Lightweight Information Integration through Partial Mapping and Query Reformulation

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades eines

Doktors der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation
von

Ing. Inf. Rodolfo Stecher

geboren am 14.06.1972, in Asunción, Paraguay

2010

Referent: Prof. Dr. techn. Wolfgang Nejdl
Korreferent: Prof. Dr. Wolf-Tilo Balke

Tag der Promotion: 04. Juni 2010

Abstract

The growing amount of structured information becoming available, fostered by the advent and development of e.g. the Semantic Web and the Web 2.0 approaches, raises the need for (semi-)automatic, flexible and adaptable integration solutions. The effort invested into this partially manually created content can be leveraged by re-use and integration, so that additional communities of users can take advantage of heterogeneous content created by or for a specific community. The implicit inclusion of semantics (by using ontologies) in the storage of such contents or their automatic discovery after creation, enables such an integration scenario. This work deals with the flexible integration of heterogeneous sources of information stored using Semantic Web standards (e.g. RDF/OWL) in a pay-as-you-go fashion (i.e. on a best effort basis based on already available information and improving integration over time). It tackles three important integration aspects: 1) The computation of initial mappings between the different content structures, expressed using ontologies, by employing a combination of lexical-, structure-, and logic-based approaches; 2) The on-the-fly reformulation of user queries, so that they can be executed on sources which are structured differently, by employing the available mappings in combination with wildcard-based relaxation rules for unknown mappings, together with a strategy for improving and learning mappings; and 3) The ranking of the results based on the confidence that a reformulated query will answer exactly what was requested in the original query. This thesis presents therefore an approach for lightweight information integration of structured sources of data. Comprehensive evaluations have been performed for the steps presented above employing real world data sets to show their feasibility and applicability.

Keywords: Information Integration, Query Reformulation, Query Ranking.

Zusammenfassung

Die wachsende Anzahl im Web verfügbarer, strukturierter, heterogener Inhalte, die von Individuen oder Gruppen erstellt werden, erfordert fexible und adaptive Datenintegrationslösungen, um eine breite Wiederverwendung dieser Inhalte zu erleichtern. Die vorliegende Arbeit stellt einen Ansatz zur fexiblen Integration heterogener strukturierte Inhalte vor. Bei der Art des gewählten Ansatzes ("Pay as you go") wird dabei schrittweise die Integrationsqualität verbessert, da neue Erkenntnisse zu Inhaltsbeziehungen dynamisch in den Integrationsprozess eingebaut werden. In dieser Arbeit werden Inhalte zugrunde gelegt, welche durch Ontologien beschrieben sind. Der entwickelte Integrationsansatz konzentriert sich auf drei Herausforderungen: 1) Die Berechnung von Abbildungen zwischen Ontologien, welche die unterschiedlichen Inhaltsstrukturen beschreiben, werden anhand einer Kombination von lexikalischen, strukturellen und logischen Methoden berechnet. 2) Eine dynamische Umformung von Benutzeranfragen wird unter Berücksichtigung der bekannten Abbildungen zwischen den Ontologien und mit Hilfe von platzhalterbasierten Regeln durchgeführt. Diese Umformung wird mit einer Strategie zur dynamischen Verbesserung der existierenden Abbildungen kombiniert. 3) Die Anfrageergebnisse werden auf der Basis der Exaktheit der Umformulierung sortiert (Ranking). Diese Sortierung verwendet die Wahrscheinlichkeit, dass eine umgeänderte Anfrage (trotzdem) korrekte Antworten im Sinne der Ursprungsanfrage liefert. Diese Arbeit präsentiert somit einen fexiblen und leichtgewichtigen Ansatz zur on-the-fly Integration von heterogen strukturierten Inhalten. Evaluationen, die für die oben vorgestellte Schritte mit realen Daten durchgeführt worden sind, zeigen die Realisierbarkeit und Anwendbarkeit des Ansatzes.

Schlagwörter: Informationsintegration, Anfragenübersetzung, Anfragenrangordnung.

Acknowledgements

I would like to thank my mentor Claudia Niederée for their help, support and guidance during the development of my research, as well as my supervisor Wolfgang Nejdl for guiding me with valuable comments and showing me the way to excellence in research. I also thank Paolo Bouquet for their help, fruitful discussions and support in the first stages of my thesis, also combined with tasty dinners and skiing activities on the Dolomite mountains. To the masters student I supervised, Karoline Kirmse, who helped design and implement the first version of the matching module many thanks for her efforts and dedication during that time. Many thanks also to all colleagues with which I had the pleasure to interact, writing papers, discussing interesting ideas or just spending some good times, and to all the friends who gave me strength and helped me staying focused. Finally, the research presented here has been partially funded by the European projects Vikef (147) and Nepomuk (102).

Contents

List of Figures **xiii**

List of Tables **xv**

1 Introduction and Motivation **1**

1.1 Social Knowledge Spaces 2

1.2 Pay-As-You-Go Integration 3

1.3 Lightweight Information Integration Approach Overview 4

1.4 Structure of the Thesis 7

2 Framing the Problem **9**

2.1 Towards Pay-as-you-go Information Integration 10

2.2 Information Integration Strategies 11

2.2.1 General Integration Strategies 12

2.2.2 Common Steps to Information Integration 13

2.3 Dimensions of Information Integration 14

2.4 Problem Statement 16

2.4.1 General Terminology and Definitions 16

2.4.2 The Problem 19

2.4.3 Addressed Challenges 19

2.5 Lightweight Information Integration Approach Overview 20

2.5.1 Main Contributions 22

3 Related Work **25**

3.1 Information Integration Approaches 25

3.1.1 Pay-As-You-Go Integration 25

3.1.2 Data Warehouses 27
3.1.3 Distributed Information Retrieval 28
3.1.4 Ontology-based Integration 31
3.2 General Information Integration Steps 32
3.2.1 Resource Description 32
3.2.2 Schema Mapping 33
3.2.3 User Query Formulation 33
3.2.4 Query Reformulation 33
3.2.5 Resource Selection 34
3.2.6 Results Handling 34

4 Mapping Ontologies **35**
4.1 Related Work on Mappings 35
4.1.1 Schema Mapping 36
4.1.1.1 Local as View vs. Global as View 38
4.1.1.2 Schema vs. ontology mapping 39
4.1.2 Ontology Mapping 40
4.1.3 Discussions 46
4.2 Mapping Approach Overview 46
4.2.1 Definitions 47
4.3 Finding the Meaning of Elements 49
4.3.1 Analysis of the Labels 50
4.3.2 Computation of the Context 52
4.3.3 Disambiguating Word Senses 53
4.3.3.1 Determining the distance weight 54
4.3.3.2 Determining the relation weight 54
4.3.3.3 Determining the type weight 56
4.3.3.4 Computing the word sense score 57
4.3.4 Representing the Meaning with a Logical Formula 59
4.3.4.1 Meaning of words as a logical formula 60
4.3.4.2 Meaning of elements as a logical formula 60
4.3.4.3 Example 62
4.3.5 Score of Element Senses 62

4.4 Find Matching Elements in Different Ontologies 65
4.4.1 Determining the Candidate Sets 66
4.4.2 Determining the Semantic Relation 66
4.4.3 Computing the Element Similarity 68
4.4.3.1 Comparing two elements 68
4.4.3.2 Comparing the modeling perspectives 70
4.4.3.3 Element similarity measure 72
4.4.4 Approximating the Semantic Relation Using Senses 73
4.4.5 Structure-based Detection of the Semantic Relation 74
4.4.6 Element Relations Computation 75
4.5 Experiments and Evaluations 75
4.5.1 Testbed 76
4.5.2 Ontology Alignment Evaluation Initiative 77
4.5.3 Evaluation parameters 79
4.5.4 Test Results 101 to 104 81
4.5.5 Test Results 201 to 210 81
4.5.6 Test Results 221-247 82
4.5.7 Test Results 301-304 82
4.5.8 OAEI Results Overview 84
4.5.9 Hands-on Empirical Experience 86
4.5.10 Experiment Conclusions 87

5 Reformulating Queries to Query Different Sources 89
5.1 Related Work on Query Reformulation 89
5.1.1 Query Rewriting 90
5.1.2 Query Relaxation 93
5.1.3 Discussions 94
5.2 Query Reformulation Approach 94
5.3 Query Rewriting and Substitution Rules 96
5.3.1 Required Definitions 97
5.3.2 Mapping-based Substitution Rules 98
5.3.3 Ontology-based Substitution Rules 99
5.3.4 Wildcards Substitution Rules 99

5.3.5 Combining Substitution Rules . . . 102
5.3.6 Query Reformulation Process . . . 102
5.3.7 Query Reformulation Example . . . 103
5.4 Refining and Learning Mappings . . . 104
5.5 Experiment and Evaluation . . . 105
5.5.1 Testbed: Ontologies and Data . . . 105
5.5.2 Experiment Setup . . . 111
5.5.3 Evaluation Results . . . 113

6 Ranking Reformulated Queries 115
6.1 Related Work on Ranking . . . 115
6.1.1 Discussions . . . 117
6.2 Query Ranking Approach . . . 117
6.2.1 Punishment Derived from Mappings . . . 118
6.2.2 Punishment Derived from Relaxation . . . 118
6.2.3 Ranking Function . . . 120
6.2.4 Computing Ranking Example . . . 121
6.3 Evaluation . . . 121
6.3.1 Evaluation Setting . . . 122
6.3.1.1 Information Sources . . . 122
6.3.1.2 Initial Mappings . . . 123
6.3.1.3 Queries . . . 123
6.3.1.4 Ground Truth . . . 123
6.3.2 Experiments . . . 123
6.3.2.1 Parameter Finding . . . 124
6.3.2.2 Evaluating Precision at Top-K . . . 125
6.3.2.3 Performance . . . 126

7 Conclusions and Future Directions 127
7.1 Summary . . . 127
7.2 Contributions . . . 128
7.3 Future Directions . . . 130

Appendix A: Short Curriculum Vitae 131

References **133**

List of Figures

1.1 Lightweight information integration approach overview showing in the upper part its runtime steps and in the lower part its preprocessing step. 5

4.1 Classification of matching techniques 42
4.2 Example user ontology 47
4.3 Part of the ontology graph 49
4.4 Context for the element Research Author 52
4.5 Example of a head-modifier tree 62
4.6 Structure-based relation detection 75
4.7 User ontology example with selected concepts 77
4.8 Part of an ontology having matching elements 78
4.9 Modeling perspective similarity of results obtained from the baseline Swoogle and the approach presented in this chapter. 87

5.1 Example Ontologies 103
5.2 Example Queries 104
5.3 Excerpt of the user's ontology O^u used for specifying Q^u 106
5.4 Simple schema for Calendar entries 106
5.5 Simple schema for Emails 107
5.6 Simple ontology for Nepomuk L3S internal Meeting Minutes 107
5.7 Simple ontology for Nepomuk L3S internal Meeting Minutes: Task items 108
5.8 Simple ontology for Nepomuk L3S internal Meeting Minutes: Information items 108
5.9 Simple ontology for Nepomuk L3S internal Meeting Minutes: Upcoming Events 109

5.10 Simple ontology for Publications 109
5.11 Simple ontology for TelCo Minutes on the Wiki 110
5.12 Precision evolution for every strategy 114
5.13 Recall evolution for every strategy 114

6.1 Finding α 125
6.2 Overall Precision 126

List of Tables

4.1 Abbreviations for word syntactic forms 51
4.2 Example of lexical information of selected labels of example Figure 4.4 . 53
4.3 Example of senses for some words in the context 53
4.4 Relation Strength 55
4.5 Senses of nouns in the glosses 57
4.6 Example weights 58
4.7 Influence example 58
4.8 Example word sense scores 59
4.9 Logic concept formula 60
4.10 Example element sense score computation 64
4.11 Parameter settings for the evaluation 80
4.12 Results of tests 101 to 104 81
4.13 Results of tests 201 to 210 82
4.14 Results of tests 221-247 83
4.15 Results of tests 301-304 84
4.16 Comparison with results of OAEI 2005 - part 1 84
4.17 Comparison with results of OAEI 2005 - part 2 84
4.18 Comparison with results of OAEI 2009 - part 1 85
4.19 Comparison with results of OAEI 2009 - part 2 85
4.20 Comparison with results of OAEI 2009 - part 3 85

5.1 Example Mappings 104
5.2 Mean number of file types per desktop 111
5.3 Quality measures for each strategy 113

6.1 Information Sources Overview 122
6.2 Example Queries 124
6.3 Precision at Top-K 126

1

Introduction and Motivation

The amount of available information as well as of sources of information has grown even faster in recent years (23), due to higher community involvement and the extended usage of new technologies and paradigms such as provided by Web 2.0 and the Semantic Web. Many of the created content collections constitute major assets due to the joint labor and collaborative intelligence invested in their creation and refinement. This gives rise to a growing interest of re-using already existing information sources leveraging the investment in the original content production. This re-use comes in different flavors, such as publishing and interlinking information sources as in the case of the Linked Data Cloud or the import of information collections as in the case of Freebase and Wikipedia. In this work, the focus is on supporting the re-use of existing information sources by enabling lightweight federation via lightweight integration of heterogeneous information sources (i.e. in a loose and semi-automatic integration vs. in a human-driven and manual integration).

This way of re-use respects the autonomy of the underlying sources and does not interfere with the community driven dynamics of these sources, which is vital for these "Social Knowledge Spaces". In addition, the interconnection of sources produced by different communities through lightweight federation, provides integrated access to these sources. Considering, for example, the area of Personal Information Management (PIM), a federation of the own desktop content with content managed (on behalf of the user) in Social Web applications such as YouTube (157) and Flickr (53) as well as controlled federation with the content of co-workers as it is, for example supported in the Nepomuk project (102), clearly can provide improved service to the individual desktop user.

However, the characteristics of today's information sources coming from Social and Semantic Web applications also impose new challenges for information integration: The sources typically exhibit only a loose schema binding, they contain noisy data of mixed quality which is quickly evolving and changing. In addition, due to the high dynamics in the information spaces, upfront integration efforts for the integration of new resources should be small, easing the federation of new sources.

A promising way of achieving such integration is denoted as pay-as-you-go. In pay-as-you-to integration, integrated information is made available with small initial integration efforts, giving the best results possible at each point in time, and increasing the quality of the integration over time. This work presents a lightweight and flexible pay-as-you-go information integration approach which is capable to deal with the heterogeneity and dynamics of today's information sources, in order to provide flexible search access without incurring in high integration costs.

1.1 Social Knowledge Spaces

Due to the success of the Social Web and the maturity of Web technologies, a growing number of community-based systems is becoming available offering different types of information services, such as Freebase (55), Wikipedia (152), The Huffington Post (115), DBPedia (34), UMBEL (144) or SAP Community Network (103) just to name a few. Such systems rely on user-generated and community-moderated content leveraging on the joint knowledge and collaborative intelligence of their target community members. The respective collaboratively created content spaces will be referred as *Social Knowledge Spaces* (SKS).

Such Social Knowledge Spaces exhibit the following properties:

- *Community defined structure:* The content stored in an SKS usually does not carry a predefined structure, but, the structures which appear in the knowledge spaces are strongly defined by the user groups which add, but also consume its content (similar to the concept of dataspaces (54)). This very important aspect makes different sources contain perhaps similar information, but considered from a different perspective and represented differently.

- *Heterogeneity and loose schema binding:* Many SKS are freely created or edited, i.e. without a fix schema, by many different people or groups of people coming from very heterogeneous backgrounds. This freedom and flexibility in knowledge articulation is one of the strengths of such knowledge spaces: Knowledge can be articulated in individual ways reflecting individual understandings of the world. This results in a low entry barrier for adding new knowledge, since no foreign schema has to be learned or adopted. The flexibility enabled by the accepted heterogeneity leads to a loose schema binding of the represented information and even to the co-existence of multiple schemata in the same repository.
- *High dynamics in content and structure:* The observable change in people's culture and behavior towards using web platforms for e.g. socializing by means of social applications, exchanging opinions in blogs, etc., make SKS become more popular and fast growing. In addition, these communities also undergo changes in their interests and focus, and in this way also the structure of the contributed information, but also the information within the SKS, which is accessed, changes.

1.2 Pay-As-You-Go Integration

Integrating information distributed across heterogeneous sources, while respecting the autonomy of the individual sources, poses several problems, which imply a set of steps for the information integration process (22). This includes 1) the computation of mappings between the descriptions of the sources to be integrated, 2) the reformulation of queries formulated for one source, so that it can be executed on the other sources, 3) the selection of the sources which will give the best or most complete answers given a query, and 4) the integration of the results obtained from different sources.

Considering the time when individual steps in the integration process are performed there are different options: it can be done in advance - before answering queries-, it can be done on the fly as part of the query processing, or, the integration tasks can be split between pre-processing and query processing time. It is this last option which is followed in the work presented in this thesis such that some steps are pre-computed but most steps are performed on the fly following a pay-as-you-go-like strategy.

Pay-as-you-go approaches and solutions (67, 76, 120, 122) aim at giving at any time the best possible results given what is already known, in a "best-result" strategy, but also consciously accepting that the integration is not fully completed or perfectly performed. In such a setting, the system accumulates evidences over time, so that 1) the integration quality is increased stepwise during system usage, and 2) the system learns from user actions or from evidences collected from the underlying data or systems, so that future user interactions give more satisfying results. Such approaches have the advantages that they can already give satisfying results even with incomplete knowledge. In addition, they have the advantage of being especially adequate for the integration of autonomous sources, exactly because they can deal with imperfect integration information as it results when the underlying sources change independently. The sources keep in this way the full flexibility and freedom to evolve based on the needs of the communities creating or managing them.

As a further characterization of the approach presented in this thesis: It deals with the lightweight integration of information sources in the area of Semantic Information Integration (38). Semantic Information Integration denotes the integration of structured and semi-structured sources where the semantics corresponding to different elements appearing in the schemata descriptions of the sources need to be resolved first, before being able to relate and integrate the data stored in those heterogeneous sources. The semantic aspect implies that the intent expressed in the schemata needs to be taken into account in order to perform the integration by allowing to bridge syntactical differences across sources or data.

1.3 Lightweight Information Integration Approach Overview

The approach in this thesis focuses on the considerable subset of the growing number of heterogeneous information sources available in existing SKS which exploit the progress that has been made in Web technology (semantic Web, Web services, etc.): they are or can be exposed in a Semantic Web standard as RDF or RDFS (149), and their structuring or schema can be expressed with more or less simple ontologies using e.g. OWL (Ontology Web Language). Furthermore, it relies on Semantic Web query languages which have been developed (e.g. SPARQL (151)), allowing to express queries in a SQL-like fashion, by expressing relations between information items in triples. These technological developments pave the ground for semantic information integration approaches by providing standards to represent the sources to be integrated using common formalisms, as well as for querying them. Using such standardized semantic technologies also paves the ground for automatic processing and machine consumption of available content.

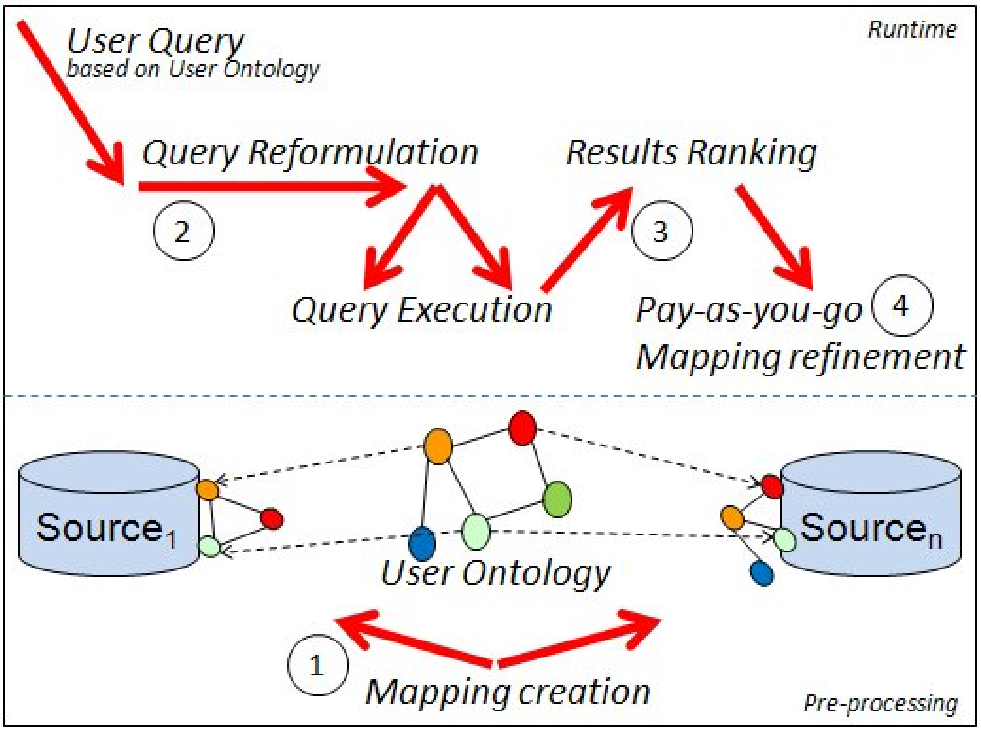


Figure 1.1: Lightweight information integration approach overview showing in the upper part its runtime steps and in the lower part its preprocessing step.

Based on the challenges of the discussed steps of the information integration process (see also (22)), the approach developed in this thesis is sketched below leaving its detailed description to corresponding dedicated chapters. In addition Figure 1.1 illustrates the major steps of the approach, presenting in the lower part of the figure the pre-processing step and in its upper part the runtime steps:

- In order to integrate heterogeneous information sources, first a mapping between corresponding elements need to be created (step 1 in Figure 1.1). In this work an automatic approach for computing mappings between ontologies describing the different information sources and the ontology describing how the user or community of users see the structure of the SKS has been developed, which takes into account the *modeling perspective* of the analyzed ontologies (i.e. the context of the ontological elements to be mapped). This approach uses a combination of several techniques (Lexical, SAT-Based and Structure-based) to compute the mappings. For capturing the *modeling perspective* it includes the context of each element into the process of matching detection. The reported experiments show in which cases this combination of methods and the inclusion of the *modeling perspective* give competitive advantages (see Chapter 4 for details) compared to other existing approaches in the field.

- After the mappings have been computed, queries expressed based on a given schema (i.e. an ontology describing the view of users on an SKS) need to be rewritten in order to be executed on sources having a different schema (step 2 in Figure 1.1), and, since the computed mappings are partial, the problem of incomplete mapping specifications needs to be addressed. For this, a flexible approach to reformulating SPARQL queries, by rewriting and relaxing them based on the existing partial mappings and on special relaxation and rewriting rules was developed. In addition, feedback on the information retrieved through the relaxation is employed to improve or learn mappings between the elements used in the original query and elements appearing in the ontologies describing the sources, in a pay-as-you-go fashion (step 4 in Figure 1.1). Experimental results show that this query reformulation approach gives good precision and short learning times for new mappings, proving its feasibility. Several query reformulation strategies are evaluated, and the ones most suitable for reducing the cold start problem (if there is only a very small number of mappings available) are discussed in Chapter 5.

- Reformulated queries are created using mappings with different confidences of being correct, and employing different relaxation strategies and degrees of relaxation. These factors introduce for the reformulated query a likeliness of allowing the retrieval of results which were not intended in the original query. For dealing with these un-solicited (and possibly wrong) results, a punishment-based ranking approach for queries is introduced. This ranking is based on the degree of relaxation introduced and the mappings employed for the rewriting in the query reformulation process. The ranking, computed by considering how a query was reformulated, allows to rank the results coming from a given source (step 3 in

Figure 1.1) depending on the confidence to obtain right results out of it, since more relaxation increases the risk to add incorrect results. This is also the case when using mappings with less confidence in the query reformulation process. The experiments on real world data sources show that the proposed ranking helps in obtaining more precise results at the first positions in the result list (see Chapter 6).

Thus, the approach focuses on the steps presented in Figure 1.1.

1.4 Structure of the Thesis

For framing the presented approach, Chapter 2 presents strategies and dimensions of information integration and gives an overview of the Semantic Information Integration area - focusing on the aspects which are relevant to this thesis - followed by the formal problem statement and the overview of the approach presented in this work. Chapter 3 presents an overview of the related work giving a general description of the information integration area, pointing to detailed related work sections in respective chapters. In Chapter 4 the details and evaluations of the Ontology Mapping approach are described and discussed as well as the area-specific related work. Chapter 5 focuses on the description, evaluation and discussion of the approach for reformulating queries, so that they can be executed on different sources even if only partial mappings are available, including the improvement of the mappings based on user feedback, as well as on the specific related work for this research area. In Chapter 6 the description of the approach for the ranking of results obtained from reformulated queries as well as its detailed related work is presented, followed by its evaluation. Finally, Chapter 7 presents the conclusions of this thesis and ideas for future work in the area of lightweight information integration.

2

Framing the Problem

This chapter presents the problem addressed by this thesis and its challenges, the areas where this work is embedded in, and an overview of the proposed approach. The problem deals with pay-as-you-go-like information integration in the context of heterogeneous, semi-structured information sources. The discussed challenges are based on an analysis of the respective literature. They focus on the transition of the traditional information integration tasks towards a pay-as-you-go integration, triggered by the increased heterogeneity of the information and the growth of the amount of information sources available nowadays (23).

Section 2.1 presents the general problem of information integration, including its evolution to today's information integration scenarios and needs. Section 2.2 highlights the existing information integration strategies and present the common steps to information integration as known from the respective literature. Section 2.3 shows several dimensions to structure the different information integration approaches and highlights the ones most related to the work presented in this thesis. Section 2.4 starts with some required definitions, continuing with the formalized problem statement, and closing the section with the challenges which will be addressed throughout this work. Section 2.5 presents an overview of the approach followed to solve the presented problem, as well as a highlight of the main contributions.

2.1 Towards Pay-as-you-go Information Integration

Information Integration is known in general as the uniform querying and later merging of data and information from disparate sources with differing conceptual or structural representations (23, 60, 158, 163). Semantic Information Integration (38) denotes the integration of structured and semi-structured sources where the semantics corresponding to different elements appearing in the schema descriptions of the sources need to be related first, before being able to integrate the data stored in those heterogeneous sources.

In recent years, the task of information integration clearly has become more dynamic due to the availability of a vastly growing number of information sources (23). Generally, the SKS's (introduced in Section 1.1) do not carry a predefined content structure, but, the structures which appear in the knowledge spaces are strongly influenced by the user groups which add, but also consume its contents (similar to the concept of dataspaces (54)). In this way, the content and its structure continuously evolve, adapting quickly to cultural changes and information needs. The advent of the Web 2.0 and the high Internet availability as well as the observable change in people's culture and behavior towards using web platforms for e.g. socializing by means of social applications, exchanging opinions in blogs, etc., make SKS become more popular. Many SKS are freely created or edited, i.e. without a fix schema, by many different people or groups of people coming from very heterogeneous backgrounds. This freedom and flexibility in knowledge articulation is one of the strengths of such knowledge spaces: Knowledge can be articulated in individual ways reflecting individual understandings of the world. This results in a low entry barrier for adding new knowledge, since no foreign schema has to be learned or adopted. Furthermore, the flexibility enabled by the accepted heterogeneity helps in representing various aspects and viewpoints on information about the described things.

This growth of available heterogeneous sources imposes new and additional challenges on semantic information integration as it is required for integrated information search and access (23). Today's users do not want to have to configure the system each time a new information source becomes available in an SKS, and also cannot do so because of this high growth rate of available sources. Anyway, they expect to be able to use information that becomes available as soon as possible, obtaining a competitive advantage out of it and gaining in this way efficiency in their everyday tasks. This leads to new requirements on semantic information integration systems such as providing best-effort results (54) at a given time, and having a trade-off between information integration quality and timely accessibility of information - paving the ground for pay as you go integration.

At the same time, a considerable subset of the presented SKS exploit the progress made in Semantic Web technologies as described in Section 1.3. The usage of such logic-based standards to describe the information sources allows to e.g. inference on them using reasoners, giving in this way more possibilities for the tasks at hand.

2.2 Information Integration Strategies

The semantic integration of information from different sources has been studied for a long time (for a survey see (38)), e.g. in the relational (e.g. (93)) and XML world, and lately in the context of the Semantic Web (e.g. (85)). A variety of approaches to tackle the semantic integration problem have been developed. Many of those approaches rely on the existence of explicit mappings between the sources to be integrated, so that the system knows which types of content corresponds to which other (see e.g. (38, 93)). One problem is that the computation of mappings between sources remains to be a complex and expensive task which typically requires manual intervention and has to be repeated for every new source becoming available as well as - at most partially - every time a source changes. Thus, the high dynamics of many of today's information integration settings makes the costs of computing and maintaining complete mappings for all information sources a limiting factor.

Different options exist for the integration of data sources, some of them assume to have one global schema which is the only one which can be used to post queries, and others where queries can be posted using the schema of any of the participating information systems. In the first case, mappings between this global schema and all participating sources need to be computed, so that queries built using this global schema can be rewritten to queries over the different sources. In the second case, mappings from each of the participating systems to other participating systems are required, which increases the cost of computing mappings, but on the other side gives more flexibility because any of the systems can request data from any of the other systems (e.g. as e.g. in peer to peer systems (33)). Several other examples for both cases are presented in the related work Section 3.1.3. In this thesis an approach relying in a global schema is assumed, as it will be described later in this chapter in Section 2.4.

2.2.1 General Integration Strategies

Based on how the integration is performed, different strategies for information integration can be identified:

- Centralized pre-request: Approaches which integrate and store all data before any information request can be answered are found e.g. in centralized data warehousing, where the data is copied to one database to be centrally processed. The centralized data warehouse approaches replicate the information at a central place, thus heavyweight storages as well as infrastructures for computations on such large information repositories (e.g. for data mining) are required. Since data is copied to a central storage, updates on the sources are not reflected on the warehouse until the data has been updated. This leads to the fact that it cannot be guaranteed to have e.g. data warehouse reports based on the latest information from the contributing sources at all times. Data warehouse as well as data mining approaches require complete mappings to be defined between the sources and the central system, which was already mentioned to have a possibly high cost and intrinsic difficulties. On the other hand, having all information in one database gives access flexibility, and makes the access to the data independent of possible single failures in different data sources, giving a consistent view on the data at each point in time. At web scale these approaches are generally not feasible, due to the amount of information to be copied, and the rate of change of today's information sources.
- Correct Per-request: These approaches, aiming at giving a correct answer or no answer at all, maintain the data at the sources and perform the integration at query or access time, avoiding in this case the need to centrally store all information and thus also the related problems discussed above. On the other side, they rely on the availability of data sources and the stability of their structure. Structural changes in any source's schema require that the integration system is reconfigured in order to offer the expected functionality. Again, complete mappings

need to be computed and recomputed whenever a source is added or changed, incurring also in the high costs for computing those complete mappings. Queries in such systems need to be reformulated, to reflect the different schemata of the sources in order to be executed on them. In such systems, queries are usually reformulated by considering the mapping information available, what makes the existence and correctness of the mappings a crucial point in such constellations.

