first, fourth) and (second, third) to have $B_1 = \{\tau^1, \tau^8\}$, $B_2 = \{\tau^3, \tau^7\}$ with MSE = 0.25.

Particle Swarm Optimization

The particle swarm optimization (PSO) [Kiranyaz et al., 2014; Kaveh, 2014; Kennedy, 2010] is a population-based stochastic algorithm. PSO is based on social psychological principles. Unlike evolutionary algorithms, in a typical PSO, there is no selection of individuals, all population members (dubbed as particles) survive from the beginning to the end of a algorithm. At the beginning of PSO, particles are randomly initialized in the problem solution space. Over successive iterations, particles cooperatively interact to improve of the fitness of the optimization problem solutions. PSO is normally used for continuous problems but that it has been extended to deal with discrete problems [Zhong et al., 2007; Cai et al., 2014] such as the one at hand.

In order to model our problem in terms of the PSO technique, we consider the input tasks $\mathcal{T}$ as the *particles*[1] to be optimized. The aim here is to to balance the size of the blocks (i.e., the total complexity of tasks included in each block) as well as possible. To adapt the idea of the PSO to load balancing, we define the *fitness function* as the task complexity difference between the most overloaded task block and least underloaded task block. Formally, The PSO fitness function is defined as

$$F = c(B^+) - c(B^-), \tag{17}$$

where $B^+ = \arg\max_{B \in \mathcal{B}} c(B)$ and $B^- = \arg\min_{B \in \mathcal{B}} c(B)$ are the most and least loaded blocks respectively, and $\mathcal{B}$ is the list of all task blocks.

Initially, the PSO based load balancing approach starts like the naïve approach (see Algorithm 2). All particles are distributed equally into the task blocks regardless of tasks' complexities, i.e., each block now contains at most $\lceil \frac{n}{2} \rceil$ particles. We dubbed the task block list as *Best Known Positions* (BKP). Afterwards, PSO computes the fitness function to the initial BKP and saves it as *Best Known Fitness* (BKF). Until a termination criterion is met, in each iteration, PSO performs the *particles migration* process. This process consists of first assigning a random velocity v to each particle p included in a block B_i, where $v \in \mathbb{N}$ and $0 \leqslant v \leqslant m$. If $v \neq i$, p is moved to the new block B_v, otherwise p stays in its block B_i. After moving all the particles, the PSO computes the new fitness F. If the new fitness F is less than BKF, PSO updates both BKF and BKP.

Note that the termination criteria can be defined independently of the core PSO algorithm. Here, we implemented two termination criteria: (1) *minimum fitness threshold* and (2) *maximum number of iterations*. If the minimum fitness threshold is reached in any iteration the algorithm terminates instantly and the BKP is returned. Otherwise, the BKF is returned after reaching the maximal number of iterations. The pseudo-code of the PSO load balancing technique in presented in Algorithm 5.

1 In the rest of the chapter we will use the terms *tasks* and *particles* interchangeably.

Back to our running example, assume we set the maximal number of iterations to 1 ($\mathbf{I} = 1$). First, the PSO initializes $B_1 = \{\tau^7, \tau^8\}$, $B_2 = \{\tau^1, \tau^3\}$ (lines 3–5) and the best known Fitness $F = 11$ (lines 9). Then, the PSO clones B_1 and B_2 to B_1^* respectively B_2^* (line 12). Assume that PSO generates random velocity $v = 1$ for τ^7 (line 16). Then, τ^7 stays in its current block B_1^*. For τ^8, assume $v = 2$ (which is different than τ^8's block B_1^*), then τ_8 migrates to B_1^* (line 18). For τ^1 and τ^3 assume $v = 1$ which make both τ^1 and τ^3 stays in B_2^*. Consequently, we have $B_1^* = \{\tau^7\}$, $B_2^* = \{\tau^1, \tau^3, \tau^8\}$ with the new fitness function $F^* = 5$ (line 21) and as $F^* < F$ then $\mathcal{B}$ and F are updated by $\mathcal{B}^*$ respectively F^* (lines 22–24). The PSO terminates as it is reached the maximum number of iterations (line 11) and returns $B_1 = \{\tau^7\}$, $B_2 = \{\tau^1, \tau^3, \tau^8\}$ with MSE = 6.25.

Deterministic Particle Swarm Optimization Load Balancer

The PSO load balancer (see Section 6.2.4) has a main drawback of being an *indeterminism* approach. This drawback is inherited from the fact that the PSO is a *heuristic* algorithm that depends up on a random selection of velocity for moving particles. In order to overcome this drawback, we propose the *Deterministic PSO* (DPSO).

The DPSO starts in the same way as the PSO by partitioning all the n tasks to m task blocks, where m equals the number of processors. In this stage, each block contains at most $\lceil n/m \rceil$ tasks regardless of tasks' complexities. Until a termination criterion is met, in each iteration the DPSO:

1. Finds the most overloaded block $B^+ = \arg\max_{B \in \mathcal{B}} c(B)$ and the least underloaded block $B^- = \arg\min_{B \in \mathcal{B}} c(B)$, where $\mathcal{B}$ is the list of all task blocks.
2. Sort tasks within B^+ based in their complexities.
3. As far as a better balancing between B^+ to B^- is met, DPSO perform *task migration*, where DPSO moves task per task in order from B^+ to B^-.
4. Compute *fitness function* as $c(B^+) - c(B^-)$.

Here, We implement two termination criteria akin with the ones defined previously for PSO: (1) *minimum fitness threshold* (**F**) and (2) *maximum number of iterations* (**I**). The pseudo code of the DPSO load balancing algorithm in presented in algorithm Algorithm 6. Note that the termination criteria can be defined independently of the core DPSO algorithm. For instance, fitness function convergence could be considered as the termination criterion.

The deterministic nature of DPSO comes from the fact that (1) DPSO only moves tasks from most overloaded block B^+ to the lest underloaded block B^-, i.e. no random particles migration as in PSO, (2)

Algorithm 5: Particle Swarm Optimization Load Balancer

input : $\mathcal{T} \leftarrow \{\tau_1, ..., \tau_n\}$: set of tasks of size n
m : number of processors
F : fitness function threshold (zero by default)
I : number of iterations

output: $\mathcal{B} \leftarrow \{B_1, ..., B_m\}$: the Best Known Particles' positions as a list of m blocks of balanced tasks

```
1  Initialize Particles' Best known Position B
2  i ← 1;
3  foreach task τ in T do
4      addTaskToBlock(τ, B_i);
5      i ← (i mod m) + 1;
6  Initialize best known Fitness F;
7  B⁺ ← getMostOverloadedBlock(B);
8  B⁻ ← getLeastUnderloadedBlock(B);
9  F ← c(B⁺) − c(B⁻);
10 i ← 1;
11 while i < I do
12     B* ← B;
13     Move each task τ (particle) to new position based on a random
         particle velocity v
14     foreach block B* ∈ B* do
15         foreach particle τ ∈ B* do
16             v ← generateRandomVelocity(0, m);
17             if B*_v ≠ B* then
18                 migrateParticleToBlock(τ, B*_v);
19                 If better fitness achieved update result
                     B*⁺ ← getMostOverloadedBlock(B*);
20                 B*⁻ ← getLeastUnderloadedBlock(B*);
21                 F* ← c(B*⁺) − c(B*⁻);
22                 if F* < F then
23                     F ← F*;
24                     B ← B*;
25                 if F == F then
26                     return B;
27     i ← i + 1;
28 return B;
```

DPSO sorts B^+ tasks before it starts the task migration process. Sorting insures migration of smaller tasks first in which away an optimal load balancing between most and least loaded blocks is achieved in each iteration.

Algorithm 6: DPSO Load Balancer

input : $\mathcal{T} \leftarrow \{\tau_0, ..., \tau_n\}$: set of tasks of size n
m : number of processors
F : fitness function threshold (zero by default)
I : number of iterations

output: $\mathcal{B} \leftarrow \{B_0, ..., B_m\}$: the Best Known Particles' positions as a list of m blocks of balanced tasks

```
1  Initialize Particles' Best known Position B
2  i ← 1;
3  foreach task τ in T do
4      addTaskToBlock(τ, B_i);
5      i ← (i mod m) + 1;
6  i ← 1;
7  while i < I do
8      B+ ← getMostOverloadedBlock(B);
9      B− ← getLeastUnderloadedBlock(B);
10     Balance B+ and B+ by migrating particles (tasks) from sorted B+
        to B−
11     B+ ← sortTasksByComplexity(B+);
12     foreach particle τ ∈ B+ do
13         migrateParticleToBlock(τ, B−);
14         if c(B+) < c(B−) then
15             break;
16         Compute fitness function F as the complexity difference between
            most and least loaded tasks
17         F ← c(B+) − c(B−);
18         if F == F then
19             return B;
20     i ← i + 1;
21 return B;
```

Assume we apply the DPSO for our running example for one iteration (**I** = 1). First, the DPSO initializes $B_1 = \{\tau^7, \tau^8\}$, $B_2 = \{\tau^1, \tau^3\}$ (lines 2–5). Then, DPSO sorts tasks within the most overloaded block B^+ which is B_1 (line 8) to be $\{\tau^7, \tau^8\}$ (line 11). Consequently, the DPSO migrates τ^7 from $B^+ = B^1$ to $B^- = B^1$ (line 13). Finally, as the DPSO finds that $c(B^+) < c(B^-)$ it breaks (line 14) and returns the result $B_1 = \{\tau^8\}$, $B_2 = \{\tau^1, \tau^3, \tau^7\}$ with MSE = 2.25.

EVALUATION

The aim of our evaluation was to quantify how well DPSO outperforms traditional load balancing approaches (i.e. naïve, greedy and

pair-based). To this end, we measured the runtime for each of the five load balancing algorithms for both synthetic and real data. In the following, we begin by presenting the algorithm and data that we used. Thereafter, we present our results on different datasets.

Experimental Setup

For our experiments, the parallel task generation was based on the ORCHID approach (See Section 2.1.2). The idea behind ORCHID is to improve the runtime of algorithms for measuring geo-spatial distances by adapting an approach akin to divide-and-conquer. ORCHID assumes that it is given a distance measure δ. Thus all pairs in the mapping M that it returns must abide by $\delta(s,t) \leqslant \theta$. Overall, ORCHID begins by partitioning the surface of the planet. Then, the approach defines a *task* as comparing the points in a given partition with only the points in partitions that abide by the distance threshold θ underlying the computation. A task is the comparison of all points in two partitions.

We performed controlled experiments on five synthetic geographic datasets[2] and three real datasets. The synthetic datasets were created by randomly generating a number of polygons ranging between 1 and 5 million polygons in steps of 1 million. We varied the synthetic dataset polygons' sizes from one to ten points. The variation of sizes of polygons was based on a *Gaussian* random distribution. Also, the (latitude, longitude) coordinates of each point are generated akin with the *Gaussian* distribution.

We used three publicly available datasets for our experiments as real datasets. The first dataset is the *Nuts*[3]. We chose this dataset because it contains fine-granular descriptions of 1,461 geo-spatial resources located in Europe. For example, Norway is described by 1981 points. The second dataset, *DBpedia*[4], contains all the 731,922 entries from DBpedia that possess geometry entries. We chose DBpedia because it is commonly used in the Semantic Web community. Finally, the third dataset was *LinkedGeoData*, contains all 3,836,119 geo-spatial objects from `http://linkgeodata.org` that are instances of the class `Way`.[5]. Further details to the datasets can be found in [Ngonga Ngomo, 2013].

All experiments were carried out on a 64-core server running *OpenJDK* 64-Bit Server 1.6.0_27 on *Ubuntu* 12.04.2 LTS. The processors were 8 quad-core *Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz* with 8192 KB cache. Unless stated otherwise, each experiment was assigned 20 GB

2 All synthetic dataset are available at `https://github.com/AKSW/LIMES/tree/master/evaluationsResults/lb4ld`

3 Version 0.91 available at `http://nuts.geovocab.org/data/` is used in this work

4 We used version 3.8 as available at `http://dbpedia.org/Datasets`.

5 We used the RelevantWays dataset (version of April 26th, 2011) of *LinkedGeoData* as available at `http://linkedgeodata.org/Datasets`.

of memory. Because of the random nature of the PSO approach we ran it 5 times in each experiment and provide the mean of the five runs' results. The approaches presented herein were implemented in the LIMES framework.[6] All results are available at the project web site.[7]

Orchid vs. Parallel Orchid

We began by evaluating the *speedup* gained by using parallel implementations of ORCHID algorithm. To this end, we first ran experiments on the three real datasets (Nuts, DBpedia and LinkedGeoData). First, we computed the runtime of the normal (i.e., non-parallel) implementation of ORCHID [Ngonga Ngomo, 2013]. Then, we evaluated the parallel implementations of ORCHID using the aforementioned five load balancing approaches. To evaluate the *speedup* gained from increasing the number of parallel processing units, we reran each of the parallel experiments with 2, 4 and 8 threads. Figure 12 shows the runtime results along with the mean squared error (MSE) results of the experiments.

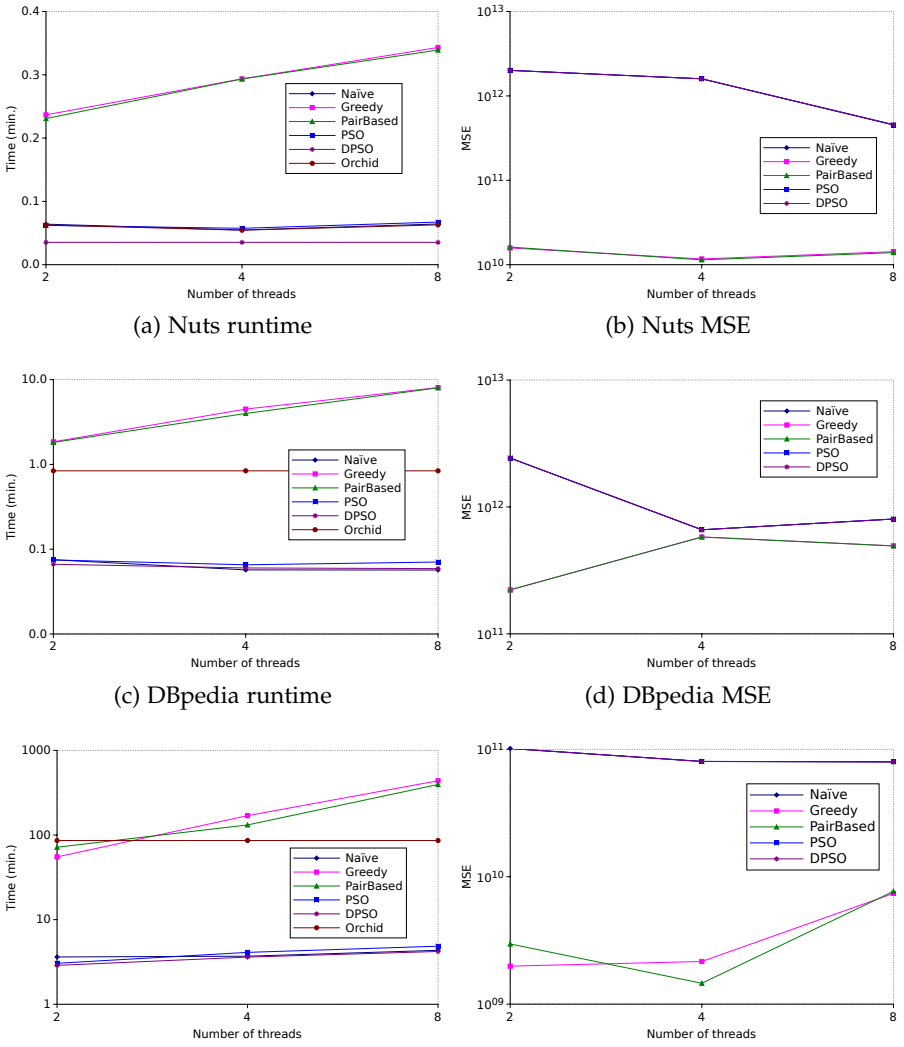
Our results show that the parallel ORCHID implementations using both PSO and DPSO outperform the normal ORCHID on the three real datasets. Particularity, when dealing with small dataset like *NUTS* (see Figure 12 (a)), PSO and DPSO achieve up to three times faster than the non-parallel version of ORCHID. When dealing with larger dataset such as *LinkedGeoData* (see Figure 12 (e)), PSO and DPSO are capable of achieving up to ten times faster than the non-parallel version of ORCHID. This fact shows that our load balancing heuristics deployed in PSO and DPSO are capable of achieving *superlinear* performance [Akl, 2004; Alba, 2002] when ran on two processors. This is simply due to the processor cache being significantly faster than RAM and thus allowing faster access to data and therewith also smaller runtimes. On the other side, greedy and pair-based load balancing fail to achieve even the run time of the normal ORCHID. This fact is due to the significant amount of time required by greedy and pair-based load balancing algorithms for sorting tasks prior to assigning them to processors.

Parallel Load balancing Algorithms Evaluation

We performed this set of experiments with two goals in mind: First, we wanted to measure the run time taken by each algorithm when applied to different datasets. Our second aim was to qualify the quality of the data distribution carried out by each of the implemented algorithm using MSE. To this end, we ran two sets of experiments. In the first set of experiments, we used the aforementioned three datasets of *Nuts*, *DBpedia* and *LinkedGeoData*. The result of this set of experiments

6 http://limes.sf.net

7 https://github.com/AKSW/LIMES/tree/master/evaluationsResults/lb4ld



(a) Nuts runtime (b) Nuts MSE

(c) DBpedia runtime (d) DBpedia MSE

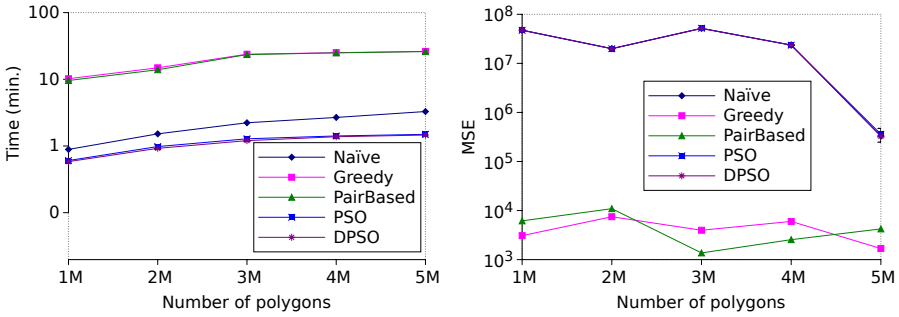
(e) LinkedGeoData runtime (f) LinkedGeoData MSE

Figure 12: Runtime and MSE generated when applying ORCHID [Ngonga Ngomo, 2013] vs. parallel implementations of ORCHID using naïve, greedy, pair based, PSO and DPSO load balancing algorithms against the three real datasets of *Nuts*, *DBpedia* and *LinkedGeoData* using 2, 4 and 8 threads

are presented in Figure 12. In the second set of experiments, we ran our five load balancing algorithms against a set of five synthetic randomly generated datasets (see Section 6.3.1 for details). The results are presented in Figure 13.

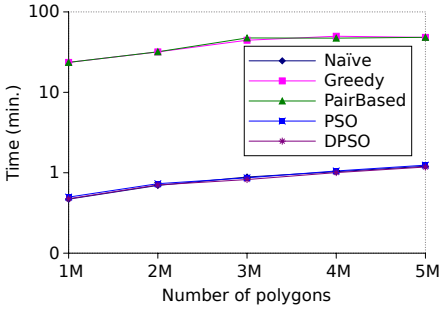
Our results suggest that DPSO and PSO outperform the naïve approach in most cases. This can be seen most clearly in Figure 13 (note the log scale). DPSO is to be preferred over PSO as it is deterministic and is thus the default implementation of load balancing currently implemented in LIMES. Still, the improvements suggest that preserving the integrity of the hypercubes generated by ORCHID still leads to a high difference in load across the processors as shown by our

MSE results. An interesting research avenue would thus be to study approaches which do not preserve this integrity while guaranteeing result completeness. This will be the core of our future work.

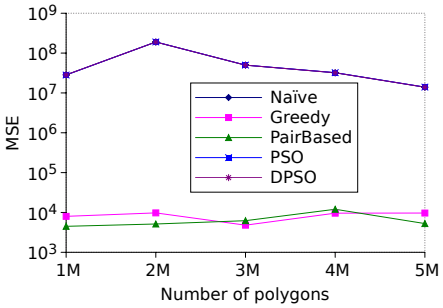


(a) 2 threads runtime

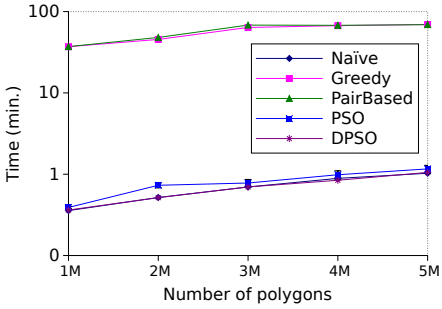
(b) 2 threads MSE



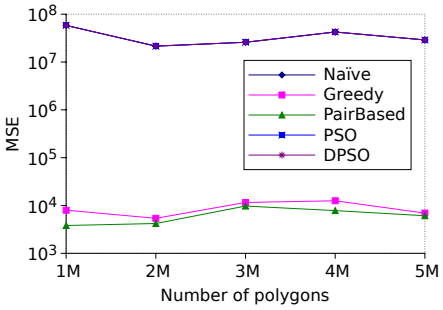
(c) 4 threads runtime



(d) 4 threads MSE



(e) 8 threads runtime



(f) 8 threads MSE

Figure 13: Runtime and MSE generated when applying parallel implementations of ORCHID using naïve, greedy, pair based, PSO and DPSO load balancing algorithms against the five synthetic datasets of sizes 1, 2, 3, 4 and 5 million polygons using 2, 4 and 8 threads

7

WOMBAT – A GENERALIZATION APPROACH FOR AUTOMATIC LINK DISCOVERY

In this chapter we present Wombat, *an approach for learning Link Specifications (LS) based on positive examples only. All the proposed algorithms in this chapter are implemented by the author, who also carried out the evaluations and co-wrote the paper.*

We studied the geospatial distance function for LD in Chapter 4. Then, we proposed the Colibri algorithm for LD across more than two knowledge bases in Chapter 5. In Chapter 6, we introduced the DPSO algorithm for LD load balancing. In this chapter, we tackle the challenge of LD in absence of negative examples.

The growth of the Data Web engenders an increasing need for automatic support when maintaining evolving data sets. One of the most crucial tasks when dealing with evolving data sets lies in updating the links from these data sets to other data sets. While supervised approaches have been devised to achieve this goal, they assume that they are provided with both positive and negative examples for links [Auer et al., 2013]. However, the links available on the Data Web only provide positive examples for relations and no negative examples.[1] The open-world assumption underlying the Web of Data suggests that given the non-existence of a link between two resources cannot be understood as stating these two resources are not related. Hence, it is impossible to construct negative examples based on existing positive examples for most relations. Consequently, state-of-the-art supervised learning approaches for LD can only be employed if the end users are willing to provide the algorithms with information that is generally not available on the LOD cloud, i.e., with negative examples.

We address this drawback by proposing the first approach for learning Link Specifications (LS) based on positive examples only. Our approach, dubbed Wombat, is inspired by the concept of generalisation in quasi-ordered spaces. Given a set of positive examples and a grammar to construct LS, we aim to find a specification that covers a large number of positive examples (i.e., achieves a high recall on the positive examples) while still achieving a high precision. A main challenge is that LSs can use various similarity metrics, acceptance threshold and nested logical combinations of those.

Our contributions in this chapter are as follows:

- We provide the first (to the best of our knowledge) approach for learning LSs that is able to learn links from positive examples only.

1 3 678 RDF data set dumps containing 714 714 370 triples analysed via LODStats (see lodstats.aksw.org) in March 2015 contained 10 116 041 `owl:sameAs` links and no `owl:differentFrom` links. Moreover, inferring `owl:differentFrom` links is often not possible due to missing schema integration and low expressiveness of knowledge bases.

- Our approach is based on an upward refinement operator for which we analyse its theoretical characteristics.
- We use the characteristics of our operator to devise a pruning approach and improve the scalability of WOMBAT.
- We evaluate WOMBAT on 8 benchmark data sets and show that in addition to needing less training data, it also outperforms the state of the art in most cases.

The rest of this chapter is structured as follows: In Section 7.1, we present preliminaries necessary to understand this chapter. We then introduce the atomic LS optimization and refinement operator underlying WOMBAT in Section 7.2. In Section 7.3, we present the WOMBAT algorithm in detail. Finally, in Section 7.4 we evaluate our approach on eight benchmark against other state-of-the-art approaches.

NOTATION

The formal specification of LD adopted herein is based on that introduced in Section 2.1. Several grammars have been used for describing LS in previous works [Ngonga Ngomo and Lyko, 2012; Isele et al., 2011a; Nikolov et al., 2012]. In general, these grammars assume that LS consist of two types of atomic components: *similarity measures* m, which allow comparing property values of input resources and *operators* op, which can be used to combine these similarities to more complex specifications. Without loss of generality, we define a similarity measure m as a function $m : S \times T \to [0, 1]$. An example of a similarity measure is the edit similarity dubbed `edit`[2] which allows computing the similarity of a pair $(s, t) \in S \times T$ with respect to the properties p_s of s and p_t of t. We use *mappings* $M \subseteq S \times T$ to store the results of the application of a similarity function to $S \times T$ or subsets thereof. We denote the set of all mappings as $\mathcal{M}$ and the set of all LS as $\mathcal{L}$. We define a *filter* as a function $f(m, \theta)$. We call a specification *atomic* when it consists of exactly one filtering function. A complex specification can be obtained by combining two specifications L_1 and L_2 through an *operator* that allows merging the results of L_1 and L_2. Here, we use the operators $\sqcap$, $\sqcup$ and $\setminus$ as they are complete and frequently used to define LS. An example of a complex LS is given in Figure 14.

We define the semantics $[[L]]_M$ of a LS L w.r.t. a mapping M as given in Table 4. Those semantics are similar to those used in languages like SPARQL, i.e., they are defined extensionally through the mappings they generate. The mapping $[[L]]$ of a LS L with respect to $S \times T$ contains the links that will be generated by L. A LS L is *subsumed* by L', denoted by $L \sqsubseteq L'$, if for all mappings M, we have

2 We define the edit similarity of two strings s and t as $(1 + lev(s, t))^{-1}$, where lev stands for the Levenshtein distance.

Table 4: Link Specification Syntax and Semantics

LS	$[[\text{LS}]]_M$
$f(m, \theta)$	$\{(s,t)\|(s,t) \in M \wedge m(s,t) \geqslant \theta\}$
$L_1 \sqcap L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \wedge (s,t) \in [[L_2]]_M\}$
$L_1 \sqcup L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \vee (s,t) \in [[L_2]]_M\}$
$L_1 \backslash L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \wedge (s,t) \notin [[L_2]]_M\}$

$[[L]]_M \subseteq [[L']]_M$. Two LS are *equivalent*, denoted by $L \equiv L'$ iff $L \sqsubseteq L'$ and $L' \sqsubseteq L$. Subsumption ($\sqsubseteq$) is a partial order over $\mathcal{L}$.

CONSTRUCTING AND TRAVERSING LINK SPECIFICATIONS

The goal of our learning approach is to learn a specification L that generalizes a mapping $M \subseteq S \times T$ which contains a set of pairs (s,t) for which $Rel(s,t)$ holds. Our approach consists of two main steps. First, we aim to derive initial atomic specifications A_i that achieve the same goal. In a second step, we combine these atomic specifications to the target complex specification L by using the operators $\sqcap$, $\sqcup$ and $\backslash$. In the following, we detail how we carry out these two steps.

Learning Atomic Specifications

The goal here is to derive a set of initial atomic LS $\{A_1, \ldots, A_n\}$ that achieves the highest possible F-measure given a mapping $M \subseteq S \times T$ which contains all known pairs (s,t) for which $Rel(s,t)$ holds. Given a set of similarity functions m_i, the set of properties P_s of S and the set of properties P_t of T, we begin by computing the subset of properties from S and T that achieve a coverage above a threshold $\tau \in [0,1]$, where the coverage of a property p for a knowledge base K is defined as

$$coverage(p) = \frac{|\{s : (s,p,o) \in K\}|}{|\{s : \exists q : (s,q,o) \in K\}|}. \tag{18}$$

Now for all property pairs $(p,q) \in P_s \times P_t$ with $coverage(p) \geqslant \tau$ and $coverage(q) \geqslant \tau$, we compute the mappings $M_{ij} = \{(s,t) \in S \times T :$

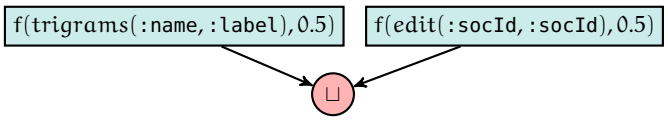


Figure 14: Example of a complex LS. The filter nodes are rectangles while the operator nodes are circles. `:socID` stands for social security number.

$m_{ij}(s,t) \geqslant \theta_j\}$, where m_{ij} compares s and t w.r.t. p and q and M_{ij} is maximal w.r.t. the F-measure it achieves when compared to M. To this end, we apply an iterative search approach. Finally, we select M_{ij} as the atomic mapping for p and q. Thus, we return as many atomic mappings as property pairs with sufficient coverage. Note that this approach is not quintessential for WOMBAT and can thus be replaced with any approach of choice which returns a set of initial LS that is to be combined.