- Best-effort per-request: The approaches mentioned before aim at answering only right and safe answers to the user, and therefore answers to a query exist or do not exist, without any intermediate scales. There are other semantic information integration approaches aiming at answering the best results possible at a given time, e.g. (67) and (160). These approaches do not give only right results, although precision is a goal. They return the best results possible at the time and state of knowledge - e.g. with respect to mappings - when the query is processed, improving if possible results over time. In this way a flexible integration is accomplished, automating as much as possible the integration process. An advantage of these approaches is that the information is immediately available once the decision to integrate a source has been made, with the disadvantage that the results of queries might contain incomplete or even incorrect results, depending on the system strategy for dealing with partially integrated sources (e.g. relaxation of the queries for unknown mappings vs. ignoring things which are still not explicitly connected or integrated). Usually such systems learn over time new connections between the sources (e.g. by statistical analysis or by explicit feedback), and improve herewith the quality of the results. The approach presented in this work is based on such idea and aims also to give the best possible answers to a given query at a certain time.

2.2.2 Common Steps to Information Integration

In spite of the differences between the information integration approaches, it is possible to identify some common steps required in some form in most of the integration approaches that do not perform information integration as a pure pre-processing task. The terminology of distributed information retrieval will be employed, which is one area of information integration, for describing these steps and challenges (22) which range from finding the right sources to how to present the results to the user. Common steps in information integration are:

- Resource description: The contents and the structure of each information source have to be described;
- Mapping computation: The mappings between elements in the different resources to be employed need to be computed;
- User query formulation: The query to be executed needs to be specified by the user following her/his view on the domain;
- Query reformulation: The user formulated query needs to be translated to the different formats supported by the available data sources;
- Resource selection: Depending on the query, the information sources possibly contributing to the results of this query have to be selected;
- Query execution: The query needs to be reformulated to be understandable by each of the selected information sources and executed; and
- Results handling: The results obtained from the queried sources have to be collected and presented to the user in the best possible way, aiming at providing the best possible user experience.

A variety of solutions employing these steps exist, usually focusing on different subsets of them. Those which are relevant for this thesis will be discussed in Chapter 3.

2.3 Dimensions of Information Integration

Information integration strategies can be structured along several dimensions, including: 1) the architectural view of the system; 2) the content and functionality of the different information sources; 3) the intended use of the integrated system (e.g. only read vs. write access), etc. (163). The two dimensions which have the highest impact on the integration approach presented here - type of source and time of integration - are discussed in some more detail:

- Type of sources: The type of involved sources is important for selecting an approach to integrate information. These types can be structured, semi- or unstructured sources of information. Considering further the structured sources, they can contain data described by the same schema across all sources, or, have heterogeneous representations of the stored data. Structured sources can also use different models of representations to store the data, as the traditional relational model, the object oriented model, or the ones arising with the Semantic Web. One representation extensively used in the Semantic Web are sources employing triple models (e.g. for storing RDF). Different underlying representations imply also different possibilities to query the single systems in the SKS. Structured and semi-structured sources require the computation of mappings between the descriptions of the sources, which can be expressed e.g. in XML by leaving relations implicitly defined, or by using Semantic Web languages such as OWL, where the relations can be typed and stated explicitly. In cases where the sources are unstructured, e.g. containing only text, one way to find a source for a given query is e.g. by a bag of words or term vectors representation of the contents of the source, while the more structure a source has, the more schema information can be employed in query answering.

- Time of the integration: The time of the integration is determined by when the integration happens. Some approaches perform the integration in advance or before any request for integrated information is posted to the system, as e.g. in data warehouses, and others perform it on-the-fly, when a request arrives as e.g. in the pay-as-you-go approaches. Each of the approaches has its advantages and disadvantages. In the pre-request approaches, all details of the integration are solved, and in some cases the information is copied to a central storage. In cases where the sources are not under the control of the integrating organization, this can be of advantage since a certain quality of service can be assured. A drawback is the cost of checking all sources for updates on the information, in order to update the central storage. Depending on the interval of updates it might be the case that the central storage delivers information which is not not up-to-date. The approaches which perform an on-the-fly integration might face some problems since all computations to enable the integration need to be performed very quickly in order not to affect the query answering time tom much. Furthermore, the sources might become unavailable in an uncontrolled or unscheduled way, hindering the delivery of all possible results. Some of the advantages of such approaches is the up-to-date information available and the possibility to integrate a new source or to update the configuration of an existing source on the fly.

A more extensive discussion about these and other dimensions for organizing the existing information integration approaches and systems will be presented in Chapter 3.

2.4 Problem Statement

This section describes the problem addressed in this thesis in a formalized way. For expressing the mentioned problem, first some definitions will be presented.

2.4.1 General Terminology and Definitions

Section 2.1 introduced the concept of an SKS, now its definition is presented in Definition 1.

Definition 1 *Social Knowledge Spaces (SKS) are distributed, heterogeneously structured collections of content across different independent web systems, produced, edited and shared among one or more users or communities.*

The structure of the data stored in each of the sources can be described by different formalisms, such as XML Schemata, ER diagrams, or Ontologies. In this thesis the information sources are assumed to be described by simple ontologies (see Definition 3). Ontologies are used to describe, interrelate and interconnect heterogeneous sources in a way that can be interpreted by humans as well as by machines. Ontologies are a "Formal specification of a shared conceptualization" (14). Studer el al. explain this definition in this way: "Conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group" (138). This work targets mainly application ontologies (146) and not so much other more general ontologies. The ontology definition presented in Definition 2 is based on the definition presented in (19), with some modifications as only allowing binary relations between concepts, including the *direct subclass* relation, and including the functions to obtain the domain and range of a relation, in order to suit the requirements of the work presented in this thesis. There exist several different definitions of ontologies in the literature (e.g. (136), (91), (66), (45)) each of them considering different aspects required for solving the respective problems at hand. But, all of them describe an ontology as a set of hierarchically organized concepts with possibly a variety of relations between them, representing a theory in some logic (25).

Definition 2 *An ontology is defined as a set of concepts, binary relations over the concepts, and hierarchy over concepts and relations. Formally, an ontology* O *is a tuple* $O = \langle C, R, H^C, H^R \rangle$ *where* C *is a set of concepts,* R *is a set of relations and* H^C*,* H^R *are hierarchies over concepts and relations defining a partial ordering over concepts and properties respectively. This partial ordering states which concept is a subclass of another concept (similarly for properties). This can be expressed as* $c_1 \leq_{H^C} c_2$*, stating that* c_1 *is the subclass of* c_2 *under the partial ordering* H^C *where* $c_1, c_2 \in C$ *(analogously for properties).* H^C *will be left out from* $\leq_{H^C}$ *whenever there is no possibility of confusion. If* $c_1 \leq c_2$ *and there is no* c_3*,* $c_1 \neq c_2 \neq c_3$ *such that* $c_1 \leq c_3 \leq c_2$*, then it will be said that* c_1 *is a direct subclass of* c_2 *and will be denoted as* $c_1 \prec c_2$ *(analogously for properties). A fixed alphabet* A *contains all concept names* (A_C) *and relation names* (A_R)*, such that* $A = A_C \cup A_R$ *and* $A_C \cap A_R = \emptyset$*. There is a function mapping every concept name or label denoted as* cn *in* A_C *to a concept in* C *and every relation label or name denoted as* pn *in* A_R *to a relation in* R*. Concepts* c *in* C *are over a domain* Δ *and relation names are mapped to binary relations* r *over* $\Delta \times \Delta$*.* c *and* r *will be overloaded to denote concept and relation names in* A *and the associated concept and relation in* O*. The function* $dom(r)$ *returns the domain* Δ *of a binary relation, while the function* $range(r)$ *returns its range* Δ*. In cases where concepts or properties of the ontology are meant without distinction, they will be denoted as ontology elements* e*,* $e \in C \cup R$*. In conformance with the terminology used in RDF (149) the relations will be also named properties.*

An SKS as presented in Definition 1 is composed of several sources, which will be denoted as presented in Definition 3. The schema information available in a SKS is influenced by the characteristics of the creation of an SKS. Either ad-hoc names are used for properties and concepts, or common schemata - although not necessarily formally described - get established by community processes (95) within a community. This happens, for example, if knowledge already existing in the SKS is used as a template for the articulation of knowledge of similar type. Usually, every user or community tends to use a specific, more or less stable vocabulary for naming specific aspects or for relating facts about a thing. The vocabulary used by each of these groups usually depends on the culture, education and background. It is also strongly influenced by the respective community. In this work it is assumed that the structure of the data stored at each information source in an SKS is described by a simple ontology as presented in Definition 3.

Definition 3 *Each Information Source storing but also providing information in a SKS is denoted as S_i. An S_i is described by a more or less formalized ontology O^{S_i}, which presents in form of concepts and properties the structure of the information stored at S_i.*

Independently of the ontologies O^{S_i} describing the sources S_i available in an SKS, the users have an understanding of the complete SKS as presented above - a general user view or ontology - as presented in Definition 4.

Definition 4 *The ontology of the user is denoted as O^u, it represents how a user or user group sees all or a portion of the SKS he/she is interested in.*

At this point the sources providing the information available in an SKS were described, now all this information needs to be retrieved. For this, queries are employed, to ask information from an SKS. These queries, as presented in Definition 5, are built using the user ontology O^u (Definition 4).

Definition 5 *The query posted by the user using a triple based query language, as it is for example the SPARQL language used to query RDF, is represented as Q^u. Queries Q^u formulated by a user or user group are stated using elements from O^u, independently of which of the S_i or how many of them contain the information required to answer the query Q^u.*

In order to complete the definitions required to state the problem, the relations between the different sources in an SKS need to be represented. This is expressed in terms of (possibly incomplete, therefore partial) mappings between the ontologies defined in Definition 4 and Definition 3, and is presented in Definition 6. The mappings used in this work relate concepts to concepts or properties to properties in different ontologies and are not of complex nature as also mainly followed in (41, 89) in order to avoid also computability issues as shown in (24).

Definition 6 *The mappings M_i between elements in the ontology of the user O^u and elements in the ontology describing the $i-th$ information source O^{S_i} specify a correspondence and a relation between elements in the two ontologies. A set of mappings is also called alignment between two ontologies. Each mapping M_i consists of a set of triples of the form (e^u, e^{S_i}, v) specifying pair-wise mappings between elements and a confidence value. The ontological element (concept or property) e^u is from the alphabet of O^u, the element e^{S_i} from alphabet of O^{S_i}, and v is a mapping confidence value between [0,1].*

2.4.2 The Problem

Given one SKS containing several information sources S_i described by more or less formalized ontologies O^{S_i}, a user preferred ontology O^u used to state structured triple-based queries Q^u on the SKS, and partial automatically computed mappings M_i between elements in O^u and elements in every O^{S_i}, enable the seamless integration of those sources S_i at query time, returning the best possible ranked query results to the user while at the same time learning from the user feedback such that future similar queries give better results.

2.4.3 Addressed Challenges

Along the steps presented in Section 2.2.2 and the problem stated in Section 2.4.2, the challenges addressed in this work are:

- Mapping computation: Partial mappings M_i between the schema used by the user O^u to express the queries Q^u and the schema describing the data sources O^{S_i} need to be automatically computed, i.e. correspondences between both elements in the ontology describing a source and the ontology used by the user to formulate the queries have to be found.
- Query reformulation: In order to enable the seamless integration of the sources in the SKS, the user formulated query Q^u needs to be translated to the different

schemata O^{S_i} describing the available data sources at query time, under consideration of the available mappings M_i. Since mappings are known to be partial this has to be considered in the process, and, in addition, the discovery of unknown mappings needs to be enabled, improving in this way the accuracy of the results of future similar queries in a pay-as-you-go fashion.

- Results handling: The results obtained from querying the different sources S_i should be ordered in a ranked fashion, taking into account the confidence of the integrating system in the fact that the returned results are actually correct results (the applied methods might introduce incorrect results e.g. due to query relaxation).

The overview of the approach described in this thesis will be presented in the next section, followed by the main contributions of the approach.

2.5 Lightweight Information Integration Approach Overview

First a general description of the approach presented in this thesis is provided, as well as pointers to the detailed descriptions to offer some guidance to the reader. Then, highlights of the main contributions of this thesis (Section 2.5.1) are presented.

The approach presented in this work deals with the integration of information and data stored based on triple models, e.g. for integrating information stored in RDF. Furthermore, the data stored in different sources is assumed to have heterogeneous representations, this is, the data is not assumed to comply to the same schema across sources.

A lightweight integration approach is proposed, which is capable of dealing with heterogeneously structured sources described by simple ontologies. It is based on the automatic computation of partial mappings. Queries posted according to an ontology describing the user view on the domain are rewritten using the existing mappings in order to retrieve data from the different sources, and relaxed to deal with the incompleteness of the mappings. The relaxed queries also serve to discover new mappings between the user ontology used to build the query and the ontologies describing the data sources by means of user feedback on the obtained results. The employed mappings and the degree of introduced relaxation are combined in order to compute a ranking. This ranking reflects the confidence in the results obtained from each source for a query, and is used to sort the obtained results.

The Lightweight Information Integration approach presented in this work aims at giving the best possible query results given the information and evidences available at a given time. This leads to answers which might be correct, but without general guarantees of correctness. User feedback on the results allows this Lightweight Information Integration approach to learn new mappings or to accumulate evidences for existing ones, so that future similar queries give more accurate results.

In order to achieve this, and considering the objectives stated in Section 2.4, some steps need to be performed. The proposed solution developed in this thesis will be shortly sketched in the following and presented in detail in the corresponding chapters:

- Automatic partial mapping computation: Computation of the initial partial mappings M_i between a user ontology describing the SKS (O^u) and the ontology describing each information source O^{S_i} to be accessed (see Chapter 4). The computation of the mapping between elements in the respective ontologies takes into account the intended meaning of the considered ontologies, in order to reduce the risk of computing wrong mappings;

- Query reformulation: Each query Q^u posted to the system is rewritten by considering the available mappings M_i so that it can be executed on sources S_i which use different schemata $O^{S_i} \neq O^u$. Since the mappings employed are assumed to be incomplete (to reflect imperfect mapping computation), the reformulated query relaxes the original query conditions for unmapped elements by replacing them with a form of wildcards following well defined strategies (see Chapter 5);

- Query ranking: Results collected after executing a query on the different sources S_i for which the query could be reformulated need to be ranked in accordance to the confidence of the query giving accurate results when executed on a specific S_i. For this a ranking function is presented which reflects the confidence of the query reformulation (see Chapter 6);

- Mapping learning: In order to give more accurate results for successive similar queries, mappings are learned from past query experiences. Feedback on the obtained results for a given query, and the actual bindings (wildcard-value) employed for obtaining correct results are combined to learn new mappings or to enhance the confidence of existing mappings (see Chapter 5).

The individual parts of this Lightweight Integration approach were presented along the general steps for information integration as described in Section 2.2.2. For each of these steps contributions beyond state of the art are provided, which will be summarized in the next section.

2.5.1 Main Contributions

The summary of main contributions developed throughout this thesis and presented along the general steps for information integration as described in Section 2.2.2 are:

- An automatic approach for computing mappings between ontologies under consideration of the *modeling perspective*. This approach uses a combination of several techniques to achieve this (Lexical, SAT-Based and Structure-based), and, in order to consider the *modeling perspective* it includes the context of each element to detect a match. The inclusion of the *modeling perspective* for the computation of the mappings and the combination of several techniques show in the performed experiments the feasibility of its use in specific cases (see Chapter 4 for details).

- A flexible approach to reformulate SPARQL queries, by rewriting and relaxing them using partial mapping information, so that several differently structured sources can be queried, and new, unknown information can be retrieved. This query reformulation approach takes advantage of user feedback on the new and unknown retrieved information to learn new mappings between the elements used in the original query and the elements appearing in the ontologies describing the sources, in a pay-as-you-go fashion. This query reformulation approach gives good precision and short learning times for new mappings in the performed experiments, showing its feasibility. Several query reformulation strategies are evaluated, and the ones most suitable for also minimizing the cold start problem are discussed in Chapter 5.

- A punishment-based ranking approach for queries, which is based on the degree of relaxation introduced and the mappings employed in the query reformulation process is also presented. This ranking approach enables to rank the results coming from a given source depending on the confidence to obtain right results out of it, since more relaxation might add more incorrect results, as also the usage of less confident mappings employed in the query reformulation process. The experiments on real world data sources show that the proposed ranking helps in obtaining more precise results first in the result list, and with this its feasibility is shown for ranking the results obtained from querying heterogeneous sources using queries reformulated with the approach presented in this thesis. The ranking approach, together with all evaluation details are discussed in Chapter 6.

The contributions presented above are the main results of this work, while smaller contributions will be presented in the respective chapters.

This chapter described in Sections 2.1, 2.2 and 2.3 the areas where the problems addressed in this thesis are positioned in the research area of information integration. Following this, Section 2.4 states the problem in a formalized way by first introducing in Section 2.4.1 some definitions which will be used throughout the thesis and which are required to formalize the problem, followed by Section 2.4.2 where the formalized problem statement is presented. Section 2.4.3 presents the most important challenges this work tackles in dealing with the identified problem. Section 2.5 presented the overview of the approach described in this thesis, followed by its main contributions in Section 2.5.1.

3

Related Work

This chapter presents an overview of the existing previous work related to the challenges - identified in Section 2.4 - which will be addressed in this thesis. An overview of Information Integration approaches is presented in Section 3.1, focusing on the work most related to the approach presented in this thesis, followed by the general underlying steps to many information integration approaches presented in Section 3.2. The specific related work for each of the information integration steps tackled in this thesis are referenced and discussed in the respective dedicated chapters.

3.1 Information Integration Approaches

In this section approaches to information integration will be described and their differentiating characteristics will be highlighted. The approaches are grouped according to the classification introduced in Section 2.2. Special focus is given to Pay-as-you-go integration, since this is most relevant for this thesis.

3.1.1 Pay-As-You-Go Integration

In this section approaches following a *Best-effort per-request* strategy are presented (see Section 2.2). This approaches are also know to follow the pay-as-you-go approach. Pay-as-you-go approaches aim at providing an “as good as possible” integration of heterogeneous and independent information sources at a given time, improving the future integration based on previous interactions of the users with the system.

Dataspaces (54) are a data management abstraction different from usual information integration management systems aiming at providing a wide integration and collaboration of heterogeneous sources of information while preserving their independence. DataSpace Support Platforms (DSSP) (67) aim at providing a suite of interrelated services that enable developers to focus on the challenges of their applications rather than on the integration details as are e.g. the consistency and access problems related to the integration of very disparate systems providing large amounts of data in different formats. Goal of these DSSPs is to provide an on-the-fly integration of disparate information sources, not as traditionally understood in integration systems but more in a data coexistence approach. One general goal of such systems is to avoid to have to define all mappings between the collaborating sources *before* the data therein is accessible, but to enable a step-wise access "as good as possible" which is improved over time in a pay-as-you-go fashion. Such systems need to deal with incomplete query answers, inconsistencies in participating data sources, reformulation of queries so that they can be executed on different systems, ranking of the obtained results, management of the feedback given to results, etc.

Some dataspace systems rely on schema matching and reference reconciliation to compute initial correspondences of data between systems, denoted as candidate matches. Since candidate matches to be presented to the user could be of very high number, e.g. the Roomba (76) dataspace component implements an approach to order them so that the user first gives feedback on matches giving maximum gain. For this the authors develop a decision-theoretic framework for ordering candidate matches using the concept of the *value perfect information* (120), having at its core an utility function that quantifies the effects of feedback on each candidate match (76). In principle, this is based on comparing the utility of a dataspace before a given match is confirmed, and after it was confirmed. Different matches give in general different utilities (e.g. amount of data which can be accessed, accuracy of the retrieved data, etc.) and the authors consider the combined utility of each match (as a matter of fact, an approximation of it) for ordering the feedback requests.

The system presented in (122), describes a self configuring data integration system providing pay-as-you-go integration using a probabilistic mediated schema computed automatically from the contributing sources. Such a mediated schema contains all possible correspondences between elements, along with the probability of each of them to be the right one. Based on this mediated schema, probabilistic mappings from the sources to the schema are also computed in an automatic fashion. So, in this approach a mediated schema is created out of the available sources, opposite to the approach presented in this thesis, where the user or community of users define its preferred view on the domain(s) of interest.

3.1.2 Data Warehouses

Data warehouses (35, 141, 155) are centralized systems used for performing comprehensive data analysis, which aim at integrating e.g. enterprise data, according to predefined dimensions, such as time or place (128).

Data warehouses (DW) present the analyzed and mined data in different abstraction depths, by arranging and combining data available in lower abstraction layers so that it can be presented in higher abstraction layers. The lowest layer contains the real source of data, usually integrating data from various heterogeneous systems. The data - coming from possibly multiple sources - is modified if needed, so that it can be represented under a common schema or view. Furthermore, its detail level is homogenized so that it can be aggregated. This homogenized data is combined and stored in specialized databases as new data, having in this way only one value for the combined data items (e.g. description, ID, etc.) at each point in time. Based on this combined data, decisions or market trends analysis can be made by the responsible stakeholders. In the process of data aggregation, summaries are computed and stored along with the aggregated data, so that reporting components which employ this data can perform faster. The creation of a DW makes some strong assumptions: it is necessary to know the sources which will provide data, to plan in advance how each source's data needs to be modified so that it can be integrated, and to enforce data quality at each step in the process.

The storage of the DW data typically is based on a multidimensional data model (65): facts representing the subject of analysis (elementary data) are organized in n-dimensional spaces or data cubes[1]. Feeding a DW database with data involves reading and transforming the data. Furthermore, it is necessary to perform data cleansing and duplicate removal, before the data can be loaded into the DW (this process is known as ETL: Extraction, Translation/Transformation, Loading). Over time, changes in the data at

[1]for an overview on technologies for storing n-cubes please refer to (156)

the sources need to be detected, so that the summaries and views on the data in the DW can be re-computed or refreshed (either completely or incrementally). Some DW are implemented as collections of materialized views (65). Since refreshing materialized views can be an expensive task, several optimization strategies have been designed and analyzed (e.g. (29), (64), (161)). Also the related problem of maintaining the consistency between views has been studied, e.g. in (162).

DW systems might have to evolve over time, because reporting requirements are changing, or - even more frequently - the underlying systems and the structure of data handled by them (119) evolve over time. As a consequence, the schema of a DW system might have to be changed, e.g. with respect to the required dimensions the DW system should support. Research dealing with such evolution challenges has been performed for example by (119), (99), (43).

According to the classification introduced in Section 2.2, the data warehouses approaches employ the *Centralized pre-request* strategy. The approach followed in this thesis is a different one to DW, namely, performing the integration at query time, and not in advance, and without any data quality control or guarantee.

3.1.3 Distributed Information Retrieval

This research area focuses on the integration of distinct and heterogeneous information sources, providing several different methods to do so. Historically this area dealt with unstructured sources, and later it included also the mixed integration of structured, semi-structured and unstructured information sources. Approaches presented in this section employ the *Correct Per-request* strategy known from Section 2.2.

Several approaches exist for distributed information integration. There are approaches relying on the usage of wrappers to enable integration of structured as well as unstructured information. In an information integration scenario, wrapper is a source-dependent piece of software which deals with the specific details and complexities of the source, and which is responsible for the conversion of methods, formats, models, etc. in order to make it conform to the overall employed methods, formats, models, etc. of the global system. In the TSIMMIS project (60) an infrastructure is provided to construct and deploy those wrappers, as well as to define different aspects required for querying and viewing the integrated results. These wrappers need to be defined and usually also implemented for each new source to be integrated, which requires a lot of manual work for each source, as well as possibly high maintenance costs if sources change frequently. In (60), each wrapper knows about the sources it wraps, and wrappers can be organized hierarchically, in order to hide details of some sources and, more importantly, to incrementally build an information model. The final information model available is built out of the wrappers model, so that no global schema exists for all the information available. In (60) it is not even required that all the information is stored using the same degree of structuring, sources providing unstructured data can be combined with sources providing structured one, and it is left to the user to understand results returned by queries executed on the system.

The work presented in (3) in the context of the CALO (Cognitive Assistant that Learns and Organizes) project aims at "developing personalized cognitive assistants [...] in an office environment where knowledge about email, schedules, people, contact information, and so on is distributed among multiple knowledge sources" (3). Its continuation in (2) presents component called Query Manager, which is responsible of managing the expansion of the conjunctive queries (expressed in the KIF language), also by performing inferencing steps based on the original query and the information available at the sources, in order to answer the queries. A central mediating ontology is used to relate the information requested by the clients with the information available at the sources, which is one of the difficulties found in this approach, i.e. the construction of such mediating ontology. An interesting aspect is the query planning and reasoning capabilities available at the Query Manager, which allows to decide among several query plans and query subgoals for executing the queries in a more efficient way. In addition, the query planing and reasoning capabilities at the Query Manager allow the integration of very different information sources, each of them having different query execution and inferencing capabilities, since the management of many of the involved aspects occurs at the Query Manager component itself.

The Building Finder system (94) also addresses the problem of integrating several heterogeneous information sources. Query results retrieved from different sources such as maps, webpages (e.g. Yahoo White Pages) and the US Census Bureaus Tigerline files, etc. are superimposed on a graphical view of streets and buildings which allows to navigate through this information. For mapping information coming from different sources a machine learning approach is used, so that the system learns e.g. how person names are represented in the different sources. The presented architecture uses wrappers for accessing the sources, and a mediator where the queries posted in e.g. the RDQL Semantic Web query language are reformulated and converted to datalog programs which are then executed on the available information sources. Results of a query are arranged in tables and sent back to the requesting agent, undergoing before a record linkage step so that same entities appearing in different sources are detected and merged or filtered out.

There also exist integration approaches in the Semantic Web area such as (85). In (85) the authors present an approach to integrate heterogeneous information sources mainly in the scientific area which is in some aspects similar to the approach presented in this thesis. They translate SPARQL (151) queries to the formats supported by the sources by relying on schema mappings and source-specific wrappers. Mappings between the schema of the query and the schema of the sources are created explicitly (manually) at the beginning of the integration of one source, and not in a pay-as-you-go fashion as it is performed in this thesis. Statistics about data available at each source are collected regularly, so that the decision on where to execute a given query is performed based on this sampling whereas in the approach presented in this thesis all sources possibly contributing to answer a query are accessed, as pointed out in the *Resource Selection* step presented in Section 2.2.2. Partial results from different sources are then combined and presented to the user, all results are correct answers to the original query under consideration of the employed mappings, this is, results obtained from the sources are determined and limited by the available mappings (if the mappings are complete then all expected results will be delivered, if the mappings are not complete then some results might be missing).

Another relevant work (69, 70) positioned in the peer data management research area (4, 13), implemented initially in the Piazza system, describes an approach to integrate RDF and XML data, by combining not only the semantic structure, but also the document structure underlying XML documents for finding data correspondences. The rationale behind this is that many systems export their schema in XML, representing implicitly the relations between attributed through nesting, opposite as in RDF where all relations are represented in an explicit fashion. The authors present a solution to this problem, by proposing a language (69) which borrows elements from XQuery (150) for mediating between these two worlds and its extension in the *Peer Programming Language (PPL)* (70). Mappings between elements are provided in local and small scale, and combined in chains until a given query can be answered. The structure of the nodes providing schema information or data (or both) in the Piazza framework reminds heavily to a peer-to-peer system of nodes, where chaining through the available nodes is used to provide the answers to a given query. A query rewriting algorithm is presented, which takes the query and the (mostly pair-wise) mappings between elements, and rewrites the query so that it can be executed on a given data-providing node.

3.1.4 Ontology-based Integration

This section collects some work performed in the logics area for information integration, which in principle could follow any of the strategies presented in Section 2.2, since they deal more with the problem of finding the mappings and how to express them, than on when to perform the data integration. Here most approaches deal with the integration of several ontologies, and study the implications and computability of different integration approaches such as the local-as-view, the global-as-view and the mixed approaches (see (87) for an overview and Section 4.1.1.1 for an explanation). In (24), the authors present a formal ontology integration system (OIS) definition. An OIS is defined as a triple consisting of a global ontology, several data sources, and mappings between the ontologies describing the data sources and the global ontologies. Mappings are considered to be defined as relations between queries over the global ontology and queries over the sources in a combination of the local-as-view and the global-as-view approaches. The authors study two types of description logics to represent such OIS, the limitations in expressing mappings as well as in expressing restrictions in the employed ontology. The used description logics are further analyzed regarding computability and complexity.

The goal in (25) is to present an approach for integrating independently developed local ontologies describing information sources into a global ontology, based on queries. This global ontology can then be used as access point to the information available in the different sources. One problem the authors see is in the specification of mappings between the local ontologies and the global one. Due to differences in e.g. conceptualization detail, the authors claim that it is more likely to need queries to relate the specifications of concepts instead of simple mappings. In this way, a mapping is not a simple relation, but a query over another ontology. A global centric approach is discussed, as well as a local centric approach (i.e. mapping each element in the global ontology as a view - query - over the local ontologies vs. mapping each element in the local ontologies to queries over the global schema). With the usage of one expressive member of the Description Logics family, the local and the global centric approaches are combined, to overcome the restrictions of having mappings only in one direction. In an extension of this work, the approach is generalized to the problem of answering non recursive datalog queries using views in the relational world, so, to compute the answer of a query only on the basis of the extension of a set of views (26). This is a relevant problem in many areas such as data warehousing or query optimization, among others. Different types of views can exist, sound views, complete views, and exact views, each of them being handled differently in the approach.

Despite the differences in the approaches to integration, several common steps can be recognized to appear in them. These steps will be presented in the following section and will be used as a basis for structuring the related work sections specially dedicated to the steps tackled in this thesis which are presented in the respective chapters.

3.2 General Information Integration Steps

Independently of the approach employed for the integration of sources, usually similar steps are performed in the different approaches to accomplish an information integration task as presented in Section 2.2.2. Approaches for some of these steps will be described in the following sections.

3.2.1 Resource Description

Resource description refers to the data stored at a source; how it is structured in case there is some structure, or in general, what kind of content is available at a source. Resources can be collaborative, this is, they expose their schema or information about their contents (e.g. statistics) to interested parties, or, can be uncollaborative, this is, no schema or other information is explicitly given to the outside world. Collaborative and structured information sources are the ones assumed to be available in this thesis, so, it is assumed that there is a schema or ontology describing the information source and it is available. Dealing with uncollaborative information sources is also possible. There are approaches used to obtain the description of a given resource (10, 22, 126) out of e.g. query sampling over some determined period in time, but this can be a difficult task. Frequencies of word occurrences is a compact and efficient way to represent a e.g. full-text information source in the area of information retrieval and integration as pointed out in (22). More details regarding the description of resources will not be presented here, since resource description has not been selected as part of the challenges to be addressed in this thesis (see Section 2.4).

3.2.2 Schema Mapping

A schema mapping is considered in the literature as a high level specification describing the relationships existing between two schemas under which data is stored or organized. Since schema mappings constitute the essential building blocks of data exchange and integration (49), extensive work on mappings, their use and foundations have been carried out. An overview of the most prominent mapping approaches are presented and discussed in Section 4.1.

3.2.3 User Query Formulation

The query to be executed needs to be specified by the user who is interested in getting an answer. Different paradigms have been explored for building queries, from loose keywords to structured query employing operators with well defined semantics. Mixed approaches for building such queries, by e.g. first issuing a full- text query and later traversing the first results by clicking on a map (94) or, in the case of structured query formulation in the Semantic Web, e.g. the Query Builder (77) which supports the user in building SPARQL (151) queries given a personal ontology, have been proposed. The details of these approaches will not be presented in this work, since query formulation is not part of the challenges presented in Section 2.4.

3.2.4 Query Reformulation

The user formulated query needs to be translated to the different formats supported by the available data sources. An overview of the existing work in this area, as well as a detailed description of the approaches known to be similar to the approach proposed in this thesis can be found in Section 5.1.

3.2.5 Resource Selection

Depending on the query, the information sources possibly contributing to the results of this query have to be selected. In the Federated Search area, many approaches exist for selecting resources appropriately for answering a query. Approaches target usually either the recommendation of information sources supporting resource selection, or the retrieval of distributed documents, so, they use different models to handle each of those different problems. For example in the Distributed Information Retrieval area, distributed information sources or databases providing textual content need to be selected accordingly to a posted query and a purpose or task. In cases where the task is to recommend a database for searching purposes, the goal is to have high recall, while in cases where the task is distributed document retrieval, the goal is to have high precision in the obtained results. In (125) a maximization framework for resource selection which considers the two tasks as two different maximization goals is presented, allowing therefore its use in both scenarios. Other approaches like (10, 105) were also proposed for the selection of the data source to be queried, given a specific query. The details of the approaches will not be further presented here, since resource selection is not part of the challenges presented in Section 2.4. A good overview and further reading regarding resource selection can be found in e.g. (126).

3.2.6 Results Handling

The results obtained from the queried sources have to be collected and presented to the user in the best possible way, aiming at providing the best possible user experience. Important parts of this task are to rank results coming from different sources, and to merge or remove duplicate items (123). Several approaches for performing deduplication e.g. (12, 39, 75, 121) exist, as well as several solution overviews are available, e.g. (38, 154). A description of the work related to ranking results of queries is presented and discussed in Section 6.1.

4

Mapping Ontologies

In this chapter the problem of finding correspondences and relations between elements in two different ontologies is addressed (see challenges in Section 2.4 and approach description in Section 2.5).

The approach for the computation of the mappings between elements in the user ontology and the ontologies describing the sources presented in Section 4.2 takes several aspects of the ontological elements (concepts and properties) into account. This approach was developed and implemented under my supervision for reusing ontologies in the ontology engineering area (80), and later extended and re-implemented in (134, 135). It combines lexical, linguistic and logic methods for finding matches between ontological elements (concepts and properties). Furthermore, it is a special characteristic of this approach to take the *modeling perspective* into account. The *modeling perspective* is the way of viewing and modeling an ontological element in a specific ontology, e.g., a person can be considered as a social being, or as a biological entity. The *modeling perspective* heavily influences the intended meaning of the considered element.

Before presenting the details of the approach, the corresponding related work will be described.

4.1 Related Work on Mappings

Mapping, matching or alignment are all terms used in the literature, to name the detection of syntactic and semantic relations existing between elements in different schemata as well as to represent them. In the ontology research area (47), "*matching* is the process of finding relationships or correspondences between entities of different ontologies, *alignment* is a set of correspondences between two or more ontologies, and *mapping* is the oriented, or directed, version of an alignment: it maps the entities of one ontology to at most one entity of another ontology"(47). In Section 4.1.1 a short overview of basic approaches in the relational and XML world will be presented, while in Section 4.1.2, approaches dealing with this problem in the world of ontologies are described, and finally, Section 4.1.3 presents a discussion about the presented related work. Regardless of being in the relational or in the ontologies world, schema matching is a problem arising in many database applications, especially where data from different sources needs to be integrated. Other areas requiring mappings or matching are the e.g. the E-business domain, data warehousing, and semantic query processing (117).

4.1.1 Schema Mapping

As mentioned in (117), a fundamental operation in the manipulation of schema information for information integration is the *match* operator, which takes two schemata as input and produces a mapping (semantic correspondence) between elements of the two schemata.

Traditionally, the task of schema matching was mainly manual. Finding correspondences between the different schemata was a tedious, error prone and therefore costly process. Graphic user interfaces supporting this process made it easier, but, they remained mostly manual. Later, several research lines exploited information available in schemata, or in the data contained in the databases described by them, in order to try to automate this process. However, the process remained error-prone, hard to compute, and therefore costly. A basic characterization of the different matching approaches is described in this section, while a more comprehensive classification of approaches is presented in Section 4.1.2.

Matching approaches can be orthogonally characterized based on the type of input they use, or on what they intend to match. Some of the resulting categories, which can be used for classifying matching approaches and which are mostly orthogonal (see (117) for a more complete overview of such categories) are:

- *Instance vs. schema:* classifies the approaches based on where the information for computing the matches comes from, from some instances complying to the schema of the database, or only from the schema describing the structuring of those instances.
- *Element vs. structure level:* takes into account if matching is performed for individual schema elements, or for combinations of elements (complex schema structures).
- *Language vs. constraint:* is about the type of information used for determining the matches; information about the names of the elements and their descriptions, or information about constraints expressed in the schemas (keys, relationships, etc).
- *Matching cardinality:* considers if the results of such a matching process relates only one element in one schema with just one element in the other schema, or if e.g. m:n relations are possible to compute and to express in the language used to represent the mappings.

A comprehensive description of the characteristics of each approach, as well as an overall classification of the basic matching techniques will be presented in Section 4.1.2. It has to be noticed that the approaches are to some extent independent of the structures to be matched, this is, relational, XML or even ontologies use approaches belonging to these different categories (with some adaptations required to deal with the specific characteristics of these structures).

Another aspect in the schema matching area is the production of an integrated schema. This problem is called schema integration in the literature (see (6)), and it is specifically the problem of creating a unified target schema based on a set of existing source schemas that relate to each other via specified correspondences or mappings. This unified schema gives a standard representation of the data, offering a way to abstract from the heterogeneity of the sources for e.g querying the systems using a query specified with elements of this unified schema. Since different unified schemas can be built out of the same set of sources, and the right one will depend on the application it is built for, a way to adaptively enumerate all interesting integrated schemas is presented in (31). In addition to the enumeration of interesting schemas, a way to reduce the possibly high number of enumerated schemas by user interaction is also presented. It filters integrated schemas by taking into account user-specified restrictions, and builds information-preserving mappings from each of these integrated schemas to the sources, so that this mappings can be used to access the content at the different sources.

Approaches aiming at the generation of integrated schemas are different from the approach presented in this chapter, since the aim of it is not to produce a unified schema out of several existing schemata, but to find the mappings between one existing schema and another existing schema. The basic methods (see Section 4.1.2 for a classification) employed for these two different tasks can and are usually similar, but the pursued goal is different. Even though schema and ontology mapping (see Section 4.1.2) use mostly the same basic matching techniques, there are some differences which will be discussed in Section 4.1.1.2.

Integration of different sources using a global schema have been classified in the literature to be local-as-view (LAV) or global-as-view (GAV) (see (87) for a comprehensive description). Both of them use mappings between a global schema, employed for defining queries, and schemas describing the different sources to be integrated. The difference between these approaches presented in Section 4.1.1.1 resides in the mappings.

4.1.1.1 Local as View vs. Global as View

In the local-as-view based integration, the mapping associates to each element of a source a query over the global schema, i.e. only the name of one of the elements in the source appears in the query while several names of elements contained in the global schema can appear in it. In this way, each mapping consists of one assertion for each element in a source with a corresponding query defined over the global schema. This makes the LAV systems very robust to additions of new sources or changes within the sources, since each element is characterized using elements in the global schema (examples of such systems are presented in (79) and (92)). On the other side, query processing in LAV systems is difficult. This difficulty arises from the fact that the knowledge about the data available at the level of the global schema can only be obtained through the views representing the sources (generally providing only partial information about the data). Since the mappings relate a view over the global schema to each source, it is not straightforward to decide how to use the sources in order to answer queries expressed over the global schema. A very good description of the LAV approach be found in (143).

The systems known as global-as-view employ mappings in which elements in the global schema are defined with queries over sources, i.e. these queries contain the name of one element in the global schema and as many from the source schema as required. In such systems each mapping contains one assertion for each element of the global schema with a query over the source schema. Most integration systems follow the GAV approach, such as e.g. TSIMMIS (61) and Garlic (27). Query processing can be regarded as easier in the GAV approach, since the elements in the global schema are mapped to views on the sources. In this way, queries defined on the global schema can be reformulated to use elements in the sources by unfolding them, which is the most extended way of handling queries in GAV systems.

The integration approach presented in Chapter 5 employs one to one mappings. In such a setting there is no difference between one or the other integration approach, since only one element from the source, and one element of the global schemas are appear in each mapping specification.

Other types of mappings were also researched, including the combination of LAV and GAV approaches denoted as GLAV (56) approaches. As presented in (56), mappings consist of conjunctive queries (i.e. views) over both the local schema and the global schema, and mappings are composed by relating conjunctive queries on a source with conjunctive queries on the global schema. This allows the most freedom to specify the mappings between sources, and query processing is still possible by transforming GLAV mapping assertions to GAV assertions (87).

As previously mentioned, even though schema mapping and ontology mapping employ mostly the same underlying matching techniques, some differences can be found. The next section discusses the major differences.

4.1.1.2 Schema vs. ontology mapping

Mapping of schemas and ontologies present many similarities, as pointed out previously. Many of the matching techniques employed for the detection of mappings between schemas are also employed for finding mappings between ontologies. Ontologies and schemas are intended to represent the structure of some data of interest, and often the same data can be represented using different schemas or ontologies (i.e. employing different notations or structure) in different applications or by different people.

However, some differences can be observed based on the formalisms used to represent one or the other, or on the intrinsic modeling differences between schemas and ontologies.

Schemas are built for a specific purpose, and are defined within an application representing constraints on the data to be stored, so that data can be validated (129). Ontologies aim to "capture consensual knowledge in a generic and formal way, and [...] may be reused and shared across applications [...] and by groups of people" (32), i.e. they are general descriptions of domains of interest, and can be used across applications (assuming their level of detail, formalization degree, etc. are appropriate). Semantic Web ontologies are mainly based on the open world assumption assuming incomplete information, i.e. there is no truth value associated to things not explicitly stated, while schemas present a close world assumption, i.e. if something is not explicitly stated as being true then it is regarded as false.

Another difference which can be observed between the mapping of schemata and the mapping of ontologies, are the logical axioms which exist in ontologies which can be used as additional source of knowledge for deciding on matches. Schemata are usually not built based on an underlying taxonomy, as it is the case in ontologies. This super-/sub-class relation, as well as other logical axioms appearing in ontologies, can be employed to help deciding on matching elements when performing ontology matching and are usually not available for deciding on schema matching.

4.1.2 Ontology Mapping

The alignment or mapping of ontologies has received much attention of the research community in recent years and several different techniques and approaches were proposed. There are different types of heterogeneity between schemata or ontologies, classified in (47) as syntactic heterogeneity, terminological heterogeneity, conceptual heterogeneity and semiotic heterogeneity. *Syntactic* heterogeneity occurs when the languages used to express the different ontologies are not the same (e.g. F-logic vs. OWL). This kind of mismatch can be resolved if there is a translation from one formalism into the other, in some cases even being equivalent and thus meaning preserving. *Terminological* heterogeneity occurs due to variations in the names used to refer to same entities in different ontologies, e.g. *Female person* vs. *Woman. Conceptual* heterogeneity also called sometimes semantic heterogeneity or logical mismatch denotes having differences in the modeling of the same domain of interest. This can occur due to the use of different axioms for defining concepts (sometimes being equivalent) or due to the use of totally different concepts. In the context of conceptual differences three possible reasons for them have been identified (11):

- Coverage difference: occurs when two ontologies describe different, possibly overlapping regions of the domain at the same level of detail or from the same perspective.

- Granularity difference: occurs when the same region of the domain is described from the same perspective, but at different levels of detail.

- Perspective difference: which is also known as difference in scope (30) occurs when two ontologies describe the same region of the world, at the same level of detail, but from a different perspective. The approach presented in this chapter aims not only at finding matching elements having *terminological* differences, but also to detect differences in the modeling perspective when deciding which elements to match.

Ontology matching approaches employ several techniques to address the matching problem, as for example machine learning, linguistics, concept lattices, formal theories and also simple heuristics among others. There are approaches that consider not only the ontology, but also the instances of concepts to determine the meaning and use of concepts. Concepts can be used in diverse ways by different communities or applications and a good way to detect this is by analyzing the corresponding instances.

A recent overview and classification of work existing in the **Ontology Matching** can be found in (47). It presents not only a variety of systems and their details, but also a comprehensive classification of all basic techniques currently used by the existing matching approaches. Figure 4.1 presents an overview of the classification as presented in (47). In the following the basic matching techniques will be briefly explained in order to place the approach presented in this chapter in the classification. The complete description of the different categories in Figure 4.1, as well as detailed descriptions of the basic techniques can be found in (47).

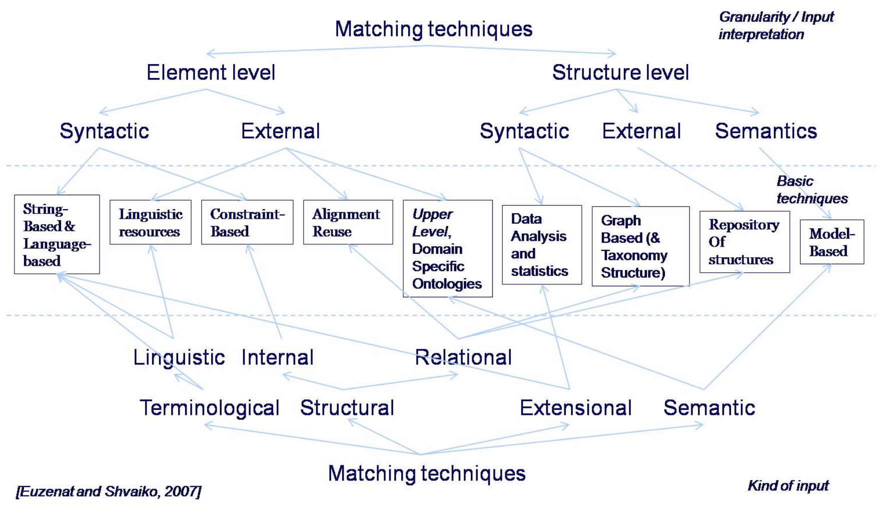


Figure 4.1: Classification of matching techniques

The most widely known basic techniques used for computing matchings are presented below. It has to be noticed that most of today's approaches combine several basic techniques in order to obtain better results, exploiting in this way all the available information for the hard task of finding matches.

- *String based* techniques are commonly employed to match labels or names and also descriptions of ontology entities. They consider these strings as sequences of letters and use different approaches to state to what extent two strings are similar/equal by comparing their letters or substrings (e.g. n-grams). Usually a similarity measure (i.e. a real number between 0 and 1) is the output of a comparison process, representing the degree of similarity.

- *Language based* techniques consider the names and descriptions of entities in the ontology as words in some natural language. These exploit also morphological properties of the words, using natural language processing.

- *Linguistic resource* based techniques employ linguistic resources such as lexicons or domain specific thesauri to match words, based on relations such as synonyms or hyponyms between them.

- *Constraint based* techniques take advantage of the internal constraints found in definitions of entities, such as type information, cardinality of attributes, referential keys, etc.

- *Alignment reuse* techniques are approaches using external resources of information containing previously computed matches in order to help in the decision of new matchings.

- *Upper level and domain specific ontologies* are techniques employing resources of common knowledge such as Cyc (86), SUMO (104), DOLCE (59), etc. to match entities, providing a grounding for taking decisions on matches (e.g. through inferencing) between different schemata. Domain specific formal ontologies can also be used to provide background knowledge, and as a grounding for deriving matching decisions.

- *Data analysis and statistics* techniques are those which take advantage of a representative sample of a population in order to find regularities and discrepancies, using this information to group items together, or to compute distances between them. Some of the techniques used are distance-based classification, formal concept analysis, and frequency distributions.

- *Graph (and Taxonomy) based* techniques consider the ontologies (as well as database schemas and taxonomies) as labeled graphs. Usually the similarity comparison between pair of nodes from two ontologies is based on the analysis of their positions within the graphs, having the underlying intuition that if two nodes from ontologies are similar, their neighborhood must also be similar. The techniques involving trees and taxonomies are special cases of this graph based approaches, where the focus is on the is-a relation.

- *Repositories of structures* store ontologies and their parts of fragments, together with pairwise similarity measures. Unlike *alignment reuse*, here only similarities between ontologies are stored, and not alignments (i.e. sets of mappings). So, whenever ontologies or parts of them need to be matched, first the repository is used for finding similar structures (where this comparison is cheaper than the computation of the full mapping), and based on the findings matching decisions are derived.

- *Model based* (or semantically grounded) techniques employ the semantic interpretation of the input ontologies, e.g. model-theoretic semantics. The idea behind these techniques is that two entities are the same if they share the same interpretations. The techniques used here are well grounded deductive methods, such as propositional satisfiability or description logics reasoning techniques.

An overview of systems will be presented shortly in the following, together with the characterization of the techniques employed. The approaches were selected from the literature with the goal of presenting example systems using the above presented techniques. As already stated, most of the systems use a combination of basic techniques, to allow its use in more than one scenario, gaining and combining evidences from different information available in different situations. Among the existing approaches, the iPrompt (107) tool suite contains tools for ontology merging and alignment. It receives as input a set of pairs of related/similar concepts from different ontologies. From this set of pairs, *AnchorPrompt* produces a set of new pairs of semantically close terms. *AnchorPrompt* views each ontology as a directed labeled graph and uses structural similarity to derive semantic similarity of concepts. *AnchorPrompt* has problems to detect similar concepts if the analyzed ontologies are structurally very different (e.g. one with a very shallow hierarchical structure compared with another with a deep structure).

The GLUE (37) approach uses a machine learning approach to find matchings by analyzing instances. The similarity of concept meanings is defined based on the joint probability distribution of the involved concepts. GLUE calculates the joint probability distribution and lets the application compute any suitable similarity measure with it. The joint similarity distribution is computed with a classifier trained for instances of concept A and then applying it to instances of a concept B of the other ontology. For the computation of the joint similarity distribution, several learners are used and their results are then combined with a meta learner (to determine the weights to be assigned to each base learner's result). GLUE also exploits available domain constraints and general heuristics to determine or improve matchings.

The QOM (44) approach is based on a combination of several basic techniques. It first analyzes the features (characteristics) of two entities from different ontologies. This feature approach assumes that identical entities have the same features. The features of ontological entities are extracted from extensional and intensional ontology definitions. For reducing the runtime complexity, it employs a search selection step where heuristics are used to reduce the number of candidate mappings to be analyzed. It also proposes a combination of pre-selection methods: label comparison, analysis of an area around an existing mapping (context), change propagation, hierarchical traversal (top-down), etc. The similarity is computed by aggregation of different similarity measures using a sigmoid function on the weights to better employ individual results, reducing the importance of those that do not contribute to the result. The approach is iterative: each iteration uses results obtained in the previous one in order to enhance the quality of the produced results.

MoA (78) is an approach to merge and align OWL ontologies. This approach uses linguistic methods to disambiguate the meaning of concepts and properties based on their local names. It presents a model to represent this lexical information, which is obtained from WordNet, and an algorithm to detect and represent in a *Semantic Bridge* semantic equivalences between concepts and properties. The Semantic Bridge can represent equivalence of concepts and properties, subconcept and subproperties and identity of instances. A merging algorithm is also presented which uses the semantic bridge and the two ontologies to create a merged version of them.

The OMEN (97) approach uses Bayesian Networks (BN) for deciding the match of concepts based on an initial hand made match. Based on known matches it analyzes the structure (e.g. domain and range of properties) to derive further matches. Nodes in the BN represent individual pairs of matches and are generated by combining pairs of concepts in a neighborhood of length 'k' starting at the initial matching nodes (this path length 'k' was introduced to avoid too much growth of the BN). Meta-rules based on the semantics of ontology relations are used to express how each mapping affects other mappings. This is expressed in Conditional Probability Tables (CPT). CPT's are used to represent how a probability distribution in one node in the BN graph affects the probability distribution in another node downstream from it. The OMEN algorithm receives as input the two source ontologies to be matched and the initial probability distributions on the nodes without a parent (root nodes) in the BN graph. The algorithm operates iteratively and produces in each iteration new matching evidences which are used in the following interactions. This approach requires interaction for specifying the seed matches, it then finds automatically matches by exploiting structural information.

The CTXMatch (18) approach is based on the semantics of the elements of schemata to be matched. This approach discovers mappings between nodes of different schemata by logical computations based on the explicit representation of the meaning of each node in the schemata in combination with additional (possible) existing background knowledge. A lexical resource such as WordNet is employed for eliciting the possible meanings of each element's label and then hierarchical structure information in combination with this semantic information is used for discarding the meanings that are not applicable in each case. As a result the method delivers a logical formula expressing the meaning of each element which is then used in a satisfiability check (by including background information) to compute the semantic relation holding among elements of different schemata. This chapter presents an approach which integrates the CTXMatch approach for collecting additional evidences in computing matching elements.

4.1.3 Discussions

Taking into account the information integration scenarios targeted in this thesis, instances are not available when the mappings are first computed, and can therefore not be considered in the computation of matching elements as they are considered in e.g. the GLUE (37) approach. The *AnchorPrompt* and OMEN (97) approaches require initial matches or probabilities, and these are not available at the time the integration is required in the scenarios targeted at in this thesis. As mentioned previously, the MoA (78) approach aims at merging the analyzed ontologies which is not the goal pursued here, and the QOM (44) employs many techniques for deriving matching elements, but do not consider explicitly the modeling intent (*modeling perspective*) nor employ satisfiability methods to compute relations between elements.

The approach presented in Section 4.2 builds upon lexical, linguistic and SAT-Based matching techniques to find matches between elements in different ontologies. Initial aim of the approach was supporting the ontology engineering process, by finding matching parts of domain-related ontologies given one initial ontology, in order to extend this initial ontology. For this purpose it takes not only each ontological element (concepts and properties) in isolation into account, but also considers its *modeling perspective* to derive matches, targeting at finding elements which not only mean the same, but are also used in a similar way, enabling in this way their easier re-use.

4.2 Mapping Approach Overview

The approach presented here has two main parts: one is the comparison of the similarity between ontological elements, and the other is the determination of the semantic relation holding between two similar elements. For determining the semantic relation between elements a logics representation of the meaning of the elements is built, and the semantic relation is computed by integrating the SAT-based CTXMatch (17) approach (based on reasoning). The computation of the similarity of elements is performed by interpreting the disambiguated intended meaning of the elements as sets of possible senses, and using set operations to compute this similarity as it will be shown later.

In the next sections the following steps will be presented in detail:

- Finding the possible meanings of ontological elements, under consideration of the *modeling perspective*;
- Constructing out of the possible found meanings a logic expression representing the meaning of the elements;

- Finding similar elements in different ontologies and computing their similarity;
- Computing the logical relation holding between similar elements (e.g. equivalence, subsumption); and
- Computing the degree of similarity of their modeling perspectives, aggregating all this information to define a similarity confidence between related elements.

In this work, the meaning of the ontological elements is specified by relating them to elements in a so called reference ontology containing words, their meanings, and semantic relations between them (e.g. synonyms, hypernyms, etc), building in this way a “frame” where meanings of elements can be compared. The *modeling perspective* will be employed to determine and measure the relative importance of each of the possible meanings found for one ontological element, improving the process of identifying the intended meaning of that element.

For better explaining the steps of this approach the user ontology (cf. Definition 4) presented in Figure 4.2 will be used as a running example.

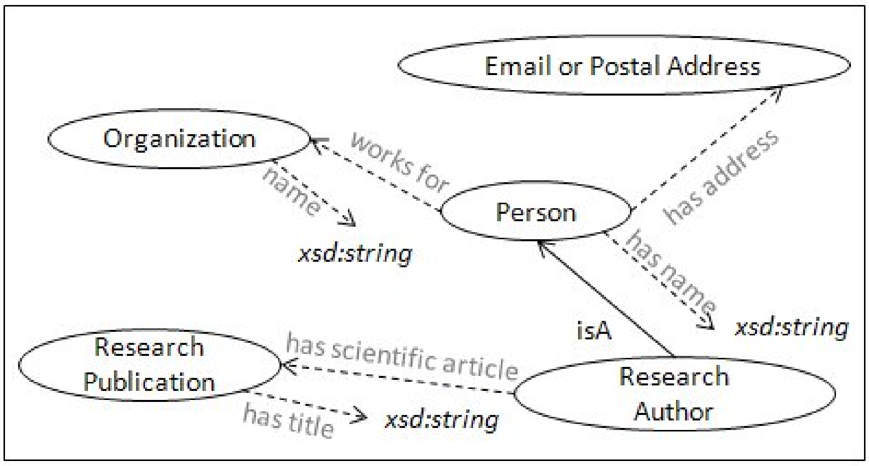


Figure 4.2: Example user ontology

In order to be able to define the meaning of the elements, as well as to determine their *modeling perspective*, some definitions are required.

4.2.1 Definitions

Recalling the definition of an ontology which was presented in Definition 2 of Section 2.4.1 additional definitions will be stated:

Definition 7 *The Ontology Graph G^O is a graph as defined in the graph theory as a set of vertices V and of edges E, $G = (V, E)$. In the definition $G^O = (V, E)$ each vertex in V is an element of $C \cup R$ from the ontology O. For each pair of distinct elements $v_i \neq v_j \in V$, if the following holds an edge between v_i and v_j is added to E:*

- *Element v_i is a concept and element v_j is a property, and v_i is in the domain of v_j or v_i is in the range of v_j, i.e. if the concept is in the domain or range of the property;*
- *Element v_i and v_j are concepts, and v_i is a direct subclass of v_j (analogously for properties).*

This is, the edges in E are defined by pairs of vertices in V, (v_i, v_j) whenever the conditions stated above hold. Formally, the definition is:

- *$V = \{C \cup R\}$, and*
- *$E = E_{(c,p)} \cup E_{(c,c')} \cup E_{(p,p')}$ where*
  - $E_{(c,p)} = \{(c, p) | c \in C, p \in R : c \in dom(p) \vee c \in range(p)\}$
  - $E_{(c,c')} = \{(c, c') | c, c' \in C : c \prec c'\}$
  - $E_{(p,p')} = \{(p, p') | p, p' \in R : p \prec p'\}$

In Ontology graphs usually found in the literature (81), vertices are generally defined by concepts, and edges by properties connecting them, resulting in a directed graph. In contrast, the definition of G^O, results in an undirected graph where concepts and properties are vertices. Edges are defined as stated above. The reason for this is that the introduced graph will be used to compute a distance between elements, independently of them being concepts or properties.

An example graph for some concepts and properties of the ontology presented in Figure 4.2 is shown in Figure 4.3. It can be noticed that the concepts and properties are represented as nodes, while the edges are defined by the domain/range or hierarchical dependencies among those nodes as defined in Definition 7.

Definition 8 *The distance between two elements (v_0, v_n), $v_0, v_n \in V$ in the ontology graph G^O is computed by measuring the length of the shortest path in the graph between these two elements. A path in G^O is the sequence of elements in V $(v_0, v_1, ..., v_n)$ such that $(v_k, v_k + 1) \in E$ for $0 \leq k < n$. The length of a path in this definition is computed by counting the edges existing in it. The distance between two elements (v_0, v_n) will be denoted as $dist(v_0, v_n) = min(|\{(v_0, v_1), (v_1, v_2), ..., (v_{n-1}, v_n)\}|)$ of the shortest path.*

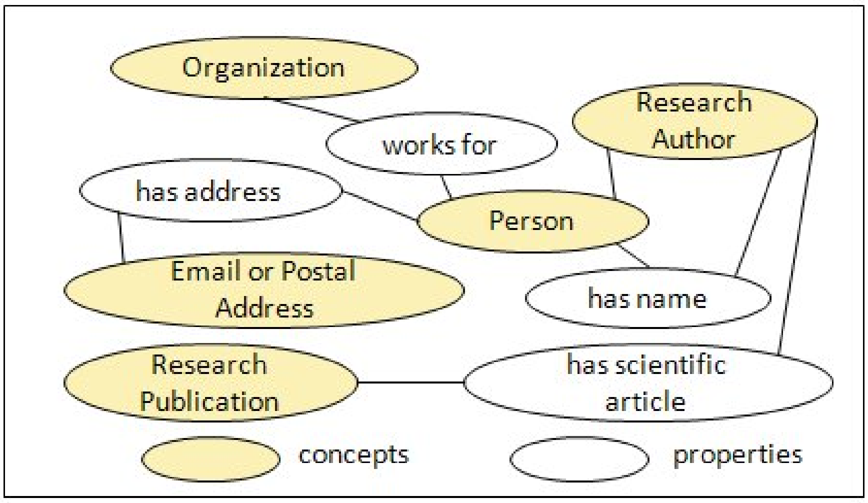


Figure 4.3: Part of the ontology graph

4.3 Finding the Meaning of Elements

The process of finding the meaning of elements in each ontology is performed by first considering each ontology by itself, independently of the other ontology to be aligned. Consider ontology O (Definition 2) to be analyzed and one of its elements e whose meaning has to be found. In order to collect information about e, the name or label of this element is extracted and analyzed, and with this information the meaning of the element is uncovered.

In more detail, the process of finding the meaning of ontological elements involves the following main steps:

- Analyzing the words employed in the labels, determining the possible senses for those words.
- Determining the *modeling perspective* of each ontological element, i.e. the context of the element.
- Disambiguating the senses of the elements based on the *modeling perspective*.
- Representing the meaning of elements in a logical concept formulae.
- Computing an importance score for each of the different senses of an element.

The following sections present the details of the above presented steps.

4.3.1 Analysis of the Labels

This section presents the analysis of the labels of elements in the ontology. This analysis aims at determining the meaning of the element by relying on an external reference ontology, by only relying on the words used in the label of elements.

The analysis of the labels of elements in the ontology assumes that labels are made of full words[1] which have a meaning, and that the used words represent the intended meaning of the element (e.g. they are not just numerical identifiers). In case the label is made of a concatenation of several words, they are identified and separated in a tokenization process. Based on these assumptions, the sequence of label words of an element in an ontology can be defined as:

Definition 9 *A label l_e of an element e is build by the label words wl_e. The sequence of words of label l_e is denoted as Wl_e, $Wl_e = \{wl_{e,1}, wl_{e,2}, ..., wl_{e,n}\}$.*

Definition 10 *The lexical information LI of a word w, is formed by the pair lemma of the word ($lemma_w$), part-of-speech (pos_w) of the word and is denoted as $LI(w) = (lemma_w, pos_w)$.*

Lexical information of the words in a label is obtained using a Part-of-Speech tagger which performs a shallow analysis of the text in the label, extracting the lemma of each word (its ground form) and the syntactic category (such as noun, verb, adverb, etc.). This information is required to better determine the possible meanings of each word, and it will be used when querying the reference ontology.

In order to continue, the reference ontology needs to be described in more detail:

Definition 11 *A reference ontology $\mathcal{RO}$ contains a set of meanings or senses S of lemmas. The senses a lemma can represent are determined by the corresponding part-of-speech. Senses $s \in S$ can also have glosses or sets of words describing them, and will be denoted as gl_s. $\mathcal{RO}$ also contains a set REL of relations rel between $s \in S$, such as synonyms, hypernyms, antonyms, etc. The relations holding between two word senses $s, s' \in S$ is defined as $REL_{(s,s')} = \{rel_1(s,s'), rel_2(s,s'), ..., rel_n(s,s')\}$.*

Once the labels of the elements in the ontology have been analyzed, the possible meanings of the words in them can be searched in the reference ontology at hand. For this a function which returns the possible senses of a word given its lexical information is defined.