Combining Atomic Specifications

After deriving atomic LS as described above, WOMBAT computes complex specifications by using an approach based on generalisation operators (See Section 2.2). The basic idea behind these operators is to perform an iterative search through a solution space based on a score function. Formally, we rely on the following definitions:

Definition 9 ((Refinement) Operator)**.** *In the quasi-ordered space* $(\mathcal{L}, \sqsubseteq)$*, we call a function from* $\mathcal{L}$ *to* $2^{\mathcal{L}}$ *an (LS)* operator*. A* downward (upward) refinement operator ρ *is an operator, such that for all* $L \in \mathcal{L}$ *we have that* $L' \in \rho(L)$ *implies* $L' \sqsubseteq L$ *(*$L \sqsubseteq L'$*).* L' *is called a* specialisation *(*generalisation*) of* L*.* $L' \in \rho(L)$ *is usually denoted as* $L \rightsquigarrow_\rho L'$*.*

Definition 10 (Refinement Chains)**.** *A* refinement chain *of a refinement operator* ρ *of length* n *from* L *to* L' *is a finite sequence* $L_0, L_1, \ldots, L_n$ *of LS, such that* $L = L_0, L' = L_n$ *and* $\forall i \in \{1 \ldots n\}, L_i \in \rho(L_{i-1})$*. This refinement chain* goes through L'' *iff there is an* i $(1 \leqslant i \leqslant n)$ *such that* $L'' = L_i$*. We say that* L'' *can be reached from* L *by* ρ *if there exists a refinement chain from* L *to* L''*.* $\rho^*(L)$ *denotes the set of all LS which can be reached from* L *by* ρ*.* $\rho^m(L)$ *denotes the set of all LS which can be reached from* L *by a refinement chain of* ρ *of length* m*.*

Definition 11 (Properties of refinement operators)**.** *An operator* ρ *is called (1)* ***(locally) finite*** *iff* $\rho(L)$ *is finite for all LS* $L \in \mathcal{L}$*; (2)* ***redundant*** *iff there exists a refinement chain from* $L \in \mathcal{L}$ *to* $L' \in \mathcal{L}$*, which does not go through (as defined above) some LS* $L'' \in \mathcal{L}$ *and a refinement chain from* L *to* L' *which does go through* L''*; (3)* ***proper*** *iff for all LS* $L \in \mathcal{L}$ *and* $L' \in \mathcal{L}$*,* $L' \in \rho(L)$ *implies* $L \not\equiv L'$*. An LS upward refinement operator* ρ *is called* ***weakly complete*** *iff for all LS* $\bot \sqsubset L$ *we can reach a LS* L' *with* $L' \equiv L$ *from* $\bot$ *(most specific LS) by* ρ*.*

We designed two different operators for combining atomic LS to complex specifications: The first operator takes an atomic LS and uses the three logical connectors to append further atomic LS. Assuming that $(A_1, \ldots, A_n)$ is the set of atomic LS found, φ can be defined as follows:

$$\varphi(L) = \begin{cases} \bigcup_{i=1}^{n} A_i & \text{if } L = \bot \\ (\bigcup_{i=1}^{n} L \sqcup A_i) \cup (\bigcup_{i=1}^{n} L \sqcap A_i) \cup (\bigcup_{i=1}^{n} L \backslash A_i) & \text{otherwise} \end{cases}$$

$$\psi(L) = \begin{cases} \{A_{i_1} \setminus A_{j_1} \sqcap \cdots \sqcap A_{i_m} \setminus A_{j_m} \mid A_{i_k}, A_{j_k} \in \mathbf{A} \\ \quad \text{for all } 1 \leqslant k \leqslant m\} & \text{if } L = \bot \\ \{L \sqcup A_i \setminus A_j \mid A_i \in \mathbf{A}, A_j \in \mathbf{A}\} & \text{if } L = A \,(\text{atomic}) \\ \{L_1\} \cup \{L \sqcup A_i \setminus A_j \mid A_i \in \mathbf{A}, A_j \in \mathbf{A}\} & \text{if } L = L_1 \setminus L_2 \\ \{L_1 \sqcap \cdots \sqcap L_{i-1} \sqcap L' \sqcap L_{i+1} \sqcap \cdots \sqcap L_n \mid L' \in \psi(L_i)\} \\ \quad \cup \{L \sqcup A_i \setminus A_j \mid A_i \in \mathbf{A}, A_j \in \mathbf{A}\} & \text{if } L = L_1 \sqcap \cdots \sqcap L_n \,(n \geqslant 2) \\ \{L_1 \sqcup \cdots \sqcup L_{i-1} \sqcup L' \sqcup L_{i+1} \sqcup \cdots \sqcup L_n \mid L' \in \psi(L_i)\} \\ \quad \cup \{L \sqcup A_i \setminus A_j \mid A_i \in \mathbf{A}, A_j \in \mathbf{A}\} & \text{if } L = L_1 \sqcup \cdots \sqcup L_n \,(n \geqslant 2) \end{cases}$$

Figure 15: Definition of the refinement operator ψ.

This naive operator is not a refinement operator (neither upward nor downward). Its main advantage lies in its simplicity allowing for a very efficient implementation. However, it cannot reach all specifications, e.g., a specification of the form $(A_1 \sqcup A_2) \sqcap (A_3 \sqcup A_4)$ cannot be reached. Examples of chains generated by φ are as follows:

1. $\bot \rightsquigarrow_\varphi A_1 \rightsquigarrow_\varphi A_1 \sqcup A_2 \rightsquigarrow_\varphi (A_1 \sqcup A_2) \setminus A_3$
2. $\bot \rightsquigarrow_\varphi A_2 \rightsquigarrow_\varphi A_2 \sqcap A_3 \rightsquigarrow_\varphi (A_2 \sqcap A_3) \setminus A_4$

The second operator, ψ, uses a more sophisticated expansion strategy in order to allow learning arbitrarily nested LS and is shown in Figure 15. Less formally, the operator works as follows: It takes a LS as input and makes a case distinction on the type of LS. Depending on the type, it performs the following actions:

- The $\bot$ LS is refined to the set of all combinations of $\setminus$ operations. This set can be large and will only be built iteratively (as required by the algorithm) with at most approx. n^2 refinements per iteration (see the next section for details).
- In LS of the form $A_1 \setminus A_2$, ψ can drop the second part in order to generalise.
- If the LS is a conjunction or disjunction, the operator can perform a recursion on each element of the conjunction or disjunction.
- For LS of any type, a disjunction with an atomic LS can be added.

Below are two example refinement chains of ψ:

1. $\bot \rightsquigarrow_\psi A_1 \setminus A_2 \rightsquigarrow_\psi A_1 \rightsquigarrow_\psi A_1 \sqcup A_2 \setminus A_3$
2. $\bot \rightsquigarrow_\psi A_1 \setminus A_2 \sqcap A_3 \setminus A_4 \rightsquigarrow_\psi A_1 \sqcap A_3 \setminus A_4 \rightsquigarrow_\psi A_1 \sqcap A_3 \rightsquigarrow_\psi (A_1 \sqcap A_3) \sqcup (A_5 \setminus A_6)$

ψ is an upward refinement operator with the following properties.

Proposition 1. *ψ is an upward refinement operator.*

Proof. For an arbitrary LS L, we have to show for any element $L' \in \psi(L)$ that $L \sqsubseteq L'$ holds. The proof is straightforward by showing that L' cannot generate less links than L via case distinction and structural induction over LS:

- $L = \bot$: Trivial.
- L is atomic: Adding a disjunction cannot result in less links (this also holds for the cases below).
- L is of the form $L_1 \setminus L_2$: $L' = L_1$ cannot result in less links.
- L is a conjunction / disjunction: L' cannot result in less links by structural induction.

□

Proposition 2. *ψ is weakly complete.*

Proof. To show this, we have to show that an arbitrary LS L can be reached from the $\bot$ LS. First, we convert everything to negation normal form by pushing $\setminus$ inside, e.g. LS of the form $L_1 \setminus (L_2 \sqcap L_3)$ are rewritten to $(L_1 \setminus L_2) \sqcup (L_1 \setminus L_3)$ and LS of the form $L_1 \setminus (L_2 \sqcup L_3)$ are rewritten to $(L_1 \setminus L_2) \sqcap (L_1 \setminus L_3)$ exhaustively. We then further convert the LS to conjunction normal including an exhaustive application of the distribute law, i.e., conjunctions cannot be nested within disjunctions. The resulting LS is dubbed L' and equivalent to L. We show that L' can always be reached from $\bot$ via induction over its structure:

- $L' = \bot$: Trivial via the empty refinement chain.
- $L' = A$ (atomic): Reachable via $\bot \rightsquigarrow_\psi A \setminus A' \rightsquigarrow_\psi A$.
- $L' = A_1 \setminus A_2$ (atomic negation): Reachable directly via $\bot \rightsquigarrow_\psi A_1 \setminus A_2$.
- L' is a conjunction with m elements: $\bot \rightsquigarrow_\psi A_{i_1} \setminus A_{j_1} \sqcap \cdots \sqcap A_{i_m} \setminus A_{j_m}$ where an element $A_{i_k} \setminus A_{j_k}$ is chosen as follows: Let the k-th element of conjunction L' be L''.
  - If L'' is an atomic specification A, then $A_{i_k} = A$ (A_{j_k} can be arbitrarily).
  - If L'' is an atomic negation $A_1 \setminus A_2$, then $A_{i_k} = A$ and $A_{j_k} = A_2$.
  - If L'' is a disjunction, the first element of this disjunction falls into one the above two cases and A_{i_k} and A_{j_k} can be set as described there.

  Each element of L'' is then further refined to L' as follows:
  - If L'' is an atomic specification A: $A \setminus A_{j_k}$ is refined to A.

- If L'' is an atomic negation $A_1 \setminus A_2$: No further refinements are necessary.
- If L'' is a disjunction. The first element of the disjunction is first treated according to the two cases above. Subsequent elements of the disjunction are either atomic LS or atomic negation and can be added straightforwardly as the operator allows adding disjunctive elements to any non-$\bot$ LS.

Please note that the case distinction is exhaustive as we assume L' is in conjunctive negation normal form, i.e., there are no disjunctions on the outer level, negation is always atomic, conjunctions are not nested within other conjunction and elements of disjunctions within conjunctions cannot be conjunctions. □

Proposition 3. *ψ is finite, not proper and redundant.*

Proof. Finiteness: There are only finitely many atomic LS. Hence, there are only finitely many atomic negations and, consequently, finitely many possible conjunctions of those. Consequently, $\psi(\bot)$ is finite. The finiteness of $\psi(L)$ with $L \neq \bot$ is straightforward.

Properness: The refinement chain $\bot \rightsquigarrow^*_\psi A_1 \sqcap A_2 \rightsquigarrow^*_\psi (A_1 \sqcup A_2) \sqcap A_2$ is a counterexample.

Redundancy: The two refinement chains $A_1 \sqcap A_3 \rightsquigarrow^*_\psi (A_1 \sqcup A_2) \sqcap A_3 \rightsquigarrow^*_\psi (A_1 \sqcup A_2) \sqcap (A_3 \sqcup A_4)$ and $A_1 \sqcap A_3 \rightsquigarrow^*_\psi A_1 \sqcap (A_3 \sqcap A_4) \rightsquigarrow^*_\psi (A_1 \sqcup A_2) \sqcap (A_3 \sqcup A_4)$ are a counterexample. □

Naturally, the restrictions of ψ (being redundant and not proper) raise the question whether there are LS refinement operators satisfying all theoretical properties:

Proposition 4. *There exists a weakly complete, finite, proper and non-redundant refinement operator in $\mathcal{L}$.*

Proof. Let C be the set of LS in $\mathcal{L}$ in conjunctive negation normal form without any LS equivalent to $\bot$. We define the operator α as $\alpha(\bot) = C$ and $\alpha(L) = \emptyset$ for all $L \neq \bot$. α is obviously complete as any LS has an equivalent in conjunctive negation normal form. It is finite as S can be be shown to the finite with an extended version of the argument in the finiteness proof of ψ. α is trivially non-redundant and it is proper by definition. □

The existence of an operator which satisfies all considered theoretical criteria of a refinement operator is an artifact of only finitely many semantically inequivalent LS existing in $\mathcal{L}$. This set is however extremely large and not even small fractions of it can be evaluated in all but very simple cases. For example, the operator α as $\alpha(\bot) = C$ and $\alpha(L) = \emptyset$ for all $L \neq \bot$ is trivially non-redundant and it is proper by definition. Such an operator α is obviously not useful as it does not help *structuring the search space*. Providing a useful way

to structure the search space is the main reason for refinement operators being successful for learning in other complex languages as it allows to gradually converge towards useful solutions while being able to prune other paths which cannot lead to promising solutions (explained in the next section). This is a reason why we sacrificed properness and redundancy for a better structure of the search space.

WOMBAT ALGORITHM

We have now introduced all ingredients necessary for defining the WOMBAT algorithms. The first algorithm, which we refer to as *simple* version, uses the operator φ, whereas the second algorithm, which we refer to as *complete*, uses the refinement operator ψ. The complete algorithm has the following specific characteristics: First, while ψ is finite, it would generate a prohibitively large number of refinements when applied to the $\bot$ concept. For that reason, those refinements will be computed stepwise as we will illustrate below. Second, as ψ is an upward refinement operator it allows to prune parts of the search space, which we will also explain below. We only explain the implementation of the complex WOMBAT algorithm as the other is a simplification excluding those two characteristics.

Algorithm 7 shows the individual steps of WOMBAT complete. Our approach takes the source data set S, the target data set T, examples $E \subseteq S \times T$ as well as the property coverage threshold and the set of considered similarity functions as input. In Line 3, the property matches are computed by optimizing the threshold for properties that have the minimum coverage (Line 7) as described in Section 7.2.1. The main loop starts in Line 13 and runs until a termination criterion is satisfied, e.g. 1) a fixed number of LS has been evaluated, 2) a certain time has elapsed, 3) the best F-score has not changed for a certain time or 4) a perfect solution has been found. Line 14 states that a heuristic-based search strategy is employed. By default, we employ the F-score directly. More complex heuristics introducing a bias towards specific types of LS could be encoded here. In Line 15, we make a case distinction: Since the number of refinements of $\bot$ is extremely high and not feasible to compute in most cases, we perform a stepwise approach: In the first step, we only add simple LS of the form $A_i \sqcap A_j$ as refinements (Line 17). Later, in Line 22, we add more complex conjunctions if the simpler forms are promising. Apart from this special case, we apply the operator directly. Line 24 updates the search tree by adding the nodes obtained via refinement. Moreover, it contains a redundancy elimination procedure: We only add those nodes to the search tree which are not already contained in it.

The subsequent part starting from Line 26 defines our *pruning procedure*: Since ψ is an upward refinement operator, we know that the set of links generated by a child node is a superset of or equal to the

Algorithm 7: WOMBAT Learning Algorithm

Input: Sets of resources S and T; examples $E \subseteq S \times T$; property coverage threshold τ; set of similarity functions **F**

```
1  A ⟵ null;
2  i ⟵ 1 ;
3  foreach property p_s ∈ S do
4      if coverage(p_s) ⩾ τ then
5          foreach property p_t ∈ T do
6              if coverage(p_t) ⩾ τ then
7                  Find atomic metric m(p_s, p_t) that leads to highest
                    F-measure;
8                  Optimize similarity threshold for m(p_s, p_t) to find
                    best mapping A_i;
9                  Add A_i to A;
10                 i ⟵ i + 1;
11 Γ ⟵ ⊥ (initiate search tree Γ to the root node ⊥);
12 F_best ⟵ 0, L_best ⟵ null;
13 while termination criterion not met do
14     Choose the node with highest scoring LS L in Γ;
15     if L == ⊥ then
16         foreach A_i, A_j ∈ A, where i ≠ j do
17             Only add refinements of form A_i \ A_j;
18     else
19         Apply operator to L;
20         if L is a refinement of ⊥ then
21             foreach A_i, A_j ∈ A, where i ≠ j do
22                 In addition to refinements, add conjunctions with
                    specifications of the form A_i \ A_j as siblings;
23     foreach refinement L′ do
24         if L′ is not already in the search tree Γ then
25             Add L′ to Γ as children of the node containing L;
26     Update F_best and L_best;
27     if F_best has increased then
28         foreach subtree t ∈ Γ do
29             if F_best > F_max(t) then
30                 Delete t;
31 Return L_best;
```

set of links generated by its parent. Hence, while both precision and recall can improve in subsequent refinements, they cannot rise arbitrarily. Precision is bound as false positives cannot disappear during

generalisation. Furthermore, the achievable recall r_{max} is that of the most general constructable LS, i.e., $\mathcal{A} = \bigcup A_i$ This allows to compute an upper bound on the achievable F-score. In order to do so, we first build a set S' with those resources in S occurring in the input examples E as well as a set T' with those resources in T occurring in E. The purpose of those is to restrict the computation of F-score to the fragment $S' \times T' \subseteq S \times T$ relevant for example set E. We can then compute an upper bound of precision of a LS L as follows:

$$p_{max}(L) = \frac{|E|}{|E| + |\{(s,t) \mid (s,t) \in [[L]], s \in S' \text{ or } t \in T'\} \setminus E|}$$

F_{max} is then computed as the F-measure obtained with recall r_{max} and precision p_{max}, i.e., $F_{max} = \frac{2p_{max}r_{max}}{p_{max}+r_{max}}$. It is an upper bound for the maximum achievable F-measure of any node reachable via refinements. We can disregard all nodes in the search tree which have a maximum achievable F-score that is lower than the best F-score already found. This is implemented in Line 28. The pruning is conservative in the sense that no solutions are lost. In the evaluation, we give statistics on the effect of pruning. WOMBAT ends by returning L_{best} as the best LS found, which is the specification with the highest F-score. In case of ties, we prefer shorter specifications over long ones. Should the tie persist, then we prefer specifications that were found early.

Proposition 5. WOMBAT *is complete, i.e., it will eventually find the* LS *with the highest F-measure within* $\mathcal{L}$.

Proof. This is a consequence of the weak completeness of ψ and the fact that the algorithm will eventually generate all refinements of ψ. For the latter, we have to look at the refinement of $\bot$ as a special case since otherwise a straightforward application of ψ is used. For the refinements of $\bot$ it is easy to show via induction over the number of conjunctions in refinements that any element in $\psi(\bot)$ can be reached via the algorithm. (The pruning is conservative and only prunes nodes never leading to better solutions.) □

EVALUATION

We evaluated our approach using 8 benchmark data sets. Five of these benchmarks were real-world data sets while three were synthetic. The real-world interlinking tasks used were those in [Köpcke et al., 2010]. The synthetic data sets were from the OAEI 2010 benchmark[3]. All experiments were carried out on a 64-core 2.3 GHz PC running *OpenJDK* 64-Bit Server 1.7.0_75 on *Ubuntu* 14.04.2 LTS. Each experiment was assigned 20 GB RAM.

3 http://oaei.ontologymatching.org/2010/

Table 5: 10-fold cross validation F-Measure results.

Data set	WOMBAT Simple	WOMBAT Complete	EUCLID Linear	EUCLID Conjunction	EUCLID Disjunction	EAGLE
Person 1	**1.00**	**1.00**	0.64	0.97	**1.00**	0.99
Person 2	**1.00**	0.99	0.22	0.78	0.96	0.94
Restaurants	**0.98**	0.97	0.97	0.97	0.97	0.97
DBLP-ACM	0.97	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**
Abt-Buy	0.60	0.61	0.06	0.06	0.52	**0.65**
Amazon-GP	0.70	0.67	0.59	0.71	**0.73**	0.71
DBP-LMDB	0.99	**1.00**	0.99	0.99	0.99	0.99
DBLP-GS	**0.94**	**0.94**	0.90	0.91	0.91	0.93
Average	**0.90**	**0.90**	0.67	0.80	0.88	**0.90**

For testing WOMBAT against the benchmark data sets in both its simple and complete version, we used the `jaccard`, `trigrams`, `cosine` and `qgrams` similarity measures. We used two termination criteria: Either a LS with F-measure of 1 was found or a maximal depth of refinement (10 resp. 3 for the simple resp. complete version) was reached . This variation of the maximum refinement trees sizes between the simple and complete version was because WOMBAT complete adds a larger number of nodes to its refinement tree in each level. The coverage threshold τ was set to 0.6. A more complete list of evaluation results are available at the project web site.[4] Altogether, we carried out 6 sets of experiments to evaluate WOMBAT.

In *the first set of experiments*, we compared the average F-Measure achieved by the simple and complete versions of WOMBAT to that of four other state-of-the-art LS learning algorithms within a 10-fold cross validation setting. The other four LS learning algorithms were EAGLE [Ngonga Ngomo and Lyko, 2012] as well as the *linear*, *conjunctive* and *disjunctive* versions of EUCLID [Ngonga Ngomo and Lyko, 2013]. EAGLE was configured to run 100 generations. The mutation and crossover rates were set to 0.6 as in [Ngonga Ngomo and Lyko, 2012]. To address the non-deterministic nature of EAGLE, we repeated the whole process of 10-fold cross validation 5 time and present the average results. EUCLID's grid size was set to 5 and 100 iterations were carried out as in [Ngonga Ngomo and Lyko, 2013]. The results of the evaluation are presented in Table 5. The simple version of WOMBAT was able to outperform the state-of-the-art approaches in 4 out of the 8 data sets and came in the second position in 2 data sets. WOMBAT complete was able to achieve the best F-score in 4 data sets and achieve the second best F-measure in 3 data sets. On average, both ver-

4 https://github.com/AKSW/LIMES/tree/master/evaluationsResults/wombat

Table 6: A comparison of WOMBAT F-Measure against 4 state-of-the-art approaches on 8 different benchmark data sets using 30% of the original data as training data.

Data set	WOMBAT Simple	WOMBAT Complete	EUCLID Linear	EUCLID Conjunction	EUCLID Disjunction	EAGLE
Person 1	**1.00**	**1.00**	0.95	0.96	0.99	0.92
Person 2	**0.99**	0.79	0.80	0.82	0.88	0.69
Restaurants	**0.97**	0.88	0.87	0.84	0.89	0.88
DBLP-ACM	**0.95**	0.91	0.88	0.89	0.91	0.85
Abt-Buy	**0.44**	0.40	0.29	0.29	0.29	0.27
Amazon-GP	**0.54**	0.41	0.31	0.30	0.32	0.32
DBP-LMDB	**0.98**	**0.98**	0.97	0.96	0.97	0.89
DBLP-GS	**0.91**	0.74	0.83	0.76	0.74	0.69
Average	**0.85**	0.76	0.74	0.73	0.75	0.69

sions of WOMBAT were able to achieve an F-measure of 0.9, by which WOMBAT outperforms the three version of EUCLID by an average of 11%. While WOMBAT was able to achieve the same performance of EAGLE in average, WOMBAT is still to be preferred as (1) WOMBAT only requires positive examples and (2) EAGLE is indeterministic by nature.

For *the second set of experiments*, we implemented an evaluation protocol based on the assumptions made at the beginning of this chapter. Each input data set was split into 10 parts of the same size. Consequently, we used 3 parts (30%) of the data as training data and the rest 7 parts (70%) for testing. This was to implement the idea of the data set growing and the specification (and therewith the links) for the new version of the data set having to be derived by learning from the old data set. During the learning process, the score function was the F-measure achieved by each refinement of the portion of the training data related to $S \times T$ selected for training (dubbed $S' \times T'$ previously). The F-measures reported are those achieved by LS on the test data set. We used the same settings for EAGLE and EUCLID as in the experiments before. The results (see Table 6) show clearly that our simple operator outperforms all other approaches in this setting. Moreover, the complete version of WOMBAT reaches the best F-measure on 2 data sets and the second-best F-measure on 3 data sets. This result of central importance as it shows that WOMBAT is well suited for the task for which it was designed. Interestingly, our approach also outperforms the approaches that rely on negative examples (i.e. EUCLID and EAGLE). The complete version of WOMBAT seems to perform worse than the simple version because it can only explore a tree of depth 3. However, this limitation was necessary to test both implementations using the same hardware.

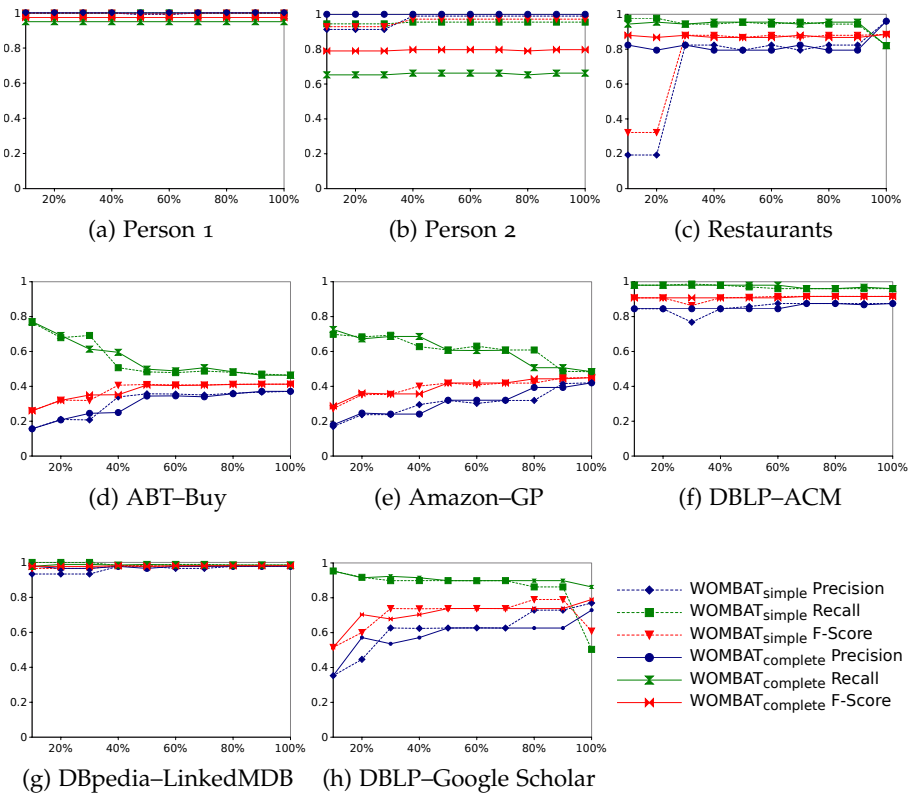


Figure 16: Precision, Recall and F-score results of applying Wombat on the benchmark data sets. The x-axis represents the fraction of positive examples used from the gold standard for training.

In the *third set of experiments*, we measured the effect of increasing the amount of training data on the precision, recall and F-score achieved by both simple and complete versions of Wombat. The results are presented in Figure 16. Our results suggest that the complete version of Wombat is partly more stable in its results (see ABT-Buy and DBLP-Google Scholar) and converges faster towards the best solution that it can find. This suggests that once trained on a data set, our approach can be used on subsequent versions of real data sets, where a small number of novel resources is added in each new version, which is the problem setup considered in this chapter. On the other hand, the simple version is able to find better LS as it can explore longer sequences of mappings.

In the *fourth set experiments*, we measured the learning time for each of the benchmark data sets. The results are also presented in Figure 17. As expected, the simple approach is time-efficient to run even without any optimization. While the complete version of Wombat without pruning is significantly slower (up to 1 order of magnitude), the effect of pruning can be clearly seen as it reduces the runtime of the algorithm while also improving the total space that the complete version of Wombat can explore. These results are corroborated by our

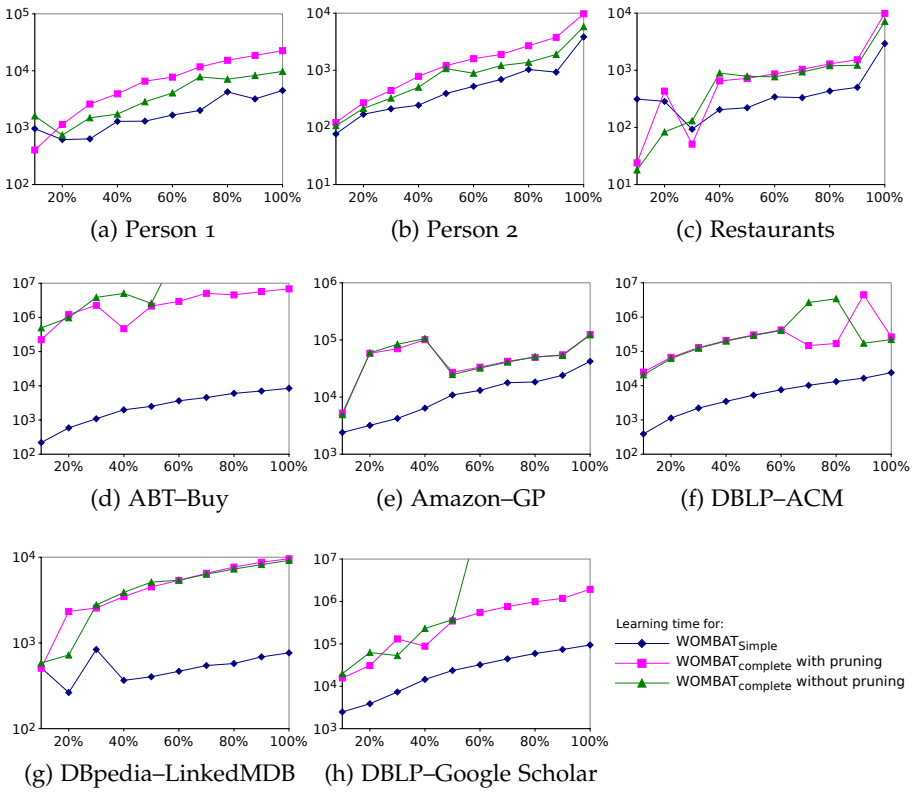


Figure 17: Runtime results of applying WOMBAT on the benchmark data sets. The x-axis represents the fraction of positive examples from the gold standard used for training, the y-axis represents the learning time in milliseconds with a time out of 10^7 ms (processes running above this upper limit were terminated). All plots are in log scale.

fifth set of experiments, in which we evaluated the pruning technique of the complete version of WOMBAT. In those experiments, for each of aforementioned benchmark data sets we computed what we dubbed as *pruning factor*. The pruning factor is the number of searched nodes (search tree size plus pruned nodes) divided by the maximum size of the search tree (which we set to 2000 nodes in this set of experiments). The results are presented in Table 7. Our average *pruning factor* of 2.55 shows that we can discard more than 3000 nodes while learning specifications.

In *a final set of experiments*, we compared the two versions of WOMBAT against the 2 systems proposed in [Kejriwal and Miranker, 2015]. To be comparable, we used the same evaluation protocol in [Kejriwal and Miranker, 2015], where 2% of the gold standard was used as training data and the remaining 98% of the gold standard as test data. The results (presented in Table 8) suggests that WOMBAT is capable of achieving better or equal performance in 4 out of the 6 evaluation data sets. While WOMBAT achieved inferior F-measures for the other 2 data sets, it should be noted that the competing systems are op-

Table 7: The *pruning factor* of the benchmark data sets.

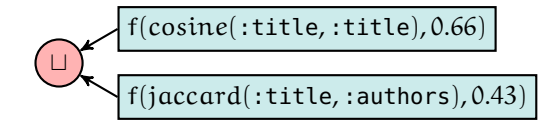
Data set	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Person 1	1.57	2.13	1.85	2.13	2.13	2.13	2.13	2.13	2.13	2.13
Person 2	1.29	1.29	1.57	1.57	1.57	1.57	1.57	1.57	1.57	1.57
Restaurant	1.17	1.45	1.17	1.45	1.45	1.45	1.45	1.45	1.45	1.45
DBLP-ACM	6.23	5.58	6.79	6.85	6.85	6.85	6.79	6.79	6.93	6.79
Abt-Buy	3.38	3.00	3.00	3.39	3.39	3.39	1.79	3.39	3.39	3.39
Amazon-GP	1.14	1.38	1.33	1.37	1.38	1.45	1.54	1.59	1.60	1.60
DBP-LMDB	1.00	1.86	2.86	1.86	1.86	2.33	2.36	2.36	2.36	2.36
DBLP-GS	1.79	1.93	2.01	2.36	2.45	1.66	2.44	2.26	1.97	2.05

Table 8: Comparison of Wombat F-Measure against the approaches proposed in [Kejriwal and Miranker, 2015] on 6 benchmarks using 2% of the original data as training data.

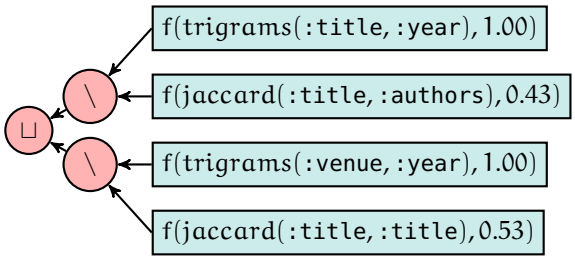
Data set	Pessimistic	Re-weighted	Simple	Complete
Persons 1	**1.00**	**1.00**	**1.00**	**1.00**
Persons 2	0.97	**1.00**	0.80	0.84
Restaurants	0.95	0.94	**0.98**	0.88
DBLP-ACM	0.93	**0.95**	0.94	0.94
Amazon-GP	0.39	0.43	**0.53**	0.45
Abt-Buy	0.36	**0.37**	**0.37**	0.36
Average	0.77	**0.78**	0.77	0.74

timised for a low number of examples and they also get negative examples as input. Overall, these results can thus be regarded as positive as they suggest that our approach can generalise a small number of examples to a sensible LS.

Overall, our results show that ψ and φ are able to learn high-quality LS using only positive examples. When combined with our pruning algorithm, the complete version of ψ achieves runtimes that are comparable to those of φ. Given its completeness, ψ can reach specifications that simply cannot be learned by φ (see Figure 18 for an example of such a LS). However, for practical applications, φ seems to be a good choice.



(a) WOMBAT simple LS



(b) WOMBAT complete LS

Figure 18: Best LS learned by WOMBAT for the *DBLP-GoogleScholar* data set.

8

DEER – AUTOMATING RDF DATASET TRANSFORMATION AND ENRICHMENT

In this chapter we present DEER*, a supervised approach for automating RDF data set transformation and enrichment. A paper about the work is published in ESWC'15 [Sherif et al., 2015]. The author shaped the main ideas and algorithms in this chapter together with the other two authors. Moreover, all the proposed algorithms in this chapter were implemented by the author, who also carried out the evaluations and co-wrote the paper.*

In the previous chapters, we proposed a set of approaches to address various challenges concerning the knowledge bases integration. In this chapter, we introduce a framework capable of combining the previously proposed approaches for automating the knowledge base enrichment and transformation.

With the adoption of Linked data come novel challenges pertaining to the integration of various knowledge bases for dedicated applications such as tourism, question answering, enhanced reality and many more. Providing consolidated and integrated data sets for these applications demands the specification of data enrichment pipelines, which describe how data from different sources is to be integrated and altered so as to abide by the precepts of the application developer or data user. Currently, most developers implement customized pipelines by compiling sequences of tools manually and connecting them via customized scripts. While this approach most commonly leads to the expected results, it is time-demanding and resource-intensive. Moreover, the results of this effort can most commonly only be reused for new versions of the input data but cannot be ported easily to other data sets. Over the last years, a few frameworks for RDF data enrichment such as LDIF[1] and DEER[2] have been developed. The frameworks provide enrichment methods such as entity recognition [Speck and Ngonga Ngomo, 2014], Link Discovery (LD) [Ngonga Ngomo, 2012] and schema enrichment [Buhmann and Lehmann, 2013]. However, devising appropriate configurations for these tools can prove a difficult endeavour, as the tools require (1) choosing the right sequence of enrichment functions and (2) configuring these functions adequately. Both the first and second task can be tedious.

In this chapter, we address this problem by presenting a supervised machine learning approach for the automatic detection of enrichment pipelines based on a refinement operator and self-configuration algorithms for enrichment functions. Our approach takes pairs of CBDs of resources $\{(k_1, k_1') \dots (k_n, k_n')\}$ as input, where k_i' is the enriched version of k_i. Based on these pairs, our approach can learn sequences of atomic enrichment functions that aim to generate each k_i' out of the corresponding k_i. The output of our approach is an enrichment pipeline that can be used on whole data sets to generate enriched versions.

1 http://ldif.wbsg.de/
2 http://aksw.org/Projects/DEER.html

Overall, we provide the following core contributions: (1) We define a supervised machine learning algorithm for learning data set enrichment pipelines based on a refinement operator. (2) We provide self-configuration algorithms for five atomic enrichment steps. (3) We evaluate our approach on eight manually defined enrichment pipelines on real data sets.

NOTATION

Let $\mathcal{K}$ be the set of all RDF knowledge bases. Let $K \in \mathcal{K}$ be a finite RDF knowledge base. K can be regarded as a set of triples $(s, p, o) \in (\mathcal{R} \cup \mathcal{B}) \times \mathcal{P} \times (\mathcal{R} \cup \mathcal{L} \cup \mathcal{B})$, where $\mathcal{R}$ is the set of all resources, $\mathcal{B}$ is the set of all blank nodes, $\mathcal{P}$ the set of all predicates and $\mathcal{L}$ the set of all literals. Given a knowledge base K, the idea behind *knowledge base enrichment* is to find an *enrichment pipeline* $E : \mathcal{K} \to \mathcal{K}$ that maps K to an enriched knowledge base K′ with $K' = E(K)$. We define E as an ordered list of *atomic enrichment functions* $e \in \mathcal{E}$, where $\mathcal{E}$ is the set of all atomic enrichment functions. $2^{\mathcal{E}}$ is used to denote the power set of $\mathcal{E}$, i.e. the set of all enrichment pipelines. The order of elements in E determines the execution order, e.g. for an $E = (e_1, e_2, e_3)$ this means that e_1 will be executed first, then e_2, finally e_3. Formally,

$$E = \begin{cases} \phi & \text{if } K = K', \\ (e_1, \ldots, e_n), \text{where } e_i \in \mathcal{E}, 1 \leqslant i \leqslant n & \text{otherwise,} \end{cases} \tag{19}$$

where ϕ is the empty sequence. Moreover, we denote the number of elements of E with $|E|$. Considering that a knowledge base is simply a set of triples, the task of any atomic enrichment function is to (1) determine a set of triples Δ^+ to be added the source knowledge base and/or (2) determine a set of triples Δ^- to be deleted from the source knowledge base. Any other enrichment process can be defined in terms of Δ^+ and Δ^-, e.g. altering triples can be represented as combination of addition and deletion.

In this chapter we cover two problems: (1) how to create self-configurable atomic enrichment functions $e \in \mathcal{E}$ capable of enriching a data set and (2) how to automatically generate an enrichment pipeline E. As a running example, we use the portion of *DrugBank* shown in Figure 19. The goal of the enrichment here is to gather information about companies related to drugs for a market study. To this end, the `owl:sameAs` links to *DBpedia* (prefix `db`) need to be dereferenced. Their `rdfs:comment` then needs to be processed using an entity spotter that will help retrieve resources such as the `Boots Company`. Then, these resources need to be attached directly to the resources in the source knowledge base, e.g., by using the `:relatedCompany` property. Finally, all subjects need to be conformed under one subject authority (prefix `ex`).

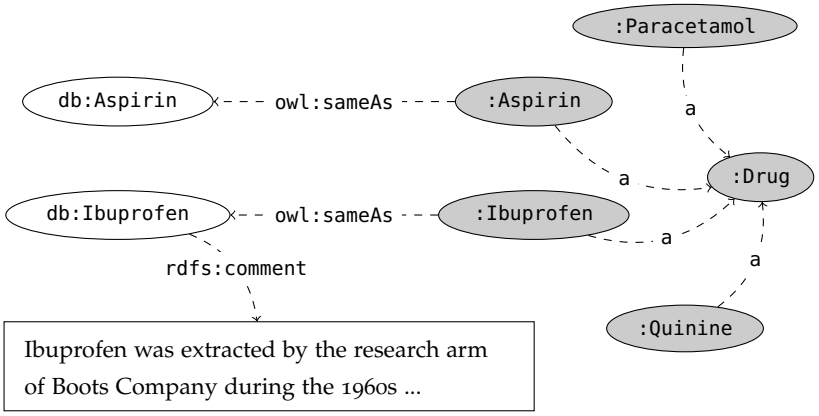
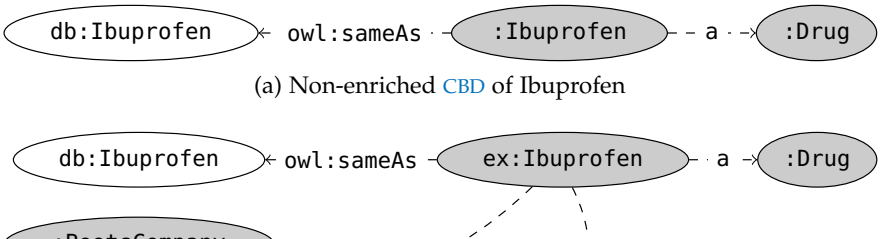


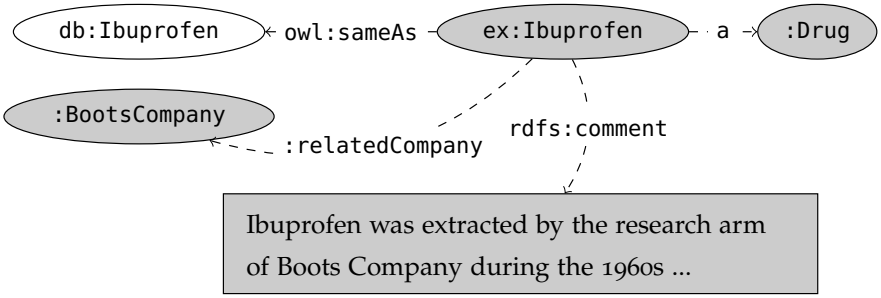
Figure 19: RDF graph of the running example. Ellipses are RDF resources, literals are rectangular nodes. Gray nodes stand for resources in the input knowledge base while nodes with a white background are part of an external knowledge base.

KNOWLEDGE BASE ENRICHMENT REFINEMENT OPERATOR

In this section, we present our refinement operator for learning enrichment pipelines and prove some of its theoretical characteristics. Our formalization is based on the general refinement operator presented in Section 2.2. Our refinement operator expects the set of atomic enrichment functions $\mathcal{E}$, the source knowledge base K as well as a set of positive examples X^+ as input, and returns an enrichment pipeline E as output. Each positive example $x^+ \in X^+$ is a pair of CBDs (k, k'), with $k \subseteq K$ and $k' \subseteq K'$, the K' stands for the enriched version of K. Note that we model CBDs as sets of RDF triples. Moreover, we denote the resource with the CBD k as *resource*(k). For our running example, the set X^+ could contain the pair shown in Figure 20 (a) as k and in Figure 20 (b) as k'.



(a) Non-enriched CBD of Ibuprofen



(b) Enriched CBD of Ibuprofen

Figure 20: Ibuprofen CBD before and after enrichment.

The set of all first elements of the pairs contained in X^+ is denoted *source*(X^+), while the set of all second elements is denoted *target*(X^+). To compute the refinement pipeline E, we employ an upward refinement operator (which we dub ρ) over the space $2^{\mathcal{E}}$ of all enrichment pipelines. We write $E \supseteq E'$ when E' is a subsequence of E, i.e., $e'_i \in E' \rightarrow e'_i = e_i$, where e_i resp. e'_i is the i^{th} element of E resp. E'.

Proposition 6 (Induced quasi-ordering)**.** $\supseteq$ *induces a quasi-ordering over the set* $2^{\mathcal{E}}$.

Proof. The reflexivity of $\supseteq$ follows from each E being a subsequence of itself. The transitivity of $\supseteq$ follows from the transitivity of the subsequence relation. Note that $\supseteq$ is also antisymmetric. □

We define our refinement operator over the space $(2^{\mathcal{E}}, \supseteq)$ as follows:

$$\rho(E) = \bigcup_{\forall e \in \mathcal{E}} E +\!\!+ e \qquad (+\!\!+ \text{ is the list append operator}) \tag{20}$$

We define precision $P(E)$ and recall $R(E)$ achieved by an enrichment pipeline on $\mathcal{E}$ as

$$P(E) = \frac{\left| \bigcup_{k \in source(X^+)} E(k) \bigcap \bigcup_{k' \in target(X^+)} k' \right|}{\left| \bigcup_{k \in source(X^+)} E(k) \right|}, \tag{21}$$

$$R(E) = \frac{\left| \bigcup_{k \in source(X^+)} E(k) \bigcap \bigcup_{k' \in target(X^+)} k' \right|}{\left| \bigcup_{k' \in target(X^+)} k' \right|}. \tag{22}$$

The F-measure $F(E)$ is then

$$F(E) = \frac{2P(E)R(E)}{P(E) + R(E)}. \tag{23}$$

Using Figure 20 (a) from our running example as source and Figure 20 (b) as target with the CBD of `:Iboprufen` being the only positive example, an empty enrichment pipeline $E = \phi$ would have a precision of 1, a recall of $\frac{3}{4}$ and an F-measure of $\frac{6}{7}$. Having defined our refinement operator, we now show that ρ is finite, proper, complete and not redundant.

Proposition 7. ρ *is finite.*

Proof. This is a direct consequence of $\mathcal{E}$ being finite. □

Proposition 8. ρ *is proper.*

Proof. As the quasi order is defined over subsequences, i.e. the space $(2^{\mathcal{E}}, \supseteq)$, and we have $|E'| = |E| + 1$ for any $E' \in \rho(E)$, ρ is trivially proper. □

Proposition 9. *ρ is complete.*

Proof. Let E resp. E' be an enrichment pipeline of length n resp. n' with $E' \supseteq E$. Moreover, let e'_i be the i^{th} element of E'. Per definition, $E +\!\!+ e'_{n+1} \in \rho(E)$. Hence, by applying ρ $n' - n$ times, we can generate E' from E. We can thus conclude that ρ is complete. □

Proposition 10. *ρ is not redundant.*

Proof. ρ being redundant would mean that there are two refinement chains that lead to a single refinement pipeline E. As our operator is equivalent to the list append operation, it would be equivalent to stating that two different append sequences can lead to the same sequence. This is obviously not the case as each element of the list E is unique, leading to exactly one sequence that can generate E. □

LEARNING ALGORITHM

The learning algorithm is inspired by refinement-based approaches from inductive logic programming. In these algorithms, a search tree is iteratively built up using heuristic search via a fitness function. We formally define a node N in a search tree to be a triple (E, f, s), where E is the *enrichment pipeline*, $f \in [0, 1]$ is the F-measure of E (see Equation 23), and $s \in \{\mathtt{normal}, \mathtt{dead}\}$ is the status of the node. Given a search tree, the heuristic selects the fittest node in it, where fitness is based on both F-measure and complexity as defined below.

Approach

For the automatic generation of enrichment pipeline specifications, we created a learning algorithm based on the previously defined refinement operator. Once provided with training examples, the approach is fully automatic. The pseudo-code of our algorithm is presented in Algorithm 8.

Our learning algorithm has two inputs: a set of positive examples X^+ and a set of atomic enrichment operators $\mathcal{E}$. X^+ contains pairs of (k, k') where each k contains a CBD of one resource from an arbitrary source knowledge base K and k' contains the CBD of the same resource after applying some manual enrichment. Given $\mathcal{E}$, the goal of our algorithm is to learn an enrichment pipeline E that maximizes $F(E)$ (see Equation 23).

As shown in Algorithm 8, our approach starts by generating an empty refinement tree τ which contains only an empty root node. Using X^+, the algorithm then accumulates all the original CBDs in k

(SOURCE(X^+)). Using the same procedure, k' is accumulated from X^+ as the knowledge base containing the enriched version of k (TARGET(X^+)). Until a termination criterion holds (see Section 8.3.3), the algorithm keeps expanding the most promising node (see Section 8.3.2). Finally, the algorithm ends by returning the best pipeline found in τ: (GetPipeline(GetMaxQualityNode(τ))).

Having a *most promising node* t at hand, the algorithm first applies our refinement operator (see Equation 20) against the most promising enrichment pipeline E_{old} included in t to generate a set of atomic enrichment functions $\mathcal{E} \leftarrow \rho(E_{old})$. Consequently, using both k_{old} (as the knowledge base generated by applying E_{old} against k) and k', the algorithm applies the self configuration process of the current atomic enrichment function $e \leftarrow$ SELFCONFIG(e, k_{old}, k) to generate a set of parameters P (a detailed description for this process is found in Section 8.4). Afterwards, the algorithm runs e against k_{old} to generate the new enriched knowledge base $k_{new} \leftarrow e(k_{old}, P)$. A dead node $N \leftarrow$ CREATENODE(E, 0, $dead$) is created in two cases: (1) e is inapplicable to k_{old} (i.e., $P == null$) or (2) e does no enrichment at all (i.e., k_{new} is isomorphic[3] to k_{old}). Otherwise, the algorithm computes the F-measure f of the generated data set k_{new}. E along with f are then used to generate a new search tree node $N \leftarrow$ CREATENODE(E, f, $normal$)). Finally, N is added as a child of t (ADDCHILD(t, N)).

Most Promising Node Selection

Here we describe the process of selecting the most promising node $t \in \tau$ as in GETMOSTPROMISINGNODE() subroutine in Algorithm 8. First, we define *node complexity* as linear combination of the node's children count and level. Formally,

Definition 12 (Node Complexity). $c(N, \tau) = \alpha \frac{|N_d|}{|\tau|} + \beta \frac{N_l}{\tau_d}$, $|N_d|$ *is number of all* N*'s descendant nodes,* $|\tau|$ *is the total number of nodes in* τ, N_l *is* N*'s level,* τ_d *is* τ*'s depth,* α *is the children penalty weight,* β *is the level penalty weight and* $\alpha + \beta = 1$.[4]

We can then define the fitness $f(N)$ of a *normal* node N as the difference between its enrichment pipeline F-measure (Equation 23) and weighted complexity. $f(N)$ is zero for *dead* nodes. Formally,

Definition 13 (Node fitness). *Let* $N = (E, f, s)$ *be a node in a refinement tree* τ, N*'s fitness is the function*

$$f(N) = \begin{cases} 0 & \text{if } s = dead, \\ F(E) - \omega \cdot c(N) & \text{if } s = normal. \end{cases} \tag{24}$$

3 http://www.w3.org/TR/rdf11-concepts/

4 Seeking for simplicity, we will use the $c(N)$ instead of $c(N, \tau)$ in the rest of this chapter.

where E *is the enrichment pipeline contained in the node* N, ω *is the complexity weight and* $0 \leqslant \omega \leqslant 1$.

Note, that we use the *complexity* of pipelines as second criterion, which makes the algorithm (1) more flexible in searching less explored areas of the search space, and (2) leads to simpler specification being preferred over more complex ones (Occam's razor [Blumer et al., 1987]). The parameter ω can be used to control the trade-off between a greedy search ($\omega = 0$) and search strategies closer to breadth first search ($\omega > 0$). The fitness function can be defined independently of the core learning algorithm.

Consequently, the most promising node is the node with the maximum fitness through the whole refinement tree τ. Formally, the most promising node t is defined as

$$t = \underset{\forall N \in \tau}{\arg\max}\, f(N), \tag{25}$$

where N is not a *dead* node. Note that if several nodes achieve a maximum fitness, the algorithm chooses the shortest node as it aims to generate the simplest enrichment pipeline possible.

Termination Criteria

The subroutine TerminationCriterionHolds() in Algorithm 8 can check several termination criteria depending on configuration: (1) optimal enrichment pipeline found (i.e., a fixpoint is reached), (2) maximum number of iterations reached, (3) maximum number of refinement tree nodes reached, or a combination of the aforementioned criteria. Note that the termination criteria can be defined independently of the core learning algorithm.

SELF-CONFIGURATION

To learn an appropriate specification from the input positive examples, we need to develop self-configuration approaches for each of our framework's atomic enrichment functions. The input for each of these self-configuration procedures is the same set of positive examples X^+ provided to our pipeline learning algorithm (Algorithm 8). The goal of the self-configuration process of an enrichment function is to generate a set of parameters $P = \{(mp_1, v_1), \ldots, (mp_m, v_m)\}$ able to reflect X^+ as well as possible. In cases when insufficient data is contained in X^+ to carry out the self-configuration process, an empty list of parameters is returned to indicate inapplicability of the enrichment function.

Algorithm 8: Enrichment Pipeline Learner

input : X^+ : Set of positive examples,
$\mathcal{E}$: Set of atomic enrichment functions
output: E : Enrichment pipeline

```
initialize refinement tree τ
τ ← CreateRootNode();
k ← Source(X+) ;
k′ ← Target(X+) ;
repeat
    Expand most promising node of τ;
    t ← GetMostPromisingNode(τ);
    E_old ← GetPipeline(t);
    ℰ ← ρ(E_old);
    Create a child of t for each e ∈ ℰ ;
    for e ∈ ℰ do
        k_old ← E_old(k);
        P ← SelfConfig(e, k_old, k′);
        k_new ← e(k_old, P);
        if P == null or k_new == k_old then
            N ← CreateNode(E, 0, dead);
        else
            f ← F(e);
            N ← CreateNode(E, f, normal);
        AddChild(t, N);
until TerminationCriterionHolds(τ);
return GetPipeline(GetMaxQualityNode(τ));
```

Dereferencing Enrichment Functions

The idea behind the self-configuration process of the enrichment by *dereferencing* is to find the set of predicates D_p from the enriched CBDs that are missing from source CBDs. Formally, for each CBD pair (k, k') construct a set $D_p \subseteq \mathcal{P}$ as follows: $D_p = \{p' : (s', p', o') \in k'\} \backslash \{p : (s, p, o) \in k\}$. The dereferencing enrichment function will *dereference* the object of each triple of k_i given that this object is an external URI, i.e. all o in k_i with $(s, p, o) \in k_i$, $o \in \mathcal{R}$ and o is not in the local namespace of the data set will be dereferenced. Dereferencing an object returns a set of triples. Those are filtered using the previously constructed property set D_p, i.e. when dereferencing o the enrichment function only retains triples with subject o and a predicate contained in D_p. The resulting set of triples is added to the input data set.

We illustrate the process using our running example: In the first step, we compute the set $D_p = \{$`:relatedCompany`, `rdfs:comment`$\}$ which consists of the properties occurring in the target but not in

the source CBD. In the second step, we collect the set of resources to dereference, which only consists of the element `db:Ibuprofen`. In the third step, we perform the actual dereferencing operation and retain triples for which the subject is `db:Ibuprofen` and the predicate is either `:relatedCompany` or `rdfs:comment`. In our example, no triples with predicate `:relatedCompany` exist, but we will find the desired triple (`db:Ibuprofen`, `rdfs:comment`, `"Ibuprofen ..."`), which is then added to the input data set.

Linking Enrichment Function

As introduced in Section 2.1, the aim of LD is as follows: Given two sets $R_s \subseteq \mathcal{R}$ of source resources and $R_t \subseteq \mathcal{R}$ of target resources, we aim to discover links $L \subseteq R_s \times R_t$ such that for any $(s,t) \in L$ we have $\delta(s,t) \geqslant \theta$ where δ is a similarity function and θ a threshold value. The goal of the linking enrichment function is to learn so called LS including a similarity function δ and a threshold θ.

Here we embed WOMBAT *within* DEER.

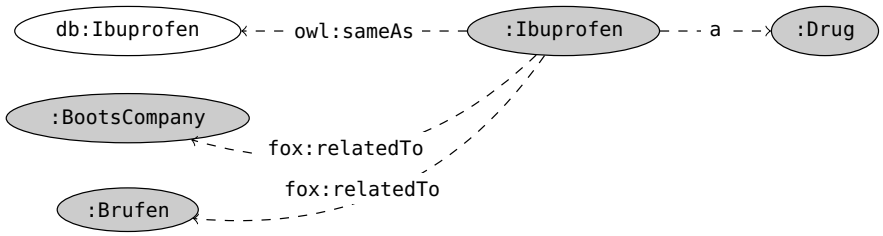
The self-configuration of the linking enrichment function stars by collecting positive linking examples L^+ from the CBDs input of DEER, where $L^+ \subseteq R_s \times R_t$ is sat of pairs of source and target resources. Also, the linking self-configuration finds the linking predicate p_l (for example `owl:sameAs`). Then, L^+ are fed to the WOMBAT algorithm (see Algorithm 7). Based on L^+, WOMBAT is capable of finding LS and generate the links between the source and target data sets. Finally, WOMBAT results are combined using the same linking predicate p_l.

NLP Enrichment Function

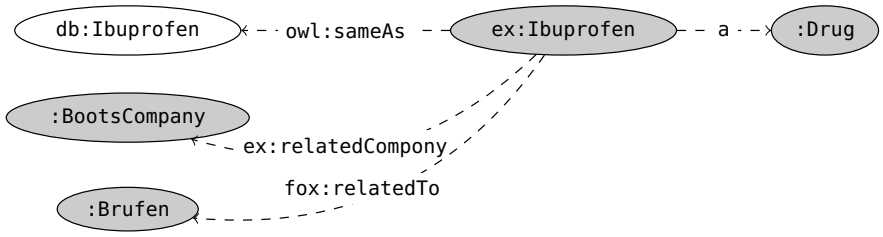
The basic idea here is to enable the extraction of all possible named entity types. If this leads to the retrieval of too many entities, the unwanted predicates and resources can be discarded in a subsequent step. The self-configuration of the Natural-Language Processing (NLP) enrichment function is parameter-free and relies on FOX [Ngonga Ngomo et al., 2011]. The application of the NLP self configuration to our running example generates all possible entities included in the literal object of the `rdfs:comment` predicate. The result is a set of related named entities all of them related to our `ex:Iboprufen` object by the default predicate `fox:relatedTo` as shown Figure 21a. In the following 2 sections we will see how our enrichment functions can refine some of the generated triples and delete others.

Conformation Enrichment Functions

The *conformation*-based enrichment currently allows for both *subject-authority-based conformation* and *predicate-based conformation*. The self-configuration process of *subject-authority-based conformation* starts by



(a) NLP enriched CBD of Ibuprofen



(b) Conformed CBD of Ibuprofen

Figure 21: Ibuprofen CBD after NLP and predicate conformation enrichment.

finding the most frequent subject authority rk in $source(X^+)$. Also, it finds the most frequent subject authority rk' in the target data set $target(X^+)$. Then this self-configuration process generates the two parameters: (`sourceSubjectAuthority`, rk) and (`targetSubjectAuthority`, rk'). After that, the self-configuration process replaces each subject authority rk in $source(X^+)$ by rk'.

Back to our running example, the authority self-conformation process generates the two parameters (`sourceSubjectAuthority`, `":"`) and (`targetSubjectAuthority`, `"ex:"`). Replacing each `":"` by `"ex:"` generates, in our example, the new conformed URI `"ex:Iboprufen"`.

We define two predicates $p_1, p_2 \in \mathcal{P}$ to be *interchangeable* (denoted $p_1 \leftrightarrows p_2$) if both of them have the same subject and object. Formally, $\forall p_1, p_2 \in \mathcal{P} : p_1 \leftrightarrows p_2 \iff \exists s, o \mid (s, p_1, o) \wedge (s, p_2, o)$.

The idea of the self-configuration process of the *predicate conformation* is to change each predicate in the source data set to its *interchangeable* predicate in the target data set. Formally, find all pairs $(p_1, p_2) \mid \exists s, p_1, o \in k \wedge \exists s, p_2, o \in k' \wedge (s, p_1, o) \in k \wedge (s, p_2, o) \in k'$. Then, for each pair (p_1, p_2) create two self-configuration parameters (`sourceProperty`, p_1) and (`targetProperty`, p_2). The predicate conformation will replace each occurrence of p_1 by p_2.

In our example, let us suppose that we ran the NLP-based enrichment first then we got a set of related named entities all of them related to our `ex:Iboprufen` object by the default predicate `fox:relatedTo` as shown in Figure 21a. Subsequently, applying the predicate conformation self-configuration will generate (`sourceProperty`, `fox:relatedTo`) and (`targetProperty`, `ex:relatedCompany`) parameters. Consequently, the predicate conformation module will replace `fox:relatedTo` by `ex:relatedCompany` to generate Figure 21 (b).

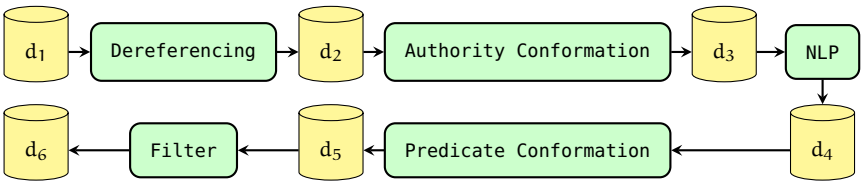


Figure 22: Graph representation of the learned pipeline of our running example, where d_1 is the positive example source presented in Figure 20 (a) and d_6 is the positive example target presented in Figure 20 (b).

Filter Enrichment Function

The idea behind the self-configuration of *filter*-based enrichment is to preserve only valuable triples in the source CBDs k and discard any unnecessary triples so as to achieve a better match to k'. To this end, the self-configuration process starts by finding the intersection between source and target examples $I = \bigcup_{(k,k')\in X^+} k \cap k'$. After that, it generates an enrichment function based on a SPARQL query which is only preserving predicates in I. Formally, the self-configuration results in the parameter set $P = \bigcup_{p\in K\cap K'\cap \mathcal{P}} p$.

Back to our running example, let us continue from the situation in the previous section (Figure 21 (b)). Performing the self-configuration of filters will generate $P = \{\texttt{fox:relatedTo}\}$. Actually applying the filter enrichment function will remove all unrelated triples containing the predicate `fox:relatedTo`. Figure 22 shows a graph representation for the whole learned pipeline for our running example.

EVALUATION

The aim of our evaluation was to quantify how well our approach can automate the enrichment process. We thus assumed being given manually created training examples and having to reconstruct a possible enrichment pipeline to generate target CBDs from the source CBDs. In the following, we present our experimental setup including the pipelines and data sets used. Thereafter, we give an overview of our results, which we subsequently discuss in the final part of this section.