[1]In cases of abbreviations it is assumed they were expanded to their extended form in a preprocessing step

Abbreviation	Syntactic form
n	nouns
v	verbs
a	adjectives
r	adverbs

Table 4.1: Abbreviations for word syntactic forms

Definition 12 *The function sense(lemma, pos) is a lexical function which finds in the reference ontology $\mathcal{RO}$ the set of possible meanings or senses for a lemma and syntactic form $sense(lemma_w, pos_w) = S_w = \{s_{w,1}, s_{w,2}, ..., s_{w,n}\}$ and $S_w \subset S$. The relation $sense^{-1}(s_{w,i}) = (lemma_w, pos_w)$ is the inverse, used to obtain given a sense the originally used lemma and syntactic form.*

Following the example introduced in Figure 4.2 and searching the reference ontology used in the implementation of this approach (WordNet (50), see Section 4.5.1), it can be seen that three different meanings are found for the word "Author", two for the noun syntactic form and one for the verb syntactic form. Each different sense in a syntactic form is identified with a different number, so, in this example two are given for "Author" as a noun: *author#*1 and *author#*2. Each of the senses has a gloss property or description part so that the intended meaning can be identified (*author#*1: *writes (books or stories or articles or the like) professionally (for pay); and author#*2*: someone who originates or causes or initiates something*). In addition, each of the senses can have synonym relations to others which represent the same intended meaning, in this example a synonym for *author#*1 is *writer#*1. In this work the senses will be represented as *<word>#<syntactic form><sense number>*, e.g. *author#n*1. Table 4.1 presents the different short forms for the syntactic forms which will be considered in such a representation.

The function presented in Definition 12 enables to find the possible senses for a given word, by searching the reference ontology. Since several senses might exist, but usually not all of them are meant for a given element in the ontology, the individual importance of each sense for the representation of the meaning of the word in a label needs to be measured.

In order to disambiguate the meaning of the words in a label, the *modeling perspective* is used in this approach. The *modeling perspective* can be captured by considering the context where the corresponding element is defined, i.e. the environment of the element within the ontology.

4.3.2 Computation of the Context

Intuitively speaking, the context of an element contains all elements connected to it in the ontology graph considering paths with a predefined maximum length. Thus, a context is defined as:

Definition 13 *The context of an element $e \in O$ are all the vertex elements $v \in V$ of G^O having a distance of maximum value r to e. This is, $ctx(e, r) = \{v \in V \mid dist(e, v) \leq r\}$. Element e is called the center element of the context $ctx(e, r)$. In the following the r parameter might be omitted, since this parameter is the same for all element contexts in one ontology, so it can be regarded as a constant.*

Since elements being far away from the considered *center element* have less influence on the meaning of the center element, only a certain radius is considered (a certain distance from the center element) for the definition of a context. This radius r determines the upper limit of the path length used for considering the influence of the meaning of elements to the meaning of the center element.

Such a context is created for concepts or properties appearing in the ontologies to be mapped (e.g. O^u and the different O^{S_i} in this case). Figure 4.4 presents an example context computed from the ontology graph (Figure 4.3) for the element "Research Author" with a radius of two ($ctx(Research\ Author, 2)$).

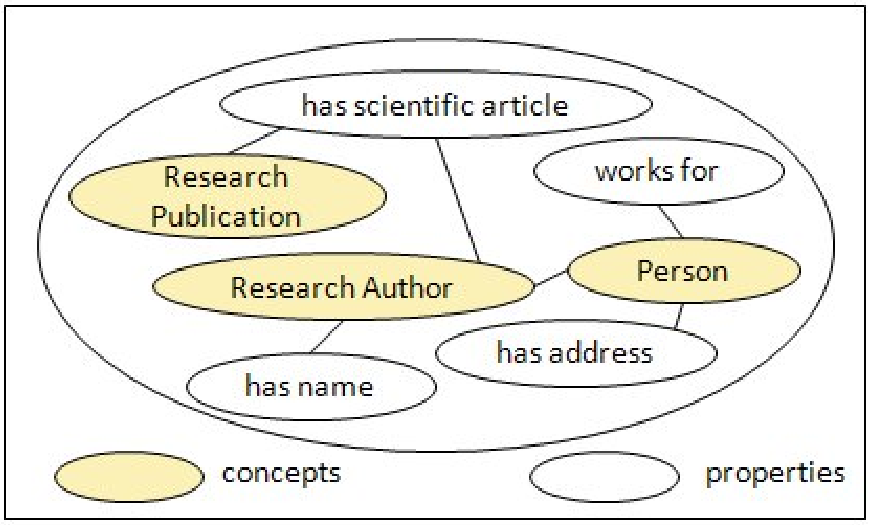


Figure 4.4: Context for the element Research Author

$$LI(Person) = (person, noun)$$
$$LI(Research) = (research, noun)$$
$$LI(Author) = (author, noun)$$
$$LI(scientific) = (scientific, adjective)$$

Table 4.2: Example of lexical information of selected labels of example Figure 4.4

w	$sense(LI(w))$
person	$\{person\#n1, individual\#n1, someone\#n1, ...\}$, $\{person\#n2\}$, ...
research	$\{research\#n1\}$, $\{research\#n2, ...\}$
author	$\{author\#n1, writer\#n1\}$, $\{author\#n2, generator\#n3, ...\}$
scientific	$\{scientific\#a1\}$, $\{scientific\#a2\}$

Table 4.3: Example of senses for some words in the context

For the example, the lexical information obtained for some of the context elements is presented in Table 4.2. This lexical information allows to search the possible senses of each of these words in the reference ontology, some of them are presented in Table 4.3.

4.3.3 Disambiguating Word Senses

The meaning of an element in the ontology can be stated by using one or more senses from the reference ontology, and these senses can be obtained via the words appearing in the label of the element. Each of the senses of a word will receive a score, denoting the relative importance it has for specifying the meaning of the word. For computing the score of the word senses, the element and its context are considered as a lexical cohesive set, and relation, distance and type weights will be combined to compute a measure of the influences between them.

Such a lexical cohesive set is known from computer linguistics (98) and means that the text used to describe the element uses related vocabulary, which makes it possible to detect relations between the words in the text. Making an analogy to the *modeling perspective* approach, an assumption in this work is that the same holds in the ontology and specially in a context.

There are two important factors to disambiguate words in a lexical cohesive text as stated in (58, 72, 127), which are the distance between two words and the relation existing between these two words.

4.3.3.1 Determining the distance weight

For the computation of the distance and relation weights it is assumed that the text is lexically cohesive, describing at least one idea using a vocabulary which is related. If the described idea changes, the surrounding vocabulary used to describe it also changes. Therefore, for determining the meaning of a certain word in the text only closely related words to the analyzed word should be used. This is the reason why the influence of words which are closer to the analyzed word will be weighted higher than other words further away. This can be done by using a weighting based on the distance (127) or by directly using the distance (58) for the disambiguation.

In this thesis, the context of an element is analogously interpreted as a cohesive construct, so the factors of distance and relation strength are adapted to this scenario for the disambiguation of the meanings of the words in the labels. Following this analogy, the distance between words in labels of elements is interpreted as the distance between those elements in the ontology graph.

Definition 14 *Given an element e and its context $ctx(e)$, the distance weight of element e' on e, $e' \in ctx(e)$ is defined as*

$$wDist(e, e') = \alpha * 2^{\frac{log_2(\frac{\alpha-1}{\alpha})}{r+1} * dist(e,e')} + (1 - \alpha) \text{ with } \alpha > 1.$$

The distance weight will be used to weight the influence of the meaning of one element on another's meaning, by considering how far or close the two elements are and the parameter α is used to determine how fast the distance influence should decrease with an increase in the distance. A value of $alpha = 1.1$, which was found to give satisfactory results in performed experiments, is used for the examples following.

4.3.3.2 Determining the relation weight

In addition to the distance between two words, the other identified factor is the relation between them. Different relation strengths between words or their senses were identified in (72), depending on the type of relation holding between them. Relations can be (72) *very strong*, between repetitions of the same word; *strong*, between synonyms and antonyms; and middle strong, if super-subclass or part-of relations hold. Additionally, the work presented in (88) states that it is possible to detect if two words are related

Relation Strength	Relation Type
very strong	synonyms
strong	hyper-, hypo-, anto-, holo-, and meronyms
weak	through the gloss gl_s

Table 4.4: Relation Strength

by comparing their descriptions (i.e. a sentence describing them). In this case, the higher the number of common words found in the respective descriptions is, the higher is the similarity of the two analyzed words. Also (153) and (63) use the description of words obtained from a lexicon to compare words, claiming that the word is related with the other words used to describe it. This approach also uses the descriptions as an additional way to compare words, but, as already pointed out in (88, 153) descriptions contain not only related words, but also general common words used to build them. These different types of words were already identified in e.g. (21), where relevant words are called *signal*s while not relevant words are called *noise*. In order to avoid that the noise affect negatively the disambiguation proposed here, the relation found over the gloss is defined as a *weak* a relation.

Following the proceeding discussions about relations and their strengths, different weights will be assigned to each different relation type which can be obtained from the reference ontology (see Definition 11) and through the gloss. The relation strength is defined here as presented in Table 4.4.

Definition 15 *The relation weights are defined as weights depending on the type of relation holding between two word senses (see Table 4.4). Weights are assigned accordingly to the strength of the relation identified in (72) and extended with the relation in the description or gloss (88). Stronger relations will end up giving more influence on the meaning than weaker relations:*

$$wRel(rel_i(s, s')) = \begin{cases} h1 & \textit{very strong} \\ h2 & \textit{strong} \\ h3 & \textit{weak} \\ 0 & \textit{no relation} \end{cases}$$

$$h1 > h2 > h3$$

Where relation weights are only considered for senses of different words, that is $s \in S_w, s' \in S_{w'}$ *with* $w \neq w'$*, in order to avoid comparing one word with itself.*

A weak relation, defined over the gloss gl_s of s, is defined whenever a sense s'' of any noun word in gl_s equals s'.

The values of $h1 = 1.0$, $h2 = 0.75$, and $h3 = 0.1$ showed satisfactory results in the experiments performed, and they will be used in the following examples.

4.3.3.3 Determining the type weight

The type weight takes into account the type (concept or property) of an element influencing another. Many guidelines for building ontologies (106, 131) state that nouns should be used whenever possible to name concepts, and verbs (often together with nouns) for labeling properties. In some cases, a simple sentence can be composed by observing the label of a property together with the label of the concept in its domain and the concept in its range (as e.g. from the introduced example in Figure 4.3: *Person works for Organization*). In such a case, the concepts will form the subject and object of the simple sentence, and the property the verb "gluing" them together. This way of naming can be exploited to weight higher concepts than properties, thus giving indirectly more importance to nouns (subject) as to verbs in the "sentence".

An additional distinction is performed when considering the influence of properties on concepts. If the noun is in the domain concept it usually represents the subject, while the noun in the range concept represents the object of the simple "sentence". Here a higher importance of the property to the subject than to the object concept is given, because it represents an aspect of it. If the element under consideration is a concept and it is in the range of a property whose influence is to be computed, this is called an incoming property, but if the concept is in the domain of the property, then it is not incoming.

Definition 16 *The element weight $wElement(e, e')$ determines the weight of an element $e' \in ctx(e)$ given an element e and the type of e':*

$$wElement(e, e') = \begin{cases} g1 & e' \in C \\ g2 & e' \in R \wedge \neg incoming(e, e') \\ g3 & e' \in R \wedge incoming(e, e') \end{cases}$$

$$g1 > g2 > g3$$

Given two elements $e, e' \in ctx(e)$ where $e' \in R$, e' is considered to be incoming for e if the following holds: $incoming(e, e') := \exists c \in C, \exists p \in R \wedge c, p \in ctx(e) \wedge c \in range(p) \wedge dist(e, c) < dist(e, p) \wedge (dist(e, e') = dist(e, c) + dist(c, p) + dist(p, e'))$

$person\#n1$	human	$\{human\#n1, man\#n4, ...\}$
$person\#n2$	body	$\{body\#n1, ...\}$, ... , $\{body\#n11\}$
$author\#n1$	article	$\{article\#n1\}$, $\{article\#n4\}$, ...
$author\#n2$	someone	$\{someone\#n1, person\#1, ...\}$
...		

Table 4.5: Senses of nouns in the glosses

The values of $g1 = 1.0, g2 = 0.75$ and $g3 = 0.5$ showed satisfactory results in the experiments performed, and they will be used in the following for the examples presented.

4.3.3.4 Computing the word sense score

The distance weight *wDist* and the element weight *wElement* are defined based on the elements in the ontology and not on the words, so, to be applicable, those weights need to be transported through the labels to their composing words.

Definition 17 *The distance and element weights of an element $e' \in ctx(e)$ are inherited by the senses of the labels of the element e', this is*
$wDist'(s, s') = wDist(e, e')$*, and* $wElement'(s, s') = wElement(e, e')$
given that $w \in Wl_e \wedge w' \in Wl_{e'} \wedge s \in S_w \wedge s' \in S_{w'}$.

Following the example presented in Figure 4.4 depicting a context with $r = 2$, the weights for some elements or senses in the context are exemplified in Table 4.6. As presented, the computation of weak relations is defined over relations of senses of nouns in the gloss, some of them are exemplified in Table 4.5.

In order to compute the score of a specific sense s of the context center element, the influence of any sense s' of the context on sense s needs to be defined first. For this the evidences found by analyzing the distance, the relation holding and the type of element are combined as presented in Definition 18.

Definition 18 *The influence of a sense s' on sense s is defined as:*
$$influence(s, s') = \sum_{i=0}^{n} wRel(rel_i(s, s')) * wDist'(s, s') * wElement'(s, s')$$

Following the example and assuming the weights presented in Table 4.6, the influence of the senses in the context on each of the senses of the context center element can be exemplified as presented in Table 4.7.

$wDist(Research\ Author, Person)$	$= 0.39$
$wDist(Research\ Author, has\ scientific\ article)$	$= 0.39$
$wDist(Research\ Author, Research\ Publication)$	$= 0.12$
$wElement(Research\ Author, Person)$	$= g1 = 1.0$
$wElement(Research\ Author, has\ scientific\ article)$	$= g2 = 0.75$
$wElement(Research\ Publication, has\ scientific\ article)$	$= g3 = 0.5$
$wRel(author\#n1, person\#n1)$	$= h2 = 0.75$
$wRel(author\#n2, person\#n2)$	$= h3 = 0.1$
$wRel(research\#n2, science\#a2)$	$= h2 = 0.75$
...	

Table 4.6: Example weights

$$influence(author\#n1, person\#n1) = 0.39 * 1.0 * 0.75 = 0.29$$
$$influence(author\#n2, person\#n2) = 0.39 * 0.75 * 0.1 = 0.03$$
$$influence(research\#n2, science\#n2) = 0.39 * 0.75 * 0.75 = 0.21$$

Table 4.7: Influence example

Before continuing, the set of words in the context, called context words, have to be defined:

Definition 19 *Context words $\mathcal{W}$ are the words of the labels of the elements appearing in a context (including the center element).*

$\mathcal{W}(e) = \{w \mid \exists e' \in ctx(e) : w \in Wl_{e'}\}$

Using the above defined influence of a sense on another sense in the context, the definition of the measure of the senses can be specified.

Definition 20 *Given the context words $\mathcal{W}(e)$ of a context $ctx(e)$ of element e, the measure of word sense score of word $w \in Wl_e$ and $s \in S_w$ can be determined as:*

$$word\_sense\_score(w, s) = \frac{\sum\limits_{w' \in \mathcal{W}(e) \setminus \{w\}} \sum\limits_{s' \in S_{w'}} influence(s, s')}{\sum\limits_{s^* \in S_w} \sum\limits_{w' \in \mathcal{W}(e) \setminus \{w\}} \sum\limits_{s' \in S_{w'}} influence(s^*, s')}$$

In case the denominator above is zero, the score is computed by giving all senses an equal measure as follows

$$word\_sense\_score(w, s) = \frac{1}{|sense(sense^{-1}(s))|}$$

As it can be seen, for computing the relative score of one sense s of one word $w \in Wl_e$, the influence on s of all senses in the context are accumulated, and divided by the sum of the influences of all senses of all context words on all senses of w in order to normalize the result.

Considering the restricted example presented, and assuming the influences depicted in Table 4.7 are the only influences in the context, the *word_sense_score* are presented in Table 4.8. The reader should remember when looking at the obtained values that only a subset of the information available in the context has been used to compute this example.

$$word\_sense\_score(author, author\#n1) = \frac{0.29}{0.29+0.03} = 0.90$$
$$word\_sense\_score(author, author\#n2) = \frac{0.03}{0.29+0.03} = 0.10$$
$$word\_sense\_score(research, research\#n2) = \frac{0.21}{0.21} = 1.0$$

Table 4.8: Example word sense scores

The word sense score function will be used later for computing the relative importance of a given sense for determining the meaning of an element, and senses with $word\_sense\_score(w, s) = 0$ are discarded since there is no detected contribution to the meaning of the word.

4.3.4 Representing the Meaning with a Logical Formula

The representation of the meaning of the elements using a logic formula allows to employ a reasoner to try to determine the relation existing to another element. Such a process of finding the relation holding between two elements can be improved by adding background knowledge (e.g. a sub-super class hierarchy of meanings or senses of words) in this process.

In order to represent the meaning of an ontological element $e \in O$ as a description logics expression, a simple description logics formula is created similarly as in (159). This is done based on the senses obtained from the reference ontology. First, the form of any logical formula using senses is defined:

Definition 21 *A logic concept formula $\mathcal{DL}_{\mathcal{RO}}$ for representing the meaning of an ontology element, using senses S from $\mathcal{RO}$ is defined as a formula where the atomic logic concepts A_s represent senses $s \in S$. Logic concepts in $\mathcal{DL}_{\mathcal{RO}}$ are constructed as presented in Table 4.9 and as defined in Chapter 2 in (7).*

A concept $C_{\mathcal{DL}}$ represents meaning using conjunctions and disjunctions of senses. In this way the meaning is not anymore just the simple collection of senses of the words

$C_{\mathfrak{DL}}, D_{\mathfrak{DL}} \longrightarrow$

$\top$	(upper concept, tautology)
$\bot$	(bottom concept, contradiction)
A_s	(atomic concept, sense from the reference ontology $\mathcal{RO}$)
$C_{\mathfrak{DL}} \sqcap D_{\mathfrak{DL}}$	(conjunction, all of them hold)
$C_{\mathfrak{DL}} \sqcup D_{\mathfrak{DL}}$	(disjunction, any of them hold)

Table 4.9: Logic concept formula

in the label, but a logical concept which expresses how these senses are combined to express the intended meaning. Such concepts $C_{\mathfrak{DL}}$ will be used for representing the meaning of each word in the labels, and then combined and used to represent the meaning of the elements.

4.3.4.1 Meaning of words as a logical formula

The meaning of each of the words in the label of elements will be represented as a concept following Definition 22:

Definition 22 *A word-concept is defined as a (possibly) non-atomic $\mathcal{DL}_{\mathcal{RO}}$ concept, which represents the meaning of a word in the label of an ontological element. It is composed by the disjunction of the lexical senses of the word having a word_sense_score measure higher than zero.*

Following the example for the element "Research Author" with the influence of each word sense (see Table 4.8), the corresponding *word-concepts* would be represented by combining atomic concepts (see Definition 21), for "author" as $author\#n1 \sqcup author\#n2$ and "researcher" as $researcher\#n2$ since those have a *word_sense_score* higher than zero.

4.3.4.2 Meaning of elements as a logical formula

Linguistic relations and dependencies between the words in the label can be used as a basis for determining how the corresponding *word-concepts* should be combined (i.e. through conjunctions or disjunctions). So called *head-modifier* relations known from computer linguistics are a good choice for this, since they are useful for specifying which is are the main words (head) representing the most relevant aspects, and which are the words which qualify or specify them (modifiers). Head-modifier relations have been widely used, e.g. to make summaries of texts (74, 82). The work presented in (74) for example describes a system to make summaries out of texts by using head-modifier relations. The system for finding head-modifier relations in a text presented in (82) was employed in this thesis, giving a head-modifier relation tree as output of its analysis. An example showing head-modifiers will be presented later in Section 4.3.4.3.

Definition 23 *A head-modifier relation tree for a label is a T_{HM} tree where heads and their modifiers are organized in a tree via the head-modifier relation. The leaves of T_{HM} are the words in the label, internal "and" nodes represent the head modifier relations forming a head expression, while internal "or" nodes represent the relation existing between two head expressions.*

Definition 24 *A parse function $pf : l_e \rightarrow T_{HM}$ is a function which creates a head-modifier relation tree T_{HM} for the label l_e of element e.*

Based on the head-modifier relations the *word-concepts* are related through conjunctions and through disjunctions. The combination of the *word-concepts* following the relations determined in the head-modifier tree will allow to represent the meaning of the element in an *element-concept.* In order to do so, first each head and its modifiers have to be combined, to later combine these expressions.

A head can be combined with its modifiers into a logical representation which will be called *head-concept.* Since the modifiers of a head qualify or specify the head, all of them restrict its interpretation and therefore a *head-concept* is formed by the conjunction of the *word-concept* of the head and all of the *word-concepts* of its modifiers.

Different *head-concepts* corresponding to the label of one element extend the interpretation of the corresponding element, and therefore the *head-concepts* are combined by disjunctions to form the *element-concept.*

Definition 25 *A logic concept building function $cbf : T_{HM} \rightarrow C_{\mathcal{DL}}$ is a function which builds a $\mathcal{DL}_{\mathcal{RO}}$ concept out of a head-modifier relation tree, where leaves in T_{HM} are converted to word-concepts. Word-concepts participating in one head-modifier relation are combined through conjunctions defined by the corresponding internal "and" nodes in T_{HM}, creating a head-concept. Different head-concepts are combined through disjunctions as defined by the corresponding internal "or" nodes in T_{HM}, giving at the end a $\mathcal{DL}_{\mathcal{RO}}$ logic concept formula expressing the meaning of the label of an element denoted as the element-concept. Such an element-concept logic concept formula for an ontological element e will be denoted as $C_{\mathcal{DL}}(e)$.*

4.3.4.3 Example

A short example representing a head-modifier tree for the label "Email or Postal Address" is presented in Figure 4.5. Assuming that the word "Email" has one sense $email\#n1$, the word "Postal" has one sense $postal\#n1$, and the word "address" has senses $address\#n1$ and $address\#n2$ with score higher than zero, the *head-concepts* will be represented as $email\#n1 \sqcap (address\#n1 \sqcup address\#n2)$ and $postal\#n1 \sqcap (address\#n1 \sqcup address\#n2)$ while the *element-concept* will be represented as $(email\#n1 \sqcap (address\#n1 \sqcup address\#n2)) \sqcup (postal\#n1 \sqcap (address\#n1 \sqcup address\#n2))$, and therefore as $(email\#n1 \sqcup postal\#n1) \sqcap (address\#n1 \sqcup address\#n2)$.

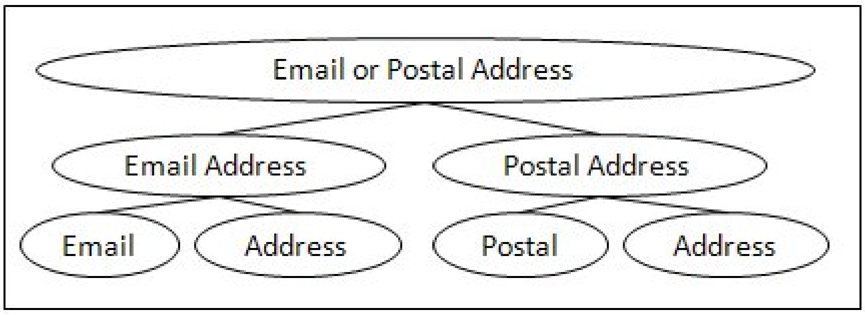


Figure 4.5: Example of a head-modifier tree

4.3.5 Score of Element Senses

Once the expression of the meaning of an element in the ontology has been specified by building an *element-concept* using Definition 25, the relative importance of each of the used senses for specifying the meaning of the described ontological element has to be computed. This relative importance of a sense will be given again as a score for each sense, an *element sense score* between $[0, 1]$, where the element sense scores for one *element-concept* sum up to one.

The word sense score has already been computed. It represents the relative importance of a sense for specifying the meaning of a word. Differing from what was presented in (80, 134, 135), the score of the senses of a word will be weighted according to the role of the word in the label (head or modifier) and will be combined to obtain a unique score for each sense describing the meaning of the element. So, first the weight of the senses at the *head-concepts* of each element will be computed, and then the final scores at the *element-concept* level. An example of these computations is presented in Table 4.10.

For computing the score of a sense at the element level, first the head-modifier word sequences have to be considered. The score of each sense in a *word-concept* detected in the label of an element is weighted depending on it belonging a head *word-concept* or to a modifier *word-concept* in a given *head-concept.* Since heads represent the essence of *head-concepts*, and modifiers specialize it, the weight will boost the senses of the head word and lower the impact of the senses of the modifier words. For computing this, each *head-concept* is considered in isolation.

Definition 26 *The modified sense score depending on a sense appearing in a word being the head or a modifier in one head-concept is defined as:*

$$word\_sense\_score'(head\text{-}concept, w, s) = word\_sense\_score(w, s) \times \lambda$$

$$where\ \lambda = \begin{cases} \gamma & if\ w\ is\ a\ head\ word \\ 2-\gamma & if\ w\ is\ a\ modifier\ word \end{cases} \quad 1 < \gamma < 2\ .$$

The experience factor γ is used for boosting scores of senses appearing in head words. The boosted/lowered partial weights computed by the $word\_sense\_score'(head\text{-}concept, w, s)$ function will be normalized back later, as part of Definition 27.

Using the modified sense score as presented in Definition 26, the score of a sense in a *head-concept* will be computed as presented in Definition 27.

Definition 27 *The score of the senses is weighted depending on if the word is in the head or in the modifier. This weighted sense score in a head-concept is defined as:*

$$head\text{-}concept\_sense\_score(head\text{-}concept, s_i) = \frac{\sum\limits_{w \in head\text{-}concept} word\_sense\_score'(w, s_i)}{\sum\limits_{\substack{s \in w \\ w \in head\text{-}concept}} word\_sense\_score'(w, s)}$$

The *head-concept_sense_score* gives a normalized sense measure which includes a weighting factor for senses belonging to head or modifier words, which now can be accumulated across all head-modifiers in the label l_e of the element to compute the *element sense score.* This denotes the relative importance of one sense for the whole element. This score can be computed by summing all *head-concept_sense_score(s)* for a given sense s in all head-modifiers (i.e. in all *head-concept*), and then dividing this sum by the number of *head-concepts* (i.e. the number of head-modifiers which build the element's definition).

Definition 28 *The element sense score, denoting the score of a sense in an element can be specified as:*

$$ess(e, s) = \frac{\sum\limits_{head\text{-}concept \in l_e} head\text{-}concept\_sense\_score(head\text{-}concept, s)}{number\ of\ head\text{-}concept \in l_e}$$

$$
\begin{array}{l}
word\_sense\_score'(author, author\#n1) = 0.90 \times 1.2 = 1.08 \\
word\_sense\_score'(author, author\#n2) = 0.10 \times 0.8 = 0.08 \\
word\_sense\_score'(research, research\#n2) = 1.0 \times 0.8 = 0.8 \\
\hline
head\text{-}concept\_sense\_score(Research\ Author, author\#n1) = 1.2 \div 2.08 = 0.58 \\
head\text{-}concept\_sense\_score(Research\ Author, author\#n2) = 0.08 \div 2.08 = 0.04 \\
head\text{-}concept\_sense\_score(Research\ Author, research\#n2) = 0.8 \div 2.08 = 0.38 \\
\hline
ess(Research\ Author, author\#n1) = 0.58 \\
ess(Research\ Author, author\#n2) = 0.04 \\
ess(Research\ Author, research\#n2) = 0.38
\end{array}
$$

Table 4.10: Example element sense score computation

The *element sense score* measure, gives a score for the importance of the different senses appearing in the element, and will be employed later in the comparison of elements in different ontologies. This score relies on the *modeling perspective* and on the function of the word in the label. It gives a measure of the importance of each of the detected senses for specifying the meaning of the element. This score can be regarded as the relative size of the sense in an element, and this interpretation will be used later to compare elements and to approximate the semantic relation between two elements.

The value of $\gamma = 1.2$ was observed to give satisfactory results, and therefore it will be used in the following examples. Considering the element "Research Author" and the word sense scores presented in Table 4.8, the scores for determining the element scores are presented in Table 4.10. It can be seen that some of the senses have a much higher value than others, for example $author\#n1$ in comparison with $author\#n2$. Referring to WordNet, the reference ontology used for the examples, and the examples presented so far, the reader can see that the meaning associated with $author\#n1$ matches closer the intended meaning of the *Research Author* concept (someone writing articles) than the meaning represented with $author\#n2$ (someone causing or initiating something).

The computation of the meaning of elements, as well as the individual scores or measures have been presented. Now it is possible to know the senses which define the meaning of an element, which can also be represented as a logic concept. The next step is to compare elements in different ontologies based on this logic representation of the meaning and on the senses and their scores.

4.4 Find Matching Elements in Different Ontologies

In this section the elements in different ontologies will be compared in order to find the relation holding between them (if any), and a measure of the confidence of how similar they are will be computed. Finding matching elements between ontology O^u and ontology O^{S_i} can become a very expensive task if the ontologies are big. Because of this, the detection of the matching elements, finding the relation holding between them, and the computation of the similarity is done on the basis of a candidate set. Having a candidate set computed using cheaper methods which do not include the full analysis presented in the previous sections, helps in narrowing the search space, improving in this way the performance.

Matching elements have a semantic relation holding between them as well as a similarity value. The semantic relation can be computed based on the logic formula as presented in Section 4.3.4 and including background knowledge from the reference ontology by using a reasoning approach (satisfiability). This is done in this work based on the CTXMatch (15, 16, 18, 159) approach. In cases where this reasoning approach does not give satisfactory results, the semantic relation will be approximated by comparing the senses of the elements and their scores (also interpreted as relevance or size for determining the meaning). The similarity between two elements is composed of two factors, the similarity of the elements and the similarity of their *modeling perspective*, which are combined to give the similarity confidence of the two elements.

In more detail, the main steps involved in finding matching elements in different ontologies are:

- Determining matching candidate sets for elements to be matched.
- Determining via reasoning the semantic relation holding between elements to be matched and elements in their candidate sets.
- Computing a similarity measure of candidate matches by analyzing their associated senses and their *modeling perspective* similarity.
- Approximating the semantic relation holding between elements using their associated senses and their score.
- Detecting possible semantic relations of unmapped elements based on structural comparisons.
- Computing the resulting semantic relation between two elements.

The details of the steps presented above will be explained in the following sections.

4.4.1 Determining the Candidate Sets

The candidate set of potentially matching elements from O^{S_i} to one given element in O^u is composed by elements which show some evidence for being related, thus, the analysis space is reduced as well as the efforts needed for the computations. Candidate elements are those, which are related to the senses of the element to be matched. In order to find which senses of the considered elements are related (through the words in the labels) the idea of the *semantic neighborhood* used in (140) is considered.

Definition 29 *The semantic neighborhood of a sense is defined as the set of senses which are related to it by considering synonym, hypernym and holonym relations.*
$SN_s = \{s^* \in S \mid \exists rel_i(s, s^*) : wRel(rel_i(s, s*)) \geq h2\}$

In this way by using the semantic neighborhood and the senses therein, the words having those senses can be obtained from the reference ontology. So, if any of the words derived from the semantic neighborhood of an element $e \in O^u$ appears in the label of an element $e' \in O^{S_i}$, then there is possibly a relation between the meanings of the two elements and e' is added to the set of candidate elements of e. The set of candidate elements for e will be denoted as $Candidate_e$.