Experimental Setup

We used three publicly available data sets for our experiments:

1. From the biomedical domain, we chose *DrugBank*[5] as our first data set. We chose this data set because it is linked with many other data sets[6], from which we can extract enrichment data using our atomic enrichment functions. For our experiments we deployed a manual enrichment pipeline E_{manual}, in which we enrich the drug data found in *DrugBank* using abstracts dereferenced from *DBpedia*, then we conform both *DrugBank* and *DBpedia* source authority URIs to one unified URI. For *DrugBank* we manually deployed two experimental pipelines:
   - $E^1_{DrugBank} = (e_1, e_2)$, where e_1 is a dereferencing function that dereferences any `dbpedia-owl:abstract` from DBpedia and e_2 is an authority conformation function that conforms the *DBpedia* subject authority[7] to the target subject authority of *DrugBank*[8].
   - $E^2_{DrugBank} = E^1_{DrugBank} +\!\!+\ e_3$, where e_3 is an authority conformation function that conforms *DrugBank*'s authority to the *Example* authority[9].

2. From the music domain, we chose the *Jamendo*[10] data set. We selected this data set as it contains a substantial amount of embedded information hidden in literal properties such as `mo:biography`. The goal of our enrichment process is to add a geospatial dimension to *Jamendo*, e.g., the location of a recording or place of birth of a musician. To this end, we deployed a manual enrichment pipeline, in which we enrich *Jamendo*'s music data by adding additional geospatial data found by applying the NLP enrichment function against `mo:biography`. For *Jamendo* we deploy manually one experimental pipeline:
   - $E^1_{Jamendo} = \{e_4\}$, where e_4 is an NLP function that find *locations* in `mo:biography`.

3. From the multi-domain knowledge base *DBpedia* [Lehmann et al., 2014] we used the class `AdministrativeRegion` for our experiments. As DBpedia is a knowledge base with a large ontology, we build a set of five pipelines of increasing complexity:

5 *DrugBank* is the Linked Data version of the DrugBank database, which is a repository of almost 5000 FDA-approved small molecule and biotech drugs, for RDF dump see `http://wifo5-03.informatik.uni-mannheim.de/drugbank/drugbank_dump.nt.bz2`

6 See `http://datahub.io/dataset/fu-berlin-drugbank` for complete list of linked data set with *DrugBank*.

7 `http://dbpedia.org`

8 `http://wifo5-04.informatik.uni-mannheim.de/drugbank/resource/drugs`

9 `http://example.org`

10 *Jamendo* contains a large collection of music related information about artists and recordings, for RDF dump see `http://moustaki.org/resources/jamendo-rdf.tar.gz`

- $E^1_{DBpedia} = \{e_5\}$, where e_5 is an authority conformation function that conforms the *DBpedia* subject authority to the *Example* target subject authority.
- $E^2_{DBpedia} = e_6 +\!\!+ E^1_{DBpedia}$, where e_6 is a dereferencing function that dereferences any `dbpedia-owl:ideology`.
- $E^3_{DBpedia} = E^2_{DBpedia} +\!\!+ e_7$, where e_7 is an NLP function that finds *all* named entities in `dbpedia-owl:abstract`.
- $E^4_{DBpedia} = E^3_{DBpedia} +\!\!+ e_8$, where e_8 is a filter function that filters for abstracts.
- $E^5_{DBpedia} = E^3_{DBpedia} +\!\!+ e_9$, where e_9 is a predicate conformation function that conforms the source predicate of `dbpedia-owl:abstract` to the target predicate of `dcterms:abstract`.

Altogether, we manually generated a set of 8 pipelines, which we then applied against their respective data sets. The evaluation protocol was as follows: Let E be one of the manually generated pipelines. We applied E to an input knowledge base K and generated an enriched knowledge base $K' = E(K)$. We then selected a set of resources in K and used the CBD pairs of selected resources and their enriched versions as examples E. E was then given as training data to DEER, which learned an enrichment pipeline E'. We finally compared the triples in K' (which we used as reference data set) with the triples in $E'(S)$ to compute the precision, recall and F-measure achieved by our approach. Generated pipelines are available at the project web site[11].

All experiments were carried out on a 8-core PC running *OpenJDK* 64-Bit Server 1.6.0_27 on *Ubuntu* 12.04.2 LTS. The processors were 8 Hexa-core *AMD Opteron* 6128 clocked at 2.0 GHz. Unless stated otherwise, each experiment was assigned 6 GB of memory. As termination criteria for our experiments, we used (1) a maximum number of iterations of 10 or (2) an optimal enrichment pipeline found.

Results

We carried out two sets of experiments to evaluate our refinement based learning algorithm. In the first set of experiments, we tested the effect of the complexity weight ω to the search strategy of our algorithm. The results are presented in Table 9. In the second set of experiments, we test the effect of the number of positive examples $|X^+|$ on the generated F-measure. Results are presented in Table 10.

Configuration of the Search Strategy.

We ran our approach with varying values of ω to determine the value to use throughout our experiments. This parameter is used for con-

11 https://github.com/GeoKnow/DEER/tree/master/evaluations/pipeline_learner

Table 9: Test of the effect of ω on the learning process using the *Drugbank* data set, where $|X^+| = 1$, E is the manually created pipeline, $|E|$ is the complexity of E, E' is the pipeline generated by our algorithm, and I_n is the number of iterations of the algorithm.

ω	E Size	E' Size	τ Size	Iter. Count	E' P	E' R	E' F
0	3	1	61	10	1.0	0.99	0.99
0.25	3	1	61	10	1.0	0.99	0.99
0.50	3	1	61	10	1.0	0.99	0.99
0.75	3	3	25	4	1.0	1.0	1.0
1.0	3	1	61	10	1.0	0.99	0.99

figuring the search strategy in the learning algorithm, in particular the bias towards simple pipelines. As shown in Section 8.3.2, this is achieved by multiplying ω with the node complexity and subtracting this as a penalty from the node fitness. To configure ω, we used the first pipeline $E^1_{DrugBank}$. The results suggest that setting ω to 0.75 leads to the best results in this particular experiment. We thus adopted this value for the other studies.

Effect of Positive Examples.

We measured the F-measure achieved by our approach on the data sets at hand. The results shown in Table 10 suggest that when faced with data as regular as that found in the data sets *Drugbank*, DBpedia and *Jamendo*, our approach really only needs a single example to be able to reconstruct the enrichment pipeline that was used. This result is particularly interesting, because we do not always generate the manually created reference pipeline described in the previous subsection. In many cases, our approach detects a different way to generate the same results. In most cases (71.4%) the pipeline it learns is actually shorter than the manually created pipeline. However, in some cases (4.7%) our algorithm generated a longer pipeline to emulate the manual configuration. As an example, in case of $E^1_{Jamendo}$ the manual configuration was just one enrichment function, i.e, NLP-based enrichment to find all *locations* in `mo:biography`. Our algorithm learns this single manually configured enrichment as (1) an NLP enrichment function that extracts all named entities types and then (2) a filter enrichment function that filters all non-location triples. Our results also suggest that our approach scales when using a small number of positive example as on average the learning time for one positive example was around 48 seconds.

Table 10: Test of the effect of increasing number of positive examples in the learning process. For this experiment we set $\omega = 0.75$. E is the manually created pipeline, $E(KB)$ is applying E to the entire data set, E' is the pipeline generated by our algorithm, and all times are in minutes.

Manual Pipeline	Examples Count	E Size	$E(KB)$ Time	E' Size	$E'(KB)$ Time	Learning Time	τ Size	Iterations Count	E' Precision	E' Recall	E' F-score
$E^1_{DBpedia}$	1	1	0.2	1	1.6	1.3	7	1	1.0	1.0	1.0
	2	1	0.2	1	1.8	1.3	7	1	1.0	1.0	1.0
$E^2_{DBpedia}$	1	2	23.3	1	0.1	0.2	7	1	1.0	0.99	0.99
	2	2	15	2	17	0.3	55	9	0.99	1.0	0.99
$E^3_{DBpedia}$	1	3	14.7	3	15.2	6.1	55	9	1.0	0.99	0.99
	2	3	15	2	15.1	0.1	55	9	0.99	0.99	0.99
$E^4_{DBpedia}$	1	4	0.4	2	0.1	0.7	13	2	0.99	0.99	0.99
	2	4	0.6	2	0.3	0.9	13	2	0.99	1.0	0.99
$E^5_{DBpedia}$	1	5	22	2	0.1	0.7	13	2	1.0	1.0	1.0
	2	5	25.5	2	0.2	0.9	13	2	1.0	1.0	1.0
$E^1_{DrugBank}$	1	2	3.5	1	4.1	0.1	61	10	0.99	0.99	0.99
	2	2	3.6	1	3.4	0.1	61	10	0.99	0.99	0.99
$E^2_{DrugBank}$	1	3	25.2	1	0.1	0.1	61	10	1.0	0.99	0.99
	2	3	22.8	1	0.1	0.1	61	10	1.0	0.99	0.99
$E^1_{Jamendo}$	1	1	10.9	2	10.6	0.1	13	2	0.99	0.99	0.99
	2	1	10.4	2	10.4	0.1	7	1	0.99	0.99	0.99

Table 11: Results of running the 7 manual generated pipelines against *DBpedia*, *DrugBank*, and *Jamendo*. All times are in minutes.

E	X^+ Size	ω	E_{manual} Size	$E_{manual}(KB)$ Time	E_{self} Size	$E_{self}(KB)$ Time	Learning Time	τ Size	Iterations Count	P	R	F
$E^1_{DBpedia}$	1	0.75	1	0.2	1	1.6	1.3	7	1	1.0	1.0	1.0
$E^2_{DBpedia}$	1	0.75	2	23.3	1	0.1	0.2	7	1	1.0	0.99	0.99
$E^3_{DBpedia}$	1	0.75	3	14.7	3	15.2	6.1	55	9	1.0	0.99	0.99
$E^4_{DBpedia}$	1	0.75	4	0.4	2	0.1	0.7	13	2	0	1.00	0
$E^5_{DBpedia}$	1	0.75	5	22	2	0.1	0.7	13	2	1.0	1.00	1.00
$E^1_{DrugBank}$	1	0.75	2	3.6	1	4.2	0.1	61	10	0.99	0.99	0.99
$E^2_{DrugBank}$	1	0.75	3	22	2	0.1	0.7	13	2	1.0	0.99	0.99
$E^1_{Jamendo}$	1	0.75	1	10.6	2	10.4	0.1	7	1	0.99	0.99	0.99

Part III

APPLICATION SCENARIOS AND CONCLUSION

In the next four chapters, we demonstrate a set of application scenarios for our proposed approaches from Part II in integration and enrichment of the *GHO*, *SemanticQuran*, *agriNepalData* and *NIF4OGGD* data sets. Finally, we conclude this thesis in Chapter 13 and discuss a set of possible future extensions.

9

GHO – PUBLISHING AND INTERLINKING THE GLOBAL HEALTH OBSERVATORY DATASET

In this chapter, we present the process of publishing and linking the GHO data set. A data set paper is published about the data set in the Semantic Web Journal [Zaveri et al., 2013b]. The author linked GHO with the other mentioned data sets. Also, he took part of the data set creation process and co-wrote the paper.

The improvement of public health is one of the main indicators for societal progress. The World Health Organization (WHO)[1], a specialized agency of the United Nations, is mainly concerned with international public health with the main aim of the attainment of the highest possible level of health by all people. Besides publishing reports on global health problems, WHO also provides access to enormous amounts of statistical data and analyses for monitoring the global health situation. The WHO's GHO publishes such statistical data and analyses for important health problems, which is categorised by either country, indicator or topic. The aim of GHO is to provide access to (1) country data and statistics with a focus on comparable estimates, and (2) WHO's analyses to monitor global, regional and country situation and trends[2].

GHO provides access to a wide variety of over 50 different data sets, such as the world health statistics, mortality and burden of disease, health expenditure per capita, deaths due to particular diseases such as HIV/AIDS, Tuberculosis, neglected tropical diseases, violence and injuries, health equity, just to name a few. Each data set contains an extensive list of indicators which capture statistical data according to a region, country or based on gender. The data covers all the 198 WHO member countries[3] and while some indicators are from the late 1970s onwards, some are prior to the mid-1990s. The data is updated as more recent or revised data becomes available or when there are changes to the methodology being used. A list of all the data sets with a description of its contents is provided in Table 12.

In this chapter, we first describe the process of the conversion of the GHO data to RDF in Section 9.1. Details of the publishing and the interlinking GHO with other data sets are presented in Section 9.2. Section 9.3 portrays a few potential application scenarios and use cases for the GHO data. A number of related initiatives and how GHO is different than what already exists is discussed in Section 9.4. Finally, we conclude with the lessons learned in Section 9.5.

1 http://www.who.int/en/
2 http://www.who.int/gho/about/en/
3 http://www.who.int/countries/en/

Table 12: Different statistical data sets available in the GHO.

Dataset	Description	Triples #
Environmental health	Number of deaths due to children health, climate change, household air pollution, UV radiation, water, sanitation and hygiene	31,012
Epidemic prone diseases	Number of reported cases of cholera, meningococcal meningitis and statistics from the Global Influenza Surveillance and Response System	255,957
Equity	Equity figures for women health, urban health and social determinants of health	324,445
Health-related Millennium Development Goals	Health indicators associated with poverty and hunger, child mortality, maternal health, environment sustainability, and global partnership for development.	784,346
Health systems	Data on healthcare infrastructure, essential health technologies, aid effectiveness, health financing, essential medicines, service delivery and health workforce	234,340
HIV/AIDS	Data on the size of the epidemic and on the HIV/AIDS response	99,476
Immunization	Country and regional data of immunisation efforts for several diseases	625,082
Injuries and violence	Number of deaths due to road traffic accidents, data on demographic and socio-economic statistics, emergency care provision and existence of a national policy for human safety	242845
Mortality and burden of disease	Number of deaths, Disability Adjusted Life Year (DALY) s, life expectancy, mortality and morbidity, disease and injury country estimates for each country	3,000,000
Neglected Tropical Diseases	Statistics on newly reported cases of each of the neglected tropical disease that is monitored	167,841
Noncommunicable Diseases	Mortality measures, risk factors and health system response and capacity for each of the noncommunicable disease that is monitored	1,409,629
Tobacco Control	Data on the prevalence of adult and youth consuming tobacco and various measures to help prevent tobacco consumption, such as policies, help, warnings, enforcing bans	379,283
Tuberculosis	Cases of incidence and mortality, diagnosis, drug regimens, treatment success for tuberculosis in each country	67,479

DATASET CONVERSION

The GHO data is published as spreadsheets describing a single data item (e.g. death, DALY) in several dimensions (e.g. country, population, disease). In order to convert the data to RDF, we used the RDF Data Cube Vocabulary [Tennison et al., 2012] which is based on the popular SDMX standard[4] and designed particularly to represent multidimensional statistical data using RDF. The vocabulary also uses the SDMX feature of Content Oriented Guidelines (COG). COG defines a set of common statistical concepts and associated code lists that can be re-used across data sets.

However, transforming these spreadsheets to RDF in a fully automated way may cause information loss as there may be dimensions encoded in the heading or label of a sheet. Thus, we implemented a semi-automatic approach by integrating the algorithm as a plug-in extension in OntoWiki [Auer et al., 2006]. OntoWiki is a tool which supports agile, distributed knowledge engineering scenarios. Moreover, it provides ontology evolution functionality, which can be used to further transform the newly converted statistical data.

Using this plug-in[5], when a spreadsheet containing multidimensional statistical data is imported into OntoWiki, it is presented as a table as shown in Figure 23. Subsequently, the user has to manually configure the (1) dimensions, (2) attributes by creating them individually and selecting all elements belonging to a certain dimension and (3) the range of statistical items that are measured. Using RDFa, the corresponding COG concepts are automatically suggested, when a user enters a word in the text box provided. The specified configurations can also be saved as a template and reused for similar spreadsheets, such as for data published in consecutive years. Then the plug-in automatically transforms the data into RDF. A presentation detailing the conversion process is available[6].

After converting the GHO data, an RDF data set containing almost 8 million triples (number of triples for each individual data set is reported in Table 12) was obtained and published at: `http://gho.aksw.org/`. The mortality and burden of disease data set in GHO alone accounts for 3 million triples. An example of the death value *127* represented as RDF using the Data Cube vocabulary is illustrated in the following listing:

DATASET PUBLISHING AND LINKING

DATASET PUBLISHING. After converting the GHO data as RDF , we published it as Linked Data using the *OntoWiki* platform [Auer et al.,

4 `http://sdmx.org`
5 Available at `aksw.org/Projects/Stats2RDF`
6 `http://goo.gl/OHDM9`

```
gho:Country rdfs:subClassOf qb:DimensionProperty;
          rdf:type rdfs:Class;
          rdfs:label "Country" .

gho:Disease rdfs:subClassOf qb:DimensionProperty;
          rdf:type rdfs:Class;
          rdfs:label "Disease" .

gho: Afghanistan rdf:type ex:Country;
              rdfs:label "Afghanistan" .

gho:Tuberculosis rdf:type ex:Disease;
              rdfs:label "Tuberculosis" .

gho:c1-r6 rdf:type   qb:Observation;
          rdf:value "127"^^xsd:integer;
          qb:dimension gho:Afghanistan;
          qb:dimension gho:Tuberculosis .
```

Listing 1: RDF representation of the death value '127' using the RDF Data Cube Vocabulary

Table 13: Technical details of the GHO RDF data set.

URL	http://gho.aksw.org/
Version date	01-11-2010
Version number	1.0
Licensing	WHO allows reproduction of its data for non-commercial purposes.
VoiD File	http://db0.aksw.org/downloads/void.ttl
DataHub entry	http://thedatahub.org/dataset/gho

2006]. OntoWiki not only allows the publishing and maintenance of the data but also provides a SPARQL Protocol and RDF Query Language (SPARQL) endpoint for the data set in combination with *Virtuoso*[7] as the storage solution for the RDF model. Additionally, it is also possible to browse the data with the HTML output of OntoWiki. Details and links of the SPARQL endpoint, the version, licensing, availability and link to the VoiD file are listed in Table 13.

DATASET LINKING. The URI's for diseases and countries are uniform for all the data sets in GHO since all the data sets utilize the same disease and country name in the original data. Thus, there was no explicit interlinking required within the data sets. A single country or disease URI was therefore automatically linked to all the instances from the different tables associated with that country or disease and vice-a-versa. Thus, when a single country or diseases is looked up, all the corresponding instances from all the tables can be retrieved. There are a total of 192 unique country URIs and 116 disease URIs in the entire data set.

7 http://virtuoso.openlinksw.com/

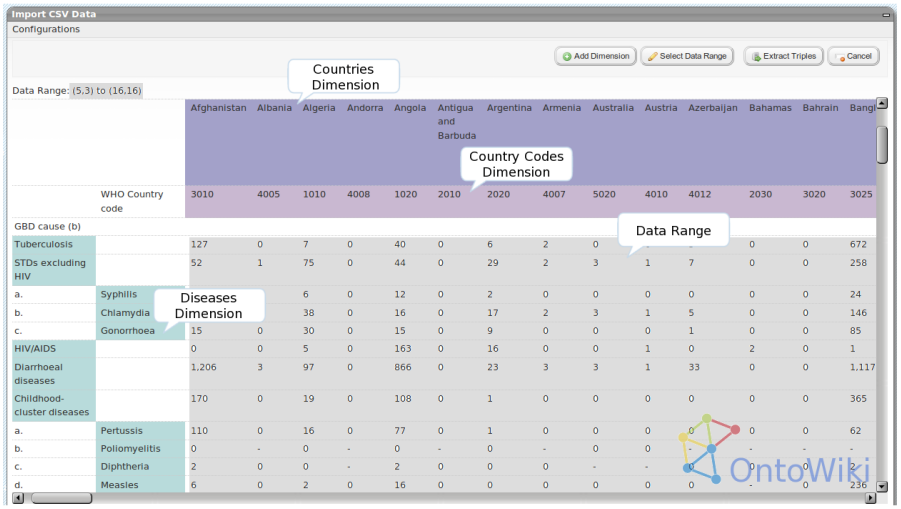


Figure 23: Screenshot of the *OntoWiki* statistical data import wizard displaying a GHO table configured for conversion into RDF.

We used the *mortality and burden of disease data set* from GHO as a test-environment for link generation and linked it with the *LinkedCT*[8] (the Linked Data version of ClinicalTrials.gov) and *PubMed*[9] (converted to Linked Data by the Bio2RDF project) data sets for diseases and countries.

We used the *Silk 2.0* [Volz et al., 2009b] tool, which is developed for discovering relationships between data items within different knowledge bases, that are available via SPARQL endpoints. Silk includes a declarative language for specifying (1) the types of RDF links that should be discovered and (2) the conditions which the data items must fulfil in order to be interlinked. We used the *Jaro distance* as string metric where applicable and two confidence value thresholds: (1) Links above 0.95 confidence were accepted and (2) links between 0.90 and 0.95 were saved to a separate file for manual inspection. The number of interlinks obtained for countries and diseases is displayed in Table 14 along with the precision for each set of links. The precision value was calculated manually by a researcher with a biomedical background by going through each links and evaluating the correctness. Then, using the formula: (correct links * total no. of links / total no. of links), the precision values were noted. Since a gold standard data set was not available, the recall values were not calculated.

Additionally, we interlinked the diseases from GHO with the diseases in BioPortal[10], in particular the ICD-10 codes[11]. We used the *LIMES* [Ngonga Ngomo and Auer, 2011] framework to create these links since the SILK tool did not provide the flexibility and efficiency that was required to generate these interlinks. There was a total of

DPSO (Chapter 6) is the default implementation of load balancing currently implemented in LIMES.

8 http://linkedct.org/
9 http://bio2rdf.org/
10 http://www.bioontology.org/
11 http://www.who.int/classifications/icd/en/

Table 14: Number of accepted (acc.) and verified (ver.) links and its precision values obtained among *GHO*, *PubMed*, *LinkedCT*, *ICD-10*, *DBpedia* and *WorldBank* for diseases and countries.

Class	predicate	Source		Target		Links		Precision	
		Dataset	Ins.	Dataset	Ins.	Acc.	Ver.	Acc.	Ver.
Diseases	owl:sameAs	LinkedCT	5000	GHO	116	163	43	0.96	-
Diseases	rdfs:subClassOf	LinkedCT	5000	GHO	116	469	45	1.00	0.99
Diseases	owl:sameAs	GHO	116	PubMed	23618	453	75	1.00	0.71
Diseases	owl:sameAs	GHO	116	ICD-10	4913	107	-	1.00	-
Locations	redd:locatedIn	LinkedCT	757341	GHO	192	300000	0	1.00	-
Countries	owl:sameAs	GHO	192	PubMed	23618	201	12	1.00	0.96
Countries	owl:sameAs	GHO	192	DBpedia	2710	192	-	1.00	-
Countries	owl:sameAs	GHO	192	WorldBank	214	189	-	1.00	-

107 interlinks that were created, of the total 116 diseases. Interlinks for diseases such as "Communicable, maternal, perinatal and nutritional conditions", "Maternal conditions" or "Nephritis and nephrosis" could not be found either because there was no match found or they were too generalized whereas ICD contained a thorough classification of the diseases. Moreover, in order to increase the interlinking between GHO and other data sets, we interlinked the countries from GHO with those in the DBpedia[12] and World Bank[13] data sets. It is to be noted that for these three interlinks (ICD-10, DBpedia and World Bank), although LIMES was able to correctly find interlinks for the diseases and countries, there were some instances for which no links were found. For these the interlinks were manually created.

In addition to the ability to explore the data by using SPARQL and the resulting lists of resources, users are able to visualize the data by using *CubeViz*. CubeViz is an OntoWiki extension, which uses DataCube resources as input. After the selection of desired dimension properties such as `gho:disease` and `gho:country` as well as the measure property `gho:incidence` CubeViz is able to generate different type of charts (e.g. bar chart, pie chart, spline chart)). As an example, the incidence of the disease "Migraine" in selected countries can be visualized with CubeViz as depicted in Figure 24.

USE-CASES

In this section, we outline selected application scenarios and use-cases for the Linked GHO data.

Monitoring Health Care Scenarios

Since GHO provides information on mortality, morbidity, health status, service coverage and risk factors for several diseases in each country,

12 http://wiki.dbpedia.org/Downloads38#links-to-gho
13 http://worldbank.270a.info/

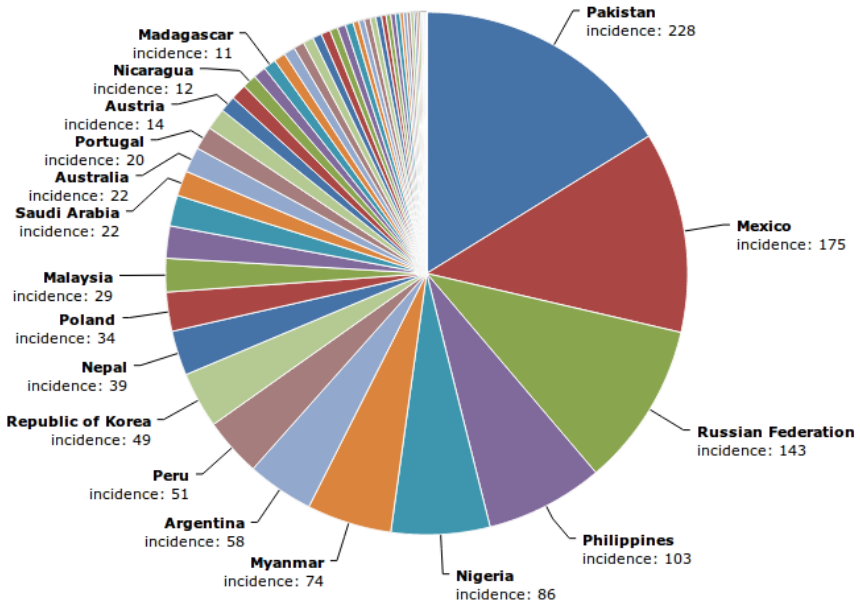


Figure 24: Screenshot of *CubeViz* displaying the pie chart of incidence of Migraine in a subset of countries.

it can be used by each country to monitor the disease prevalence for any given year and to compare prevalence as well as the effect of counter-measures with similar or neighbouring countries.

For example, Listing 2 shows the SPARQL query for retrieving the number of deaths due to Measles in all countries. Listing 3 shows the SPARQL query for retrieving the measles immunization coverage among 1-year-olds (%)[14]. The values retrieved from these SPARQL queries can be used to compare the number of deaths and the immunization services carried out in a country for a particular year to gauge its effectiveness.

```
1  SELECT ?deaths ?diseasename ?countryname
2  FROM <http://ghocountry.org>
3  FROM <http://interlinks.org>
4  WHERE {?item a qb:Observation.
5  ?item gho:Country ?country .
6  ?country rdfs:label ?countryname.
7  ?item gho:Disease ?disease.
8  ?disease rdfs:label ?diseasename.
9  ?item att:unitMeasure gho:Measure2 .
10 ?item eg:incidence ?deaths .
11 FILTER regex(?diseasename, "Measles")}
```

Listing 2: SPARQL query for retrieving the number of deaths due to Measles in all countries.

14 All prefixes can be found on prefix.cc

This can help to implement either precautionary measures if the mortality is high or curb health expenditures for diseases which seem to have adequate treatment options.

```
SELECT DISTINCT ?countryname ?incidence ?whichyear
WHERE {?item  a qb:Observation.
?item ex:incidence ?incidence.
?item gho:Country ?country.
?item gho:Year ?year.
?year rdfs:label ?whichYear.
?country rdfs:label ?countryname
FILTER regex(?whichYear, "2004")}
```

Listing 3: SPARQL query for retrieving the measles immunization coverage among 1-year-olds (%).

Disparity Analysis

Another application of the GHO data set is evaluating the disparity between the availability of treatment options and the global burden of disease, as illustrated in the ReDD-Observatory project [Zaveri et al., 2011]. This project interlinks GHO with the PubMed and LinkedCT data sets so as to enable the evaluation of the disparity. This disparity is partially caused due to the limited access to information that would allow health care and research policy makers make more informed decisions regarding health care services. The hindrance lies in reliably obtaining and integrating data regarding the disease burden and the respective research investments. Therefore, as the Linked Data paradigm provides a simple mechanism for publishing and interlinking structured information on the Web, an opportunity is created to reduce this information gap that would allow for better policies in response to these disparities.

Listing 4 provides an example of a SPARQL query which retrieves the number of deaths (from GHO) and the number of clinical trials (from LinkedCT) for the disease Tuberculosis and HIV/AIDS in all countries.

Primary Source Providing Ground Truth

GHO enables direct linking to the ground truth data for secondary (e.g. scientific publications) or tertiary (e.g. encyclopedias) sources. This enables improved provenance tracking in those sources. It also allows automatic syndication of the data using SPARQL or simple REST queries, which enables a simpler verification of statements compared to the manual work, which would be necessary without Linked Data. For example, the Wikipedia entry for Disease[15] (a tertiary source)

15 http://en.wikipedia.org/wiki/Disease

```
SELECT ?countryname ?diseasename ?value count(?trial)
FROM <http://gho.aksw.org/>
FROM <http://linkedct.org/>
WHERE {?item a qb:Observation ;
          gho:country ?country ;
          gho:disease ?disease ;
          att:unitMeasure gho:Measure ;
          gho:incidence ?value .
  ?country rdfs:label ?countryname .
  ?disease rdfs:label ?diseasename .
  ?trial  a ct:trials ;
          ct:condition ?condition ;
          ct:location ?location .
  ?condition owl:sameAs ?disease .
  ?location shv:locatedIn ?country .
FILTER (?diseasename("Tuberculosis", "HIV/AIDS")).}
```

Listing 4: SPARQL for retrieving the number of deaths and number of trials for Tuberculosis and HIV/AIDS in all countries.

uses the statistical values from Global Health Observatory (GHO), in particular from the *mortality and burden of disease* data. The Years of Potential Life Lost (YPLL) and DALY metrics are derived from GHO to provide information about them for several diseases categories in different regions of the world. Similarly, the corresponding DBpedia entry[16] (also a tertiary source) also links to the GHO page about the burden of diseases[17].

Human Development Data Warehouse

Just as data warehouses and business intelligence are now integral parts of every larger enterprise, the linked GHO data can be the nucleus for a human development data warehouse. In such a human development data warehouse, a large number of statistical data and indicators are published by different organizations that could be integrated automatically or semi-automatically in order to obtain a more interactive picture of the human development.

Currently, the indicators (e.g. the Human Development Index (HDI)) are very coarse-grained, mainly referring to countries. Using linked data, such indicators could be computed on a much more fine-grained level, such as for cities and regions as well as with regard to different groups of people (e.g. per gender, ethnicity, education level). Policy making would be based on more rational, transparent and observable decisions as it is advocated by evidence-based policy.

For example, Listing 5 shows the SPARQL query for retrieving the public health expenditure (from the World Bank data set) and the

16 http://dbpedia.org/page/Disease

17 http://www.who.int/healthinfo/global_burden_disease/2004_report_update/en/index.html

number of DALY s (Disability-Adjusted Life Years)[18] caused by all diseases (from GHO) in the year 2004. The results from these queries can thus be compared per country per year to obtain an overview of the human development problems affecting each country.

```
SELECT ?countryNameGHO ?daly (?obsValue AS ?publicHealthExpenditure)
WHERE {
  GRAPH g-indicators: {
    ?obs property:indicator indicator:SH.XPD.PUBL;
        sdmx-dimension:refArea ?countryWB;
        sdmx-dimension:refPeriod <http://reference.data.gov.uk/id/year/2004>;
        sdmx-measure:obsValue ?obsValue.
  }
  GRAPH g-meta: { ?countryWB a dbo:Country .}
  SERVICE <http://gho.aksw.org/sparql> {
    SELECT DISTINCT ?countryNameGHO ?daly {
      ?item a qb:Observation.
      ?item gho:Country ?countryGHO .
      ?countryGHO owl:sameAs ?countryWB.
      ?countryGHO rdfs:label ?countryNameGHO.
      ?item gho:Disease ?disease.
      ?disease rdfs:label "All Causes".
      ?item att:unitMeasure gho:Measure2 .
      ?item eg:incidence ?daly .
    }
  }
}
```

Listing 5: SPARQL query for retrieving the public health expenditure (from World Bank data set) and number of DALY s caused by all disease (from GHO) in the year 2004.

RELATED INITIATIVES

There are already a number of efforts to convert health care and life science related data sets to Linked Data such as LODD, LinkedCT, OBO ontologies and the World Wide Web Consortium (W3C) 's Health Care and Life Sciences Working Group, each of which is discussed in this section along with the importance of converting and publishing the GHO data sets.

LODD, i.e. the Linking Open Drug Data project[19], mainly converts, publishes and interlinks drug data that is available on the web, ranging from impacts of drugs on gene expression to results of the clinical trials. A number of data sets have been converted in this project[20] including DrugBank, DailyMed, SIDER to name a few. However, these data sets are restricted to drug data and even though they do contain disease data (from the Diseasome data set), they do not connect the number of deaths or the health expenditure or the status of the health

18 http://dbpedia.org/page/Disability-adjusted_life_year
19 http://www.w3.org/wiki/HCLSIG/LODD/
20 http://www.w3.org/wiki/HCLSIG/LODD/Data

system in each country for each of the diseases that are included (as provided by GHO).

LinkedCT is the Linked Data version of ClinicalTrails.gov which publishes data about clinical trials in RDF and links it to other data sets such as PubMed. Even though, in LinkedCT each trial is associated with a disease and a drug, it does not provide information about the prevalence of the disease in a particular country, which is provided in GHO.

OBO is the Open Biological and Biomedical Ontologies project[21] which aims to create a suite of interoperable reference ontologies in the biomedical domain. It brings together biology researchers and ontology developers who work together to develop a set of ontologies as well as design principles that can help develop interoperable ontologies. However, most of the ontologies developed are at the experimental level or organismal level and are not yet sufficiently interlinked with other data sets available as Linked Data. Additionally, the *NCBO's BioPortal*[22] contains a large collection of controlled medical terminologies, all available as Linked Data.

The Semantic Web Health Care and Life Sciences (HCLS Interest Group)[23] is established by the W3C to support the use of Semantic Web technologies in health care, life sciences, clinical research and translational medicines. The group focuses on aiding decision-making in clinical research, applying the strengths of Semantic Web technologies to unify the collection of data for the purpose of both primary care (electronic medical records) and clinical research (patient recruitment, study management, outcomes-based longitudinal analysis, etc.). Subgroups, on the other hand, focus on making the biomedical data available in RDF , dealing with biomedical ontologies, focus on drug safety and efficacy communication and support researchers in the navigation and annotation of the large amount of potentially relevant literature.

SUMMARY AND OUTLOOK

Although we were able to successfully convert the GHO data set and utilize one of the data sets in a use case, we encountered some problems such as cumbersome conversion, low interlinking quality and lack of time series capability in the data sets. We discuss these problems in the sequel.

CONVERSION. The conversion process was cumbersome and time consuming because, first of all, each individual Excel files needed to be downloaded from the GHO web portal. Then, each file had to

21 http://obofoundry.org/
22 http://bioportal.bioontology.org/
23 http://www.w3.org/blog/hcls/

be converted into Comma-Separated Values (CSV) so that it could be appropriately displayed as an HTML table in OntoWiki. Since the conversion method was semi-automated, one had to individually selected the dimensions, attributes and data range for each of the files. While some of the required steps such as the annotation of the CSV files for conversion are not automatable, other steps, such as the Excel to CSV conversion can be performed more efficiently (e.g. in a batch or through bulk processing).

Extension of WOMBAT *Chapter 7.*

COHERENCE. The number of links obtained between the data sets for diseases was relatively low, as presented in Table 14. The main reason was the different use of identifiers for the naming of the disease. For example, 'heart attack' in GHO could not be matched with 'cardiac arrest' in LinkedCT using the basic string similar functionality of SILK. In order to address this problem, we plan to extend WOMBAT (see Chapter 7) in such a way, that background knowledge in the form of gazetteers can be also taken into account. Also, we plan to link the disease names with their corresponding ICD codes so as eliminate the need for entity recognition and also to improve precision and recall.

TEMPORAL COMPARABILITY. The data in GHO is not published regularly every year. Also, since the health data recording and handling systems differ between countries, comparability of the data is limited. This is mainly due to the differences in definitions and/or time periods and incomplete data provision from different countries. Therefore, computing time trends is not possible using GHO, which would be a good indicator of the health scenario in each country over a number of years. We expect, however, that the increased visibility and transparency of a Linked Data version of GHO together with the enhanced possibility of annotation and linking (when compared to simple Excel sheets) will contribute to standardization and increased temporal comparability in the future.

EXPLORING GHO. Using OntoWiki or similar tools (such as Disco or Tabulator etc.) to browse the RDF data helps users to gain new insights. CubeViz as an OntoWiki extension provides visualization of statistical data (such as GHO) in an user friendly way by means of displaying the data in various types of diagrams and charts. However, a limitation of such generic visualization tools is their limited scalability.

Automating GHO update using WOMBAT *Chapter 7 and* DEER *Chapter 8.*

UPDATING GHO. Although GHO is not published regularly, we plain to automate both processes of transforming and linking the upcoming versions of GHO. Given the high quality of the currently existing links in GHO, we plan to use these links as positive examples to feed

WOMBAT to generate new versions of GHO. Moreover, we intend to use the current CBD structure of GHO resources as positive examples to feed DEER (see Chapter 8) in order to automate the whole process of transforming GHO new versions.

SUMMARY. By providing the GHO data as Linked Data and linking it with other data sets, it is possible to not only obtain information on important health related topics in each country but also ease the work of health care professionals for data analysis in providing easy access to data. Moreover, it provide opportunities to link to related data and thus perform analyses for current priority health issues. The Linked Data publishing and linking of the GHO data is a first milestone in a larger research and development agenda: The creation of a global human development data warehouse, which allows to interactively monitor social, societal and economic progress on a global scale.

10

SEMANTIC QURAN – A MULTILINGUAL RESOURCE FOR NATURAL-LANGUAGE PROCESSING

This chapter describes the Semantic Quran data set, a multilingual RDF representation of translations of the Quran. A paper about the data set is published in the Semantic Web Journal [Sherif and Ngonga Ngomo, 2015b]. To create the data set, the author first designed an ontology for representing multilingual data. Then, he aligned data from two semi-structured sources to the created ontology. Also, the author linked Semantic Quran with the other mentioned data sets and co-wrote the paper.

In this chapter, we present the *Semantic Quran* data set, by which we aim to contribute towards the realization of the vision of a multilingual LOD and thus support the adaptation of NLP tools for languages for which only a limited amount of Linked Data exist. The Semantic Quran data set consists of all chapters of the Quran in 43 different languages including rare languages such as *Divehi*, *Amazigh* and *Amharic*. The data included in our data set was extracted from two semi-structured sources: the *Tanzil* project and the *Quranic Arabic Corpus* (cf. Section 10.3). We designed an ontology for representing this multilingual data and their position in the Quran (i.e., numbered chapters and verses). In addition to providing aligned translations for each verse, we provide morpho-syntactic information on each of the original Arabic terms utilized across the data set. Moreover, we linked the data set to three versions of *Wiktionary* as well as *DBpedia* and ensured therewith that our data set abides by all Linked Data principles[1].

In the following, we present the data sources that we used for the extraction (Section 10.1). Thereafter, we give an overview of the ontology that underlies our data set (Section 10.2). Section 10.3 depicts the extraction process that led to the population of our ontology. We present our approach to interlinking the Semantic Quran and Wiktionary in Section 10.4. Finally, we present several usage scenarios for the data set at hand (Section 10.5).

DATA SOURCES

Two web resources were used as raw data sources for our data set. The first web resource is the data generated by the *Tanzil Project*[2], which consists of the original verses in Arabic as well as 42 manual translations of the entire book. Our second web resource, *the Quranic Arabic Corpus*[3], was used to obtain morpho-syntactic information on each of the words contained in the Arabic version of the Quran.

1 http://www.w3.org/DesignIssues/LinkedData.html
2 http://tanzil.net/
3 http://corpus.quran.com

Tanzil Project

The Tanzil Project[4] was motivated by inconsistencies across the different digital versions of the Quran. These were mainly due to missing/incorrect diacritics, Arabic text conversion problems, and missing encoding for some Arabic characters.

Tanzil was launched in early 2007 with the aim of producing a curated unicode version of the Arabic Quran text that can serve as a reliable standard text source on the web. To achieve this goal, then Tanzil team developed a three-step data quality assurance pipeline which consists of (1) an automatic text extraction of the Arabic text, (2) a rule-based verification of the extraction results and (3) a final manual verification by a group of experts.

The result of this process was a set of data sets that were made available in several versions and formats.[5] In addition to the original Arabic sources, Tanzil provides sentence-parallel translations of the Quran in 42 different languages by different translators[6]. We manually selected one translation per language for the extraction process.[7] Note that all Tanzil data sets are distributed under the terms of Creative Commons Attribution 3.0 License.[8]

The Quranic Arabic Corpus Project

The Quranic Arabic Corpus is an open-source project, which provides Arabic annotated linguistic resources which shows the Arabic grammar, syntax and morphology for each word in the Quran. This is a valuable resources for the development of NLP tools for the Arabic language, in which a single word can encompass the semantics of entire English sentences. For instance the Arabic word *"faja'alnāhum"* can be translated into the entire English sentence "and we made them". The compact syntax of Arabic leads to that a single word being separable into distinct morphological segments. For example, *"faja'alnāhum"* can be subdivided into:

- *fa* – a prefixed conjunction (engl. "and"),
- *ja'al* – the stem, a perfect past tense verb (engl. "made") inflected as first person masculine plural,
- *nā* – a suffixed subject pronoun (engl. "we") and
- *hum* – a suffixed object pronoun (engl. "them").

4 http://tanzil.net/wiki/Tanzil_Project
5 For more details on available formats and data sets, please see http://tanzil.net/download/.
6 http://tanzil.net/trans/.
7 The list of translations used can be found at http://goo.gl/s5RuI
8 http://creativecommons.org/licenses/by/3.0/

A RDF and Natural Language Processing Interchange Format (NIF) [Hellmann et al., 2012] representation of this rich morphology promises to further the development of integrated NLP pipelines for processing Arabic. In addition, given that this corpus was curated manually by experts, it promises to improve the evaluation of integrated NLP frameworks. We thus decided to integrate this data with the translation data available in the Tanzil data sets. Here, we used the Quranic Arabic Corpus Version 0.4[9] in its delimited text file version under the *"GNU General Public License"*.[10]

ONTOLOGY

To represent the data as RDF, we developed a general-purpose linguistic vocabulary. The vocabulary[11] was specified with the aim of supporting data sets which display a hierarchical structure. It includes four basic classes: Chapter, Verse, Word and LexicalItem.

The *Chapter class* provides the name of chapters in different languages and localization data such as chapter index and order. Additionally, the chapter class provides metadata such as the number of verses in a chapter and provenance information. Finally, the chapter class provides properties that allow referencing the verses it contains. For example each chapter provides a `dcterms:tableOfContents` for each of its verses in the form `qrn:quran<chapter>- <verse>`.

The *Verse class* contains the verse text in different languages as well as numerous localization data such as verse index and related chapter index. Additionally, this class provides related verse data such as different verse descriptions and provenance information. Finally, it contains referencing properties similar to those of chapters.

The *Word class* encompasses the next level of granularity and contains the words in the verse text in different languages as well as numerous localization data such as related verse and chapter verse indexes. Additionally, the word class provides word provenance information and some referencing properties.

Currently, the *LexicalItem class* provides morphological data on the Arabic words only. Several ontologies can be used to represent such information. In our data set, we relied on the RDF representation of the *GOLD linguistic ontology*[12] [Farrar and Langendoen, 2003] to provide linguistic properties of lexical items such as acoustic, root, part of speech, gender, number, and person. We chose to use GOLD in contrast to other ontologies because it belongs to the most exhaustive ontologies for modeling linguistic properties. Thus, it will allow us to easily extend this data set in future work. All the objects of the previ-

9 http://corpus.quran.com/download/
10 http://www.gnu.org/licenses/gpl.html
11 http://mlode.nlp2rdf.org/datasets/qvoc.owl.ttl
12 http://linguistics-ontology.org/

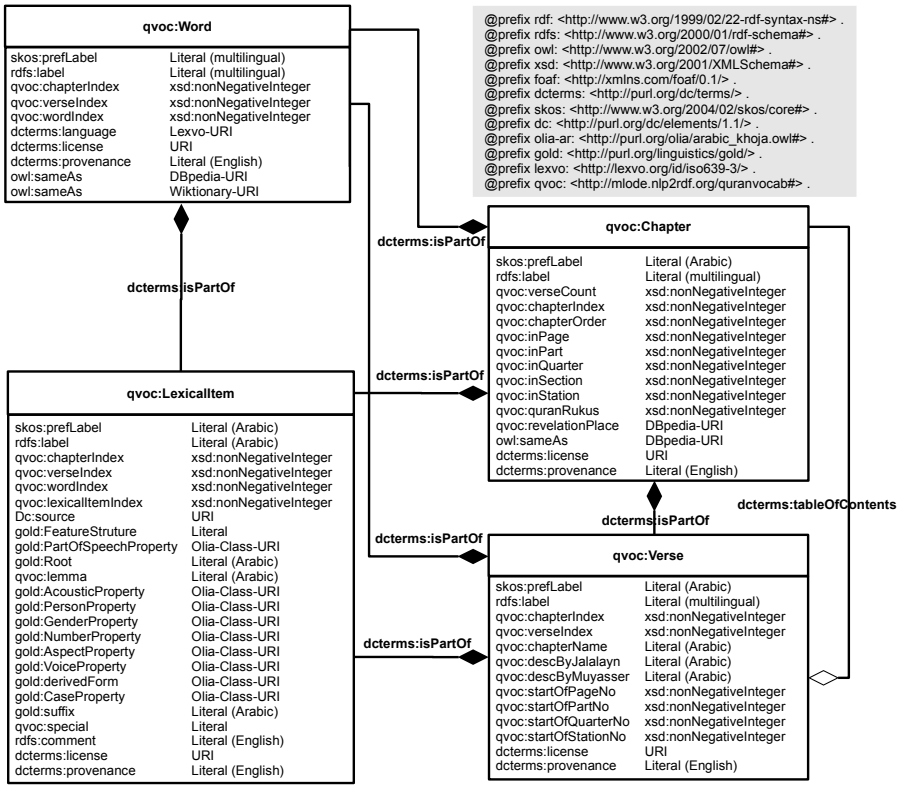


Figure 25: UML class diagram of the Semantic Quran ontology.

ously mentioned properties are URIs from the *OLIA Arabic Linguistic ontology*[13] . Analogously to the other classes, LexicalItem provides provenance information and referencing properties. A UML class diagram of the four basic ontology classes of the Semantic Quran data set with inter-class internal relations is shown in Figure 25.

EXTRACTION PROCESS

The original Tanzil Arabic Quran data and translations are published in various formats. For the sake of effectiveness, delimited text files were selected as the basis for the RDF extraction. The format of the delimited file is `chapterIndex|verse|verseText`. For example, the first verse of the first chapter of the English translation of the Quran is `1|1|In the Name of Allah, the Most Beneficent, the Most Merciful.` On the other hand, the Quranic Arabic corpus is available as tab-separated text file of the form "`LOCATION FORM TAG FEATURES`":

- The `LOCATION` field consists of 4-part numbering scheme of the form (Chapter : Verse : Word : Segment). For example, the first segment of the first word of the first verse of the first chapter has the form `(1:1:1:1)`.

13 http://nachhalt.sfb632.uni-potsdam.de/owl/

- The FORM field contains the text of the current segment in the Extended *Buckwalter transliteration*[14]. For example the corresponding FORM to (1:1:1:1) is bi (engl. "In").
- The TAG field contains the part-of-speech tag for the current segment. For example the corresponding TAG to (1:1:1:1) is p which stands for preposition.
- The FEATURES field contains a complete morphological analysis of the current segment such as root, case and person-number-gender properties. For example the corresponding FEATURES to (1:1:1:1) is PREFIX|bi+ which stands for preposition prefix ("by", "with", "in") with acoustic property "bi".

Given the regular syntax used in the text file corpus at hand, we were able to carry out a one-to-one mapping of each fragment of the input text file to resources, properties or data types as explicated in the ontology shown in Figure 25. We relied on the *Apache Jena Framework*[15] for the conversion. The part-of-speech information and morphological characteristics of each segment of the Arabic Quranic Corpus were extracted and integrated with the words found in the Tanzil data set. The merged data is now available in the RDF format. In order to simplify the interoperability of the generated data set, we followed the specifications of the NIF. Currently, the original Arabic and four different translations of the Quran (Arabic, English, German, French and Russian) abide by the NIF formalization. Details of the Semantic Quran data set CKAN entry, its SPARQL endpoint, version and license are listed in Table 15.

Table 15: Technical details of the Quran RDF data set.

Name	SemanticQuran
Example Resource	http://mlode.nlp2rdf.org/resource/semanticquran/quran1-1
data set Dump	http://mlode.nlp2rdf.org/datasets/semanticquran.nt.gz
Sparql Endpoint	http://mlode.nlp2rdf.org/sparql
data set Graph	http://thedatahub.org/dataset/semanticquran
Ontology	http://mlode.nlp2rdf.org/datasets/qvoc.owl.ttl
Ver. Date	29.11.2012
Ver. No	1.0
Licence	Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)
DataHub Entry	SemanticQuran

14 The Buckwalter transliteration uses ASCII characters to represent the orthography of the Arabic language. For the conversion table, see http://www.qamus.org/transliteration.htm

15 http://jena.apache.org/

LINKING

DPSO (Chapter 6) is the default implementation of load balancing currently implemented in LIMES.

We aimed to link our data set with as many data sources as possible to ensure maximal reusability and integrability in existing platforms. We have generated links to 3 versions of the RDF representation of Wiktionary as well as to DBpedia. All links were generated by using the LIMES framework [Ngonga Ngomo, 2012]. The link specification used was essentially governed by fragments similar to that shown in Listing 6. The basic intuition behind this specification is to link words that are in a given language in our data set to words in the same language with exactly the same label. We provide 7617 links to the English version of DBpedia, which in turn is linked to non-English versions of DBpedia. In addition, we generated 7809 links to the English, 9856 to the French and 1453 to the German Wiktionary. Links to further versions of DBpedia and Wiktionary will be added in the future.

```
<SOURCE>
   <ID>quran</ID>
   <ENDPOINT>http://mlode.nlp2rdf.org/sparql</ENDPOINT>
   <VAR>?x</VAR>
   <PAGESIZE>-1</PAGESIZE>
   <RESTRICTION>?x a qvoc:Word</RESTRICTION>
   <PROPERTY>rdfs:label AS lowercase->nolang RENAME label </PROPERTY>
</SOURCE>
<TARGET>
   <ID>wiktionary</ID>
   <ENDPOINT>http://wiktionary.dbpedia.org/sparql</ENDPOINT>
   <VAR>?y</VAR>
   <PAGESIZE>-1</PAGESIZE>
   <RESTRICTION>?y rdf:type lemon:LexicalEntry</RESTRICTION>
   <RESTRICTION>FILTER langMatches( lang(?v0), "en" )</RESTRICTION>
   <PROPERTY>rdfs:label AS lowercase->nolang RENAME label </PROPERTY>
</TARGET>
<METRIC>trigrams(x.label,y.label)</METRIC>
```

Listing 6: Fragment of the link specification to the English Wiktionary.

We evaluated the quality of the links generated by manually checking 100 randomly selected links from each of the three languages. The manual check was carried out by the two authors. A link was set to be correct if both authors agreed on it being correct. Overall, the linking achieve a precision of 100% for the English version, 96% for the French and 87% for the German. The error in the French links were due homonymy errors. For example, “Est” (engl. East) was linked to “est” (engl. to be) in some cases. Similarly in the German, “Stütze” (engl. support) was linked to “stütze” (engl. imperative singular form the verb “to support”). In the next version of the data set, we will add context-based disambiguation techniques to improve the quality of the links. Especially, we will consider the type of the expression to link while carrying out the linking to ensure that verbs cannot be

matched with nouns for example. Still, the accuracies we achieve in these three languages are sufficient to make the data set useful for NLP applications. The recall could not be computed manually. While these values are satisfactory, they can be improved further by devising a disambiguation scheme based on the context within which the words occured. To achieve this goal, we aim to combine the results of LIMES with the AGDISTIS disambiguation framework[16] in future work.

USE-CASES

The availability of a multilingual parallel corpus in RDF promises to facilitate a large number of NLP applications. In this section, we outline selected application scenarios and use cases for our data set.

Data Retrieval

The Quran contains a significant number of instances of places, people and events. Thus, multilingual sentences concerning such information can be easily retrieved from our data set, for example for the purpose of training NLP tools. Moreover, the aligned multilingual representation allows searching for the same entity across different languages. For example, Listing 7 shows a SPARQL query which allows retrieving Arabic, English and German translations of verses which contain *"Moses"*.

```
SELECT DISTINCT ?chapterIndex ?verseIndex ?verseTextAr ?verseTextEn ?verseTextGr
WHERE{
  ?word rdfs:label "Moses"@en;
        dcterms:isPartOf ?verse.
  ?verse a qvoc:Verse;
         skos:prefLabel ?verseTextAr;
         qvoc:verseIndex ?verseIndex;
         dcterms:isPartOf ?chapter;
         rdfs:label ?verseTextEn;
         rdfs:label ?verseTextGr.
  FILTER ( lang(?verseTextEn) = "en" && lang(?verseTextGr) = "de")
  ?chapter qvoc:chapterIndex ?chapterIndex.
}
```

Listing 7: Verses that contains moses in (i) Arabic (ii) English and (iii) German.

Arabic Linguistics

The RDF representation of Arabic morphology and syntax promises to facilitate the retrieval of relevant sub-corpora for researchers in

16 http://github.com/AKSW/AGDISTIS

linguistics. For example, Listing 8 provides an example of a SPARQL query witch retrieves all Arabic prepositions as well as an example statement for each of them.

```
SELECT ?preposition ( sql:SAMPLE ( ?verseTextAr ) AS ?example )
WHERE{
  ?s gold:PartOfSpeechProperty olia-ar:Preposition;
     skos:prefLabel ?preposition;
     dcterms:isPartOf ?verse.
  ?verse a qvoc:Verse;
         skos:prefLabel ?verseTextAr.
}GROUP BY ?preposition
```

Listing 8: List all the Arabic prepositions with example statement for each.

Another example is provided by Listing 9, which shows a list of different part-of-speech variations of one Arabic root of the word read *"ktb"* (engl. "write"); note that in this example we use the Arabic root *"ktb"* witten in The Buckwalter transliteration.

```
SELECT DISTINCT ?wordText ?pos
WHERE{
  ?wordPart a qvoc:LexicalItem ;
            gold:Root "ktb";
            gold:PartOfSpeechProperty ?pos;
            dcterms:isPartOf ?word.
  ?word a qvoc:Word;
        skos:prefLabel ?wordText.
}
```

Listing 9: List of different part of speech variations of one Arabic root of the word read *"ktb"*.

Interoperability using NIF

Using the interoperability capabilities provided by NIF, it is easy to query all occurrences of a certain text segment without using the verse, chapter, word, or lexical item indexes. For instance, Listing 10 lists all the occurrences of *"Moses"* with no need to have an extra index.

```
SELECT ?textSegment ?verseText {
  ?s str:occursIn ?verse;
     str:isString ?verseText.
  ?textSegment str:referenceContext ?s;
               str:anchorOf "Moses"@de.
}
```

Listing 10: List of all occurrences of "Moses" using NIF

Information Aggregation

The interlinking of the Quran data set with other RDF data sources provides a considerable amount of added value to the data set. For example, the interlinking with Wiktionary can be used as in Listing 11 to get the different senses for each of the English words contained in the first verse of the first chapter *"qrn:quran1-1"*.

```
SELECT DISTINCT ?wordTextEn ?sense
FROM <http://thedatahub.org/dataset/semanticquran>
FROM <http://en.wiktionary.dbpedia.org>
WHERE{
  ?word a qvoc:Word;
        rdfs:label ?wordTextEn;
        dcterms:language lexvo:eng ;
        dcterms:isPartOf qrn:quran1-1;
        owl:sameAs ?wiktionaryWord.
  FILTER ( lang(?wordTextEn) = "en" )
  ?wiktionaryWord lemon:sense ?sense
}
```

Listing 11: List of all senses of all English words of the first verse of the first chapter "qrn:quran1-1".

SUMMARY AND OUTLOOK

In this chapter, we presented the *Semantic Quran*, an integrated parallel RDF data set in 42 languages. This multilingual data set aims to increase the availability of multilingual data in LOD and to further the development of NLP tools for languages that are still under represented, if not absent, from the LOD cloud. Thanks to its RDF representation, our data set ensures a high degree of interoperability with other data sets. For example, it provides 26735 links overall to *Wiktionary* and *DBpedia*. As demonstrated by our use cases, the data set and the links it contains promise to facilitate research on multilingual applications. Moreover, the availability of such a large number of languages in the data set provides opportunities for linking across the monolingual data sets on the LOD Cloud and thus perform various types of large-scale analyses.

To improve the ease of access to our data set, we aim to extend the TBSL framework [Unger et al., 2012] to allow users to gather sensible information from the data set. Moreover, we aim to automatically provide links to the upcoming versions of *Wiktionary* using WOMBAT (see Chapter 7). Additionally, we will link the Semantic Quran data set with many of the publicly available multilingual *Wordnets*. We already provided NIF for the five languages Arabic, English, French, German and Russian. We will extend the NIF content of the data set to the remaining 38 languages. Given that *Quran* is originally in Arabic,

Automatically link Semantic Quran to upcoming versions of Wiktionary using WOMBAT *(see Chapter 7.*

we plan to include as many translations as possible for each language in which we intend to apply COLIBRI (see Chapter 5) to detect erroneous and missing links.

Apply COLIBRI *(see Chapter 5) to detect erroneous and missing links*

11

AGRINEPALDATA – ONTOLOGY BASED DATA ACCESS AND INTEGRATION FOR IMPROVING THE EFFECTIVENESS OF FARMING IN NEPAL

This chapter describes the AgriNepalData data set aiming to improve the the faming in Nepal [Pokharel et al., 2014]. The author linked AgriNepalData with other mentioned data sets. Also, he co-designed the AgriNepalData ontology, took part of the data set creation and co-wrote the paper.

Information and communication technologies (ICT) have gained significant importance in our lives across several domains. Agriculture is no exception and the coining of the term *E-agriculture* roots back to the rather recent *World Summit of the Information Society* in 2003[1]. The key characteristics of E-agriculture are the dissemination, access and exchange of information. ICT can play a vital role to boot up a farmers' living standard by providing relevant information. Nevertheless, in Nepal (a country with agricultural based economy) information such as crop geographical location, properties of soil, climate information and crop production normally are not publicly available. It is difficult for farmers to obtain access to such information and, therefore, they cannot benefit from for planning and decision making.

In the agriculture domain, various aspects have to be integrated to build a fully functioning system with all the information related to agriculture such as weather measurements, soil characteristics, new research results and findings, government policies, market information and inventory. All of such different data are produced by different bodies of the government and all of these departments are working rather independently with limited integration between them.

Taking the rice crop as an example, not only irrigation alone can improve its productivity, there are other factors[2] such as soil status, weather conditions and rice water requirements during each of its sub-seasons. Due to the lack of integration between such heterogeneous data, extraction of information like how much irrigation is required for rice in a particular region on a particular day is difficult to obtain, which in turn leads to reduced farming efficiency.

Recently, many different agriculture related projects were established in Nepal, in particular by the *Ministry of Irrigation*[3] and the *Ministry of Agriculture Development*[4]. For instance, the *Ground Water Irrigation Project*[5] was launched improve the rice productivity in *Chitwan* district, Nepal. While those initiatives provide relevant information, it is not published using established standards. For this reason, we convert information to LOD [Bizer et al., 2009; Auer et al., 2013] using the RDF data model and established vocabularies. This allows not

1 http://www.e-agriculture.org/e-agriculture
2 http://cals.arizona.edu/pubs/water/az1220/
3 http://www.doi.gov.np
4 http://www.doanepal.gov.np
5 http://www.doi.gov.np/projects/project.php?pid=25

only to publish data conforming to W3C standards, but also to establish links between data sources, thereby enabling analysis methods going beyond those possible when using the original data sources in isolation.

In this chapter, we can draw on existing ontologies. In particular, *AGROVOC*[6] is a controlled RDF vocabulary with around 32.000 concepts covering all of the Food and Agriculture Organization of the United Nations (FAO) areas of interest including food, nutrition, agriculture, fisheries, forestry and environment. AGROVOC thesaurus is already mapped to many ontologies such as the *FAO Biotechnology Glossary*, *EUROVOC*, *GEMET*, *Library of Congress Subject Headings* (LCSH), *NAL Thesaurus*, *Thesaurus for Economics* (STW), *Thesaurus for the Social Sciences* (TheSoz), *Geopolitical ontology*, *Dewey Decimal Classification* (DDC), *DBpedia* [Lehmann et al., 2014] and *GeoNames*.