4.4.2 Determining the Semantic Relation

The determination of the semantic relation holding between two elements is based on the *element-concept* $\mathcal{DL}_{\mathcal{RO}}$ description of them. It is based on the satisfiability approach defined in the CTXMatch approach (15, 16, 18, 159) with the difference that CTXMatch targets at finding matches of elements in classification hierarchies, while this work aims at find elements matching in different ontologies, which makes the process of constructing the logical representation of elements different because of the possible richness in ontologies. Another difference is the type of background knowledge available in hierarchies in comparison to the one available in ontologies which can include much more relation types between elements. The computation of the semantic relation between two of the defined *element-concepts* can be solved by using a reasoner, posting the task as a satisfiability proof. Starting point for the definition of the background knowledge is to have computed the *element-concept* for all involved elements, i.e. the element to be matched and all elements in the corresponding candidate set. These *element-concepts* are all added to the reasoner. Additionally, other senses (represented as atomic logic concepts) from the reference ontology can be added to the reasoner as background knowledge for each of the involved *element-concepts*, such as super/sub-class relations or equivalence/disjunction relations to other senses in the reference ontology. Since here only satisfiability is checked, and there are no quantified relations (roles), sub-/super-classes are enough for this computations.

Definition 30 *The knowledge base KB is built by the logical concepts (derived from the senses) from the reference ontology used to specify the meaning of the ontological elements to be compared, the concepts of the elements in the context or* modeling perspective*, the relations existing between the senses, as well as all related sub-/super-classes of the employed senses:*

$$
\begin{aligned}
KB := \{ \quad & C_{\mathfrak{DL}}(e) \mid \forall e \in O^u \ ; \\
& C_{\mathfrak{DL}}(e') \mid \forall e \in O^u \wedge \forall e' \in Candidate_e \ ; \\
& C_{\mathfrak{DL}}(e') \sqsubseteq C_{\mathfrak{DL}}(e'') \mid \exists e', e'' \in O^u : e' \leq e''; \\
& C_{\mathfrak{DL}}(e') \sqsupseteq C_{\mathfrak{DL}}(e'') \mid \exists e', e'' \in O^u : e' \geq e''; \\
& C_{\mathfrak{DL}}(e') \equiv C_{\mathfrak{DL}}(e'') \mid \exists e', e'' \in O^u : e', e'' \ equivalent; \\
& C_{\mathfrak{DL}}(e') \sqcap C_{\mathfrak{DL}}(e'') = \perp \mid \exists e', e'' \in O^u : e', e'' \ disjoint; \\
& A_s \sqsubseteq A_{s'} \mid \forall e \in O^u \wedge A_s \in C_{\mathfrak{DL}}(e) \wedge \exists s' \in S : s, s' \ hyponyms; \\
& A_s \sqsupseteq A_{s'} \mid \forall e \in O^u \wedge A_s \in C_{\mathfrak{DL}}(e) \wedge \exists s' \in S : s, s' \ hypernyms; \\
& A_s \equiv A_{s'} \mid \forall e \in O^u \wedge A_s \in C_{\mathfrak{DL}}(e) \wedge \exists s' \in S : s, s' \ synonyms; \\
& A_s \sqcap A_{s'} \mid \forall e \in O^u \wedge A_s \in C_{\mathfrak{DL}}(e) \wedge \exists s' \in S : s, s' \ antonyms;
\end{aligned}
$$

Having defined the KB, finding the relations between the elements can be defined as a satisfiability problem on KB.

Definition 31 *The satisfiability problem for detecting a semantic relation between two elements is:*

$$
\begin{aligned}
KB &\implies C_{\mathfrak{DL}}(e) \equiv C_{\mathfrak{DL}}(e') \\
KB &\implies C_{\mathfrak{DL}}(e) \sqsubseteq C_{\mathfrak{DL}}(e') \\
KB &\implies C_{\mathfrak{DL}}(e) \sqsupseteq C_{\mathfrak{DL}}(e') \\
KB &\implies C_{\mathfrak{DL}}(e) \sqcap C_{\mathfrak{DL}}(e') = \perp
\end{aligned}
$$

In this way a reasoner can be feed as presented in Definition 30 and the relations (see Definition 31) between two elements can be asked. By doing this, and if there is enough evidence, the reasoner gives an answer of the relation existing between the $C_{\mathfrak{DL}}$ concepts, and therefore between the ontological elements for which they stand.

In addition to the relation holding between the elements, the similarity between them can be computed by analyzing the scores of the senses following a set-based approach which compares the overlap of the senses and their relative score (which can be interpreted as relative size) which will be presented in the next section.

4.4.3 Computing the Element Similarity

The element similarity is a measure of how similar elements are, considering their disambiguated senses and the relative size or score of each of them. It goes in the direction of considering each element a set of senses where each sense has a different importance or size to define the meaning of the element. Additionally, the contexts where the elements appear are compared, comparing in this way the respective *modeling perspective*, which helps in having more accurate measures of their intended usage and meanings.

4.4.3.1 Comparing two elements

There are cases where the reasoning approach for computing the relation between two elements cannot give an answer. This happens e.g. when there is not enough knowledge available to derive one of the considered relations. In order to cope with these cases, and also to have a measure of how similar two given elements are, the approach presented in this section computes the similarity of elements based on how much common or related meaning they have. This approach builds on the ideas presented in (73), where the attempt to compute the degree of generalization or specialization between two concepts is made based on the set theoretic structure of a taxonomy (an element is composed by all its subsumed elements). For this purpose the intersection of sets is used, corresponding to elements in taxonomies or ontologies. The larger the size of the intersection, the higher is the evidence of being in a sub-super class relation.

In this thesis the score of each sense is interpreted as its set size when comparing two elements. These sizes are employed for computing a sort of intersection of meaning between elements.

It must be recalled that a logical concept formula $C_{\mathfrak{DL}}(e)$ representing the meaning of an ontological element e is composed by atomic concepts A_s, and these A_s stand for senses s from the reference ontology (see Definition 21). In the following definitions and for easing the notation, whenever $s \in C_{\mathfrak{DL}}(e)$ is presented it means a sense s from which A_s has been created, and with $A_s \in C_{\mathfrak{DL}}(e)$.

Definition 32 *The sense overlap between senses s, s' of two ontological elements e, e', where $s \in C_{\mathfrak{DL}}(e)$ and $s' \in D_{\mathfrak{DL}}(e')$ is defined as:*

$$s\_o(s, s') = \begin{cases} min\,\{ess(e, s), ess(e', s')\} & \textit{if } s = s' \\ 0 & \textit{if } s \neq s' \end{cases}$$

Where $ess(e, s)$ denotes the relative size interpretation of the importance of a given sense for specifying the meaning of an element (see Definition 28).

The *sense overlap* defines a measure of the overlap of senses, considering cases where the senses are the same, but without considering cases where the senses are not exactly the same but closely related. Since this should be considered as well, a *sense similarity overlap* is defined which considers relations between the senses.

Definition 33 *The sense similarity overlap between two senses s, s' of two ontological elements e, e', where $s \in C_{\mathfrak{DL}}(e)$ and $s' \in D_{\mathfrak{DL}}(e')$ given a relation $rel(s, s') \in REL_{(s,s')}$ is defined as the size of the intersection multiplied by a relationship factor f:*

$$s\_o_{sim}(s, s', rel(s, s')) = \begin{cases} min\,\{ess(e, s), ess(e', s')\} * f & \textit{if } wRel(rel(s, s') \geq h2 \\ 0 & \textit{else} \end{cases}$$

$$f = \begin{cases} 1 & \textit{if } s \textit{ is synonym of } s' \\ \sigma & \textit{if } s \textit{ is hypernym or hyponym of } s' \end{cases}$$

It has to be noticed that for the computation of the *sense similarity overlap* the factor correlates with the strength of the relation, synonyms (the strongest relation after identity) have the strongest factor. This reflects that (i) even if different words are used in the label, if they were detected to have synonym senses describing their meanings they will be considered with the strongest factor (which is a simplification since it is known that synonyms might have slightly different meanings), (ii) any sense s which has the hypernym relation with a sense s' is more specific than the meaning of s', restricting its interpretation, and (iii) the other way around for hyponyms. Therefore synonyms are reflected by a stonger factor than the other two types of relationships.

Since there might be more than one relation between two senses s and s' and all of them should contribute to the *similarity overlap* (see Definition 34), an iterative algorithm for its computation is presented in Algorithm 4.1. In this computation the strongest relation holding between senses is taken at each iteration and the results are accumulated, discarding the relations already considered in previous iterations until no relations are left.

Definition 34 *The similarity overlap $sim\_o(s, s')$ is defined as the addition of the sense similarity overlap between senses s and s' considering all relations as described in Algorithm 4.1.*

Algorithm 4.1 for computing the *similarity overlap* of senses s, s' of elements e, e'

```
1: $sim\_o \leftarrow 0.$
2: $REL \leftarrow REL_{(s,s')}$
3: while $| REL | > 0$ do
4:     $rel \leftarrow rel_i(s, s') \mid \exists i \forall j, wRel(rel_i(s, s')) \geq wRel(rel_j(s, s')), 1 \leq i, j \leq | REL |$
5:     $sim\_o \leftarrow sim\_o + s\_o_{sim}(s, s', rel)$
6:     $REL \leftarrow REL - \{rel\}$
7: end while
8: return $sim\_o$
```

Based on the two sense overlap and the similarity overlap definitions presented in Definition 32 and Definition 34, the element overlap can be computed by combining these two overlap measures.

Definition 35 *The element overlap of two element-concepts is defined as the additions of senses and similarities overlaps, and is computed as:*

$$elm\_overlap(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) = \sum_{s \in C_{\mathfrak{DL}}(e), s' \in D_{\mathfrak{DL}}(e')} (s\_o(s, s') + sim\_o(s, s'))$$

It is this $elm\_overlap(C, D)$ function which is used to compare two elements and to obtain a measure of how similar they are, by considering the elements and their senses. In the next step, the modeling perspectives of the two elements need to be compared, so that also a measure is obtained for showing how similar the corresponding perspectives are.

4.4.3.2 Comparing the modeling perspectives

It was presented before how the *modeling perspective* is captured, by using the context of the element, i.e. its surrounding elements in the ontology. For comparing the modeling perspectives of two elements, their contexts are compared without considering the two elements themselves. Since the task is to find elements in a sources' ontology which match elements in the user ontology, an asymmetric approach is used here. This approach is oriented to the user ontology, i.e. the elements in the context in the user ontology are used to look for similarities. This means that, the information in the user ontology is used to determine the similarity, and if the ontology describing a source has more elements in the context as the context from the user ontology, this does not cause the similarity of the perspective to be lower. Since elements in the context of the element to be matched can have similarities with more than one element in the compared context, only the one with the higher similarity is taken into account for the comparison of the contexts. This is done in this way, to avoid that one element having many similar elements in the other context influences the results too much, providing misleading results. Additionally, modeling differences are also taken into account. If for example, one element is a concept in one context and has as its most similar element a property in the other context, this implies a modeling difference which is also considered when comparing the modeling perspectives.

The comparison of the contexts requires all elements in both contexts to have their meanings specified as *element-concept*. In the first step, the most similar element in the target context is found for each element in the user context.

Definition 36 *Given two contexts $ctx(e')$ and $ctx(e'')$ where $e' \in O^u$ and $e'' \in O^{S_i}$, the most similar element $MSE(e) \in ctx(e'')$ for element $e \in ctx(e')$ is defined using the elm_overlap and computed as presented in Algorithm 4.2.*

Algorithm 4.2 for computing the *MSE*

```
1: $max\_ext_overlap \leftarrow 0$
2: $e^* = null$
3: for all $e_i \in ctx(e'')$ do
4:     $ext\_overlap \leftarrow ext\_overlap(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e_i))$
5:     if $ext\_overlap > max\_ext\_overlap$ then
6:         $ext\_overlap \leftarrow max\_ext\_overlap$
7:         $e^* \leftarrow e_i$
8:     end if
9: end for
10: return $e^*$
```

Definition 37 *Given contexts $ctx(e')$ and $ctx(e'')$ where $e' \in O^u$ and $e'' \in O^{S_i}$, context elements e, e^* where $e \in ctx(e')$ and $e^* \in ctx(e'')$ and $e^* = MSE(e)$, and given the distance and element weights $dist(e', e)$, $dist(e'', e^*)$, and $wElement(e', e)$, $wElement(e'', e^*)$, the modeling differences of e and e^* are defined as:*

$$wDiff_{Dist}(e, e^*) = 2^{-|dist(e',e)-dist(e'',e^*)|} \textit{ and}$$

$$wDiff_{Element}(e, e^*) = \begin{cases} 1 & \textit{if } wElement(e', e) = wElement(e'', e^*) \\ \upsilon & \textit{if } wElement(e', e) \neq wElement(e'', e^*) \end{cases}$$

The modeling differences can be seen as the degree of deviation of the different modeling perspectives. So, the similarity of the elements show how much similar they are, the differences punish the deviations in the modeling, and their combination give a measure of how much each element contributes to the similarity of the contexts.

Definition 38 *The contribution of an element e^* to the similarity of two contexts $ctx(e')$ and $ctx(e'')$, where $e^* \in ctx(e'')$ and $e^* = MSE(e)$ is defined as follows:*
$contribution(e^*) = elm\_overlap(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e^*)) * wDiff_{Dist}(e, e*) * wDiff_{Element}(e, e^*)$

It has to be noticed that for the computation of the similarities of two contexts, a kind of recall computation is performed, where more similar and less similar elements are considered, in contrast to relevant or not relevant. Another difference is that each element which contributes to the result (because it is similar) does not contribute with 1, but with a contribution dependent on this similarity degree.

Definition 39 *The similarity of two modeling perspectives, represented by contexts $ctx(e')$ and $ctx(e'')$, is computed by the mean value of the sum of the contributions of the most similar elements, and is defined as:*
$ctxSim(ctx(e'), ctx(e'')) = \frac{\sum\limits_{e \in ctx(e') \setminus \{e'\}} contribution(MSE(e))}{|ctx(e') \setminus \{e'\}|}$
In case $ctx(e') \setminus \{e'\}$ does not have elements, the similarity of the contexts is defined as 1 (there is no context for the element to be matched).

If there is no context for the element to be matched, any context matches (complete recall), so, the similarity is set to 1. This is due to the asymmetric comparison of contexts. Differently as done in (80), the center element is not considered in *ctxSim*, since this is already considered in *elm_overlap*.

It has to be mentioned that, due to the computation process of the similarity of two contexts, the similarity value can result in a very small value. Even a small value here means there is a similarity, and such a small value should not be discarded without further consideration.

4.4.3.3 Element similarity measure

The measure of the similarity of two elements in different ontologies, under consideration of the existing local element similarity, as well as the context or *modeling perspective* similarity is defined (in addition to what was presented in (80)) as the combination of those measures.

Definition 40 *Given two elements $e \in O^u$ and $e' \in O^{S_i}$, the measure of the similarity of e and e′ given the local comparison of the possible senses, as well as the context or* modeling perspective *where they are located in the corresponding ontologies is defined as:*

$$sim(e,e') = min\{elm\_overlap(e,e'), ctxSim(ctx(e), ctx(e'))\} + \frac{|elm\_overlap(e,e') - ctxSim(ctx(e),ctx(e'))|}{2}$$

The overlap of the elements and the similarity of the respective contexts are combined in a lineal way. There is no need to additionally weight these factors because all considerations regarding the importance of the different employed measures have been added throughout the computations.

4.4.4 Approximating the Semantic Relation Using Senses

In cases where the reasoning process is not able to give an answer, or, in cases where there is the need to compare two different ways of obtaining the semantic relation, an approximation of the relation can be performed. The presented approximation is based on the idea that an element whose meaning is represented using more general senses than senses describing the meaning of another element will probably imply that this relation holds also for the compared elements. Since this holds only up to a certain point, in addition to determining the semantic relation, a measure of the confidence of this approximated relation is computed. The computation is based again on the scores for the different senses, since they provide a good measure of how much of the meaning corresponds to a more or less general sense. Three sets are required for this computation, the equivalence overlap with the similarity of the synonyms; the generalization set, which contains the hypernyms of the senses of the considered element; and the specialization set, containing the hyponyms of the considered element.

Definition 41 *Equivalence overlap $overlap_{\equiv}$ between two senses s, s' of two elements e, e', $s \in C_{\mathfrak{DL}}(e)$ and $s' \in D_{\mathfrak{DL}}(e')$ is defined as:*

$$overlap_{\equiv}(s,s') = \begin{cases} min\,\{ess(e,s), ess(e',s')\} & \textit{if } s = s' \textit{ or } s \textit{ is synonym of } s' \\ 0 & \textit{if } s \neq s' \textit{ and } s \textit{ is not synonym of } s' \end{cases}$$

Definition 42 *The generalization overlap $overlap_{\sqsupseteq}$ between two senses s, s' of two elements e, e', $s \in C_{\mathfrak{DL}}(e)$ and $s' \in D_{\mathfrak{DL}}(e')$ is defined as:*

$$overlap_{\sqsupseteq}(s,s') = \begin{cases} min\,\{ess(e,s), ess(e',s')\} & \textit{if } s \textit{ is hypernym of } s' \\ 0 & \textit{else} \end{cases}$$

Definition 43 *The specialization overlap $overlap_{\sqsubseteq}$ between two senses s, s' of two elements e, e', $s \in C_{\mathfrak{DL}}(e)$ and $s' \in D_{\mathfrak{DL}}(e')$ is defined as:*

$$overlap_{\sqsubseteq}(s,s') = \begin{cases} min\,\{ess(e,s), ess(e',s')\} & \textit{if } s \textit{ is hyponym of } s' \\ 0 & \textit{else} \end{cases}$$

The overlap degree is then defined for each of the three measures as follows:

Definition 44 *The equivalence overlap degree is defined as the addition of the equivalence overlaps as:*

$$degree_{\equiv}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) = \sum_{s \in C_{\mathfrak{DL}}(e), s' \in D_{\mathfrak{DL}}(e')} overlap_{\equiv}(s,s')$$

Definition 45 *The generalization degree is defined as the addition of the generalization and equivalence overlaps as:*

$$degree_{\sqsupseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) = \sum_{s \in C_{\mathfrak{DL}}(e), s' \in D_{\mathfrak{DL}}(e')} overlap_{\sqsupseteq}(s,s') + overlap_{\equiv}(s,s')$$

Definition 46 *The specialization degree is defined as the addition of the specialization and equivalence overlaps as:*

$$degree_{\sqsubseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) = \sum_{s \in C_{\mathfrak{DL}}(e), s' \in D_{\mathfrak{DL}}(e')} overlap_{\sqsubseteq}(s,s') + overlap_{\equiv}(s,s')$$

The semantic relation existing between the two elements e and e' can now be approximated by looking at the highest of the above presented degrees. If all degrees are equally sized (equivalence, generalization and specialization) no semantic relation can be identified.

Definition 47

$$SemRel = \begin{cases} \text{“equivalent”} & \textit{if } degree_{\equiv}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) = 1 \\ \text{“more general”} & \textit{if } degree_{\sqsupseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) < degree_{\sqsubseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) \\ \text{“more specific”} & \textit{if } degree_{\sqsupseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) > degree_{\sqsubseteq}(C_{\mathfrak{DL}}(e), D_{\mathfrak{DL}}(e')) \end{cases}$$

Where a measure of the confidence of the approximation is given by the corresponding highest degree.

4.4.5 Structure-based Detection of the Semantic Relation

In a similar approach to the one presented in (107) or (97) (and extending (80)), earlier detected matches are used for deducing other matches by taking into account the structural information from the respective ontologies. If an element $e \in O^u$ without a match is detected in the is-a hierarchy between two other elements in the ontology O^u which do have matching elements in O^{S_i}, and if there is a non-matched element $e' \in O^{S_i}$ in the same relative hierarchical position, then it can be deduced that it is likely to be a relation between this two elements. Figure 4.6 depicts an example where the the existence of a relation between element B and element B' is deduced based on the fact that there is a detected relation between elements A and A' and C and C'. For this cases only the existence of a relation is stated, but not which kind of semantic relation it is.

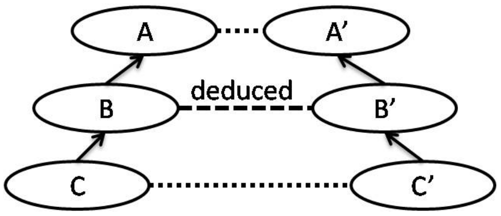


Figure 4.6: Structure-based relation detection

4.4.6 Element Relations Computation

For computing the final relation produced by this approach, the relations obtained via the SAT approach, the sense-set approximation, and the evidence of a relation coming from the structure-based approximation are combined. If relations coincide the result is trivial and the same result stays, if conflicts occur then the degrees obtained for approximating the relation are considered. If relations are approximated with a high enough threshold, then the sense-set approximation result is taken, if the approximation threshold is not high enough, then the SAT-based result is considered as the result. In cases where none of the results have a confidence which is high enough, neither the SAT-based approach nor the sense-set approximation, the relation is left unspecified and only the “related” relation type is returned as the output.

4.5 Experiments and Evaluations

The experiments performed aim at detecting the strengths and weaknesses of the matching approach presented in this chapter. For this evaluation, the experiments use an existing evaluation framework proposed by the Ontology Alignment Evaluation Initiative (48) (OAEI), which presents a framework for systematically evaluating ontology alignment approaches. In the following some implementation details, as well as details of the evaluation framework will be described. In addition to the results using OAEI 2005, the overall results of running the OAEI 2009 (124) version of the benchmark are presented, as well as a small hands on experience which was performed in order to gain a feeling of the advantages of using the approach compared to a plain search in a search engine specialized for ontologies.

4.5.1 Testbed

The matching approach presented in this chapter aims at finding matches between selected elements in the user ontology O^u and elements in a source ontology O^{S_i}, under consideration of the modeling perspective as presented before. This approach has been implemented as a Java prototype. Detailed class and packages architecture, as well as other implementation details of its first implementation are presented in (80). The approach was modified and extended, and several parts of it re-implemented as described in this chapter and in (135).

The software prototype allows to select an ontology from the local disk and displays it in a graph layout structure by using the JGraph (`www.jgraph.com`) library. The user can then select a set of concepts for which the user is interested to have a mapping. The labels of these selected elements are extracted, tokenized and lemmatized using the TreeTagger (142), and, the meaning disambiguation approach presented in this chapter is carried out.

The reference ontology employed is WordNet (50). WordNet (96), is considered one of the biggest and most successful resources for lexical information. Such a lexical resource groups words with the same meaning in named sets, each of them denoting a sense. So, a word having different meanings will be found in different sets, and synonyms will be found in one set representing one meaning or sense. The relations between senses are lexical and semantic relations. Examples of such relations are synonym, antonym, part-of relations, etc. WordNet makes a difference between lexical and semantic relations (96), where lexical relations are relations between word forms, and semantic relations between the associated senses. For this work all of them were considered as *lexical relations* (because they come from the lexical resource). In this setting, the lexical resource can be seen as a valid reference ontology. There exist many other resources of this type providing lexical information about words such as GermaNet (71) for German language or MultiWordNet (113) which combines several languages.

The HeadModifier expressions were built using the EP4IR Dependency Parser (83). The framework used to handle the ontologies was Jena (28) in its version 2.4, and the reasoning for determining the semantic relations was performed using Racer (116).

Available ontologies, preselected based on synonyms of the selected concepts, are analyzed in detail for finding mappings between the selected elements and the elements in those ontologies. The pool of available ontologies can be a set of ontologies in a local directory, or an ontology search engine such as Swoogle (36, 42), which is integrated in this implementation.

The matching approach is carried out on the available ontologies, and matches for the selected elements are computed and presented to the user. Screenshots exemplifying some of the different parts of this prototype are presented in Figure 4.7, where the user ontology and some selected concepts are shown. In Figure 4.8 the details of parts of a matching ontology are presented, with some highlighted related elements.

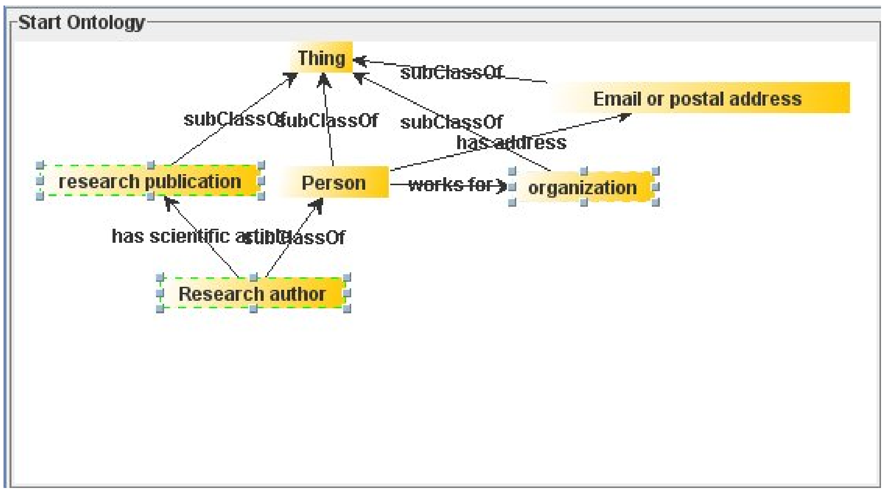


Figure 4.7: User ontology example with selected concepts

4.5.2 Ontology Alignment Evaluation Initiative

The Ontology Alignment Contest was introduced in 2004 by Jérôme Euzenat from INRIA (`http://www.inrialpes.fr`) as a way to test and compare different alignment algorithms. The goal was to have a common way to evaluate such algorithms, in order to better know their strengths and weaknesses. In 2005 the EON evolved and changed its name to the Ontology Alignment Evaluation Initiative (OAEI).

The evaluation (46) consists of a set of OWL (148) ontologies to be compared and matched with one OWL reference ontology, whereas for all of these comparisons the same configuration of the algorithm should be employed. Together with the ontologies, reference alignments are provided, as well an API to interact with an output format specified by the benchmark. The provided reference alignments, the ground truth,

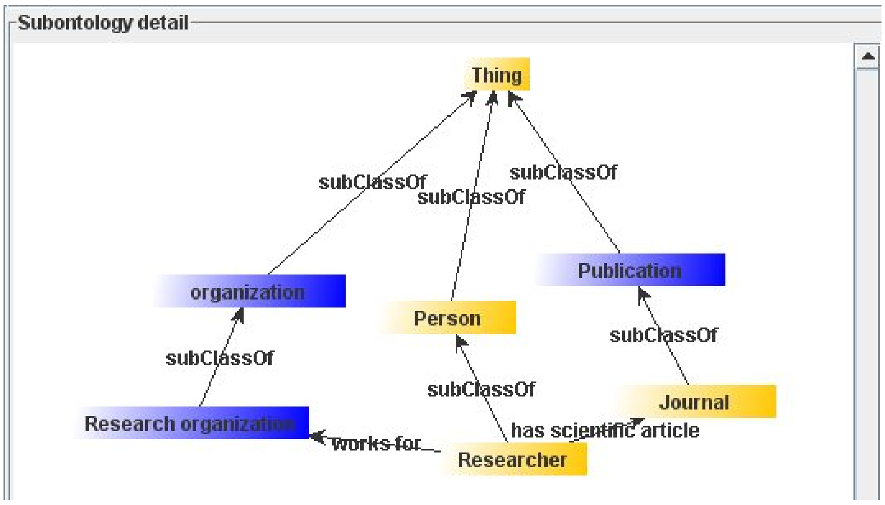


Figure 4.8: Part of an ontology having matching elements

expresses pairs of mapped elements, a confidence value, and a relation holding for the respective mappings. Most of the provided results have the highest confidence value possible (1), and the equivalence relation holds for the mapping, although some few results include the subsumption relation. It is worth mentioning here that OAEI aims at evaluating approaches finding matchings for all elements in the ontology, regardless of its specific usage or of the differences in modeling between the two compared ontologies.

The given ontologies are mostly about one field, bibliographic references. Three testing sets of ontologies are provided, which are (48):

- simple tests (Test numbers 1xx), comparing the reference ontology with itself, with another irrelevant ontology, or the same ontology in a restricted representation of OWL.

- systematic tests (Test numbers 2xx), obtained by discarding some features of the ontologies, leaving the other features untouched. The considered features were (names, comments, hierarchy, instances, relations, restrictions, etc.). This approach aimed at recognizing what kind of information is exploited by different algorithms, showing where their strengths and weaknesses are. In addition, ontologies discarding combinations of the above are available, leading to a better characterization of different scenarios.

- "real life" ontologies (Test numbers 3xx) for bibliographic references which were collected from the web and were left mostly untouched (modifications made were only for making them readable by the different software existing for managing ontologies, such as Jena (28)).

The details about which information has been removed for which test in the *systematic tests* can be found in (48).

The reference ontology provided by OAEI 2005 contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals, and 20 anonymous individuals. This reference ontology was put in the context of the Semantic Web by including other, external, resources for representing non-bibliographic information, using the FOAF (`http://xmlns.com/foaf/0.1/`) and the iCalendar(`http://www.w3.org/2002/12/cal/`) ontologies for expressing People, Organization and Event concepts. The available tests enable a variety of different comparisons, trying to represent very different types of ontologies to make clear where different approaches perform better.

For executing the experiments all elements in the reference ontology are analyzed for a matching element in each of the test ontologies. The execution of the experiments was performed automatically, by loading one of the target matching ontologies after the other, and at the end writing the summary of the precision and recall results obtained by each of these comparisons, as well as computing an overall quality measure over all results.

In addition to OAEI 2005, the overall results of running the OAEI 2009 (124) version of the benchmark (110) are also presented. The OAEI 2009 version of the benchmark is structured in a similar way, as the 2005 version. The same groups and types of modifications hold, but the authors claim that some ontologies with errors have been corrected (e.g. circular references removed), and that the tests have increased difficulty. Since the same type of modifications have been performed for the different test groups (1xx, 2xx, and 3xx), the details of the benchmark OAEI 2009 will not be further described here. The interested reader can find all details and results in (108, 110, 124).

4.5.3 Evaluation parameters

The evaluation of the matching approach presented in this chapter employed the OAEI evaluation set. This allows to detect where the strengths and the weaknesses of the approach are. Several parameters required for this approach have to be specified first, before being able to perform the evaluation, and the results presented here will show the best results obtained for several tries with different parameter configurations.

The configuration of the parameters used are presented in Table 4.11.

For the discussion of the results, the benchmark tests are classified in categories accordingly to the characteristics of the employed ontologies (48) and presented in the

Parameter Name	Parameter Description	Value
r	Radius for the contexts in O^u and O^{S_i}	2
α	Parameter for $wDist$)	1.1
$h1$	$wRel$ weight for very strong relations	1.0
$h2$	$wRel$ weight for strong relations	0.75
$h1$	$wRel$ weight for weak relations	0.1
$g1$	$wElement$ weight for concepts	1.0
$g2$	$wElement$ weight for not incoming properties	0.75
$g3$	$wElement$ weight for incoming properties	0.50
σ	Similarity overlap factor f value (Definition 33)	0.75
υ	Modeling difference distance factor value for different elements (Definition 37)	0.5
γ	Boosting factor for senses in the head of a head-modifier expression (Definition 26)	1.2
-	Threshold value for considering two elements as similar (Definition 35)	0.75
-	Threshold value for considering two contexts as similar (Definition 39)	0.25

Table 4.11: Parameter settings for the evaluation

following sections. This results in 5 categories, tests 101-104, tests 201-210, tests 221-247, tests 248-266, and tests 301-304. Tests 248-266 were not included in the evaluation because they aim at finding matchings between some meaningful English label and some random string (e.g. "c56ui2"). Finding such matching elements is something which is not only not supported by this approach, but also not desired at all, since an ontology engineer trying to find some ontology to be used as an inspiration to extend his/hers ontology would not have any advantage in receiving such a result. Precision and recall results were computed by comparing the obtained results with the golden standard as described in the evaluation benchmark suite guidelines.