The data management efforts performed are the first steps on a larger research agenda, which we publish in the context of *ArgiNepalData* project[7]. In general, this article presents an application of web intelligence methods. A major contribution is the conversion and integration of data from five different sources (cf. Section 11.2.1). In addition to providing the farming data sets as RDF, we designed an ontology for representing and aligning those heterogeneous data sets. This alignment enables inference of new knowledge from converted data. Moreover, we linked the data set to *DBpedia* as well as *AGROVOC* and ensured therewith that our data set abides by all Linked Data principles[8].

The remainder of this chapter is structured as follows: In the subsequent section, we present a detailed description of the the framework used in our data conversion. Then, in Section 11.2, we describe each of the data sources used in our data sets. Moreover, we give an overview of the ontology that forms the background structure of our data sets (Section 11.3). We present the approach used to link the farming data sets in Nepal with different external data sets in Section 11.4. Based on this, we present several usage scenarios for the data sets at hand (Section 11.6). Finally in Section 11.7, we summarise the work done in AgriNepalData and present a set of lesson learned.

METHODOLOGY

In order to generate the *AgriNepalData* data sets, we have adapted a data management framework (see Figure 26). For the data management in *AgriNepalData*, we use the Linked Data Life-cycle vision as a

6 http://aims.fao.org/standards/agrovoc/linked-open-data
7 http://agrinepaldata.com
8 http://www.w3.org/DesignIssues/LinkedData.html

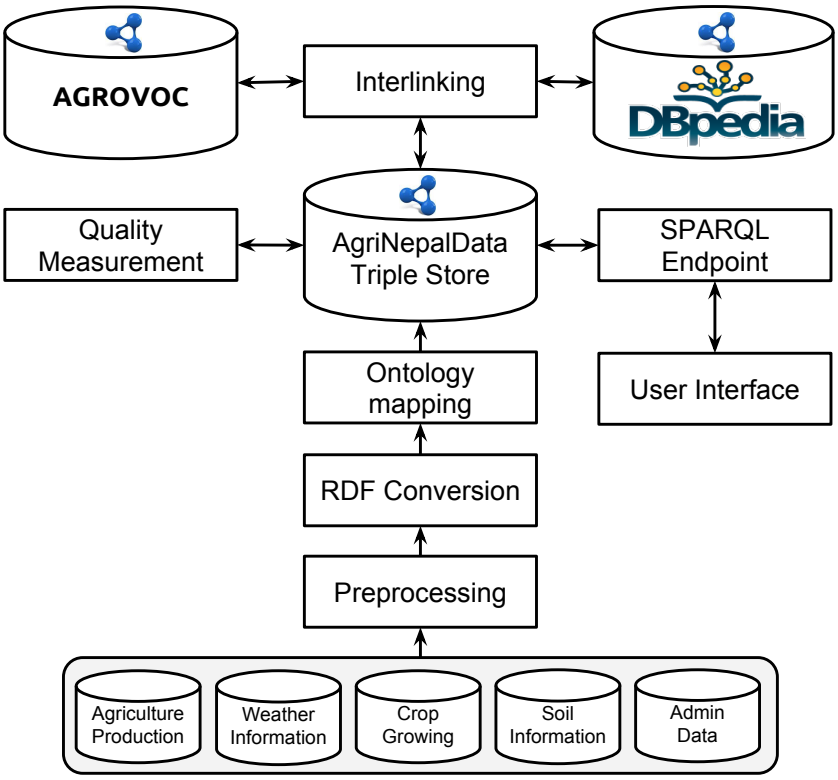


Figure 26: *AgriNepalData* data management framework.

base [Auer and Lehmann, 2010][9] (see Figure 27). Below, we discuss each of the 8 lifecycle phases in the context of *AgriNepalData*:

- **Extraction:** The first step is the extraction of RDF from CSV, HTML and shape files. We have used *OpenRefine, TripleGeo and Sparqlify* tools for this process. Detailed descriptions are given in Section 11.2.2.

- **Storage and Querying:** For hosting the *agriNepalData* we need a triple store which can handle not only (1) different data types such as strings, numbers, dates and spatial point sets, but also (2) continuously growing size as more data sets converted and added. In order to fulfill the aforementioned requests we chose *Virtuoso*[10]. *Virtuoso* provides backward chaining Web Ontology Language (OWL) reasoning, geospatial/text indexing and query functionality through SPARQL endpoint. For querying the *AgriNepalData* user can use the provided endpoint[11].

- **Manual revision and Authoring:** In order to minimize the error rate in the converted data we apply manual test cases. In

9 http://stack.linkeddata.org/
10 http://virtuoso.openlinksw.com/rdf-quad-store/
11 http://agrinepaldata.com/sparql

some cases the manual testing led us to discover some discrepancies either in our conversion framework or in the data itself. In former case we refine our framework and in the later case we refined the data preprocessing phase. For example, when applying a manual test case for checking for the RDF district data, we notice missing some data for fruit production of the *Kabhrepalanchok* District. We surprisingly found this district to have two different names in agriculture production data set; *Kavre* for wool production (which was already considered as the name used by this data set to this district) and *Kavrepalanchok* for fruit production (which is missing). Therefore, in the preprocessing phase, we added the *Kavrepalanchok* name to the list of synonyms of *KabhrepalanchokDistrict*.

- **Interlinking:** The AgriNepalData data sets are interlinked with both *DBpedia* and *AGROVOC* data sets using LIMES (for more details see Section 11.4).

- **Classification and Enrichment:** In this phase, we applied the DEER (see Chapter 8) to enrich our data set with additional geospatial data from other data sets like *DBpedia* and *LinkedGeoData*.

Enriching AgriNepalData using DEER *from Chapter 8.*

- **Quality and Analysis:** As any data set is as good as its quality, we applied *RDFUnit* [Kontokostas et al., 2014] tool to measure the quality of data as well as set of manual verifications (see Section 11.5 for more details).

- **Evolution and Repair:** After applying the manual test cases and the automated data quality tools we discovered a set of discrepancies, which we needed to repair. Once we repaired the discovered errors we re-ran the manual test cases as well as the automatic tools to increase the quality of the data sets.

- **Search and Browsing:**

  The *Facete (Faceted Browser)* [Stadler et al., 2014] tool is used to provide a visual searching and browsing interface. More detailed descriptions are given in Section 11.6.3.

DATASET DESCRIPTION

In this section, we first describe each of raw sources in detail. Since we obtain the source data from different sources, we cannot expect them to be homogeneous, which leads to challenges in the RDF conversion process. We illustrate the data conversion and those challenges in the second subsection.

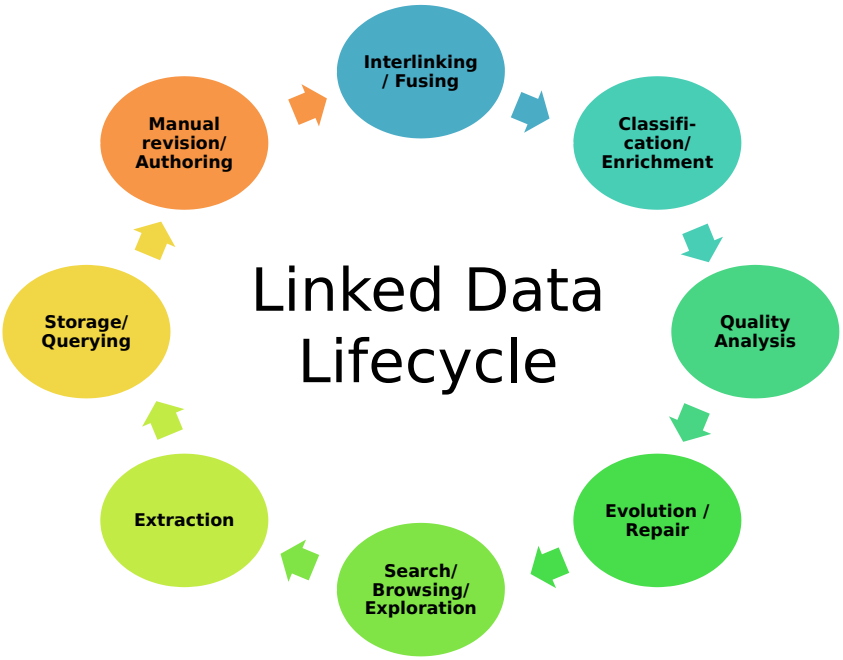


Figure 27: Linked Data Lifecycle.

Data Sources

The raw data sets have been collected from five semi-structured sources

1. **Agriculture Production Statistics of Nepal:** This data set is collected from the *Ministry of Agricultural Development*[12] in Nepal. It contains the information about yearly production of different crops since 1990 to 2012. Furthermore, it provides details of the production of crops and livestocks in each of Nepal's districts in 2011/12. The raw data set is freely available online in PDF format and as CSV file on request to ministry.

2. **Weather Information:** This data set is collected from the *Department Of Hydrology and Meteorology*[13] in Nepal. It contains information about daily rainfall of stations for *Babai* from 1980 to 2008 and *West Rapti* from 1980 to 2006. Additionally, it includes hourly weather information of *Banepa* from 2011 to 2012. The raw data set is freely available online in HTML format and in CSV file format on request.

3. **Crop Growing Days Information:** This data set is collected from FAO website[14]. It contains the information about crop growing days in each stage as well as crop coefficients in each stages. The raw data set is freely available as HTML file.

12 http://www.moad.gov.np/
13 http://www.dhm.gov.np/
14 http://www.fao.org/docrep/s2022e/s2022e00.htm

4. **Soil Information Of Nepal:** The original data set was called *SOTER_Nepal*, which is collected from the *ISRIC - World Soil Information*[15]. The SOTER_Nepal database provides generalized information on landform and soil properties at a scale 1:1 million. It consists of 17 SOTER units and characterized by 56 representative and four synthetic profiles for which there are no measured soil data. The raw data set was in form of shape files. ISRIC encourages the provision and use of all its data for research, education and policy support.

5. **Administrative Data of Nepal :** This data set is collected from the *International Centre for Integrated Mountain Development (ICIMOD)*[16], Nepal. This is based on topographic zonal map published by department of survey in different dates, which are more then 20 thematic layers covering entire Nepal. It contains the information about each of Nepal's development regions, zones, districts, villages development committees (VDCs), wards, national parks, peaks and roads. The raw data set was in form of shape files. ICIMOD offers free access for its data for free registered users.

Extraction Process

The data was extracted from various sources using three different formats (CSV, Shapefiles, HTML) for each of which we describe the conversion process below.

1. **CSV to RDF Conversion:** The crop statistics, weather information and crop growing days data sets are available in CSV format, but do not have any uniform structure beyond using the same format. First, we did some preprocessing such as removing special characters, unifying measurement units and filling missing data. Afterwards, The CSV files were converted to RDF using *OpenRefine*[17] and *Sparqlify* [Ermilov et al., 2013; Stadler et al., 2015].

   Listing 12 shows an example of converting one row of data of a CSV file to RDF format. Originally, the raw CSV data row was: `PaddyTaplejung2011 Taplejung 10477 22167 2116 Paddy 2011 /12`, which shows statistics about the *Paddy* produced in from August 2011 to July 2012 in *Taplejung* district.

2. **Shape to RDF Conversion:** The original data sets of soil and administrative information are stored as shape files. Shape files hold spatial data information in form of polygon or points as

15 http://www.isric.org/data/soil-and-terrain-database-nepal
16 http://geoportal.icimod.org/downloads/
17 http://openrefine.org/

```
agrd:PaddyTaplejung2011
   a agro:CerealCropProduction, time:TemporalEntity ;
   agro:inDistrict agrd:TaplejungDistrict ;
   agro:produce agrd:Paddy ;
   agro:production "22167"^^dbo:tonne ;
   agro:yield "2116"^^dbo:perHectare ;
   quty:area "10477"^^dbo:hectare ;
   time:hasBeginning "2011-08-01"^^xsd:date ;
   time:hasEnd "2012-07-31"^^xsd:date .
```

Listing 12: RDF Conversion for Paddy produced in year 2011/12 in Taplejung district

well as some non-spatial information. The spatial information of the shape files are converted to RDF using *TripleGeo* [Patroumpas et al., 2014], while the non-spatial information of shape files is first extracted to CSV by using *QGIS*[18] and then converted to RDF using *OpenRefine*.

Listing 13 shows an example of the conversion of the information contained in an *ESRI* shape file. The example shows the conversion of information on the district of *Gorkha* consisting of both spatial (polygon information in WKT format) as well as non spatial (name, area, region, zone, dcode) facts.

```
agrd:GorkhaDistrict
   a agro:District ;
   rdfs:label "Gorkha district"@en ;
   agro:dcode "36" ;
   agro:hasPart agrd:Gandaki ;
   agro:inDistrict agrd:GorkhaDistrict ;
   agro:inZone agrd:Gandaki ;
   agro:region agrd:Hill ;
   quty:area "3645.866"^^dbo:squareKilometre ;
   gsp:hasGeometry agrd:Geom_polygon_GorkhaDistrict .

agrd:Geom_polygon_GorkhaDistrict
   a opgis:Polygon ;
   gsp:asWKT "POLYGON ((85.10174531999999 28.456713989999997, 85.10162976
        28.454921459999998...))"^^gsp:wktLiteral .
```

Listing 13: Example of spatial and non-spatial RDF conversion of information for Gorkha district from an ESRI shapefile.

3. **HTML to RDF Conversion:** The crop growing days Information raw data was in form of HTML files. First we apply some manual selection of interesting pieces of data, which in most cases was in form of tables. Afterworlds, in a manner akin to the one used to convert CSV to RDF, the manually selected tables are converted to RDF using open *OpenRefine*.

18 http://www.qgis.org/en/site/

After converting all data sets, the resulting RDF files contain more than 1.4 million triples with 327475 distinct subjects. Table 16 shows the number of triples as well as distinct subjects of the RDF conversion for each of the aforementioned data sets. Table 17 provides technical details about *AgriNepalData*. Also, it includes version and license information.

Table 16: *AgriNepalData* triples details.

Source data set	**# Triples**	**# Subjects**
Agriculture Production Statistics of Nepal	27623	2887
Weather Information	404808	42003
Crop Growing Days Information	1030	125
Soil Information Of Nepal	21666	942
Administrative Data of Nepal	978288	281302
Ontology Related	216	216
Total	1433631	327475

Table 17: Technical details of the *AgriNepalData*.

Dataset Name	AgriNepalData
Project Website	http://agrinepaldata.com
	http://aksw.org/Projects/AgriNepalData
SPARQL Endpoint	http://agrinepaldata.com/sparql
Dataset Dump	http://agrinepaldata.com/download/agrinepal.zip
Ontology	http://agrinepaldata.com/download/agrinepaldataont.owl
Version Date	15-03-2014
Version Number	1.0
Licensing	(CC BY-NC-SA 3.0)
VoiD File	http://agrinepaldata.com/download/void.ttl
DataHub Entry	AgriNepalData

ONTOLOGY

To integrate the data on schema level, we developed an extensible ontology vocabulary for our data set. The vocabulary[19] was specified

19 http://agrinepaldata.com/download/agrinepaldataont.owl, also available in ecosystem of LOV http://lov.okfn.org/dataset/lov/index.html

with the aim of supporting any data set dealing with agricultural aspects. Currently, our ontology includes 38 classes (see Figure 28) covering production, geography and weather aspects. More additional classes can be added to the ontology at hand to cover more aspects if necessary.

The *Production* class is the super class for all other sub-production classes. Currently, there are eight sub-classes of the *Production* class covering different productions types found so far in our data set. Naturally, extending this part of our ontology is straightforward by adding more *Production* sub-classes for more production types. Each of the production sub-classes contains properies to keep track of its product date, quantity and location (for more details, see the right part of Figure 28). For example, *mandarin* is an instance of class *Fruit*, which its production is handled by the *FruitProduction* class, which is a sub-class of *Production* class. These are modelled in OWL as standard local range restrictions using universal quantifiers.

To keep track of various geographical information, starting from *Country* class our ontology contains a chain of derived classes to represent the hierarchical structure of the administrative regions of the country (Nepal in our case). The *Country* class and all of its sub-classes represent geographical information using WKT datatypes. Also, this part of the ontology is extensible through inheritance of new classes (for more details, see the left part of Figure 28). For example, *Goldhunga 1* ward is part of the VDC of *Goldhunga*, which is a part of the district of *Kathamandu*, which is a part of *Mid-Western development region*, which is a part of the country *Nepal*.

Finally, our ontology models weather statistics coming from weather stations through the *Station* class which is the super class of three sub-classes dubbed *RainfallStation*, *MeteorologyStation* and *ETOStation* to collect respectively rainfall, meteorology, and evapotranspiration statistics (for more details, see the bottom part of Figure 28). For example, *Kusum407*, located in the *Banke* district, is an instance of the *RainfallStation*, which a subclass of the *Station* class.

LINKING

We aimed to link our data set with as many data sources as possible to ensure maximal reusability and integrability in existing platforms. All links are generated by using the LIMES framework [Ngonga Ngomo and Auer, 2011]. In this framework, heuristics can be defined for the similarity of RDF resources and all similarity values exceeding a particular threshold are considered links. So far, we have generated links to *DBpedia* as well as *AGROVOC*.

DPSO (Chapter 6) is the default implementation of load balancing currently implemented in LIMES.

For example, Listing 14 shows a LIMES LS for linking Nepal districts in *AgriNepalData* to equivalent resources found in DBpedia. The LS used for other spatial resources such as zones and VDCs were es-

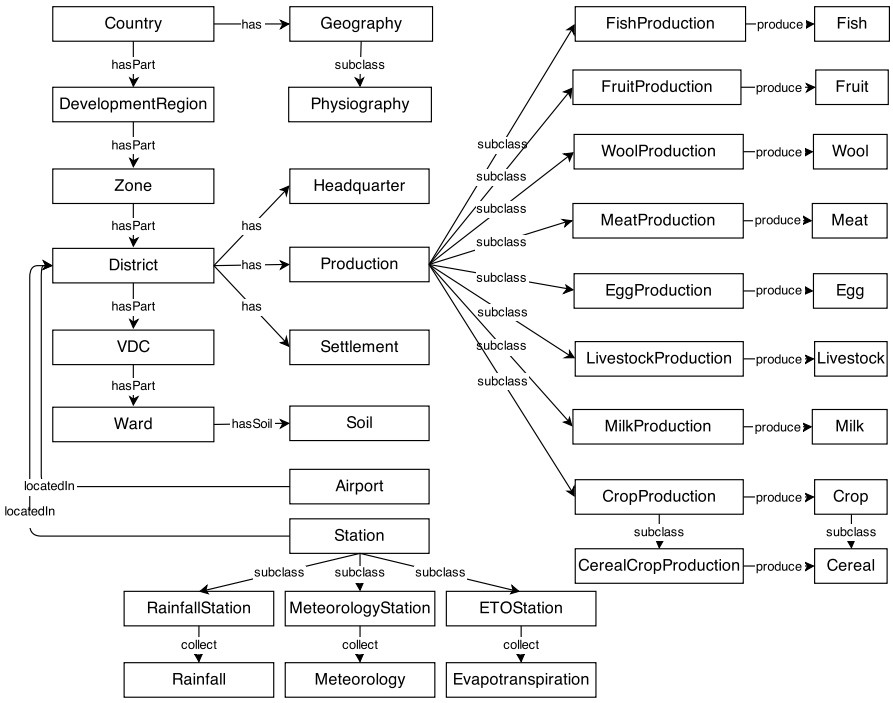


Figure 28: *AgriNepalData* ontology structure.

sentially governed by similar metrics. Table 18 shows details of links between *AgriNepalData* and both *DBpedia* and *AGROVOC*, where the `owl:sameAs` is used as the linking predicate.

Table 18: Number of inter-links and precision values obtained between *AgriNepalData* and other data sets, where the `owl:sameAs` is used as the linking predicate.

Link Class	Source Data set	Source Instances	Target Data set	Target Instances	Accepted Links	Verified Links	Precision
Places	AgriNepal	37161	DBpedia	754450	524	100	0.97
Species	AgriNepal	1265	DBpedia	239194	192	100	0.93
Airport	AgriNepal	43	DBpedia	12688	27	27	1.00
Species	AgriNepal	1265	AGROVOC	32294	53	53	0.91

QUALITY MEASUREMENT

Link Verification

For each class that contains links, we evaluated the quality of the links generated by LIMES by manually checking 100 randomly se-

```
<SOURCE>
   <ID>AgriNepalData</ID>
   <ENDPOINT>http://agrinepaldata.com/sparql</ENDPOINT>
   <VAR>?x</VAR>
   <PAGESIZE>1000</PAGESIZE>
   <RESTRICTION>?x a agro:District</RESTRICTION>
   <PROPERTY>geos:hasGeometry/geos:asWKT RENAME polygon</PROPERTY>
   <PROPERTY>rdfs:label AS nolang->lowercase</PROPERTY>
</SOURCE>
<TARGET>
   <ID>DBpedia</ID>
   <ENDPOINT>http://dbpedia.org/sparql</ENDPOINT>
   <VAR>?y</VAR>
   <PAGESIZE>1000</PAGESIZE>
   <RESTRICTION>?y a dbpedia-owl:Settlement</RESTRICTION>
   <PROPERTY>geo:geometry RENAME polygon</PROPERTY>
   <PROPERTY>rdfs:label AS nolang->lowercase</PROPERTY>
</TARGET>
<METRIC>AND(hausdorff(x.polygon,y.polygon)|0.7,
            trigram(x.rdfs:label,y.rdfs:label)|0.7)</METRIC>
```

Listing 14: Fragment of the LS for linking districts of Nepal between AgriNepalData and DBpedia.

lected links[20]. The manual check was carried out by the first two authors. A link was set to be correct if both authors agreed on it being correct. The results are shown in Table 18.

For linking places and airports, the linking achieves a precision between 0.97% and 100%. This high precision value is because we configure LIMES to use a combination of two metrics: (1) the string matching between resources labels, and (2) the geo-spatial matching between resources' WKT using the *Hausdorff* (see Section 4.3.9 for details) point distance set metric (see Listing 14 for an example of combining string and spatial metrics). The recall could not be computed manually due to the absence of ground truth.

Here we use Hausdorff, one of the evaluated point sets distance measures in Chapter 4.

For linking species, the linking achieves a precision between 0.91% and 0.93%. In this case, we used exact string matching as otherwise the precision turned out to be too low. As its name implies the exact match gives us only species with identical names to be linked.

Dataset Verification

For data set verification, we used the *RDFUnit*[21] framework. *RDFUnit* generates 956 test cases for all vocabularies used within *AgriNepalData*. Among 956 test cases, the results provided by *RDFUnit* shows that 935 test cases are passed, 3 are failed and 18 time out. Additionally, It shows that 65417 triples contain errors from a total of 1433631 triples, with average error rate of 0.045 per triple. Given that there are

20 In cases where there are less than 100 links, we checked all the links

21 http://aksw.org/Projects/RDFUnit.html

327475 distinct subject in *AgriNepalData*, the average error per distinct subject is 0.199.

All the failed test cases were due to some errors in the raw data. For example, one test case detected that all airports have latitude values out of the valid range $[-90°, 90°]$, which leads us to review the original data, finding a bug in raw data as there were missing floating point symbols (for instance the value of $28.78°$ was saved as $2878° > 90°$). Therefore, we manually fixed this bug. We iterated this process until all of the test cases passed.

USE-CASES

In this section, we outline selected application scenarios and use-cases for *AgriNepalData*.

Irrigation In Field

Example: *Mr. Bhandari* who lives in *Bhairawa*, Nepal wants to know *"How much irrigation water is required for a wheat plant which was planted in November 1 through out the life time of plant (120 days)?"*. To answer this, he first needs to know the weather condition. Therefore, he needs to look for the rainfall of each of the 120 days. Also, he needs to know the maximum and minimum temperature, humidity, wind status and sunshine hours. In addition, wheat as well as any other crops have their own crop specific water requirements. Finally, he needs to know the current water contained in soil which depends on soil type and previous rainfall status. To do so, he has to gather all of these pieces of information manually from different sources and update the information daily. Not only the process of finding all this information is tedious, but also the resulting information storage, update, and integration is hard.

Lacking of timely access to such necessary information may lead to less productivity and constraint him to his traditional methods of farming. In addition, it is possible to develop a farming mobile application so that farmer can access these information from the farm without any prior technical knowledge.

The crop water [Allen et al., 1998] is calculated as follows:

$$ET_{crop} = ET_o \times K_c \tag{26}$$

where ET_{crop} is the *crop evapotranspiration* or crop water need (mm/day), K_c is the crop factor and ET_o is reference evapotranspiration (mm/day). Each crop has its own growing stages during its season. In the case of wheat, it has an initial-season of 15 days, development-season of 25 days, mid-season of 50 days and late-season of 30 days. Moreover, each place has its specific ET_o for every month.

Using our data set, Listing 15 provides a SPARQL query for computing ET_{crop} for each of the 120 days of the wheat season, thereby answering the question of Mr. Bhandari.

```
SELECT ?place AS ?WheatPlace
((0.5*xsd:float(?int)+0.5*xsd:float(?dev))*xsd:float(?etoNov)) AS ?WaterPerDayNov
((0.5*xsd:float(?dev)+0.66*xsd:float(?mid))*xsd:float(?etoDec)) AS ?WaterPerDayDec
(xsd:float(?mid) * xsd:float(?etoJan)) AS ?WaterPerDayJan
(xsd:float(?lat) * xsd:float(?etoFeb)) AS ?WaterPerDayFeb
WHERE {
 agrd:Eto707 bio:place ?place.
 cros:cropKcEachSatageWheat agro:kcForInitialStage ?int;
                    agro:kcForDevelopmentStage ?dev;
                    agro:kcForMidSeasonStage ?mid;
                    agro:kcForLateSeasonStage ?lat.
 agrd:Eto707 agro:etoOfNepalInNovember ?etoNov;
          agro:etoOfNepalInDecember ?etoDec;
          agro:etoOfNepalInJanuary ?etoJan;
          agro:etoOfNepalInFebruary ?etoFeb.
}
```

Listing 15: How much irrigation water is required for a wheat plant which was planted in November 1 through out the life time of plant (120 days)?

Agriculture Planner, Policy Maker

The process of agriculture planning requires a significant amount of diverse knowledge to be available. A part of such knowledge is related to various *crops' statistics*, e.g. information like how many types of crops are planted in a particular district in a particular year? Also, which district has the maximum agri-production of a particular crop? Furthermore, *temporal information* is essential for long term agriculture planning, such as information related to a particular crop production in the last 10 years. Another part of agriculture planning information has *geographic nature*, such as the size/population/location of each district. Of course, not all those pieces of information can be found in our data set. Nevertheless, thanks to the Linked Data principles, we can acquire the missing information from other data sets through links. For instance, to answer the a question like *which districts are self dependent in their agri-products?*, Listing 16 provides a SPARQL query to collect the requested information pieces about a crop, paddy, production in each district as ratio of production per person (tonne/person) as well as production per district unit (tonne/km^2) not only from our data set, but also from *DBpedia* using federated query services provided by SPARQL 1.1.

```
SELECT DISTINCT ?district ?productionyear ?cropProduction ?yieldProduction
 ?districtAreaKmSq ?population
 xsd:float(?cropProduction)/xsd:float(?districtAreaKmSq) AS ?cropProdPerSqDistrict
 xsd:float(?cropProduction)*1000.00/xsd:float(?population) AS ?cropProdPerPerson
WHERE{
  SERVICE <http://dbpedia.org/sparql> {
    SELECT ?districtAreaKmSq ?population ?dbpediauri
    FROM <http://dbpedia.org>
    WHERE{
       ?dbpediauri dbp:area ?districtAreaKmSq;
                dbp:population ?population;
                dbp:title "Districts of Nepal"@en .
    }
  }
  ?districturi owl:sameAs ?dbpediauri.
  ?s ?p agro:CerealCropProduction;
    gnd:dateOfProduction ?productionyear
    qty:area ?cropingArea;
    agro:produce agrd:Paddy;
    agro:production ?cropProduction;
    agro:yield ?yieldProduction;
    agro:inDistrict ?districturi.
  ?districturi rdfs:label ?district.
    FILTER (lang(?district) = "en")
}ORDER BY ASC(?cropProdPerPerson)
```

Listing 16: Which districts are self dependent in their agri-products?

Agriculture Spatial Data Visualization

In order to understand the data, the spatial part of agriculture data like rainfall stations, airports and district are visualized using the *Facete* [Stadler et al., 2014] tool. Facete is a web-based exploration and visualization application enabling the spatial faceted browsing of data with a spatial dimension. Figure 29 demonstrates the information about the rainfall station location. The left section of the figure contains the selection field where we selected station properties and below the facet value can be seen. The middle section of the figure contain the information about data which is displayed according to the selection from left section. For example, a agriculture planner wants to know the numbers of stations and their location in a specific area for collecting weather information. A planner may also interested for finding the nearby locations and visualized them. In the figure, the data `0407, 0408, 0413 etc.` are rainfall station numbers. The right part of the figure shows the location of different spatial location and the details can be seen by clicking the marker. The location of the *Lumbini* station is clicked which is marked by blue color.

SUMMARY AND OUTLOOK

By providing the *AgriNepalData* data sources as Linked Data and combining them with other data sets, it is now possible to obtain a variety

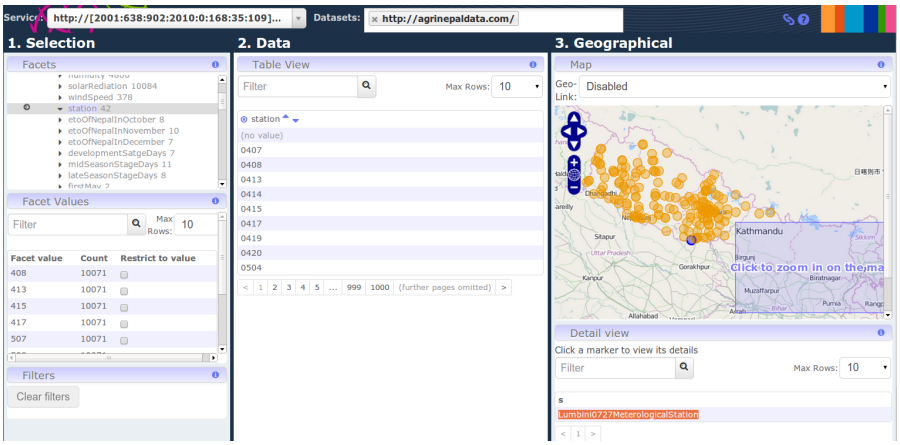


Figure 29: Facete visualization of the *Lumbini* rainfall station.

of related agricultural information from one structured data set. We have done an initial demonstration via SPARQL queries or tool deployments that the resulting data enables several relevant use cases. This is the first step on a larger research agenda aiming at an increase of productivity and efficiency of farming in Nepal. While our study is limited to Nepal, it can also be generalised to other countries in the mid and long term.