4.5.4 Test Results 101 to 104

The results of this group of tests can be found in Table 4.12. Results of Test 102 comparing the ontology provided in Test 101 with another ontology of a complete different domain show that as expected there were no mappings found by this approach. It can be observed that in Test 104 modifications in the employed language in the names of elements affect the capability of detecting matches, due to limitations of the employed reference ontology (WordNet (50)).

Test	Precision	Recall	Description
101	1.0	1.0	Reference alignment
102	-	-	Irrelevant ontology
103	1.0	1.0	Language generalization
104	0.99	0.97	Language restriction

Table 4.12: Results of tests 101 to 104

4.5.5 Test Results 201 to 210

These tests include the modification of the information attached to elements, removing labels of elements or comments attached to them, translating them to languages other than English, or replacing them by synonyms. Table 4.13 presents the results obtained for this category of tests, showing expected worst results whenever names are removed, or translated to other languages. The very low results in test 202 are caused by the cheap method (simple term search on names) employed to build the set of candidate matches, which is used as a basis for later comparison (presented in Section 4.4.1). Since the candidate matches cannot be detected, no further comparisons are possible. On the opposite, Test 203 shows good results, since the comments do not play a fundamental role in this approach, so removing them does not have a great negative impact. Names translated to other languages cannot be found in the reference ontology, so a simple fall-back representation for their meaning is used, which is the cause for low recall results in Tests 206-207, and 209-210.

Test	Precision	Recall	Description
201	0.83	0.25	No names
202	0.00	0.00	No names, comments removed
203	0.95	0.92	No comments
204	0.98	0.95	Different naming
205	0.85	0.42	Synonym replacements
206	0.92	0.23	Translation complete
207	0.88	0.23	Translation label and identifiers
208	0.94	0.86	Different naming, comments removed
209	0.72	0.21	Synonym replacements, comments removed
210	0.88	0.23	Translation label and identifiers, comments removed

Table 4.13: Results of tests 201 to 210

4.5.6 Test Results 221-247

This group of tests show very clearly one important aspect of this approach, which is the usage of the modeling perspective as a means for detecting matching elements. As it can be clearly seen in Table 4.14, Tests 228, 233, 236, and 239-247 where properties have been removed present lower precision since the properties are important in determining the modeling perspective of concepts. The lack or modification of hierarchies as in Tests 233, and 239-247 in combination with the removal of the properties impacts also the obtained precision. The hierarchy is also considered as an important factor, as can be seen in Tests 222, 230, 239, 240-247, since it is also captured as an aspect determining the modeling perspective. Since this approach does not consider instances at all, its removal does not have any influences on the results as can be seen in Test 224.

4.5.7 Test Results 301-304

The results of this last group of tests presented in Table 4.15 show again that this approach considers the modeling perspective also as an important aspect to be considered for deciding on a match or not. If the ontologies are modeled differently, and elements are used in different ways, then this approach fulfills its goal by concluding that there

Test	Precision	Recall	Description
221	0.98	0.90	No specialization
222	0.93	0.91	Flattened hierarchy
223	0.95	0.94	Expanse hierarchy
224	0.99	0.97	No instances
225	0.99	0.98	No restrictions
228	0.87	0.85	No properties
230	0.85	0.87	Flattened classes
231	0.99	0.97	Expanse classes
232	0.98	0.90	No hierarchy, no instances
233	0.89	0.76	No hierarchy, no properties
236	0.87	0.85	No instances, no properties
237	0.94	0.92	Flattened hierarchy, no instances
238	0.95	0.94	Expanse hierarchy, no instances
239	0.93	0.90	Flattened hierarchy, no properties
240	0.79	0.67	Expanse hierarchy, no properties
241	0.89	0.76	Removed hierarchy, instances and properties
246	0.93	0.90	Flattened hierarchy, no instances nor properties
247	0.79	0.67	Expanse hierarchy, no instances nor properties

Table 4.14: Results of tests 221-247

should not match. Furthermore, these tests employ "external" ontologies containing a variety of additional elements modeled differently, and not only modifications of the ontology used in Test 101 as performed in the presented Tests 103-247.

Test	Precision	Recall	Description
301	0.85	0.54	BibTeX/MIT
302	0.76	0.33	BibTeX/UMBC
303	0.49	0.39	Karlsruhe
304	0.83	0.75	INRIA

Table 4.15: Results of tests 301-304

4.5.8 OAEI Results Overview

Finally, an overview of the mean precision and recall values of this approach, in comparison with other approaches participating in OAEI 2005 are presented in Table 4.16 and Table 4.17 (only the results of the above presented test have been considered for computing this overview). The *MODPERSP* entry corresponds to the approach presented in this chapter.

Approach:	EDNA	FALCON	FOAM	CTXMATCH	DUBLIN
Test	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.
1xx	0.96 1.0	1.00 1.00	0.65 0.65	0.87 0.34	1.00 0.99
2xx	0.64 0.90	0.97 0.99	0.93 0.91	0.72 0.44	0.97 0.97
3xx	0.48 0.80	0.92 0.81	0.91 0.68	0.22 0.06	0.65 0.59

Table 4.16: Comparison with results of OAEI 2005 - part 1

Approach:	CMS	OMAP	OLA	MODPERSP
Test	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.
1xx	0.74 0.30	0.96 1.00	1.00 1.00	1.0 0.99
2xx	0.62 0.31	0.62 0.31	0.62 0.31	0.88 0.71
3xx	0.90 0.39	0.88 0.93	0.93 0.93	0.73 0.50

Table 4.17: Comparison with results of OAEI 2005 - part 2

It has to be noticed again that the purpose of the approach presented in this chapter is not to find all matching elements regardless its usage but to find only those which are employed in a similar way. The other approaches presented in Table 4.16 and Table 4.17 aim at finding all matching elements, regardless of the element's usage. Therefore, care should be taken when comparing the obtained values. Nevertheless, finding elements which have the same intent is one mayor goal of this approach, and having such mappings is something which is required by the query reformulation approach presented in Chapter 5. Furthermore, experiments using the 2009 version of the OAEI benchmark (110) have been performed and its overview presented in Table 4.18, Table 4.19, and Table 4.20 (mean values computed considering the tests presented above and the preliminary results available in (108)). As mentioned before, the OAEI 2009 benchmark is structured similarly as the OAEI 2005 benchmark and the tests have similar characteristics. Therefore, the detailed results presented previously also apply.

Approach:	EDNA	AFLOOD	AGRMAKER	AROMA	ASMOV
Test	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.
1xx	0.96 1.0	1.00 1.00	0.98 0.98	1.0 1.0	1.00 1.0
2xx	0.63 0.85	0.98 0.96	0.98 0.89	0.97 0.94	0.99 0.99
3xx	0.47 0.80	0.90 0.78	0.92 0.77	0.83 0.76	0.79 0.80

Table 4.18: Comparison with results of OAEI 2009 - part 1

Approach:	MAPPSO	RIMOM	SOBOM	TAXOMAP	MODPERSP
Test	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.
1xx	1.00 1.00	1.00 1.00	0.98 0.97	1.00 0.34	0.98 0.98
2xx	0.95 0.95	0.99 0.98	0.97 0.84	0.85 0.46	0.90 0.74
3xx	0.53 0.24	0.80 0.80	0.86 0.51	0.77 0.30	0.71 0.51

Table 4.19: Comparison with results of OAEI 2009 - part 2

Approach:	DSSIM	GEROME	KOSIMAP	LILY
Test	Prec. Rec.	Prec. Rec.	Prec. Rec.	Prec. Rec.
1xx	1.00 1.00	1.00 1.00	0.99 0.99	1.00 1.00
2xx	0.98 0.88	0.96 0.96	0.93 0.87	0.99 0.98
3xx	0.92 0.65	0.70 0.58	0.69 0.47	0.82 0.79

Table 4.20: Comparison with results of OAEI 2009 - part 3

4.5.9 Hands-on Empirical Experience

In addition to the evaluation using the OAEI benchmarks, a hands-on experience has been performed and evidences collected. This was done by employing the presented approach for searching for ontologies which could be used to extend user ontologies. The following areas have been considered:

- Ontologies containing concepts: Persons, Projects and Organizations.
- Ontologies containing concepts: Publications, Authors, Persons.
- Ontologies containing concepts: Body, Artery, Heart, Human.
- Ontologies containing concepts: Wine, Drink, Grape.

For each of these areas different user ontologies have been constructed, and the prototype presented in this chapter has been used for searching for candidates to extend these ontologies. A total of seven user ontologies were employed, and the number of elements searched for varied between 2 and 4. In addition, the Swoogle (36) ontology search engine has been used to search for ontologies containing such elements, to obtain a baseline for comparison. Since it is very subjective and therefore difficult to measure automatically which result has a higher modeling perspective similarity, a simple heuristic measure has been computed based on three evidences:

1. The number of matching elements vs. the number of elements searched for,
2. The number of matching elements having the same type as the ones used for searching, and
3. The modeling similarity, by taking for properties their domains and ranges, and for concepts the properties they carry into account.

The arithmetic mean of these evidences has been computed as a measure of the *Modeling Perspective Similarity.*

Since both Swoogle (42) and the results of this approach return ranked results, the modeling perspective similarity for the Top-1, Top-2, and Top-3 obtained results has been computed and are presented in Figure 4.9.

Figure 4.9 shows that the approach presented in this chapter returns in the first places of the result list more satisfying results than a search in a specialized ontology search engine. These results give a hint of the applicability and feasibility of employing this approach for easing the way of finding ontologies which are modeled similarly.

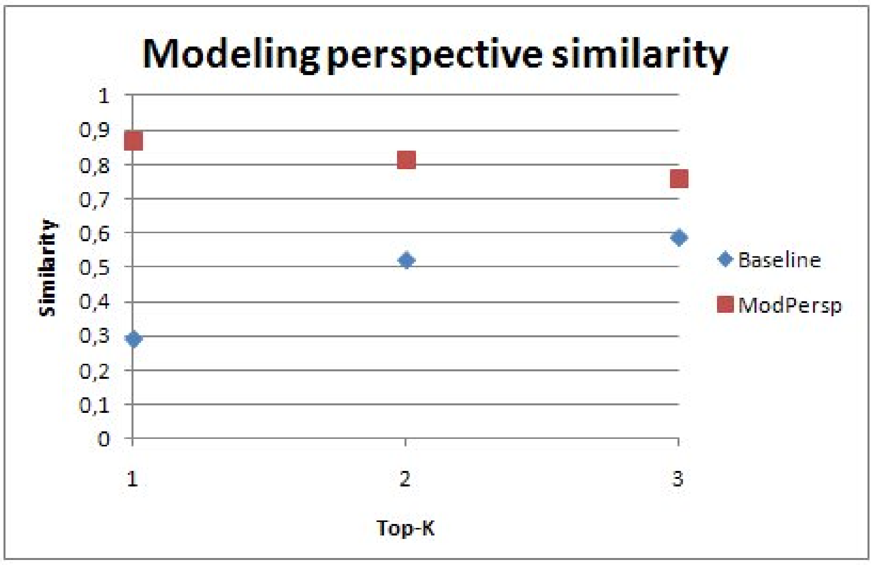


Figure 4.9: Modeling perspective similarity of results obtained from the baseline Swoogle and the approach presented in this chapter.

4.5.10 Experiment Conclusions

The experiments shows that the presented approach performs acceptably good (superseding the results obtained in (80)) in a variety of cases compared with the results of other benchmarked approaches which are mostly tailored to achieve good results in the benchmark. As already mentioned, the lower recall which can be observed in some cases is caused by the pre-selection process used, to avoid that all selected elements in the user ontology need to be compared to all elements in the candidate ontology (presented in Section 4.4.1). This pre-selection is based on the senses detected for elements, so, if the labels of elements do not carry meaningful English words, they are not considered in the matching process. Although there are specialized approaches with higher results in specific cases, the general (mean) result show that this approach is performant and flexible enough to find the expected matches. The high precision obtained in most of the cases shows that the elements for which matchings are found are correct, and this is important for enabling the approach presented in Chapter 5.

The performed hands on experience (Section 4.5.9) presents a hint that the proposed approach is better than a specialized search engine to detect ontologies modeled similarly to the ones used in the search, which is an evidence that the claim of capturing the *modeling perspective* is achieved as expected.

In this chapter an approach for finding matching elements in different ontologies was presented. This approach differs from other existing approaches by taking into account the *modeling perspective* of the ontologies, and taking the decision of a matching or not by considering this *modeling perspective*. The experiments show good precision results for the cases where lexical information is available, as required by the approach. An interesting extension would be to also include instances in the computation of the matching, which has proven to be a good source of evidences for matching. In the scenario considered in this thesis the instances are not available at the moment of the computation of the mapping, but are employed later to learn new mappings as part of the process described in Chapter 5.

5

Reformulating Queries to Query Different Sources

This chapter addresses the problem of reformulating a query formulated following one ontology (O^u) to be used for querying an information source S_i which is described by another ontology O^{S_i}. Challenges presented in Section 2.4 are dealt with, and the sketched approach presented in Section 2.5 is described in detail.

The query reformulation approach (132, 133) presented in this chapter presents a reformulation of the formulated user query Q^u by using existing partial mappings M_i between ontologies, and strategies to deal with the un-mapped elements (for which no mapping exist) so that information source S_i can be queried. The approach aims at finding in a pay-as-you-go fashion all possible answers, and, by relying on feedback, to learn and give better results in successive similar queries on S_i.

Before presenting the details of the approach in Section 5.2, the related work for the research area will be described.

5.1 Related Work on Query Reformulation

This section presents approaches used to rewrite and to relax queries in order to execute them over sources having different structures, this is, to handle heterogeneity of sources. The related work related to rewriting queries will be presented first, followed by approaches related to relaxing queries, and finalizing the section with a discussion.

5.1.1 Query Rewriting

Rewriting of queries means to use other terms (e.g. another vocabulary or formalism) to express the query, such that the query intent is not modified, i.e. the semantics are preserved as much as possible. Rewriting of queries involves also to consider different spans for value restrictions in queries, e.g. when querying a range of temperature data in Celsius or Fahrenheit. Rewriting can be of different kinds, among others a) using syntactical rewritings, where the vocabulary in the query is replaced without further modifications; b) involving the translation into another data model with possibly different complexity, as it is performed e.g. in (158); c) it can involve splitting the original query in subqueries by analyzing the sources and their contents as presented e.g. in (6, 118); or d) employing an ontology, taking advantage of the inferencing capabilities offered by such formalisms for rewriting and also relaxing the original query as e.g. in (136). There are many different approaches using and proposing query rewriting, some of them will be described in the following.

One of the goals of the SIMS project (6) was to provide access to heterogeneous and distributed information sources. SIMS relies on a domain model whose elements are used to refer to elements describing the contributing sources. The mappings between the sources and the domain model ensure to have a minimal span, i.e. mappings are complete and not redundant, assuring that all information can be accessed by employing the mappings. In its query reformulation component (6), several semantics-preserving reformulation operators are introduced, in order to reformulate query terms from the domain model to terms appearing in the information sources. Queries are expressed using Loom (90). The semantics-preserving reformulation operators allow to select a given information source for executing parts of a given query, to decompose a query in several subqueries (e.g. when the required information is distributed among several sources), etc. Queries are reformulated according to source-selection rules, so that the best reformulation is chosen out of many possible reformulations, and, at the end of the reformulation process a query which contains only source elements is obtained. It might be that after a reformulation process the system notices that a query cannot be answered, so the process ends, else, the different query plans are executed on the different sources to compute the results of the query.

An overview of the field of answering queries using views can be found in the survey (68), where different approaches to query rewriting are presented. In this overview, the problem of distributed information integration is presented as an extreme case of answering queries using views, where different sources can provide partial answers to the query. Generally, there is the physical storage, then there are one or more views representing the stored data (e.g. precomputing aggregations of the data at the physical storage), and there is the query, and it is not trivial to decide which resource to use for answering the query most efficiently, the storage, the precomputed views, or a combination of both, but it is known that the answers for a query come from data in one physical system. This becomes harder in the distributed information integration scenario, where there can be many different sources, increasing the number of views. In such settings, and as pointed out in (68), the views on the sources can cover all information required in the query or can cover only a subset of the query, providing in the latter case partial query answers.

In (118), the authors present an approach to answer conjunctive queries using views on different sources, so that the original conjunctive query is reformulated to fit the part of the information covered by a given view. It is assumed to have query templates (kind of predefined queries) where attribute values are fed for parameterizing it. These templates are based on the global representation of the available information spectrum to be queried. In addition, elements appearing in the query, also understood as a view, are declared to need to be bound with a value before executing the query (i.e. are a required parameter for using the template), and other elements are left unbound or free for returning the query results. With this information it is possible to know in advance which elements in the query are meant for providing an answer and which ones are meant for receiving parameters (bound query elements). Underlying each of these query templates, in the query body, there is a customized program responsible of obtaining the results for this query template for a specific source (this program knows how to interact with the specific source). Query templates can be combined to provide solutions to subgoals or subqueries, in order to answer more complex queries or to create views of the different sources. In this way the actual representation used by a given source is encapsulated, being those query bodies or programs the ones responsible of collecting the actual data from the sources for a given query template or view.

In (158) an approach is presented to rewrite queries formulated using elements in a global schema, for integrating both XML and relational sources, extending work performed in the relational world. It is assumed that partial mappings relating elements in the global schema to elements in the schemata describing the sources are available, and it is this mapping information which is employed in the rewriting process. A nested relational representation is used as a common data model to represent both types of schemas. Mappings are expressed as queries, depicting the correspondences and relations between the elements in the global schema and the elements in the schema of the source. Whenever a query is posted, its elements are rewritten using the mappings, as union of queries (in case more than one source can provide data for one element). In this process the problem of having partial mappings is dealt with, by obtaining data from more than one source. In addition to the actual correspondences between elements in the mappings, also data constraints are available, which depict in which cases data can be merged once obtained (e.g. primary key equality as a criteria to merge data from two different sources). Data obtained from more than one source is then combined under consideration of the constraints, and in this way missing mappings can be overcome. A query optimization process is responsible of removing (sub) queries which are known to be unsatisfiable and to remove duplicates before query execution. Depending on the type of the source, queries (or subqueries) are translated to the target query language and executed. The queries supported by this approach for XML sources are a subset of XQuery (150) including nested queries.

Several rewriting algorithms using views containing only conjunctions exist, which can be classified to be bucket-based or based on the use of inverse rules (also used in the case where views contain disjunctions). The main idea of the inverse rule based approaches is to construct a set of inverse rules inverting the view definitions, and then rewrite the queries by composing it with this inverse rules (1, 52, 68). The bucket-based approaches collect the views which are useful for answering a relation stated in the query, and in a next step analyze all of these buckets in order to find a solution for the query. This makes the combination of possibilities to be analyzed to grow exponentially. In (8) an extension of one well known bucket-based algorithm is presented. This extension allows to use a bucket-based approach also in presence of inclusion dependencies (i.e. given relationships between different schema relations) to rewrite conjunctive queries. It aims at finding the maximal rewriting, this is, the maximum rewriting which is possible under the given mappings so that the rewritten query gives valid answers to the original query (which could be a subset of the optimal results). An extension of this bucket-based work to handle views containing disjunctions is presented in (9). Here, tuples which are in the view but are not needed for answering the query are automatically discarded for reducing the number of options to be considered, thus improving performance.

Other approaches which do not rely on views use for example the so called malleable schemas (40). The work presented in (160) is based on malleable schemas, where relations between schemas are discovered based on co-occurrences of data (attributes and values). This allows to cope with some vagueness and redundancy in the integration of all participating sources and is the basis for rewriting queries based on those relations. Since a relaxation using malleable schema might also introduce some results which are not expected as results to a query (due to this relaxation), a ranking or ordering of the obtained results is required, so that results which are *closer* to the intent of the original query are presented first to the user (160). A similar idea to ranking results of relaxed queries is presented in Chapter 6.

5.1.2 Query Relaxation

Query relaxation is the process of making the query broader, or removing restrictions from the query, so that it gives more results than the original query, e.g. more general results. In some cases of query rewriting, when the rewritten query is too restrictive, the relaxation of some constraints in the query is required for obtaining any results. Independently of being rewritten or not, an area where relaxation of queries come into play is in the case of cooperative answering systems. Such systems cooperate with the user in giving answers to queries with no results by relaxing constraints appearing in the query.

Cooperative answering of queries can be performed in different ways, providing the user with additional information to help him/her to formulate better the intended query, giving intermediate answers, or, as proposed in (57) by relaxing the posted query so that neighboring information can also be retrieved. In (57) a relaxation approach is presented, which a) is based on the explicit inclusion of user constraints to restrict the relaxation (e.g. this result is not desired); b) uses heuristics to order the relaxed queries; and c) integrates semantic information from thesauri. The goal of this relaxation strategy is to help the user to avoid posting queries which fail in giving results. The existing database constraints are used to deduce if a given query can give results at all, and are used to deduce which parts of the query need to be relaxed and in which order. The user given constraints are used to avoid some relaxation possibilities, by knowing what the user does not want to obtain as an answer, and the thesauri is used to generalize some query constraints.

Relaxing a query which failed to give results can be a tedious process, since usually many different possibilities need to be tried out before obtaining satisfying results. Therefore, in (100) an approach is presented to relax the query automatically, based on a given query and a database. The relaxation process is not precomputed, i.e. relaxation options are not defined in advance, but computed online after analyzing the constraints specified in a given query. This analysis is performed on a trained Bayesian model of the implicit relations between attributes, learned from the available sources. This Bayesian model of the attribute dependencies is then employed to analyze which constraints should be relaxed to obtain the most accurate results possible under the given circumstances.

In (84) another approach to relax queries is presented, which intends also to perform as less modifications as possible in order to obtain results for a given query which failed to give results in its original form. Precise semantics is given to the modification or relaxation of numeric conditions in a query (e.g. $salary < 10.000$); a lattice-based ordering of all possible relaxations is presented aiming at selecting the best possible combination of relaxations; and different algorithms to traverse this lattice are described, aiming at optimizing query response time.

5.1.3 Discussions

The described query rewriting approaches do not present strategies for a pay-as-you-go integration, as it is mentioned as one of the goals in this thesis. Either rewritings are performed by relying on complete mapping information, and query optimization can be partly performed at the mediator up to the point of deciding if a query will not have any answer (e.g. (6), (118)), or even having only partial mappings where it is assumed that the query will only have an answer if the combination of data from the sources gives a valid tuple for answering it (e.g. (158)). Approaches such as (160) employ rewriting in combination with relaxation, as the approach presented in this chapter, but require to inspect the instances, the data and not only its schema, to find correspondences and to decide on which restriction in the query to lessen in order to obtain results to a query, which is not required in the approach described in this chapter. Other relaxation approaches (e.g. (57), (84)) target mainly single databases or rely on user-specified constraints which need to be considered in the relaxation process, requiring user intervention.

The flexible information integration approach presented in Section 5.2 addresses the dynamic integration needs in SKS where only partial mappings between the sources to be integrated exist. In addition to exploiting defined mappings, substitution strategies, inspired by the idea of using wildcards in querying and filtering tasks, are employed for reformulating (rewriting and relaxing) queries. In addition, feedback evidences are employed for refining the existing mapping based on the results of executing the reformulated queries.

5.2 Query Reformulation Approach

In heterogeneous information integration settings as the ones motivating this work, both the user query (based on his view of the information space) and the various information sources to be queried are using ontological elements (property and concept names) based on different ontologies. An important challenge in answering queries in such settings is to translate between the ontological elements used in the query and the ontological elements available in the information source. A complete mapping would provide adequate translation rules for all ontological elements available in the user query.

However, this work targets the cases where only partial mappings are available, i.e. there can be ontological elements in the user query that cannot be translated into ontological elements of the targeted information source. There are different causes for the incompleteness of the mapping: 1) there might exist semantic mappings between the ontological elements of the query and of the information source which are not specified in the mapping (e.g. due to failures in the mapping detection process), and 2) there is no corresponding ontological element in the information source, i.e. the respective ontological element of the user query cannot be mapped to the information source. This approach focuses on dealing with case 1). For dealing with case 2) methods are required for combining results from different information sources to jointly cover the user query.

For querying an information source S_i given a user query Q_u there are several ways to deal with the situation where no mapping exists in M_i for one or several of the ontological elements used in the query. For the discussion is is assumed that the queries consists of conjunctive combinations of query predicates. Other query options such as projection and disjunctive combination of query terms are not considered in this work. The following three strategies for dealing with partial mappings have been identified:

- *Focus on precision*: replace the ontological elements in the query for which there is a mapping in M_i with the corresponding mapping element and leave the other parts of the query as originally specified. Following this case will mean that, if at least one ontological element in the query does not have a mapping available, then the query will return no results when evaluated on S_i (since a conjunctive combination of query terms is assumed). Therefore this strategy will not be further considered.

- *Focus on Recall*: remove all query predicates that contain an ontological element that cannot be mapped to an ontological element of the targeted information source. This strategy considers only elements that have a mapping. By relaxing the query (deletion of predicates) this strategy improves recall. However, it might considerably reduce precision depending on the importance and selectivity of the deleted query predicates.

- *Controlled query relaxation*: replace a selected set of the ontological elements in the user query, for which no mapping element exists in M_i, with variables. The use of variables instead of e.g. properties in triple style queries, as considered here, introduces a "wildcard" type of property: They can be potentially matched with any property in the target information source. Of course, it might also introduce incorrect answers in the results. Selecting the situations in which variables are used to replace not-mapped query ontological elements defines a trade-off between precision and recall.

In this work the *controlled query relaxation* approach presented above is followed, exploring different strategies in order to find a good balance between how much recall is gained and to which extent precision is lost. In addition, the results of this strategy will also be used to refine the existing mapping: The variable v introduced for a property p will be matched to another property, say q, in the targeted information source. In case results based on this match become part of the query result, this is an evidence for the assumption that there might be a mapping between p and q. Such evidences will be accumulated, in order to learn further mappings, or presented to the user for getting user feedback on further possible mappings (see Section 5.4). This support for the refinement of existing partial mappings is a second important feature of the lightweight information integration and query rewriting approach presented in this chapter.

Algorithm 5.1 Process overview for querying a new source in the SKS

```
1: Find out any new data source type and discover its schema.
2: Find out initial schema mappings between the user's ontology $O^u$ and the different data source's schemas
   $O^{S_i}$
3: Use the different mappings to rewrite queries posted accordingly to $O^u$
4: Query the data sources, retrieve and merge the results
5: Present the results to the users and expect feedback/refinement about the used mappings
6: Consider the user feedback to refine the mappings
7: Go to step 4 or end
```

5.3 Query Rewriting and Substitution Rules

The core of the problem tackled in this chapter is the identification of a set of substitution rules for transforming a given query Q^u into a query Q^{S_i} that can be understood by the target information source S_i. The substitution rules will be based on the existing (partial) mappings. In addition, they will also be driven by a controlled substitution of ontological (or schema) elements such as properties and concept names by variables (*wildcard substitution rules*).

Since the focus is on analyzing the effect of "wildcard" substitution on query reformulation, the following simplifying assumptions are made:

1. No syntactical transformations between source and target query are required or have already been performed in a pre-processing step.
2. The information sources to be accessed are described by simple ontologies.
3. The mappings relate only concept to concept or property to property.
4. The queries contain only selective predicates combined by conjunction.

Before discussing the substitution rules and their application to queries some required definitions are introduced.

5.3.1 Required Definitions

The definition of *Ontologies* considered this chapter is the one presented in Definition 2, and the reader is referred to it (see Section 2.4.1).

Information Sources are defined in Definition 3, where the information sources to be accessed are described by simple ontologies. A set $\mathcal{S}$ of n local sources and a set $\mathcal{OS}$ of n ontologies such that each S_i is described by O^{S_i} with an alphabet A^{S_i}, $i \leq n$ is assumed to be available. Each S_i contains the set $ext(O^{S_i})$, the extension or instances of O^{S_i}.

The *Personal Information View* of the user on the domain is assumed to also be given in form of an ontology, the personal ontology of the user O^u as defined in Definition 4. O^u reflects the user's mental organization of the data explicated as a simple ontology, and represents the vocabulary the user employs for querying the system.

Definition 48 *The alphabet of O^u is denoted as A^u and this alphabet is used by the user to construct any query Q^u, where $A^u \cap A^{S_i} = \emptyset$ for all $i \leq n$ is assumed.*

Definition 49 *Extending the query definition presented in Definition 5, individual query terms are expressed as triples $< s, p, o >$, which may contain properties (for p), references to resources (for s and o), and literals (for o) from an infinite set of literals (constants) $\mathcal{L}$. The properties and concepts used in the query are from the alphabet A^u. In addition variables can be used in the query triples in all positions to* *express connections between triples (e.g. paths), to cover irrelevant values and to name elements to be included into the query result.* VAR *will be used to denote an infinite set of variables that can be used in any query and as* VAR_{Q^u} *the set of variables actually used in query* Q^u*. The actual elements from* A^u *used in query* Q^u *will be referred as* A^{Q^u}*, i.e. the ontological elements appearing in the user query.*

Triple based query languages also contain statements that refer to the typing of the searched instances. In RDF the special property $rdf : type$ is used for this purpose. It connects the instance to the schema level (or the A-Box to the T-Box). It is assumed that the employed query language also supports such a property denoted as τ. A triple $< s, \tau, o >$ expresses the condition that s is an instance of the concept denoted by o.

As mentioned above, the work presented in this chapter focuses on the part of the query language where the query conditions are defined, and the conditions are restricted to conjunctive combinations of predicates. Thus the user query Q^u is composed of a set of triples $< s, p, o >$, with the option for s, p and o as described above.

Mappings were presented in Definition 6. Between O^u and O^{S_i} there is a mapping denoted as M_i (Chapter 4 presents a method to compute such mappings). As mentioned, the mappings are restricted to relate concept to concept or property to property. The mapping does not have to be complete, i.e. not for all ontological elements in O^u there have to be mappings in M_i. Additional semantic mappings might exist between elements of O^u and O^{S_i} which are not specified in M_i (e.g. due to failures or incompleteness in the mapping detection process).

5.3.2 Mapping-based Substitution Rules

As described before, it is assumed that a set of mappings M_i is defined between the personal information view of the user and the target information sources. These mappings are used to define a basic set of substitution rules.

The decision on which elements from M_i to use for rewriting the query Q^u to query source S_i is based on the confidence value for each pair-wise mapping in M_i and the threshold β:

Definition 50 *The mappings* M_i' *employed for rewriting the query* Q^u *to query source* S_i *is*

$$M_i' := \{(o, e, v) | \exists o, v : (o, e, v) \in M_i \land v > \beta \land o \in A^{Q^u}\}$$

The symbol β represents the threshold value used for deciding on using a given mapping or not. The confidence values can be modified based on the results of evaluating the query (feedback), so that mappings that were used in the past could cease to be used due to low confidence or non-used mappings could start to be used due to rise in their confidence (see Section 5.4).

Having defined M'_i, two first sets of substitution rules - the mapping-based substitution rules SUB_{M1} and SUB_{M2} for concepts and for properties, respectively - can be defined:

Definition 51 *The mapping-based substitution rules SUB_{M1} for concepts are*

$$SUB_{M1} := \{(o, e) | \exists s, p :< s, p, o > \in Q^u \wedge p = \tau \wedge (o, e, v) \in M'_i\}$$

Definition 52 *The mapping-based substitution rules SUB_{M2} for properties are*

$$SUB_{M2} := \{(p, e) | \exists s, o :< s, p, o > \in Q^u \wedge (p, e, v) \in M'_i\}$$

Definition 53 *The substitution rules which directly apply the available mappings are*

$$SUB_M := SUB_{M1} \cup SUB_{M2}$$

These substitution rules directly apply the mappings defined in M_i.