To extend this work, we plan to convert more data from previous years to enable large-scale temporal analysis. Furthermore, we intend to include other data from other domains that can influence the agricultural process like transportation and trade. As an important extension to our work, we aim to implement question answering techniques to enable non-experts to access our data set from the project web site and mobile applications. Developing automatised solutions for each of the manual data conversion/verification tasks is one of the remaining general research challenges given the heterogeneous nature of data published by a number of different bodies.

12

NIF4OGGD – NLP INTERCHANGE FORMAT FOR OPEN GERMAN GOVERNMENTAL DATA

This chapter describes the NIF4OGGD, a data set for integrating open German governmental data with geospatial data [Sherif et al., 2014]. The author integrated NIF4OGGD with the other mentioned data sets. Also, he co-designed the NIF4OGGD ontology, took part of the data set creation and co-wrote the paper.

The open data movement has become increasingly important as a key driver for economical success. For instance, the German government has passed an *E-Government-Law*[1], which emphasises the importance of machine-readable data provided by official agencies and other German government organizations. To use this potential we present our multi-data set mashup NIF4OGGD[2].

Many open data platforms, however, still provide data in (often proprietary, non-standard) formats that lack machine-readability. There is a growing number of data repositories. For example, the city of *Berlin* (see Section 12.1) provides more than 200 data sets from kindergarten locations across city districts to ozone pollution distributions. While this diversity allows to create a wide range of applications and mashups, the integration of several sources remains a challenging problem. In order to provide a standardized solution to gather and correlate open data documents, we propose to use formats and tools that achieve interoperability between Natural Language Processing (NLP) tools, language resources and annotations. A format that fits these requirements is the Linked Data-based NIF [Hellmann et al., 2012, 2013]. We employ NIF to connect three different government data repositories and interlink it with spatial information in the Web of Data. The resulting language resource is published along with a user interface for browsing it.

The contributions of this work are: (1) we describe a process for creating a novel data set comprising several open data sets across Germany, which we (2) made publicly available. Furthermore, we (3) use Linked Data via NIF as multi-lingual interchange format to allow queries across data sets. Additionally, we (4) offer a simple search engine interface for end users. Finally, we (5) provide use cases that show the potential impact of NIF4OGGD.

OPEN GERMAN GOVERNMENTAL DATA

In general, in Germany it is not allowed to publish person-centred data without the consent of the described persons. Taking this restriction into account, most data portals upload statistical data or textual data from public hearings. This data is difficult to understand without

1 http://www.bmi.bund.de/SharedDocs/Downloads/DE/Themen/OED_Verwaltung/Informationsgesellschaft/egovg_verkuendung.pdf

2 http://aksw.org/Projects/NIF4OGGD

an extra layer of structural information and most often proprietary, unstructured, not standardized and thus not readable by a machine. We aim to overcome these problems via NIF, the NLP interchange format which is based on the principles of *Linked Data*.

Tim-Berners Lee postulated the *5 Star principle*[3] for sharing open data as Linked Data. The first step is to make data available on the web via an open licence and put it as a second step online in an structured format like Excel. The data will get a third star if it is in a non-proprietary format like CSV and a fourth star if URIs are used to denote resources. Last, it is 5 Star data if it links to other data sets creating a richer context. We analysed three local portals providing open data with respect to their 5 Star quality. Moreover, we look for the availability of geodata, SPARQL [Prud'hommeaux and Seaborne, 2008] endpoint availability and whether the portal has a visual interface for analysing the data on-site, e.g., a heat map function. The results can be seen in Table 19.

Berlin[4] is the pioneer portal for open data in Germany. Although it does not provide 5 Star data, it comes up with an API and several well structured and non-proprietary data files about, e.g., public wireless LAN locations, events or a list of all memorials. Overall, there are 289 data sets in 21 categories.

Bonn[5] is currently not able to deliver content neither via download nor API. The city of Bonn is discussing how and what data should be delivered.

Cologne[6] offers 172 data sets from 9 categories for download. Although, the portal provides several open license data sets, no data set exists following the Linked Data paradigm. Like in the case of *Bonn*, administrative data is available via an administration management system[7] that already has been scraped by the Cologne Open Data Portal[8].

Portal	★	★★	★ ★★	★★ ★★	★★ ★★★	geodata	SPARQL	visual analytics
Berlin	(✓)	(✓)	(✓)	(✓)	(✓)	✓	✗	✗
Bonn	✓	✗	✗	✗	✗	✗	✗	✗
Cologne	(✓)	(✓)	(✓)	✗	✗	✓	✗	✗
NIF4OGGD	✓	✓	✓	✓	✓	✓	✓	✓

Table 19: Different data portals, their 5 Star classification and further features. (✓) means that not all data is available at this particular star level.

3 http://www.w3.org/DesignIssues/LinkedData.html
4 http://daten.berlin.de/, 22 October 2013
5 http://www.bonn.de/rat_verwaltung_buergerdienste/aktuelles/open_data, 22 October 2013
6 http://www.offenedaten-koeln.de/, 22 October 2013
7 http://ratsinformation.stadt-koeln.de/infobi.asp
8 http://offeneskoeln.de/

DATASET

In this section, we briefly present *LinkedGeoData* [Stadler et al., 2012] and our data extraction from E-Government data portals. Moreover, we point out current problems and how we overcome each of them by using NIF and the Linked Data paradigm.

LinkedGeoData

The OpenStreetMap (OSM)[9] project offers a freely available and rich source of spatial data. OSM consists of more than 1 billion nodes and 100 million ways stored in a relational database. LinkedGeoData (LGD)[10] provides a transformation of OSM data into RDF [Auer et al., 2009], which comprises approximately 20 billion triples. LGD is available according to the Linked Data principles and interlinked with *DBpedia* [Auer et al., 2008] and *GeoNames*[11]. LGD provides its RDF data not only in form of free dump files[12], but the data can also be queried via a SPARQL endpoint[13]. LGD provides an ontology for structuring the information in *OpenStreetMap*. For instance, it contains more than forty subclasses of `HighWay`. As an example, Listing 17 is a SPARQL query, which retrieves all streets of the city of *Berlin*, along with latitude and longitude information.

For obtaining the relevant data sets, we downloaded OSM dumps[14] for *Berlin* and *North Rhine-Westphalia* and applied the LGD conversion[15] to them. The resulted data sets are stored in the project endpoint[16].

Data Extraction

To enrich the spatial data, government data was retrieved from administration management systems of Bonn and Cologne, see Section 12.1. All of the data is document based, containing *PDF documents* with administrative decisions and documented enquiries of citizens. Each document serves as a resource in the administration management systems and features a title and minor meta-data. To obtain the data, the portals where queried by custom web scrapers.

9 http://openstreetmap.org
10 http://linkedgeodata.org/
11 http://www.geonames.org/ontology/documentation.html
12 http://downloads.linkedgeodata.org/
13 http://linkedgeodata.org/sparql
14 http://geofabrik.de
15 https://github.com/GeoKnow/LinkedGeoData
16 http://mlode.nlp2rdf.org/sparql

```
PREFIX lgd: <http://linkedgeodata.org/ontology/>
PREFIX geovocab: <http://geovocab.org/geometry#>
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
SELECT DISTINCT ?s ?streetLabel ?lat ?long
FROM <http://thedatahub.org/dataset/lgd-berlin>
WHERE{
  ?s a lgd:HighwayThing;
          rdfs:label ?streetLabel;
          geovocab:geometry ?geometry.
  ?geometry lgd:posSeq ?posSeq.
  ?posSeq ?posSeqP ?posSeqO.
  ?s2 geovocab:geometry ?posSeqO;
          geo:lat ?lat;
          geo:long ?long.
}
```

Listing 17: Select all streets of Berlin along with latitude and longitude.

In the case of *Bonn*, the PDF documents where downloaded via `curl`[17] and converted to textual data via `pdf2text`[18]. Because of this procedure, the original formatting of the documents and any information contained in it (like tabular data) was lost. The negative effect of this is limited in our case, since we only perform text search over the documents. However, keeping this information and performing advanced extraction methods is one of our steps in a larger research agenda.

In case of *Cologne*, the above mentioned open data portal already performed this step and allowed us to use the textual data of the documents and the titles of the resources.

ARCHITECTURE

The NIF4OGGD architecture shown in Figure 30 has three main modules: Conversion of documents to NIF, Enrichment and Visualization & Search. The modules provide a flexible solution to integrate multiple web data sources using (semantic) web standards and NIF.

Conversion of Documents to NIF

For the conversion to NIF, two resources where established for every document: First, the textual content of a document was added to a resource of the type `nif:Context` with the `nif:isString` predicate as a literal. The URL of the source document was also included in the metadata to ensure full traceability of the original data. Second, the title string of the document was in turn added to a resource of the type `nif:Title` with the `nif:anchorOf` predicate and linked to the

17 http://curl.haxx.se/
18 http://www.cyberciti.biz/faq/converter-pdf-files-to-text-format-command/

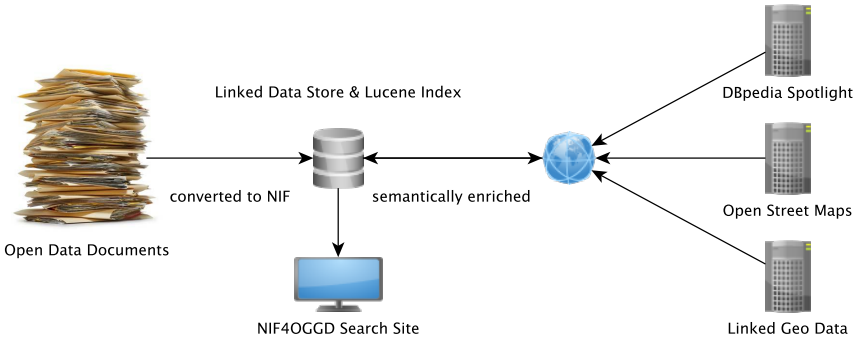


Figure 30: Architecture of the NIF4OGGD system.

Field	IdfpoPSVBNtxx#txxDtxx	Norm	Value
city	Idfp--S--Nnum--------	1.0	Koeln
description	Idfp--S--Nnum--------	1.0	Neumarkt
document	Idf---SV-Nnum--------	0.005E	Antrag auf Aufstellung eines Bebauungsplanes in KölnFlittard, Pützl
document	Idf---SV-Nnum--------	0.005E	Eigenbetriebsähnliche Einrichtung Veranstaltungszentrum Köln Wirt
document	Idf---SV-Nnum--------	0.005E	Baubeschluss zur Realisierung der Pilotanwendung einer umweltser
document	Idf---SV-Nnum--------	0.005E	Ergebnis der Lärmmessungen am Brüsseler Platz@deStellungnahm
document	Idf---SV-Nnum--------	0.005E	Aufwertung der Veedel im Stadtbezirk Innenstadt zu Stadtteilen@de
latitude	Id----S--------------	---	50.9358#50.9359#50.9358#50.9358#50.9358#50.9358#50.9358
longitude	Id----S--------------	---	6.94708#6.94709#6.94799#6.94839#6.94839#6.94633#6.94896
types	Idfp--S--Nnum--------	1.0	DBpedia:Street
url	Id----S--------------	---	http://linkedgeodata.org/triplify/way10807067

Figure 31: *Lucene* index.

context resource via `nif:referenceContext`. Listing 18 provides an example of a NIF conversion of a document presented in Figure 32.

Enrichment

We used the approaches introduced in Chapter 8 to enrich NIF4OGGD by geospatial data.

In order to integrate geographical and governmental data, the governmental data is enriched by geospatial data via DEER (see Chapter 8). The output of this extracting is stored as standardized NIF files. The collection of LGD locations and government data is additionally stored as a set of documents in a *Lucene*[19] index. Furthermore, we built an in-memory dictionary for the data provided by LGD. This dictionary is used for performing an analysis in all NIF files that contain government data. Specifically in the data stored in `nif:isString` property, when a location name occurs in a document, all the data is stored in the *Lucene* document corresponding to that location. Figure 31 shows all fields stored for an indexed document.

Visualization & Search

Aiming to allow an easy integration of NIF4OGGD into external web processes, we implemented RESTful and SOAP web services for the search process. The web service interface allows access to query a

19 http://lucene.apache.org

```
<http://offeneskoeln.de/dokumente/3819-2011/#char=0,1596>
  a nif:Context , nif:RFC5147String ;
  nif:isString "Baubeschluss zur Realisierung der Pilotanwendung einer
       umweltintensiven [...]"@de ;
  nif:sourceUrl <http://offeneskoeln.de/dokumente/3819-2011/> ;
  nif:sourceUrl <http://ratsinformation.stadt-koeln.de/vo0050.asp?__kvonr=29978> .
<http://offeneskoeln.de/dokumente/3819-2011/#char=0,40>
  a nif:Title , nif:RFC5147String ;
  nif:beginIndex "0"^^xsd:nonNegativeInteger ;
  nif:endIndex "12"^^xsd:nonNegativeInteger ;
  nif:referenceContext <http://offeneskoeln.de/dokumente/3819-2011/#char=0,1596> ;
  nif:anchorOf "Baubeschluss"@de .
```

Listing 18: Example NIF resources

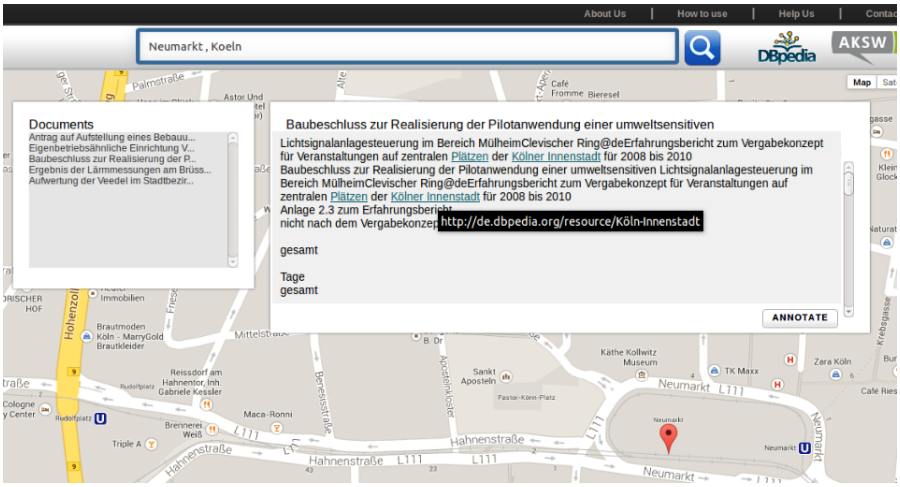


Figure 32: Searching for governmental documents mentioning *Neumarkt* in *Cologne*.

Lucene index and returns *JSON* and NIF format. NIF4OGGD is deployed as a web service and has a very simple user interface for demonstration. The source code is available at our project repository[20]. The user interface[21] was built using the Google Maps API, allowing to search locations by name. Once the location is selected by the user, the map shows it on a map and displays related government documents stored in the index. It is also possible to annotate the texts using DBpedia Spotlight [Mendes et al., 2011] and use the annotated resources to discover more information in a graph database. Figure 32 shows the NIF4OGGD user interface.

USE-CASES

In this section, we outline selected application scenarios and use cases for *NIF4OGGD*.

20 https://github.com/aksw/nif4oggd
21 http://nif4oggd.aksw.org

Data Retrieval.

An example use case for data retrieval are citizens searching for events in their neighbourhood. NIF4OGGD provides data about what governmental events happen in a specific area. For instance, Figure 32 shows a user query for governmental documents mentioning *Neumarkt* in *Cologne*. As shown, there are many documents about the search topic, in which the user is free to browse.

Interoperability using NIF

The aligned governmental documents using NIF representation enable searching for the same entity across different authorities' documents. NIF's interoperability capabilities make it easy to query all occurrences of a certain text segment in all available documents without the need of any additional indexing. Especially, sophisticated SPARQL queries can posed to interlinked NIF documents, e.g., to find all places across Berlin where demonstrations were declared about some particular political issue.

Listing 19 introduces an example of using NIF to retrieve all documents mentioning *Baubeschluss*.

```
PREFIX str: <http://nlp2rdf.lod2.eu/schema/string/>
SELECT ?document
WHERE{
  ?s str:isString ?document.
  ?textSegment str:referenceContext ?s;
            str:anchorOf "Baubeschluss".
}
```

Listing 19: List of all occurrences of "Baubeschluss" using NIF

Information Aggregation

Using *DBpedia Spotlight*, NIF4OGGD annotates the governmental documents using the cross-domain data set of *DBpedia*, which provides added value to the data. As an example, in Figure 32 user can use the annotated entities of the presented document to get detailed data about *Kölner Innenstadt*.

SUMMARY AND OUTLOOK

NIF4OGGD is a novel data set providing geospatial data that is integrated with governmental information. We presented an extraction process for creating NIF4OGGD and made it freely available. In addition, our project constitutes a new central sharing point for Open

German Governmental Data which is published following the 5 Star principles.

In the future, we plan to extend our data sets and enrich it with more data from the LOD-cloud. Furthermore, we plan to employ additional NLP-algorithms, e.g., to just show documents with a valid time range, to provide more trustful documents via NIF4OGGD.

13 CONCLUSION AND FUTURE WORK

In Part II we proposed a set of approaches for RDF data sets integration and enrichment. Also, we showed a set of use cases and application scenarios of our approaches in the last four chapters. In this chapter, we present conclusion and future extensions for each of our approaches.

POINT SET DISTANCE MEASURES FOR GEOSPATIAL LD

In Chapter 4, we presented an evaluation of point set distance measures for Link Discovery (LD) on geo-spatial resources. We evaluated these distances on sample from three different data sets. Our results suggest that while different measures perform best on the data sets we used, the *mean* distance measure is the most time-efficient and overall best measure to use for LD. We also showed that all measures apart from the *Fréchet* distance can scale even on large data sets when combine with an approach such as ORCHID (see Section 2.1.2). While working on this survey, we realized the need for a full-fledged benchmark for geo-spatial LD.

In future work, we will devise such a benchmark and make it available to the community. All the point sets measures presented in this work were integrated in the LIMES framework available at `http://limes.sf.net`. Additionally, we will extend this framework with dedicated versions of ORCHID for the different measures presented herein. Moreover, we will aim to devise means to detect the best measure for any given geo-spatial data set.

UNSUPERVISED LD THROUGH KNOWLEDGE BASE REPAIR

In Chapter 5, we introduced COLIBRI, the first unsupervised LD approach which attempts to repair instance knowledge in n knowledge bases ($n \geqslant 2$) to improve its linking accuracy. COLIBRI relies on the deterministic approach EUCLID for detecting links between knowledge bases. We compare EUCLID with the state-of-the-art and showed that it outperforms the state-of-the-art while being deterministic. We then presented the core of COLIBRI, which relies on voting, error detection and error correction approaches. We showed how COLIBRI can combine these steps to improve the quality of instance knowledge in input knowledge bases effectively. Our evaluation suggests that our approach is robust and can be used by error rates up to 50% when provided with at least three knowledge bases. In addition, our results

show that COLIBRI can improve the results of EUCLID by up to 14% F-measure. While our approach is automatic, the repair step could be refined to be semi-automatic for high-accuracy domains such as medical care.

In future work, we plan to extend our evaluation further and analyse our performance on real data as well as on knowledge bases of different size. We plan to deploy our approach in interactive scenarios within which users are consulted before the knowledge bases are updated. The voting procedure implemented by COLIBRI can be used to provide users with a measure for the degree of confidence in a predicted link and in the need for a repair within an interactive learning scenario. To this end, we will use the entries in the voting matrices to provide scores about how sure we are that data should contains errors. Finally, we aim to devise PFMs that perform well even in scenarios where one-to-one links are not given. So far, COLIBRI does not take the naming conventions within the knowledge bases into consideration. We thus plan to combine COLIBRI with pattern learning approaches for data integration to ensure that the updated knowledge bases remain consistent w.r.t. the naming conventions they employ.

LOAD BALANCING FOR LD

In Chapter 6, we presented and evaluated load balancing techniques for LD on parallel hardware based on particle-swarm optimization. In particular, in the PSO approach, we applied *particle-swarm optimization* to optimize the distribution of tasks of different sizes over a given number of processors. While the PSO outperforms classical load balancing algorithms, it has the main drawback of being indeterministic in nature. Therefore, we proposed the DPSO, where we altered the task selection of PSO for ensuring deterministic load balancing of tasks. We combined the PSO approaches with the ORCHID algorithm. All the implemented load balancing approaches were evaluated on real and artificial data sets. Our evaluation suggests that while naïve approaches can be super-linear on small data sets, our deterministic particle swarm optimization outperforms both naïve and classical load balancing approach such as greedy load balancing on large data sets as well as a data sets which originate from highly skewed distributions.

Although we achieve reasonable results in terms of scalability, we plan to further improve the time efficiency of our approaches by enabling the splitting of one task over more than one processor. As an extension of DPSO, we plan to implement a caching technique, which enables DPSO to be used on larger data sets that can not be fitted in memory [Hassan et al., 2015]. While DPSO was evaluated in combination with ORCHID in Section 6.3, we will study the combination of

our approach with other space tiling and/or blocking algorithms for generating parallel tasks.

A GENERALIZATION APPROACH FOR AUTOMATIC LD

In Chapter 7, we proposed (to the best of our knowledge) the first approach to learn link specifications from positive examples via generalisation over the space of link specifications. We presented a simple operator φ that aims to achieve this goal as well as the complete operator ψ. We evaluated φ and ψ against state-of-the-art LD approaches and showed that we outperform them on benchmark data sets. The completeness of ψ proved to be an advantage pertaining to its performance on complex benchmarks. We also considered scalability and showed that ψ can be brought to scale similarly to φ when combined with the pruning approach we developed.

In future work, we aim to parallelize our approach as well as extend it by trying more aggressive pruning techniques for even better scalability.

AUTOMATING RDF DATASET ENRICHMENT AND TRANSFORMATION

In Chapter 8, we presented an approach for learning enrichment pipelines based on a refinement operator. To the best of our knowledge, this is the first approach for learning RDF based enrichment pipelines and could open up a new research area. We also presented means to self-configure atomic enrichment pipelines so as to find means to enrich data sets according to examples provided by an end user. We showed that our approach can easily reconstruct manually created enrichment pipelines, especially when given a prototypical example and when faced with regular data sets. Obviously, this does not mean that our approach will always achieve such high F-measures. What our results suggest is primarily that if a human uses an enrichment tool to enrich his/her data set manually, then our approach can reconstruct the pipeline. This seems to hold even for relatively complex pipelines.

Although we achieved reasonable results in terms of scalability, we plan in the future to improve time efficiency by parallelising the algorithm on several CPUs as well as load balancing. The framework underlying this study supports directed acyclic graphs as enrichment specifications by allowing to split and merge data sets. In future work, we will thus extend our operator to deal with graphs in addition to sequences. Moreover, we will look at pro-active enrichment strategies as well as active learning.

Part IV

APPENDIX

A

CURRICULUM VITAE

Mohamed Ahmed Mohamed Sherif

Straße des 18. Oktober 28, Wh. 91
04103 Leipzig, Germany.
(+49) 15751432713
sherif@informatik.uni-leipzig.de
http://aksw.org/MohamedSherif.html

Personal Data

Name: Mohamed Ahmed Mohamed Sherif
Birth date: December 5th, 1980
Birth place: Gharbya, Egypt
Nationality: Egyptian
Marital status: Married

Education & Work

2012 – Present
University of Leipzig (Leipzig, Germany)
Ph.D., Faculty of Mathematics and Computer Science, Department of Computer Science.
Thesis title: *Automating Geospatial RDF Dataset Integration and Enrichment*.

2009 – 2011
Suez Canal University (Ismailia, Egypt)
Research Assistant, Faculty of Computers and Informatics, Department of Information Systems.

2004 – 2009
Menoufia University (Shepen El-Kom, Egypt)

M.Sc., Faculty of Computer and Information, Department of Information Systems.
Thesis title: *Web based 3D Geographical Information System*.

1998 – 2002
Suez Canal University (Ismailia, Egypt)
B.Sc., Faculty of Computer and Information, Department of Computer Science. **Very good grade with degree of honour.**
Graduation Project: *Design and Implementation of Hotel Management System*. **Excellent grade, best project award.**

Research Interests

- Semantic Web
- Artificial Intelligent
- Data Integration
- Data Enrichment

Selected Publications

1. **Sherif, M. A.**, Ngonga Ngomo, A.-C., and Lehmann, J. (2015). Automating RDF dataset transformation and enrichment. In 12th Extended Semantic Web Conference, Portoroz, Slovenia, 31st May - 4th June 2015. Springer.

2. **Sherif, M. A.** and Ngonga Ngomo, A.-C. (2014). Semantic quran: A multilingual resource for natural-language processing. Semantic Web Journal, Special Call for Linked Dataset descriptions.

3. **Sherif, M. A.** and Ngonga Ngomo, A.-C. (2015a). An optimization approach for load balancing in parallel link discovery. In Proceedings of the 11th International Conference on Semantic Systems (SEMANTICS '15).

4. **Sherif, M. A.**, Coelho, S., Usbeck, R., Hellmann, S., Lehmann, J., Brümmer, M., and Both, A. (2014). NIF4OGGD - NLP interchange format for open German governmental data. In The 9th edition of the Language Resources and Evaluation Conference, 26-31 May, Reykjavik, Iceland.

5. Grange, J. J. L., Lehmann, J., Athanasiou, S., Rojas, A. G., Giannopoulos, G., Hladky, D., Isele, R., Ngonga Ngomo, A.-C., **Sherif, M. A.**, Stadler, C., and Wauer, M. (2014). The GeoKnow generator: Managing geospatial data in the linked data web. In Proceedings of the Linking Geospatial Data Workshop.

6. Ngonga Ngomo, A. N., **Sherif, M. A.**, and Lyko, K. (2014). Unsupervised link discovery through knowledge base repair. 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings, pages 380–394.

7. Pokharel, S., **Sherif, M. A.**, and Lehmann, J. (2014). Ontology based data access and integration for improving the effectiveness of farming in Nepal. In Proc. of the International Conference on Web Intelligence.

8. Stadler, C., Unbehauen, J., Westphal, P., **Sherif, M. A.**, and Lehmann, J. (2015). Simplified RDB2RDF mapping. In Proceedings of the 8th Workshop on Linked Data on the Web (LDOW2015), Florence, Italy.

9. Zaveri, A., Kontokostas, D., **Sherif, M. A.**, Bühmann, L., Morsey, M., Auer, S., and Lehmann, J. (2013a). User-driven quality evaluation of DBpedia. In Proceedings of 9th International Conference on Semantic Systems, I-SEMANTICS '13, Graz, Austria, September 4-6, 2013, pages 97–104. ACM.

10. Zaveri, A., Lehmann, J., Auer, S., Hassan, M. M., **Sherif, M. A.**, and Martin, M. (2013b). Publishing and interlinking the global health observatory dataset. Semantic Web Journal, Special Call for Linked Dataset descriptions(3):315–322.

Technical and Programming Skills

- **Programming Languages Skills:**
  - Java, C / C++ (Professional).
  - PHP, Javascript, .NET, VBscript (Intermediate).
- **Database Systems:**
  - Oracle, MySQL, MongoDB, SQL Server

Selected Projects

- **DEER**: `http://aksw.org/Projects/DEER`
  RDF data extraction and enrichment framework.
- **LIMES**: `http://aksw.org/Projects/LIMES`
  Link discovery framework for metric spaces.
- **GeoKnow**: `http://geoknow.eu`
  Making the web an exploratory for geospatial knowledge.

Language Skills

- **Arabic**: Native
- **English**: Advanced
- **German**: Intermediate (B2 Certificate)

Research Community Service

- **Program Committee** The 1st International Conference On Advanced Intelligent System and Informatics (AISI) 2015
- **Organizer** for Leipziger Semantic Web Tag (LSWT) 2013
- **Reviewer** for i-challenge 2013, ESWC 2016
- **Presenter** for *ESWC 2014*, *ESWC 2015*, *WIC 2014*

BIBLIOGRAPHY

Abel, F., Gao, Q., Houben, G.-J., and Tao, K. (2011). Semantic enrichment of twitter posts for user profile construction on the social web. In *Proc. of ESWC*, pages 375–389. Springer. (Cited on page 16.)

Akl, S. G. (2004). Superlinear performance in real-time parallel computation. *The Journal of Supercomputing*, 29(1):89–111. (Cited on pages 14 and 60.)

Alba, E. (2002). Parallel evolutionary algorithms can achieve superlinear performance. *Information Processing Letters*, 82(1):7–13. (Cited on pages 14 and 60.)

Ali, L., Janson, T., and Schindelhauer, C. (2014). Towards load balancing and parallelizing of rdf query processing in p2p based distributed rdf data stores. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*, pages 307–311. IEEE. (Cited on page 14.)

Allen, R. G., Pereira, L. S., Raes, D., Smith, M., et al. (1998). Crop evapotranspiration-guidelines for computing crop water requirements-fao irrigation and drainage paper 56. *FAO, Rome*, 300:6541. (Cited on page 130.)

Alt, H. and Godau, M. (1995). Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91. (Cited on pages 11, 26, and 27.)

Atallah, M. J. (1983). A linear time algorithm for the hausdorff distance between convex polygons. Technical report, Purdue University, Department of Computer Science. (Cited on page 11.)

Atallah, M. J., Ribeiro, C. C., and Lifschitz, S. (1991). Computing some distance functions between polygons. *Pattern recognition*, 24(8):775–781. (Cited on page 11.)

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2008). DBpedia: A nucleus for a web of open data. In *Proceedings of the 6th International Semantic Web Conference (ISWC)*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer. (Cited on page 136.)

Auer, S., Dietzold, S., and Riechert, T. (2006). OntoWiki - A Tool for Social, Semantic Collaboration. In Isabel, C., Stefan, D., and Allemang, D., editors, *The Semantic Web - ISWC 2006, 5th International*

Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, Proceedings, volume 4273 of *Lecture Notes in Computer Science*, pages 736–749, Berlin / Heidelberg. Springer. (Cited on page 98.)

Auer, S. and Lehmann, J. (2010). Making the web a data washing machine—creating knowledge out of interlinked data. *Semantic Web Journal*. (Cited on page 121.)

Auer, S., Lehmann, J., and Hellmann, S. (2009). LinkedGeoData - adding a spatial dimension to the web of data. In *Proc. of 8th International Semantic Web Conference (ISWC)*. (Cited on pages 1 and 136.)

Auer, S., Lehmann, J., Ngonga Ngomo, A., and Zaveri, A. (2013). Introduction to linked data and its lifecycle on the web. In *Reasoning Web*, pages 1–90. (Cited on pages 1, 3, 63, and 119.)

Badea, L. (2000). Perfect refinement operators can be flexible. In Horn, W., editor, *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 266–270. IOS Press. (Cited on page 15.)

Badea, L. and Nienhuys-Cheng, S.-H. (2000). A refinement operator for description logics. In Cussens, J. and Frisch, A., editors, *Proceedings of the 10th International Conference on Inductive Logic Programming*, volume 1866 of *Lecture Notes in Artificial Intelligence*, pages 40–59. Springer-Verlag. (Cited on page 15.)

Badea, L. and Stanciu, M. (1999). Refinement operators can be (weakly) perfect. In Džeroski, S. and Flach, P., editors, *Proceedings of the 9th International Workshop on Inductive Logic Programming*, volume 1634 of *Lecture Notes in Artificial Intelligence*, pages 21–32. Springer-Verlag. (Cited on page 15.)

Barequet, G., Dickerson, M., and Goodrich, M. T. (2001). Voronoi diagrams for convex polygon-offset distance functions. *Discrete & Computational Geometry*, 25(2):271–291. (Cited on page 11.)

Barequet, G., Dickerson, M. T., and Goodrich, M. T. (1997). Voronoi diagrams for polygon-offset distance functions. In *Algorithms and Data Structures*, pages 200–209. Springer. (Cited on page 11.)

Bartoň, M., Hanniel, I., Elber, G., and Kim, M.-S. (2010). Precise hausdorff distance computation between polygonal meshes. *Comput. Aided Geom. Des.*, 27(8):580–591. (Cited on page 11.)

Bhattacharya, B. K. and Toussaint, G. T. (1983). Efficient algorithms for computing the maximum distance between two finite planar sets. *Journal of Algorithms*, 4(2):121 – 136. (Cited on page 23.)

Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data-the story so far. *International journal on semantic web and information systems*, 5(3):1–22. (Cited on page 119.)

Bizer, C. and Schultz, A. (2010). The R2R Framework: Publishing and Discovering Mappings on the Web. In *COLD Workshop*. (Cited on page 16.)

Bloehdorn, S. and Sure, Y. (2007). Kernel methods for mining instance data in ontologies. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, pages 58–71. (Cited on page 13.)

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1987). Occam's razor. *Inf. Process. Lett.*, 24(6):377–380. (Cited on page 85.)

Böhm, C., de Melo, G., Naumann, F., and Weikum, G. (2012). LINDA: distributed web-of-data-scale entity matching. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 2104–2108. (Cited on page 13.)

Bowring, B. (1984). The direct and inverse solutions for the great elliptic line on the reference ellipsoid. *Bulletin géodésique*, 58(1):101–108. (Cited on pages 20 and 29.)

Buhmann, L. and Lehmann, J. (2013). Pattern based knowledge base enrichment. In *12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*. (Cited on pages 4 and 79.)

Cai, Q., Gong, M., Ma, L., Ruan, S., Yuan, F., and Jiao, L. (2014). Greedy discrete particle swarm optimization for large-scale social network clustering. *Information Sciences*. (Cited on page 55.)

Caragiannis, I., Flammini, M., Kaklamanis, C., Kanellopoulos, P., and Moscardelli, L. (2011). Tight bounds for selfish and greedy load balancing. *Algorithmica*, 61(3):606–637. (Cited on page 53.)

Chambers, E. W., Colin de Verdière, É., Erickson, J., Lazard, S., Lazarus, F., and Thite, S. (2010). Homotopic fréchet distance between curves *or, walking your dog in the woods in polynomial time*. *Computational Geometry*, 43(3):295–311. (Cited on page 11.)

Cheatham, M. and Hitzler, P. (2013). String similarity metrics for ontology alignment. In *International Semantic Web Conference (2)*, pages 294–309. (Cited on pages 2 and 18.)

Choudhury, S., Breslin, J. G., and Passant, A. (2009). *Enrichment and ranking of the youtube tag space and integration with the linked data cloud*. Springer. (Cited on page 16.)

Chrisman, N. and Girres, J.-F. (2013). First, do no harm: Eliminating systematic error in analytical results of gis applications. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XL-2/W1:35–40. (Cited on pages 20 and 29.)

Cook IV, A. F., Driemel, A., Har-Peled, S., Sherette, J., and Wenk, C. (2011). Computing the fréchet distance between folded polygons. In *Algorithms and Data Structures*, pages 267–278. Springer. (Cited on page 12.)

d'Amato, C., Fanizzi, N., and Esposito, F. (2008). Non-parametric statistical learning methods for inductive classifiers in semantic knowledge bases. In *Proceedings of the 2th IEEE International Conference on Semantic Computing (ICSC 2008), August 4-7, 2008, Santa Clara, California, USA*, pages 291–298. (Cited on page 13.)

Denis, F., Gilleron, R., and Letouzey, F. (2005). Learning from positive and unlabeled examples. *Theoretical Computer Science*, 348(1):70 – 83. Algorithmic Learning Theory (ALT 2000) 11th International Conference, Algorithmic Learning Theory 2000. (Cited on page 14.)

Dietze, S., Sanchez-Alonso, S., Ebner, H., Yu, H. Q., Giordano, D., Marenzi, I., and Nunes, B. P. (2013). Interlinking educational resources and the web of data: A survey of challenges and approaches. *Program: electronic library and information systems*, 47(1):60–91. (Cited on page 16.)

Doan, A., Madhavan, J., Dhamankar, R., Domingos, P. M., and Halevy, A. Y. (2003). Learning to match ontologies on the semantic web. *VLDB J.*, 12(4):303–319. (Cited on page 13.)

Driemel, A., Har-Peled, S., and Wenk, C. (2012). Approximating the fréchet distance for realistic curves in near linear time. *Discrete & Computational Geometry*, 48(1):94–127. (Cited on page 11.)

Duda, R. O., Hart, P. E. P. E., and Stork, D. G. (2001). *Pattern classification*. Wiley, pub-WILEY:adr, second edition. (Cited on page 23.)

Eiter, T. and Mannila, H. (1997). Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133. (Cited on pages 11, 25, and 26.)

Ermilov, I., Auer, S., and Stadler, C. (2013). Csv2rdf: User-driven csv to rdf mass conversion framework. In *Proceedings of the ISEM '13, September 04 - 06 2013, Graz, Austria*. (Cited on page 124.)

Esposito, F., Fanizzi, N., Iannone, L., Palmisano, I., and Semeraro, G. (2004). Knowledge-intensive induction of terminologies from metadata. In *The Semantic Web - ISWC 2004: Third International Semantic Web Conference, Hiroshima, Japan, November 7-11, 2004. Proceedings*, pages 441–455. Springer. (Cited on page 15.)

Euzenat, J. (2008). Algebras of ontology alignment relations. In *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, pages 387–402. (Cited on page 13.)

Euzenat, J. and Shvaiko, P. (2007). *Ontology matching*. Springer-Verlag, Heidelberg (DE). (Cited on page 16.)

Fanizzi, N., Ferilli, S., Mauro, N. D., and Basile, T. M. A. (2003). Spaces of theories with ideal refinement operators. In Gottlob, G. and Walsh, T., editors, *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 527–532. Morgan Kaufmann. (Cited on page 15.)

Farrar, S. and Langendoen, T. (2003). A linguistic ontology for the semantic web. *GLOT INTERNATIONAL*, 7. (Cited on page 111.)

Ferrara, A., Montanelli, S., Noessner, J., and Stuckenschmidt, H. (2011). Benchmarking matching applications on the semantic web. In *The Semanic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29 - June 2, 2011, Proceedings, Part II*, pages 108–122. (Cited on pages 28 and 46.)

Fréchet, M. (1906). Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo*, 22(1):1–72. (Cited on page 26.)

Getoor, L. and Taskar, B. (2007). *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press. (Cited on page 13.)

Grange, J. J. L., Lehmann, J., Athanasiou, S., Rojas, A. G., Giannopoulos, G., Hladky, D., Isele, R., Ngonga Ngomo, A.-C., Sherif, M. A., Stadler, C., and Wauer, M. (2014). The geoknow generator: Managing geospatial data in the linked data web. In *Proceedings of the Linking Geospatial Data Workshop*.

Guthe, M., Borodin, P., and Klein, R. (2005). Fast and accurate hausdorff distance calculation between meshes. *Journal of WSCG*, 13(2):41–48. (Cited on page 12.)

Hartung, M., Groß, A., and Rahm, E. (2013). Composition methods for link discovery. In *Datenbanksysteme für Business, Technologie und Web (BTW), 15. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), 11.-15.3.2013 in Magdeburg, Germany. Proceedings*, pages 261–277. (Cited on page 13.)

Hasan, S., Curry, E., Banduk, M., and O'Riain, S. (2011). Toward situation awareness for the semantic sensor web: Complex event processing with dynamic linked data enrichment. *Semantic Sensor Networks*, page 60. (Cited on page 16.)

Hassan, M., Speck, R., and Ngonga Ngomo, A.-C. (2015). Using caching for local link discovery on large data sets. In *Engineering the Web in the Big Data Era*, volume 9114 of *Lecture Notes in Computer*

Science, pages 344–354. Springer International Publishing. (Cited on page 143.)

Hassanzadeh, O., Pu, K. Q., Yeganeh, S. H., Miller, R. J., Popa, L., Hernández, M. A., and Ho, H. (2013). Discovering linkage points over web data. *PVLDB*, 6(6):444–456. (Cited on pages 39 and 40.)

Hefny, H. A., Khafagy, M. H., and Wahdan, A. M. (2014). Comparative study load balance algorithms for map reduce environment. *International Journal of Computer Applications*, 106(18):41–50. (Cited on page 14.)

Hellmann, S., Lehmann, J., and Auer, S. (2012). Linked-data aware uri schemes for referencing text fragments. In *EKAW 2012*, Lecture Notes in Computer Science (LNCS) 7603. Springer. (Cited on pages 111 and 134.)

Hellmann, S., Lehmann, J., Auer, S., and Brümmer, M. (2013). Integrating nlp using linked data. In *12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*. (Cited on page 134.)

Hoang, H. H., Cung, T. N.-P., Truong, D. K., Hwang, D., and Jung, J. J. (2014). Semantic information integration with linked data mashups approaches. *International Journal of Distributed Sensor Networks*, 2014. (Cited on page 16.)

Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. (2013). Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61. (Cited on page 1.)

Huttenlocher, D. P., Kedem, K., and Kleinberg, J. M. (1992). On dynamic voronoi diagrams and the minimum hausdorff distance for point sets under euclidean motion in the plane. In *Proceedings of the Eighth Annual Symposium on Computational Geometry*, SCG '92, pages 110–119, New York, NY, USA. ACM. (Cited on page 26.)

Iannone, L., Palmisano, I., and Fanizzi, N. (2007). An algorithm based on counterfactuals for concept learning in the semantic web. *Applied Intelligence*, 26(2):139–159. (Cited on page 15.)

Isele, R. and Bizer, C. (2011). Learning linkage rules using genetic programming. In *Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011*. (Cited on pages 2, 12, and 16.)

Isele, R., Jentzsch, A., and Bizer, C. (2011a). Efficient multidimensional blocking for link discovery without losing recall. In *Proceedings of the 14th International Workshop on the Web and Databases 2011, WebDB 2011, Athens, Greece, June 12, 2011*. (Cited on pages 2 and 64.)

Isele, R., Jentzsch, A., and Bizer, C. (2011b). Efficient Multidimensional Blocking for Link Discovery without losing Recall. In *WebDB*. (Cited on pages 3, 5, 51, and 52.)

Isele, R., Jentzsch, A., and Bizer, C. (2012). Active learning of expressive linkage rules for the web of data. In *Web Engineering - 12th International Conference, ICWE 2012, Berlin, Germany, July 23-27, 2012. Proceedings*, pages 411–418. (Cited on page 12.)

Jiang, X., Huang, Y., Nickel, M., and Tresp, V. (2012). Combining information extraction, deductive reasoning and machine learning for relation prediction. In *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, pages 164–178. (Cited on page 13.)

Jin, X., Zhao, J., Sun, Y., Li, K., and Zhang, B. (2004). Distribution network reconfiguration for load balancing using binary particle swarm optimization. In *Power System Technology, 2004. PowerCon 2004. 2004 International Conference on*, volume 1, pages 507–510. IEEE. (Cited on page 14.)

Joshi, D., Samal, A., and Soh, L.-K. (2009). A dissimilarity function for clustering geospatial polygons. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 384–387. ACM. (Cited on page 12.)

Kaveh, A. (2014). Particle swarm optimization. In *Advances in Metaheuristic Algorithms for Optimal Design of Structures*, pages 9–40. Springer. (Cited on page 55.)

Kejriwal, M. and Miranker, D. P. (2015). Semi-supervised instance matching using boosted classifiers. In *The Semantic Web. Latest Advances and New Domains*, pages 388–402. Springer. (Cited on pages xiv, 76, and 77.)

Kennedy, J. (2010). Particle swarm optimization. In *Encyclopedia of Machine Learning*, pages 760–766. Springer. (Cited on page 55.)

Kiranyaz, S., Ince, T., and Gabbouj, M. (2014). Particle swarm optimization. In *Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*, pages 45–82. Springer. (Cited on page 55.)

Kitchenham, B. (2004). Procedures for performing systematic reviews. Technical report, Joint Technical Report Keele University Technical Report TR/SE-0401 and NICTA Technical Report 0400011T.1. (Cited on pages 4 and 20.)

Kolb, L. and Rahm, E. (2013). Parallel entity resolution with dedoop. *Datenbank-Spektrum*, 13(1):23–32. (Cited on pages 3, 14, and 51.)

Kolb, L., Thor, A., and Rahm, E. (2012). Load balancing for mapreduce-based entity resolution. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 618–629. IEEE. (Cited on pages 14 and 54.)

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500. (Cited on page 13.)

Kontokostas, D., Westphal, P., Auer, S., Hellmann, S., Lehmann, J., Cornelissen, R., and Zaveri, A. J. (2014). Test-driven evaluation of linked data quality. In *Proceedings of the 23rd international conference on World Wide Web*. to appear. (Cited on page 122.)

Köpcke, H., Thor, A., and Rahm, E. (2010). Evaluation of entity resolution approaches on real-world match problems. *PVLDB*, 3(1):484–493. (Cited on pages 40 and 72.)

Lehmann, J., Athanasiou, S., Both, A., Buehmann, L., Garcia-Rojas, A., Giannopoulos, G., Hladky, D., Hoeffner, K., Grange, J. J. L., Ngonga Ngomo, A., Pietzsch, R., Isele, R., Sherif, M. A., Stadler, C., Wauer, M., and Westphal, P. (2015). The geoknow handbook. Technical report.

Lehmann, J. and Haase, C. (2009). Ideal downward refinement in the EL description logic. In *Inductive Logic Programming, 19th International Conference, ILP 2009, Leuven, Belgium*. (Cited on page 15.)

Lehmann, J. and Hitzler, P. (2007). Foundations of refinement operators for description logics. In *ILP*, volume 4894 of *Lecture Notes in Computer Science*, pages 161–174. Springer. (Cited on page 15.)

Lehmann, J. and Hitzler, P. (2010). Concept learning in description logics using refinement operators. *Machine Learning journal*, 78(1-2):203–250. (Cited on pages 10 and 15.)

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2014). DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*. (Cited on pages 1, 90, and 120.)

Lopez, V., Unger, C., Cimiano, P., and Motta, E. (2013). Evaluating question answering over linked data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 21:3–13. (Cited on page 16.)

Ludwig, S. A. and Moallem, A. (2011). Swarm intelligence approaches for grid load balancing. *Journal of Grid Computing*, 9(3):279–301. (Cited on page 14.)

Mackaness, W. A., Ruas, A., and Sarjakoski, L. T. (2011). *Generalisation of geographic information: cartographic modelling and applications*. Elsevier. (Cited on page 28.)

Madhavan, J. and Halevy, A. Y. (2003). Composing mappings among data sources. In *VLDB*, pages 572–583. (Cited on page 13.)

McKenna, M. and Toussaint, G. T. (1985). Finding the minimum vertex distance between two disjoint convex polygons in linear time. *Computers & Mathematics with Applications*, 11(12):1227–1242. (Cited on page 24.)

McMaster, R. B. (1987). Automated line generalization. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 24(2):74–111. (Cited on page 28.)

Mendes, P. N., Jakob, M., García-Silva, A., and Bizer, C. (2011). Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 1–8. ACM. (Cited on page 139.)

Millard, I., Glaser, H., Salvadores, M., and Shadbolt, N. (2010). Consuming multiple linked data sources: Challenges and experiences. In *COLD Workshop*. (Cited on page 16.)

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., and PRISMA Group (2009). Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *PLoS medicine*, 6(7). (Cited on pages 4 and 20.)

Muggleton, S. (1997). Learning from positive data. In *Inductive logic programming*, pages 358–376. Springer. (Cited on page 14.)

Nentwig, M., Hartung, M., Ngonga, A. N., and Rahm, E. (2015). A survey of current link discovery frameworks. *Semantic Web Journal*. (Cited on page 19.)

Ngonga Ngomo, A. (2012). On link discovery using a hybrid approach. *J. Data Semantics*, 1(4):203–217. (Cited on pages 4, 13, 79, and 114.)

Ngonga Ngomo, A. (2013). ORCHID - reduction-ratio-optimal computation of geo-spatial distances for link discovery. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*, pages 395–410. (Cited on pages 2, 5, 9, 10, 12, 18, 26, 28, 52, 59, 60, and 61.)

Ngonga Ngomo, A. and Auer, S. (2011). LIMES - A time-efficient approach for large-scale link discovery on the web of data. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 2312–2317. (Cited on pages 8, 100, and 127.)

Ngonga Ngomo, A., Kolb, L., Heino, N., Hartung, M., Auer, S., and Rahm, E. (2013a). When to reach for the cloud: Using parallel hardware for link discovery. In *The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings*, pages 275–289. (Cited on pages 3 and 51.)

Ngonga Ngomo, A. and Lyko, K. (2012). EAGLE: efficient active learning of link specifications using genetic programming. In *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, pages 149–163. (Cited on pages 64 and 73.)

Ngonga Ngomo, A. and Lyko, K. (2013). Unsupervised learning of link specifications: deterministic vs. non-deterministic. In *Proceedings of the 8th International Workshop on Ontology Matching co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 21, 2013.*, pages 25–36. (Cited on pages 2, 12, 13, 38, 39, 41, and 73.)

Ngonga Ngomo, A., Lyko, K., and Christen, V. (2013b). COALA - correlation-aware active learning of link specifications. In *The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings*, pages 442–456. (Cited on pages 2, 12, and 42.)

Ngonga Ngomo, A.-C. (2012). On link discovery using a hybrid approach. *J. Data Semantics*, 1(4):203–217. (Cited on pages 3, 8, and 51.)

Ngonga Ngomo, A.-C., Heino, N., Lyko, K., Speck, R., and Kaltenböck, M. (2011). Scms - semantifying content management systems. In *ISWC 2011*. (Cited on page 87.)

Ngonga Ngomo, A.-C., Sherif, M. A., and Lyko, K. (2014). Unsupervised link discovery through knowledge base repair. In *Extended Semantic Web Conference (ESWC 2014)*. (Cited on page 37.)

Nickel, M., Tresp, V., and Kriegel, H. (2012). Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pages 271–280. (Cited on page 13.)

Nickerson, B. G. and Freeman, H. (1986). Development of a rule-based system for automatic map generalization. In *Proceedings of the Second International Symposium on Spatial Data Handling*, pages 537–556. (Cited on page 28.)

Nienhuys-Cheng, S.-H., Laer, W. V., Ramon, J., and Raedt, L. D. (1999). Generalizing refinement operators to learn prenex conjunctive normal forms. In *ILP*, volume 1634 of *Lecture Notes in Artificial Intelligence*, pages 245–256. (Cited on page 15.)

Nienhuys-Cheng, S.-H., van der Laag, P. R. J., and van der Torre, L. W. N. (1993). Constructing refinement operators by decomposing logical implication. In *AI∗IA*, volume 728 of *LNAI*, pages 178–189, Torino, Italy. Springer. (Cited on page 15.)

Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134. (Cited on page 14.)

Niiniluoto, I. (1987). *Truthlikeness*. Synthese Library. Springer. (Cited on page 24.)

Nikolov, A., d'Aquin, M., and Motta, E. (2012). Unsupervised learning of link discovery configuration. In *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, pages 119–133. (Cited on pages 2, 12, 39, 42, and 64.)

Nikolov, A., Uren, V. S., Motta, E., and Roeck, A. N. D. (2009). Overcoming schema heterogeneity between linked semantic repositories to improve coreference resolution. In *The Semantic Web, Fourth Asian Conference, ASWC 2009, Shanghai, China, December 6-9, 2009. Proceedings*, pages 332–346. (Cited on page 16.)

Nutanong, S., Jacox, E. H., and Samet, H. (2011). An incremental hausdorff distance calculation algorithm. *Proc. VLDB Endow.*, 4(8):506–517. (Cited on page 11.)

Oddie, G. (1978). Verisimilitude and distance in logical space. *The logic and epistemology of scientific change, Acta Philosophica Fennica*, 30(2-4):227–42. (Cited on pages 24 and 25.)

Pan, J.-S., Wang, H., Zhao, H., and Tang, L. (2015). Interaction artificial bee colony based load balance method in cloud computing. In *Genetic and Evolutionary Computing*, pages 49–57. Springer. (Cited on page 14.)

Patroumpas, K., Alexakis, M., Giannopoulos, G., and Athanasiou, S. (2014). Triplegeo: an etl tool for transforming geospatial data into rdf triples. In *EDBT/ICDT Workshops*, pages 275–278. (Cited on page 125.)

Pérez-Solà, C. and Herrera-Joancomartí, J. (2013). Improving relational classification using link prediction techniques. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part I*, pages 590–605. (Cited on page 13.)

Phuoc, D. L., Polleres, A., Hauswirth, M., Tummarello, G., and Morbidoni, C. (2009). Rapid prototyping of semantic mash-ups through semantic web pipes. In *WWW*, pages 581–590. (Cited on page 16.)

Pokharel, S., Sherif, M. A., and Lehmann, J. (2014). Ontology based data access and integration for improving the effectiveness of farming in Nepal. In *Proc. of the International Conference on Web Intelligence*. (Cited on page 119.)

Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL Query Language for RDF. W3C Recommendation. (Cited on page 135.)

Quinlan, S. (1994). Efficient distance computation between non-convex objects. In *In Proceedings of International Conference on Robotics and Automation*, pages 3324–3329. (Cited on page 12.)

Ramon, J. and Bruynooghe, M. (2001). A polynomial time computable metric between point sets. *Acta Informatica*, 37(10):765–780. (Cited on page 11.)

Saleem, M., Ngonga Ngomo, A., Parreira, J. X., Deus, H. F., and Hauswirth, M. (2013). Daw: Duplicate-aware federated query processing over the web of data. In *International Semantic Web Conference (1)*, pages 574–590. (Cited on page 1.)

Salman, A., Ahmad, I., and Al-Madani, S. (2002). Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, 26(8):363–371. (Cited on pages 3 and 51.)

Saykol, E., Gülesir, G., Güdükbay, U., and Ulusoy, Ö. (2002). Kimpa: A kinematics-based method for polygon approximation. In *Advances in Information Systems*, pages 186–194. Springer. (Cited on page 12.)

Schwarte, A., Haase, P., Hose, K., Schenkel, R., and Schmidt, M. (2011). Fedx: Optimization techniques for federated query processing on linked data. In *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, pages 601–616. (Cited on page 16.)

Shapiro, E. Y. (1991). Inductive inference of theories from facts. In Lassez, J. L. and Plotkin, G. D., editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 199–255. The MIT Press. (Cited on page 15.)

Sherif, M., Ngonga Ngomo, A.-C., and Lehmann, J. (2015). Automating RDF dataset transformation and enrichment. In *12th Extended Semantic Web Conference, Portoroz, Slovenia, 31st May - 4th June 2015*. Springer. (Cited on page 79.)

Sherif, M. A., Coelho, S., Usbeck, R., Hellmann, S., Lehmann, J., Brümmer, M., and Both, A. (2014). NIF4OGGD - NLP interchange format for open german governmental data. In *The 9th edition of the Language Resources and Evaluation Conference, 26-31 May, Reykjavik, Iceland*. (Cited on page 134.)

Sherif, M. A. and Ngonga Ngomo, A.-C. (2015a). An optimization approach for load balancing in parallel link discovery. In *SEMANTiCS 2015*. (Cited on page 51.)

Sherif, M. A. and Ngonga Ngomo, A.-C. (2015b). Semantic quran: A multilingual resource for natural-language processing. *Semantic Web Journal*, 6:339–345. (Cited on page 109.)

Sherif, M. A. and Ngonga Ngomo, A.-C. (2015c). A systematic survey of point set distance measures for link discovery. *Semantic Web Journal*. (Cited on page 18.)

Speck, R. and Ngonga Ngomo, A. (2014). Ensemble learning for named entity recognition. In *Proc. of ISWC (International Semantic Web Conference) 2014*, pages 519–534. (Cited on pages 4 and 79.)

Stadler, C., Lehmann, J., Höffner, K., and Auer, S. (2012). Linkedgeodata: A core for a web of spatial open data. *Semantic Web*, 3(4):333–354. (Cited on page 136.)

Stadler, C., Martin, M., and Auer, S. (2014). Exploring the Web of Spatial Data with Facete. In *Companion proceedings of 23rd International World Wide Web Conference (WWW)*, pages 175–178. (Cited on pages 122 and 132.)

Stadler, C., Unbehauen, J., Westphal, P., Sherif, M. A., and Lehmann, J. (2015). Simplified RDB2RDF mapping. In *Proceedings of the 8th Workshop on Linked Data on the Web (LDOW2015), Florence, Italy*. (Cited on page 124.)

Suchanek, F. M., Abiteboul, S., and Senellart, P. (2011). PARIS: probabilistic alignment of relations, instances, and schema. *PVLDB*, 5(3):157–168. (Cited on pages 12, 38, and 39.)

Sutskever, I., Salakhutdinov, R., and Tenenbaum, J. B. (2009). Modelling relational data using bayesian clustered tensor factorization. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pages 1821–1828. (Cited on page 13.)

Tănase, M., Veltkamp, R. C., and Haverkort, H. (2005). Multiple polyline to polygon matching. In *Algorithms and Computation*, pages 60–70. Springer. (Cited on page 11.)

Tang, M., Lee, M., and Kim, Y. J. (2009). Interactive hausdorff distance computation for general polygonal models. *ACM Trans. Graph.*, 28(3):74:1–74:9. (Cited on page 11.)

Tennison, J., Cyganiak, R., and Reynolds, D. (2012). The RDF Data Cube vocabulary. Technical report, W3C Working Draft 05 April. http://www.w3.org/TR/vocab-data-cube/. (Cited on page 98.)

Toussaint, G. T. and Bhattacharya, B. K. (1981). Optimal algorithms for computing the minimum distance between two finite planar sets. In *Pattern Recognition Letters*, pages 79–82. (Cited on page 24.)

Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A., Gerber, D., and Cimiano, P. (2012). Template-based question answering over RDF data. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pages 639–648. (Cited on page 117.)

van der Laag, P. R. J. and Nienhuys-Cheng, S.-H. (1994). Existence and nonexistence of complete refinement operators. In Bergadano, F. and Raedt, L. D., editors, *ECML*, volume 784 of *Lecture Notes in Artificial Intelligence*, pages 307–322. Springer-Verlag. (Cited on page 15.)

Volz, J., Bizer, C., Gaedke, M., and Kobilarov, G. (2009a). Discovering and maintaining links on the web of data. In *ISWC*, pages 650–665. (Cited on page 8.)

Volz, J., Bizer, C., Gaedke, M., and Kobilarov, G. (2009b). Discovering and Maintaining Links on the Web of Data. In Bernstein, A., Karger, D., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., and Thirunarayan, K., editors, *ISWC 2009, Proceedings of the 8th International Semantic Web Conference, Chantilly, VA, USA, October 25-29, 2009*, volume 5823, pages 650–665, Berlin, Heidelberg. Springer-Verlag. (Cited on page 100.)

Yan, D., Cheng, J., Lu, Y., and Ng, W. (2015). Effective techniques for message reduction and load balancing in distributed graph computation. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 1307–1317, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee. (Cited on page 14.)

Zaveri, A., Kontokostas, D., Sherif, M. A., Bühmann, L., Morsey, M., Auer, S., and Lehmann, J. (2013a). User-driven quality evaluation of DBpedia. In *To appear in Proceedings of 9th International Conference on Semantic Systems, I-SEMANTICS '13, Graz, Austria, September 4-6, 2013*, pages 97–104. ACM.

Zaveri, A., Lehmann, J., Auer, S., Hassan, M. M., Sherif, M. A., and Martin, M. (2013b). Publishing and interlinking the global health observatory dataset. *Semantic Web Journal*, Special Call for Linked Dataset descriptions(3):315–322. (Cited on page 96.)

Zaveri, A., Pietrobon, R., Auer, S., Lehmann, J., Martin, M., and Ermilov, T. (2011). ReDD-Observatory: Using the web of data for evaluating the research-disease disparity. In Hübner, J. F., Petit, J.-M., and Suzuki, E., editors, *Proceedings of the 2011 IEEE/WIC/ACM*

International Joint Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2011, Campus Scientifique de la Doua, Lyon, France, August 22-27, 2011, volume 1, pages 178–185. IEEE Computer Society. (Cited on page 103.)

Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., and Auer, S. (2015). Quality assessment for linked data: A survey. *Semantic Web Journal*. (Cited on page 2.)

Zhong, W.-l., Zhang, J., and Chen, W.-n. (2007). A novel discrete particle swarm optimization to solve traveling salesman problem. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 3283–3287. IEEE. (Cited on page 55.)

Zhou, K., Gui-Rong, X., Yang, Q., and Yu, Y. (2010). Learning with positive and unlabeled examples using topic-sensitive plsa. *Knowledge and Data Engineering, IEEE Transactions on*, 22(1):46–58. (Cited on page 14.)

DECLARATION

This thesis is a presentation of my original research work. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions.

Universität Leipzig, Augustusplatz 10, 04109, Leipzig,

Mohamed Ahmed Mohamed Sherif