5.3.3 Ontology-based Substitution Rules

Ontological knowledge is a further source of substitution rules. For example, unmapped concepts and properties in the query Q^u might be replaced by super-classes and super-properties from Q^u, respectively, for which a mapping has been defined. In (24), Calvanese and colleagues present detailed approaches on how to use ontological knowledge and reasoning in query rewriting and query processing.

5.3.4 Wildcards Substitution Rules

One of the core ideas of the approach presented here is to replace ontological elements, i.e. concepts and properties, in the user query for which no matching exists, with variables. This is possible since triple based query languages, as considered here, can deal with variables in all triple positions (subject, predicate and object). For this approach it is important to find the right balance between substituting unmapped ontological elements by variables and omitting entire tuples, which contain unmapped ontological elements. Therefore different relaxation strategies have been designed and evaluated as part of the experiments in Section 5.5:

1. *Strategy 1 - Substitute concepts without mapping:* This strategy replaces concepts used in query Q^u for which no mapping exists with variables. Thus, the respective statement $< s, \tau, c >$, which says s has to be of type c is relaxed to the case that it can be of any type as long as there is a type statement. This is important if two objects in Q^u are supposed to be of the same type. Triples containing unmapped properties are deleted from the query.

2. *Strategy 2 - Substitute properties without mapping:* Any property appearing in Q^u without a mapping defined in $M'i$ is replaced by a variable. Thus the triples, in which the property appears are relaxed to potentially match triples with arbitrary properties in S_i. However the real matches in the query evaluation process are further restricted by the subject and object of the triple as well as by the rest of the query conditions. Triples containing unmapped concepts are deleted from the query.

3. *Strategy 3 - Substitute all unmapped elements:* Replace with a variable all ontological elements in the query Q^u which do not have a mapping in M_i. This is a combination of the first two strategies, but no triples are deleted from the query (as a consequence of the substitution strategy).

4. *Strategy 4 - Substitute unmapped properties of mapped concepts:* If a concept in Q^u is mapped to another concept in S_i by M_i there is typically also a need for mapping the properties associated to the concept. This strategy replaces these properties by variables if no mapping is defined for them. *Strategy 4* is more restrictive than *Strategy 2* with respect to substitution, since only selected unmapped properties are replaced by variables. Triples containing properties that are neither mapped nor replaced by variables and triples containing unmapped concepts are deleted from the query.

5. *Strategy 5 - Substitute unmapped properties if the property values are Literals:* In queries, properties with literal values typically play an important role, since they are used to formulate value conditions for the query. *Strategy 5* focuses on this type of properties and tries to enable value-based matches, even if different property names are used in O^u and O^{S_i} and no mapping exists in M_i. This is done by replacing exactly these properties by variables. *Strategy 5* is also more restrictive than *Strategy 2* with respect to substitution, since only selected unmapped properties are replaced by variables. Again, triples containing properties that are neither mapped nor replaced by variables and triples containing unmapped

concepts are deleted from the query.

6. *Strategy 6 - Substitute unmapped properties of mapped concepts and those linking to literals:* This is a combination of *Strategy 4* and *Strategy 5*. It is also more restrictive than *Strategy 2* with respect to substitution, since only selected unmapped properties are replaced by variables.

Using the different strategies will result in different sets of substitution rules. For a compact definition of the substitution rules the following definitions are required:

Definition 54 *The set of subjects for which the type information is explicitly changed (according to SUB_{M1}) is*

$$TS_i := \{s|\exists p, o : (o, e, v) \in M_i' \wedge < s, p, o > \in Q^u\}$$

Definition 55 *The set of unmapped properties is*

$$UMP_{M_i} := \{p|\exists s, o :< s, p, o > \in Q^u \wedge \neg\exists q : (p, q, v) \in M_i'\}$$

Definition 56 *The set of the un-mapped concepts is*

$$UMC_{M_i} := \{c|\exists s, p :< s, p, c > \in Q^u \wedge \neg\exists d : (c, d, v) \in M_i'\}$$

For the variables introduced by the wildcard substitution rules some conditions have to be fulfilled. When ontological element e_1 and e_2 (property or concept) are replaced by variables v_{e_1} and v_{e_2}:

- the same variable is used in all places where the element is replaced, even if this is done by different substitution rules.
- $v_{e_1}, v_{e_2} \notin VAR_{Q^u}$ (i.e. no variables from the query are re-used)
- $(v_{e_1} = v_{e_2}) \Rightarrow (e_1 = e_2)$ (i.e. different ontological elements are replaced by different variables)

Using the definitions above the substitution rules for the six strategies can be described as follows:

Definition 57 *Substitution rules for Strategy 1:*

$$SUB_w^1 := \{(c, var_c)|\exists s, p :< s, p, c > \in Q^u \wedge c \in UMC_{M_i}\}$$

Definition 58 *Substitution rules for Strategy 2:*

$$SUB_w^2 := \{(p, var_p) | \exists s, o :< s, p, o > \in Q^u \wedge p \in UMP_{M_i}\}$$

Definition 59 *Substitution rules for Strategy 3:*

$$SUB_w^3 := SUB_w^1 \cup SUB_w^2$$

Definition 60 *Substitution rules for Strategy 4:*

$$\begin{aligned} SUB_w^4 := \{(p, var_p) | \exists s, o :< s, p, o > \in Q^u \wedge \\ s \in TS_i \wedge p \in UMP_{M_i}\} \end{aligned}$$

Definition 61 *Substitution rules for Strategy 5:*

$$\begin{aligned} SUB_w^5 := \{(p, var_p) | \exists s, o :< s, p, o > \in Q^u \wedge \\ p \in UMP_{M_i} \wedge o \in \mathcal{L}\} \end{aligned}$$

Definition 62 *Substitution rules for Strategy 6:*

$$SUB_w^6 := SUB_w^4 \cup SUB_w^5$$

5.3.5 Combining Substitution Rules

Groups of substitution rules have been identified, those based on the defined mappings (see Section 5.3.2), and those based on the introduction of wildcards (variables) (see Section 5.3.4). For the rewriting process (see Section 5.3.6) these types of substitution rules are combined. Depending on the selected wildcard strategy, there are six alternative sets of substitution rules, namely SUB_w^1 to SUB_w^6 (see Section 5.3.4).

Definition 63 *Six different combined sets of substitution rules are created $\mathcal{SUB}^1$ to $\mathcal{SUB}^6$ as (for j= 1, .. 6) which will be used to evaluate the impact of the different wildcard strategies on the query rewriting process and its results:*

$$\mathcal{SUB}^j := SUB_M \cup SUB_w^j$$

5.3.6 Query Reformulation Process

All sets of substitution rules defined in the previous section contain pairs of elements. Within the pairs, the first element refers to the element to be replaced in the query and the second element refers to the respective replacement. Thus, the presence of a pair

(e_1, e_2) in a substitution set implies that $Q^u[e_1, e_2]$ will be computed for the considered user query Q^u, i.e. all occurrences of e_1 in the query will be replaced by e_2. For an entire set of substitution rules the substitutions have to be sequentially applied. Since $A^u \cap A^{S_i} = \emptyset$ is assumed and there are no rules that replace variables, the sequence of application will not influence the result of the substitution.

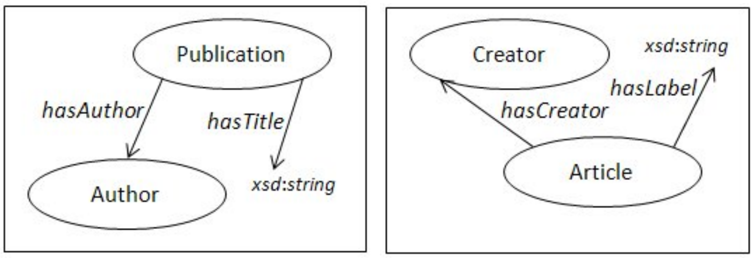
With each of the six sets of combined substitution rules $\mathcal{SUB}^1$ to $\mathcal{SUB}^6$ defined in Section 5.3.5 a query Q^u can be rewritten into a target query using in the following way:

$$\begin{aligned} Q_i^{\mathcal{SUB}^j} &:= Q^u[e_1 : sub_1]...[e_m : sub_m]\ where \\ (e_i, sub_i) \in \mathcal{SUB}^j \wedge (i \neq k \Rightarrow (e_i, sub_i) &\neq (e_k, sub_k)) \\ &\wedge |\mathcal{SUB}^j| = m \end{aligned}$$

Definition 64 $Q_i^{\mathcal{SUB}^j}$ *denotes the target query created by applying the substitution rules of rule set* $\mathcal{SUB}^j$ *to the query* Q^u.

5.3.7 Query Reformulation Example

In Figure 5.1(a) a simplified version of the example user ontology O^u and in Figure 5.1(b) a simple ontology describing one source S_i to be queried are presented.



(a) User ontology O^u (b) Ontology describing S_i

Figure 5.1: Example Ontologies

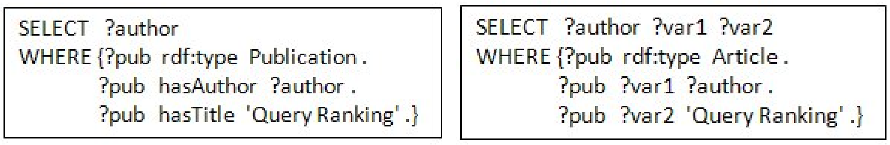
The available partial mapping M_i between these two ontologies is presented in Table 5.1.

A simple user query Q^u, constructed using elements of the user ontology O^u, is presented in Figure 5.2(a).

Figure 5.2(b) shows the rewritten form of this user query, employing $\mathcal{SUB}^4$ and using the threshold $\beta = 0.70$. The reader can notice that the concept *Publication* is rewritten

From	To	Confidence
Publication	Article	0.75
hasTitle	hasLabel	0.65

Table 5.1: Example Mappings



```
SELECT  ?author
WHERE {?pub  rdf:type  Publication .
         ?pub  hasAuthor  ?author .
         ?pub  hasTitle  'Query Ranking' .}
```

(a) User query Q^u

```
SELECT  ?author  ?var1  ?var2
WHERE {?pub  rdf:type  Article .
         ?pub  ?var1  ?author .
         ?pub  ?var2  'Query Ranking' .}
```

(b) Reformulated query Q^{S_i}

Figure 5.2: Example Queries

to concept *Article*, since there is a mapping specification (Table 5.1) with high enough confidence relating the two concepts. For property *hasTitle* the confidence is not high enough and for property *hasAuthor* there is no mapping defined, and so, according to $\mathcal{SUB}^4$, they are replaced with a new variable each (notice that same elements are replaced with the same variable).

5.4 Refining and Learning Mappings

The previous discussions in this chapter focused on the application of this query reformulation approach. However, the approach also has the potential of contributing evidences for the refinement and extension of the existing partial mapping M_i.

If an ontological element, let's say a property p, is replaced by a variable v_p the respective query triple is relaxed. During query execution the variable is iteratively bound to different properties in O_u. Only some of these bindings will contribute to the query results, since also all other query conditions have to be fulfilled given the binding. (p, q) is called a *successful coupling*, when p is replaced by a variable v_p and the binding of v_p to property q contributes to the query result.

If a successful coupling happens frequently for a pair p and q of properties (or a pair of concepts) this suggests that there is a not yet specified mapping between these two ontological elements. To give the intuition behind this idea: Assume the information source contains a property q with literal values. A property p used in the query will often form a successful coupling with q, if the values specified for p in the query are often included in the values available for q in the information source S_i. The assumption made here is that properties with a similar set of employed values have a higher probability of being the same property. Of course, this is a heuristic, which will not work out in all cases.

In this approach, individual (small) evidences are aggregated whenever successful couplings occur, which are defined by user feedback. The collected evidences are then employed for updating the mappings, e.g. inserting the mapping (p, q). Not only new mappings are learned in this way, but also collected evidences are used to increase the confidence values for already existing mappings (as they have e.g. be computed by automatic mapping methods).

In this way, the more evidences for correct bindings of variables in queries to values in results are found, the better the mappings get, and in this way a pay-as-you-go like integration is enabled. The more different queries are posted, the more the system will learn or improve different mappings, leading to eventually have a complete mapping definition over time. It will be clearly shown in the results in Section 5.5.3 how this process of learning mappings or improving the confidence of mapping influences the obtained precision.

5.5 Experiment and Evaluation

The different query rewriting strategies presented in Section 5.3 have been evaluated with respect to their impact on precision and recall. Furthermore, the potential for mapping refinement as presented in Section 5.4 has been systematically analyzed. This section describes the employed datasets, the setup of the experiments and the obtained results.

5.5.1 Testbed: Ontologies and Data

A virtual personal desktop (VPD), a special kind of SKS introduced originally in (133), was used for the experiments. In this VPD, local desktop information sources are originally based on one predefined set of ontologies (109), but the users have the freedom to extend and modify them. Other external sources have their own ontologies describing the structure of the data stored therein, without any user control. The queries are not directly evaluated on the desktop resources, but on a semantic layer of metadata that describes and relates the desktop resources. This layer is created by extraction technology as it was, for example, developed in the Nepomuk project (5, 20).

In the experiments five information sources S_1, S_2,... S_5 are employed, reflecting five areas of this VPD. The ontology used by the user O^u for query formulation in the experiments is connected to the ontologies O^{S_i} describing the information sources S_i by partial mappings. An excerpt of the user ontology O^u which was used for constructing the different queries for this experiment is presented in Figure 5.3.

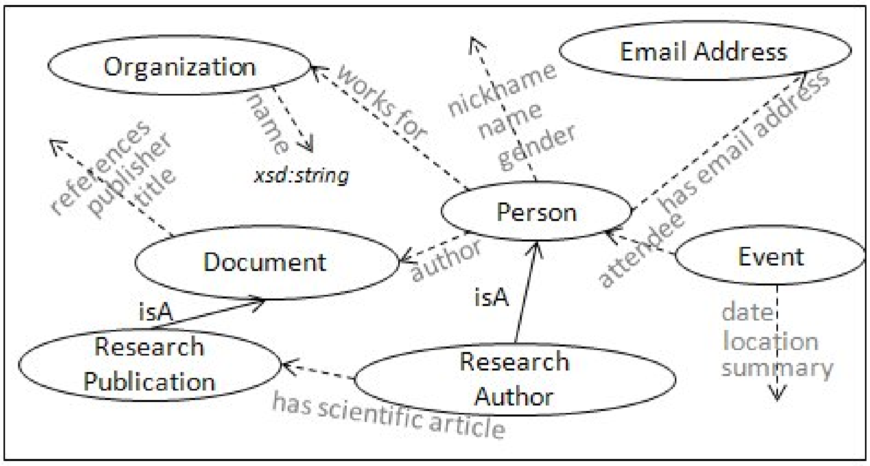


Figure 5.3: Excerpt of the user's ontology O^u used for specifying Q^u

In the following the information sources S_i are described in more detail.

Information source $\mathbf{S_1}$ contains metadata about **Calendar Items**, **Email**, **Presentations** and **Documents**. Exemplary aspects of the representation of this information source are shown in the ontologies in Figures 5.4 and 5.5 which are part of a set of ontologies to describe desktop resources as designed in the project *Nepomuk Information Ontologies*[1].

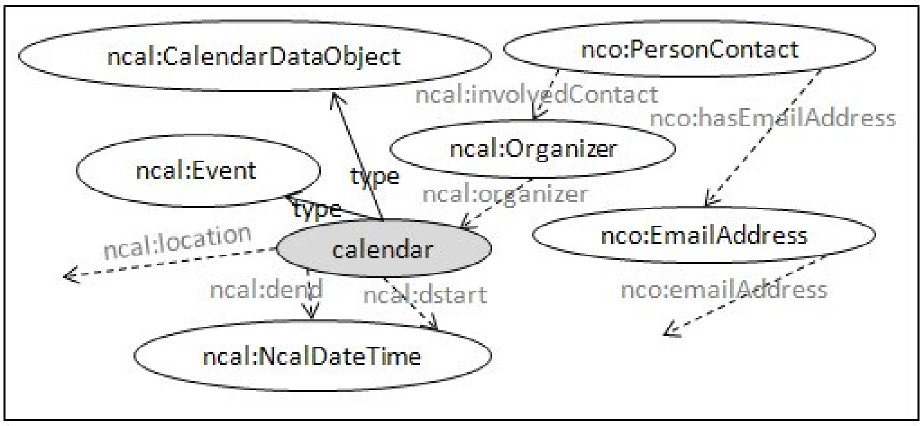


Figure 5.4: Simple schema for Calendar entries

[1]`http://nepomuk.semanticdesktop.org/ontologies/`

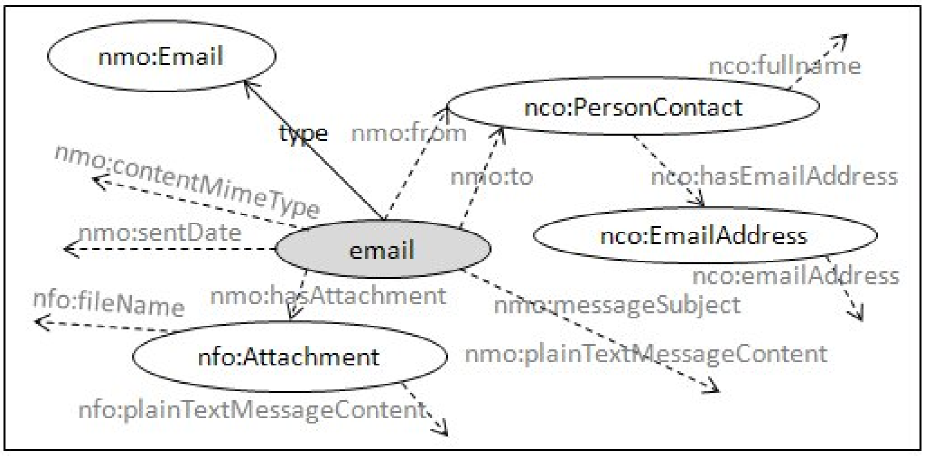


Figure 5.5: Simple schema for Emails

Information source $\mathbf{S_2}$ contains information about **Meeting Minutes**. Project minutes are *Documents* with a predefined structure, containing the minutes of internal Nepomuk project meetings at L3S as presented in Figure 5.6. The assigned tasks and their deadlines (see Figure 5.7), as well as informative items and events are contained (see Figure 5.8 and Figure 5.9 respectively). The metadata extracted from this special kind of documents does not follow the Nepomuk ontologies previously presented in this section, but the *L3S Meeting Minutes* ontology (mtm), which was created independently of the Nepomuk ontologies.

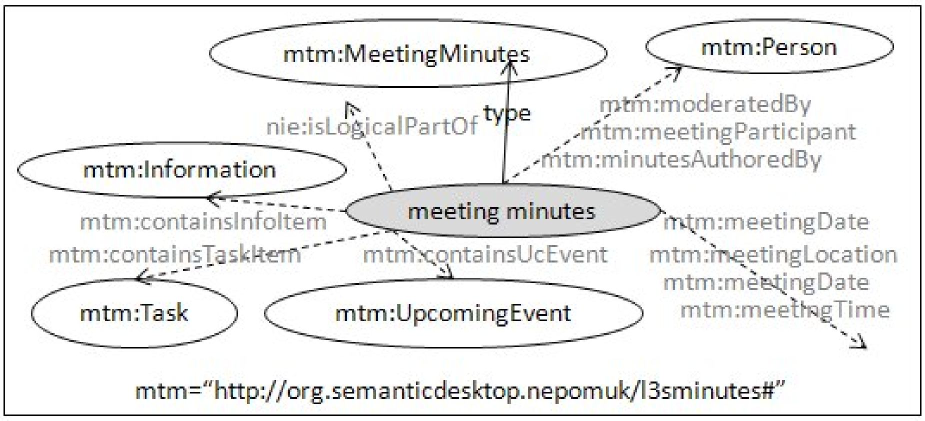


Figure 5.6: Simple ontology for Nepomuk L3S internal Meeting Minutes

Information source $\mathbf{S_3}$ contains information about **Publications**. Scientific publications, usually as PDF files, are also present in the desktops. For this kind of pub-

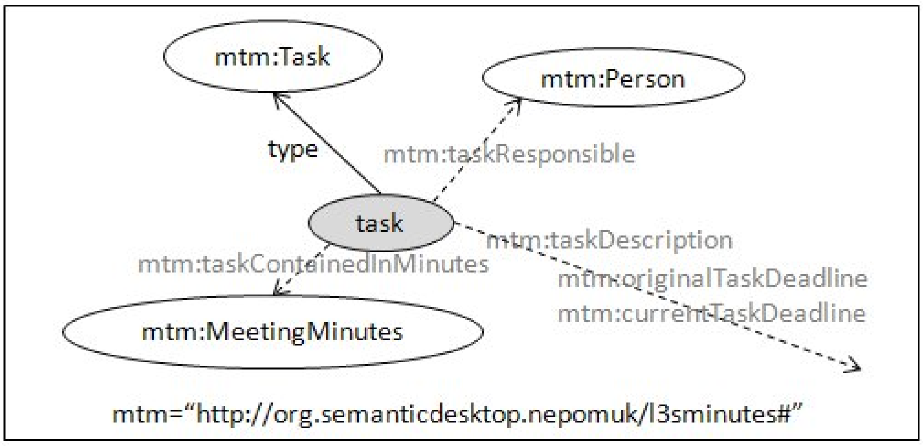


Figure 5.7: Simple ontology for Nepomuk L3S internal Meeting Minutes: Task items

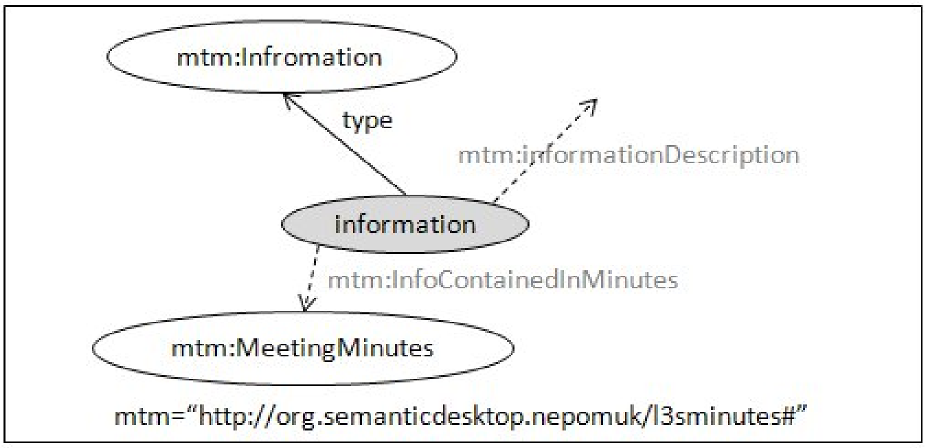

mtm:Infromation
type
mtm:informationDescription
information
mtm:InfoContainedInMinutes
mtm:MeetingMinutes
mtm="http://org.semanticdesktop.nepomuk/l3sminutes#"

Figure 5.8: Simple ontology for Nepomuk L3S internal Meeting Minutes: Information items

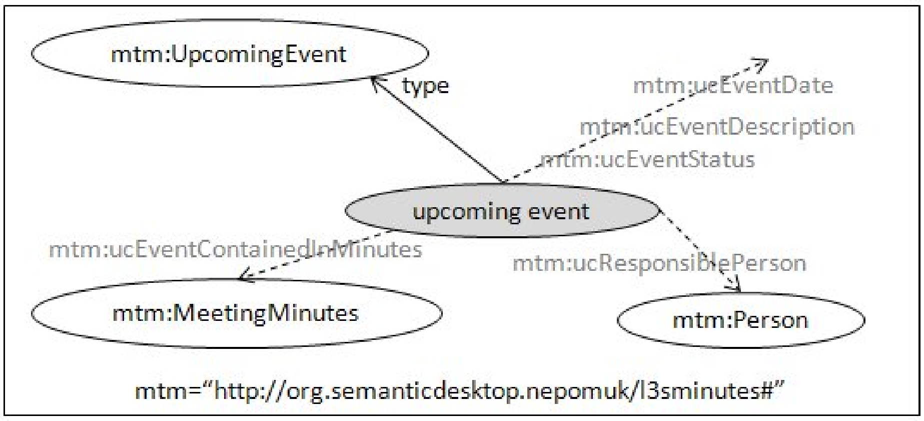


Figure 5.9: Simple ontology for Nepomuk L3S internal Meeting Minutes: Upcoming Events

lications, metadata depicting authors, editors, proceedings where they were published, and a publication date among others can be extracted. A partial view of the extracted metadata is shown in Figure 5.10.

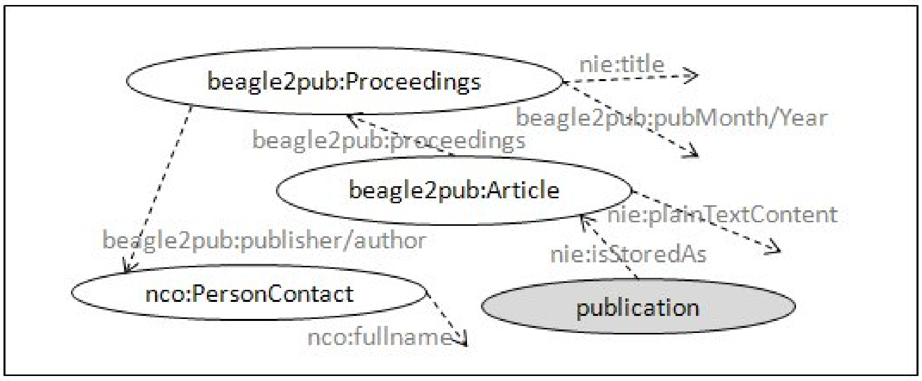


Figure 5.10: Simple ontology for Publications

Information source S_4 contains information about **Persons** extracted from **FOAF** files. Another kind of data sources are the own FOAF profile and the FOAF profiles found when visiting web pages. Standard FOAF profiles usually contain information about the email address, the place of work, postal address, known people, etc. Also information relating to "friend" profiles are provided, so these are also considered as additional sources of information, by traversing the user's network of friends. The FOAF files crawled are files describing the current user of the system as well as the people he knows, obtained by traversing up to k steps the references to files of known people. For crawling FOAF files and storing them in the Nepomuk repository, a special crawler was built which stores the files by using the FOAF ontology[1].

Information source $\mathbf{S_5}$ contains information about **Telephone Conferences** extracted from project **Wiki Content**. In Nepomuk, XWiki[2] was employed, a collaborative platform where all project relevant shared content is stored. In the Nepomuk XWiki there is a record of all telephone conferences held, each in a different page and with similar structure. These pages on the wiki were accessed on a regular basis and its contents crawled. The extracted metadata was then stored it in the local repository. The *Wiki Telco Minutes* (xwt) ontology used for representing this information was also developed independently of the Nepomuk ontologies and an excerpt of it is presented in Figure 5.11.

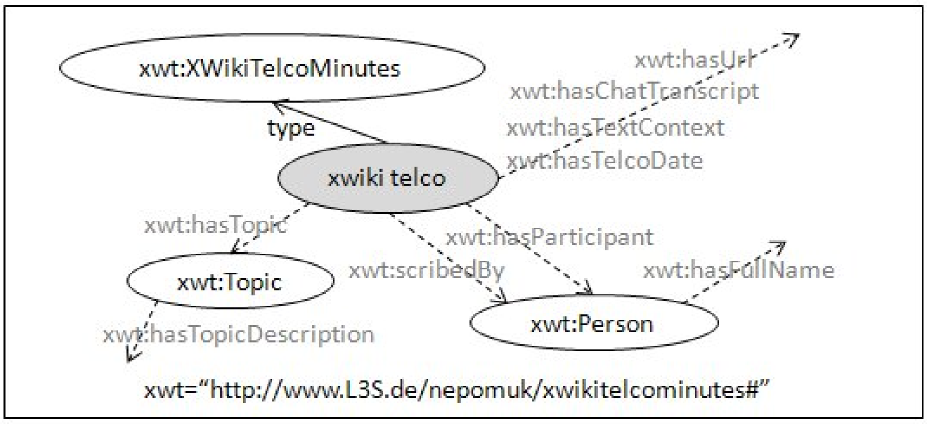


Figure 5.11: Simple ontology for TelCo Minutes on the Wiki

All the generated or extracted metadata is stored in a RDF store, as available in the Nepomuk project.

Data from 14 different desktops of people working at L3S[3] was used for the experiments. These desktops contained different file types such as MS-Office files, Emails, PDF's, structured files e.g. LaTeX and BibTeX files, as well as plain text files. Each different desktop was crawled separately with the technology available in Nepomuk, and, for each of them metadata in form of RDF triples was generated as a result of the crawling process. In addition, the metadata available for each of the desktops was enriched with metadata automatically extracted from internal L3S Nepomuk project meetings using, user-related FOAF descriptions, as well as metadata related to tele-

[1]`http://xmlns.com/foaf/0.1/`
[2]`http://www.xwiki.com`
[3]`http://www.L3S.de`

Pictures	Documents	Structured files
120	174	188
PDF	**Presentations**	**Emails**
405	12	3379

Table 5.2: Mean number of file types per desktop

phone conferences from the Nepomuk XWiki . The mean quantity of different file types available for each desktop is presented in Table 5.2.

5.5.2 Experiment Setup

The goal of the experiments was to evaluate the different query substitution strategies presented in Section 5.3 over several iterations, providing at each iteration feedback on the confidence of the mappings by analyzing the results of the queries. The experiments were performed by using an evaluation framework interacting with e.g. the Nepomuk *InfoIntegration* component which is integrated into the Nepomuk infrastructure providing the functionality described in Algorithm 5.1.

Datasets: The datasets (S_i) described in Section 5.5.1 were used.

Ground Truths: A ground truth G^{S_i} was computed for each dataset S_i, by representing its instances in terms of the user-view (O^u) on the domain. For this a corresponding transformation has been defined and executed on the complete content of each S_i, obtaining in this way all possible correct answers to each query.

Mappings: A set of initial mappings was computed from the ontology defining the user-view on the domain O^u and the ontologies presented in Section 5.5.1. The automatic computation of this first mapping (M_i) was performed by using the *PhaseAlignment* Nepomuk component based on the work described in (145).

Queries: Eighty queries were defined on terms of the user-view on the domain having between 1 and 4 conjunctive conditions. The queries were executed on each dataset's ground truth G^{S_i}, and the ones returning no results were not applied for this dataset. From now on when referring to the queries (each of them denoted Q^u) in the context of a dataset only those giving at least one result when querying the corresponding ground truth will be considered.

The evaluation of the query substitution strategies ($\mathcal{SUB}^j$ as defined in Section 5.3) was performed on each dataset, and then computed the overall quality measure of each substitution strategy. Quality was measured in terms of precision and recall, by comparing the results of evaluating the query on the ground truth G^{S_i} with the results obtained when querying the dataset with the rewritten query using a specific rewriting strategy. Algorithm 5.2 presents an overview of the evaluation process by relying on the notation previously introduced.

Algorithm 5.2 Experiment execution process

```
1:  Compute mappings $M_i$
2:  for each feedback iteration do
3:    for each different substitution strategy $\mathcal{SUB}^j$ do
4:      for each $S_i$ do
5:        for each $Q^u$ do
6:          Evaluate $Q^u$ on $G^{S_i}$
7:          for each computed mapping do
8:            Compute $Q_i^{\mathcal{SUB}^j}$ by applying $\mathcal{SUB}^j$ to $Q^u$
9:            Accumulate results of evaluating $Q_i^{\mathcal{SUB}^j}$ on $S_i$
10:         end for
11:         Compute quality measures and mapping feedback
12:       end for
13:       Compute quality measures over all queries
14:     end for
15:     Compute quality measures over all $S_i$
16:     Store quality measures for $\mathcal{SUB}^j$ for this iteration
17:   end for
18:   Update mappings with mapping feedback
19: end for
20: return quality measures for $\mathcal{SUB}^j$ at each feedback iteration
```

In the experiments presented here the user feedback was simulated. The values of the variables for properties or concepts in the evaluated queries were taken, and, by comparing them with the expected ones, a decision on an answer being right or wrong was made. In case of a correct “binding” of variables to concepts or properties in a query, the corresponding mapping confidence value was incremented by 0.1 for the next iteration. In case no mapping existed between these elements, a mapping with confidence value 0.0 was added, this mapping could then be increased in following query evaluations as already explained. It would remain future task to investigate on using the evidence on false “bindings” to lower existing mapping confidence values. For the experiments presented in this chapter a value of $\beta = 0.7$ was used. The results of the experiments are presented in Section 5.5.3.

5.5.3 Evaluation Results

The substitution strategies $\mathcal{SUB}^j$ evaluated first are the ones defined in Section 5.3. The reader might recall that different strategies imply different usage of wildcards in the query rewriting process, and in this way different relaxation strategies.

As presented in Sections 5.5.1 and 5.5.2, the different strategies over all datasets were evaulated. In Table 5.3 the precision values for each of the strategies without considering any mapping feedback are presented. This is, mappings were used as initially generated. It can be noticed that all strategies present low precision values, and most of them also low recall values, meaning that the generated mappings were mostly not complete enough to allow the retrieval of the requested information. The next paragraph presents the evolution of the precision and recall over several iterations, showing that the feedback on the mappings is an important aspect in the presented approach.

	$\mathcal{SUB}^1$	$\mathcal{SUB}^2$	$\mathcal{SUB}^3$	$\mathcal{SUB}^4$	$\mathcal{SUB}^5$	$\mathcal{SUB}^6$
Precision	0.28	0.14	0.16	0.19	0.25	0.19
Recall	0.34	0.62	0.81	0.62	0.24	0.59

Table 5.3: Quality measures for each strategy

Figure 5.12 shows the variation of precision and Figure 5.13 of recall for the different strategies over the different iterations. At each iteration, feedback on mappings was given for each strategy separately, so that the evolution of precision and recall values by using the different strategies with feedback can be observed. Strategy $\mathcal{SUB}^1$ and $\mathcal{SUB}^5$ gave the best precision throughout the iterations, but the worst recall when the mappings were sparse or low in confidence (cold start problem). After very few iterations this changed due to the feedback given to the mappings. This suggest that the best way to proceed would be to use a mix of strategies, starting with a strategy that returns less precise results but higher recall like $\mathcal{SUB}^2$, $\mathcal{SUB}^3$ or $\mathcal{SUB}^6$ and then, after some iterations when the mappings were improved based on the feedback, move to use $\mathcal{SUB}^1$ or $\mathcal{SUB}^5$. The low recall that can be observed for some of the strategies is caused by the incompleteness of the mappings at the first iterations. Once the system learns and increases confidence of the correct mappings, recall also improves. An interesting aspect is the small decrease in precision for some strategies at the last iteration. This happened because wrong mappings have been learned for those strategies. As mentioned earlier, it would remain an interesting task to evaluate the problem of detecting not only right but also wrong mappings via user feedback, and to use this information to avoid misslearning of mappings.

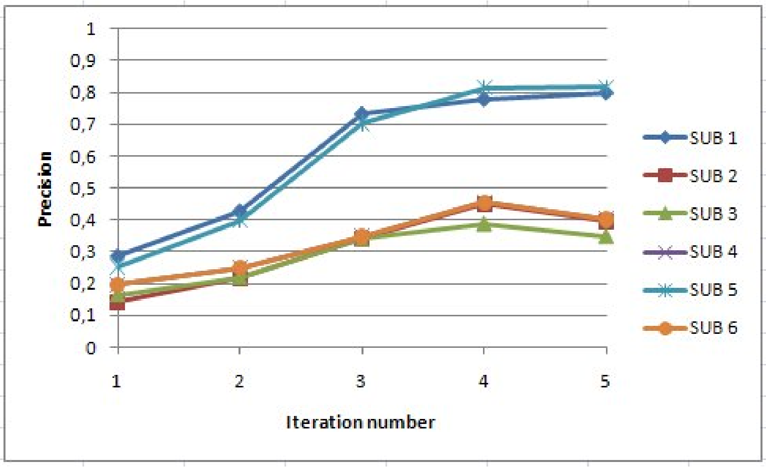


Figure 5.12: Precision evolution for every strategy

Regarding the mappings and the discovery of new ones, evidence shows that all newly discovered mappings are added in the course of the first iterations. Subsequent feedback increases the confidence of these mappings, so that eventually after few iterations (the third in the presented experiments) most of them are used for rewritings. Regarding the existing mappings which had a too low confidence in order to be used in the query rewriting processes, positive feedback evidence makes their confidence raise in every iteration so that at some point they are used in the query rewriting process. Different strategies make confidence of different mappings grow faster than others, which is the reason that, given the employed datasets and queries, some strategies perform better than others.

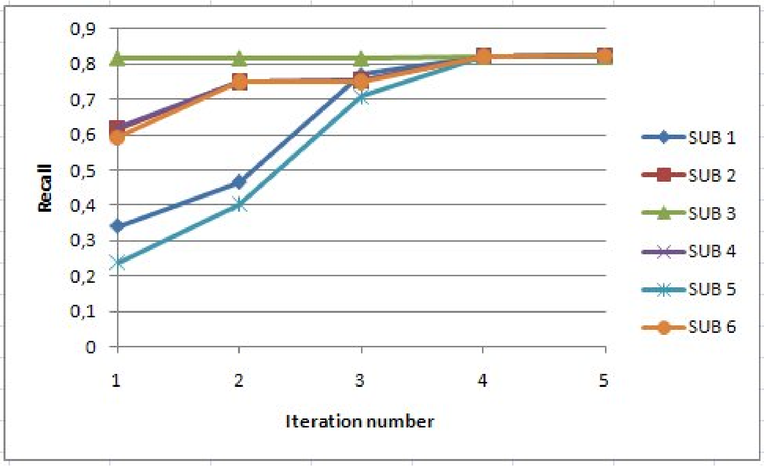


Figure 5.13: Recall evolution for every strategy

6

Ranking Reformulated Queries

This chapter presents a way to deal with the results obtained from querying information sources using the Query Reformulation approach presented in Chapter 5. This chapter addresses one of the challenges presented in Section 2.4 and describes in detail the solution sketched in Section 2.5. The example introduced in Section 5.3.7 will be used to exemplify the approach.

An approach (130) to rank the results obtained from querying different information sources based on the confidence that a query gives right results is presented. This approach takes into account the relaxation degree and confidence of the employed mappings introduced in the reformulated query to compute a ranking for the query so that results coming from information sources are ranked by this query ranking result. First, the related work to ranking is presented in Section 6.1, followed by the ranking approach in Section 6.2, and its evaluation in Section 6.3.

6.1 Related Work on Ranking

Ranking is mostly known from Information Retrieval approaches. There are, however, also some other approaches which rank the results from structured queries.

The approach to answer imprecise queries over autonomous web databases presented in (101), computes the ranking based on the percentage of common attribute-value pairs between the original query and each relaxed query. Relaxation of queries is performed by removing selected attribute-value pairs from the original query, until the expected results are obtained. This is similar to the approach presented in this thesis in assessing the degree of relaxation for computing the ranking. The mapping confidence is also additionally considered in the ranking function presented in this chapter.

Ranking is also computed on relaxed or malleable queries as done in (160), by considering the quantification of the correlations existing between attributes, ranking higher the results based on relaxations using higher correlated attributes than other relaxations employing less correlated attributes. Correlations are detected by analyzing the data sources, detecting entity duplicates described by attribute values of different schemata. In this way correlation of schema elements is discovered, by analyzing these duplicate entities in the data sets, their describing values and the schema elements used to name them. This is more similar to the idea presented here on using the mapping confidence in computing the ranking function.

There are several other approaches e.g applied in XML databases, as (114), where approximate tree structured answers on XML documents are computed by taking advantage of the analysis and summarization of the XML tree structure (*treesketch summaries*). Here, a measure of the quality of answers is computed using the analyzed tree structure and the distribution of document edges, which can be used for ranking obtained answers. The user inspects the summarized results, and, if interested, the query is executed on the real XML document so that real matching content is retrieved.

In addition, ranking is also used in the context of Semantic Web technology (see e.g. (36, 136)). In (136), for example, an interesting technique for ranking the results for a query on the semantic web takes into consideration the inferencing processes that led to each result. In this approach, the relevance of the returned results for a query is computed based upon the specificity of the relations (links) used when extracting information from the knowledge base, i.e. the degree of inferencing for obtaining each of the elements in the query is accumulated in a measure, in order to state how much the result is a direct answer to the query or was inferred by using the knowledge in the ontology about the domain. This approach is complementary to the approach presented in this chapter, since the specificity values from the inferencing process could be considered in the computation of the ranking function presented in this thesis, in addition to the confidence in the used mapping and the degree of relaxation applied. The work presented in (137) aims also at integrating distributed sources containing RDF data which are described by ontologies. An important difference is that only rewriting without relaxation of queries is produced, so, the ranking for the results of queries only considers the confidence of the used mappings and not other aspects (such as relaxation degree) as done in the work presented in Section 6.2.

6.1.1 Discussions

The described ranking approaches employ the information about how much the relaxed query deviates from the original query in order to compute the ranking. Approaches like (160) take into account the instances available in order to quantify the correlation of attributes, and rank results of queries based on this aspect (relaxations using less correlated attributes imply lower ranking). Other approaches like (136) employ information from the inferencing process followed to obtain a replacement, by measuring the “distance” between the original query elements and their replacements. It can be noticed that each of the approaches employ information which is directly derived from the approach used to produce the query reformulation in order to compute the ranking of the results.

The approach presented in Section 6.2 employs rewriting in combination with relaxation to reformulate the query, so, both aspects need to be taken into account for defining the ranking function. Partial solutions to similar approaches were presented in e.g. (137) and (101), but no approach combining the confidence of the employed mappings with the degree of relaxation introduced could be found. Therefore, a novel ranking function has been developed and is presented in Section 6.2.

6.2 Query Ranking Approach

The application of the query reformulation strategies presented in Chapter 5 might introduce errors into the query result set. At the same time they define different levels of confidence in query results for each query over different information sources, based on the reformulation level of the original query. For reflecting this, e.g. when presenting the results to the user, a way to rank the results coming from different information sources accordingly to the confidence on the query used to compute them is needed. Therefore, the approach presented in this chapter uses the ranking of the different *queries* to rank results from different information sources (vs. ranking single results at each information source).

Ranking of query (retrieval) results is a long studied issue in Information Retrieval (IR), where the goal is to present to the user the most relevant query results among the first results (top-k). Current approaches to ranking, as they are used e.g. in Web search engines, do not rely on document content alone, but also take into account aspects such as popularity and authoritativeness typically relying on a variation of the PageRank approach (111).

Structured queries as considered here and as used in the database area have the traditional boolean model. An object either matches the query and is included into the query result or it is excluded from the result. These models do not have a notion of results with higher relevance for the query and thus no ranking is required.

In the Lightweight Information Integration approach presented here, queries are relaxed, giving in this way the possibility to obtain results, independently of their correctness, which do not strictly correspond to the originally specified query[1].

Considering that complete and perfectly confident mappings can only give correct results (the ideally rewritten query), the approach presented in this section for a *query* ranking function (as opposed to ranking individual query results) is based on introducing punishments to each of the aspects which might introduce errors when executing the reformulated query over a specific information source. Intuitively, the higher the difference of the resulting query from the ideally rewritten query, the more "punishment" is added to the results obtained from executing it.

6.2.1 Punishment Derived from Mappings

Query rewriting based on mappings contributes to making the query results less reliable by the lack of confidence in the used mappings. The higher the confidence in the correctness of a mapping, the lower the probability of introducing an error when this mapping is used to rewrite the user query. For reflecting this, and also considering that mappings are independent, a factor called **Query Mapping Confidence** (Qmc) is defined in Eq. 6.1 on the basis of the definition of M'_i introduced in Section 5.3.2. This factor reflects the probability of introducing errors by the product of the confidences of the mappings used in the rewriting of Q^u (as also done in (137)).

$$Qmc = \prod v|(e, e', v) \in M'_i \tag{6.1}$$

The rationale for this is that each successfully applied mapping will avoid that this element is relaxed with a wildcard, but it will also possibly add some incorrect results, depending on the confidence of this mapping.

6.2.2 Punishment Derived from Relaxation

The query relaxation might introduce errors by allowing bound expressions in Q^u to become unbound in Q^{S_i}. For the query example presented in Section 5.3.7 this would

[1]In ideal case of a complete mapping specification, all query results obtained will be correct results (i.e. the boolean model mentioned before will be obtained).

mean that, if there were no mapping from "Publication" to "Article", "Publication" would have been replaced by a wildcard, and therefore the constraints applied to the type of the variable "pub" would have been relaxed to something more general.

The computation of the punishment considers the number of introduced variables which remain bound (by some value constraint) and the number of variables introduced which do not even have a value constraint. The computation relies on the following definitions:

Definition 65 *$\mathfrak{LU}$ are the element names in Q^u not part of $A^u \cup \tau \cup VAR_{Q^u}$ (i.e. Literals and URI's not belonging to the schema).*

Definition 66 *nb denotes the number of bound variables introduced in the relaxation process: For $< s, p, o >$, o is a literal or a URI and p is replaced with var_p: $< s, var_p, o > \wedge o \in \mathfrak{LU}$, then $nb = nb + 1$ once for var_p. This is, nb will be incremented by one the first time var_p is introduced, even though any other occurrence of p will also be replaced with the same var_p.*

The idea in Definition 66 is that a wildcard is being introduced, but the values it can take are restricted by s and the given value of o which is fixed.

Definition 67 *nu denotes the number of unbound variables introduced in the relaxation process.*

The computation of the number of introduced unbound variables (nu) (Definition 67) has two options, depending on the elements of the triples:

1. For $< s, \tau, o >$, o is replaced with var_c: $< s, \tau, var_c > \wedge o \in A_C$, $nu = nu + 1$ for *every* usage of var_c. nu is increased by one for every occurrence, since more relaxation is added with every replacement. In this case s is allowed to be of any type, as well as any other thing originally specified to have the same type as s, so the "essence" specification of things is being relaxed.

2. For $< s, p, var_o >$, p is replaced with var_p: $< s, var_p, var_o > \wedge var_o \in VAR_{Q^u}$, $nu = nu + 1$ for *every* occurrence of var_p. The value of nu is increased by one since each occurrence increases the degree of relaxation. This occurs because there was only the relation p as a restriction between the values that s and var_o could get, and now even this last restriction has been relaxed. In this case the relaxed query expresses the fact that there must be some connection between s and var_o, but without saying which connection it is.

Definition 68 *qL represents the number of triple elements in Q^u: qL = Number of triples $*3$.*

With these measures, a punishment for bound variables can be computed as:

$$P_{nb} = 1 - \frac{nb}{qL} \tag{6.2}$$

Even though the variables are bound, this factor is needed because their number influences the relaxation relative to the query joins (therefore the normalization depending on the query length). Also, the correlation between nb and the ranking is indirect, since the less bound variables are introduced, the higher that query can be ranked. The introduction of unbound variables has a higher influence on the accurateness of the query results than the bounded ones, because they introduce more relaxation to the query, and therefore more penalty is needed. The punishment for unbound variables can be computed as:

$$P_{nu} = \alpha^{1-\sqrt{\frac{nu}{qL}}} - 1, \tag{6.3}$$

where the value of α, the relaxation penalty, still has to be defined. This factor grows differently than the one computed for bound variables, in the sense that the errors introduced by unbound variables make the query less accurate than introduction of bound variables, and therefore this factor should have a more dramatic decrease when the number of unbound variables is high. Different variations of the exponent factor in the function used for the computation of P_{nu} have been tried out, but they all behaved similarly.

6.2.3 Ranking Function

Based on the factors presented above, a ranking function is defined so that the query results over all available information sources can be sorted accordingly, giving higher values to the ones with likely less errors. The factors presented in Equations 6.1, 6.2 and 6.3 are combined and the ranking function can be computed as a product of the possible punishments (errors) introduced:

$$R(Q^u) = Qmc * P_{nb} * P_{nu} \tag{6.4}$$

$R(Q^u)$ gives us a good estimate about the expected correctness of the reformulated query results, by taking into account the confidence of the employed mappings, and the effects of applying a certain “wildcard” strategy on the original query. The combination of these values leads to a measure of how much the reformulated query “deviates” from the ideally rewritten query, by combining the confidence of the used mappings, and the punishments for the applied relaxation.

6.2.4 Computing Ranking Example

The computation of the ranking for one query will be exemplified using the example introduced in Section 5.3.7. Considering the reformulated query presented in Figure 5.2(b) and $\alpha = 2$ (more on this in Section 6.3.2.2), the computation yields $Qmc = 0.75$, $nb = 1$ (relaxation $hasTitle \rightarrow ?var2$), $nu = 1$ (relaxation $hasAuthor \rightarrow ?var1$), and $qL = 9$. With these values the ranking can be computed using Eqs. 6.1, 6.2, 6.3, and 6.4 as:

$$R(Q^u) = Qmc * P_{nb} * P_{nu} = 0.75 * 0.89 * 0.589 = 0.39$$

When the same query is reformulated for being executed on another source, if the mappings have higher confidence, or the relaxation is not so high, the ranking will be higher, so that results can be ranked using this obtained value based on the belief that different sources will give more accurate results than others for the same user query.

6.3 Evaluation

In this evaluation, the ranking method validity for efficiently ordering the results is shown, by computing the precision at different cuts of the top results - top-k. First different values for α were investigated and an optimal α was computed. Based on this configuration, the precision of this ranking approach was evaluated using a larger query set. Since for the ideal query, results exist and are equally correct, or no result exists, the obtained results could not be compared against a ranked ground truth. Instead of this, the top-k results were compared with the complete ground truth (the set of expected right results). In this comparison, it was checked which of the top-k results were correct, getting at the end the precision of results at several top-k values. As strategies $\mathcal{SUB}^2$ and $\mathcal{SUB}^4$ presented similar behavior as $\mathcal{SUB}^6$ they will not be discussed in detail. Also, brief assessments regarding the query execution time are presented, dependent on the employed underlying storage.

Source	VPD	UMBEL	YAGO	DBPediaPersons	DBPediaInfoboxes
No. triples	$3,329,376$	$6,740,428$	$460,418,591$	$812,339$	$9,877,564$

Table 6.1: Information Sources Overview

6.3.1 Evaluation Setting

In this section the details about the information sources, the initial mappings, queries and ground truth used for the evaluation are presented.

6.3.1.1 Information Sources

The heterogeneous information sources considered with the number of contained triples are presented in Table 6.1. Each of the sources was used to create a different Sesame repository (`www.openrdf.org/`). In detail the employed sources are:

Virtual Personal Desktop (VPD) obtained from crawling desktops (PDFs, Word documents, Emails, Wiki pages, friends' FOAF profiles residing in the Internet) of colleagues using the crawling approach and ontologies presented in Section 5.5.1. These 16 sources have their own user ontology and query set (see Section 5.5.2 for details). The total number of triples across all 16 VPDs are presented in Table 6.1.

UMBEL (Upper Mapping and Binding Exchange Layer (144)) provides an ontology and corresponding instances, along with many definitions of equality (using "sameAs" relations) to instances in the other datasets used for this experiment - YAGO and DBPedia. Therefore, the instances provided by UMBEL were used to compute the ground truth. The provided ontology (*UMBEL Ontology*) was taken as the *user ontology* (in a filtered version containing only concepts and properties which actually appear in the instances), and will be denoted from now on as user ontology or O^u. In order to construct the *UMBELInstances* data source and its ontology out of it, original references to concepts and properties, as well as to resources in the instances were all modified programmatically, simulating in this way a new source which will be denoted from now on as UMBEL.

YAGO (139) in its RDFS version 2008-w40-2 was used. From the provided instances a simple ontology (*YAGO Ontology*) describing them was extracted, by querying them and extracting all classes and properties appearing therein.

DBPedia (34) in its version 3.2 was used. Out of it two sources were created: 1) the *DBPediaPersons* source containing all available "Persondata" files which are represented using the *FOAF* ontology, and 2) the *DBPediaInfoboxes* source containing the "Infoboxes" and "Types" files which are represented using the *DBPedia Ontology* provided by this version of DBPedia.

6.3.1.2 Initial Mappings

Mappings were computed between the *UMBEL Ontology* and its modified version presented above, the *DBPedia Ontology*, the *FOAF* ontology, and the *YAGO Ontology* with the approach implemented in Nepomuk and described in (145). These alignments were randomly modified, deleting some mappings, and used as initial *partial* pair-wise mappings between the *UMBEL Ontology* and the ontologies describing the sources. Same type of partial mappings were also computed for the VPDs, between the user ontology specified in 5.5.1 and the ontologies describing the different crawled sources.

6.3.1.3 Queries

Two sets of queries were used in the experiments. A set of 5 queries, $QSet1$, was used for finding the α parameter of the ranking function, also focusing on more complex queries (including e.g. regular expressions) for avoiding an over-adaptation of the parameter to simple queries. The main query set $QSet2$, an extension of the first set and of the queries used in 5.5, consists of more than 70 queries containing mainly 1 to 3 joins, which are used to evaluate the performance of the algorithm. In more detail, they are queries for searching entities by either specifying direct properties of the entities (e.g. name) or by specifying properties of related entities (e.g. name of the city a person was born in). Table 6.2 shows on the left hand side an example of the most complex queries, and on the right side an example of a simpler query (in SPARQL syntax, prefixes were removed to ease readability).

6.3.1.4 Ground Truth

For each query there is a ground truth, which contains all correct results expected. In order to have comparable query results from the different sources, explicit equivalences between resources appearing in the different sources have been exploited (e.g. the ones provided in the UMBEL site with a “sameAs” relationship) or manually created whenever needed.

6.3.2 Experiments

In this experiments, and for each source, relaxation strategies (wildcard based) together with rewriting strategies (mapping based) (introduced in Chapter 5) are used. For each of the strategies, four user feedback iterations were simulated, this is, after

`SELECT DISTINCT ?rally ?name ?person`
`WHERE { {?rally rdf:type umbel:Rally .`
`  ?rally skos:prefLabel ?name`
`  Filter(!isURI(?name) && Regex(str(?name),`
`  "rallye deutschland", "i")) .`
`  Optional {?rally <umbel:rallyParticipant> ?person .`
`           ?person rdf:type umbel:Person . }}`
`UNION {?rally rdf:type umbel:Rally .`
`?rally semset:hasSemset ?semset .`
`?sems skos:altLabel ?name`
`Filter(!isURI(?name) && Regex(str(?name),`
`"rallye deutschland", "i")) .`
`Optional {?rally <umbel:rallyParticipant> ?person .`
`          ?person rdf:type umbel:Person . }}}`
	`SELECT DISTINCT ?person`
	`WHERE {`
	`{?person skos:prefLabel "John Coltrane" .`
	` ?person  rdf:type umbel:Artist. }`
	`UNION {`
	`  ?person semset:hasSemset ?sems  .`
	`  ?sems skos:prefLabel "John Coltrane" .`
	`  ?person rdf:type umbel:Artist . }}`

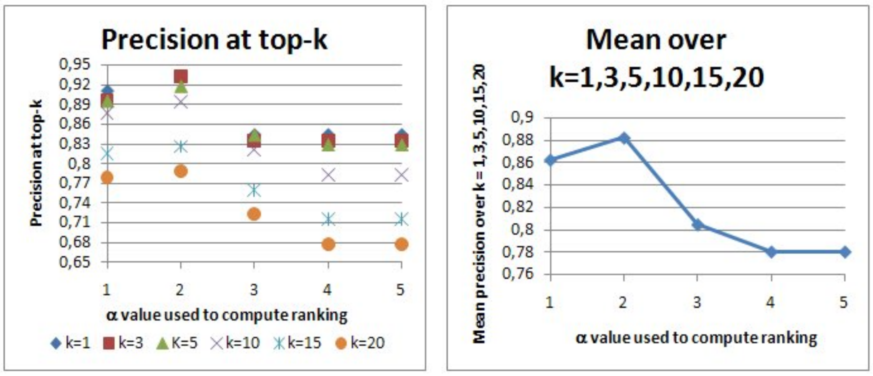
Table 6.2: Example Queries

each iteration over the queries, the evidences found for learning new mappings or for modifying the confidence of existing mappings were evaluated and mappings were accordingly modified. The evidences were automatically obtained by comparing results with the ground truth. The next iteration used the available mappings as updated in the previous iteration. The details about the mapping learning process were presented in Section 5.4 and will not be repeated here.

To evaluate the obtained results, a global evaluation was performed, i.e. the top-k *integrated* results of the same query over all available information sources were compared with the ground truth. Two experiments were performed, one for computing the required α value in the ranking function, and another for evaluating the ranking algorithm performance, with the determined α value.

6.3.2.1 Parameter Finding

In the first part of the experiments α was determined using the $QSet1$ query set. Results of each query, strategy and information source were accumulated and ranked, and compared with the ground truth. Figure 6.1(a) presents how different α values influence the precision of the results. It can be seen that the precision obtained by using an $\alpha = 2$ value gives the best results. As expected, an $\alpha = 1$ value gives less good results, since in this case one factor in the ranking function is disregarded. Also, precision is best for k values roughly between k=3 and k=10, showing that the ranking function is tailored for finding correct results in the top of the list. Figure 6.1(b) presents the mean precision across different k values, and it shows a similar behavior as in Figure 6.1(a), proving that regardless of the k value to be used, the best value obtained is $\alpha = 2$.



(a) Results per k (b) Mean Results

Figure 6.1: Finding α

6.3.2.2 Evaluating Precision at Top-K

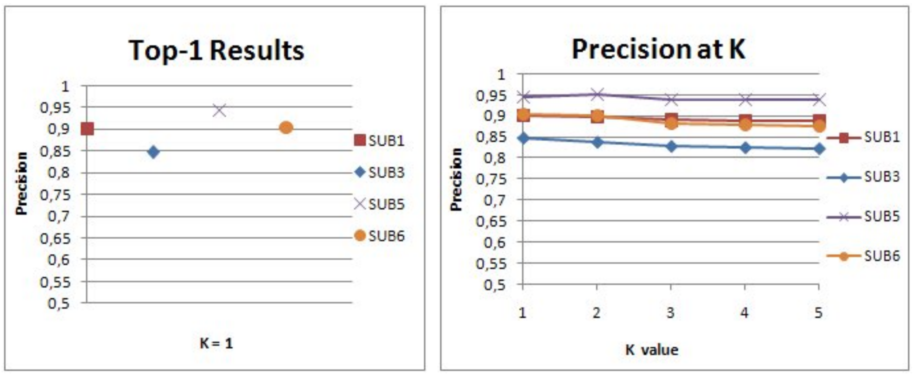
The second part of the evaluation uses the bigger set of queries, $QSet2$, and evaluates the precision at top-k of the presented approach using the obtained ranking parameter α. Precision at top-1 to top-5 for the query results was inspected. The mean results obtained from running this evaluation over all presented datasets, iterations and strategies is presented in Table 6.3, and by strategy in Figure 6.2(b). The precision was computed by considering all queries, also the ones having less than k results. It can be seen that there is not much difference between the obtained precision results at top-1 to top-5, all of them being around 0.9, which is considered to be a good precision for this lightweight integration approach. In Figure 6.2(a) the results for the top-1 precision for each of the considered strategies is presented. As the reader can notice, most of the strategies have high precision, being notable that strategies $\mathcal{SUB}^1$ and $\mathcal{SUB}^5$ give the best results. Strategy $\mathcal{SUB}^3$ shows the worst precision results at top-1, which is an indicator that the usage of this strategy for the presented settings needs to be revised (strategy success depends partly on the confidence and completeness of the initial mappings and the way the strategy proceeds to relax the query).

K	1	2	3	4	5
Precision	0.901	0.898	0.883	0.880	0.879

Table 6.3: Precision at Top-K

6.3.2.3 Performance

The execution time was measured by query on each information source, and the mean value over all queries and information sources was computed: 1.2 seconds. Worst case execution occurred at the first mapping-learning iteration (new mappings are still not learned) for strategy $\mathcal{SUB}^3$ - 3.38 minutes. It is worth noticing that in the worst case for this strategy (no mapping found for any element), all ontological elements (concepts and properties) in the query are converted to wildcards, leading to a very intense form of query relaxation (for a discussion about SPARQL complexity please see (112)). Also, the slow execution time in the worst case was also due to the fact that the query time depends a lot on the underlying used repository, Sesame in this case.



(a) Precision at Top-1 for Each Strategy (b) Precision for Top-K

Figure 6.2: Overall Precision

7 Conclusions and Future Directions

This thesis presented a lightweight information integration approach that relies on the pay-as-you-go idea for dealing with partial mappings or incomplete information. This chapter concludes the work by summarizing the developed approaches and highlighting the main research contributions. Furthermore, this chapter presents possible future research directions in the areas covered by this thesis, which came up as a result of the investigations performed during this work.

7.1 Summary

The large amount of information sources arising every day, and their fast change, make it necessary to develop lightweight integration methods, for accessing and taking advantage of this valuable resources. In this thesis an approach for solving the problem of integrating heterogeneous structured and semi-structured information sources (containing information in e.g. RDF) in a pay-as-you-go fashion was presented. The approach relies on the automatic computation of partial mappings between the user view on the domain of the Semantic Knowledge Spaces (SKS) and the different information sources available in the SKS expressed as simple ontologies. A novel approach to compute those mappings was presented in Chapter 4. This approach combines lexical, structure-based, and SAT-based methods for finding matches between ontologies, taking into account the *modeling perspective*. The *modeling perspective* is used to better detect the intended meaning of the ontological elements, by considering their modeling contexts. The approach was evaluated using a state of the art evaluation framework. The evaluation showed good results for the targeted cases, proving its applicability.

Subsequently, a flexible information integration approach which addresses the dynamic integration needs in a SKS, where only partial mappings are defined between the sources to be integrated, was presented in Chapter 5. In addition to exploiting defined mappings, substitution strategies are employed, which are inspired by the idea of using wildcards in querying and filtering tasks. Starting from a triple based query language, mappings are used to rewrite query elements, while unmapped ontological elements appearing in the query are substituted in a controlled way with variables, leading to a controlled form of query relaxation. In addition, the approach also takes advantages of the obtained evidences for refining the existing mapping in a pay-as-you-go fashion. This is based on the results of executing the relaxed queries over the different information sources in the SKS. Different strategies for replacing non-matched ontology elements with variables were discussed and evaluated over real-world data sets with very promising results, not only in precision, but also in the improvement of the mappings, showing its feasibility.

Chapter 6 completes the work presented in this thesis by describing a novel approach for ranking results of such rewritten and relaxed queries over different data sources in an SKS. The ranking function is based on the confidence of the rewriting process, and the degree of introduced relaxation. For computing this ranking, the confidence of each mapping employed in the query rewriting process - in combination with the amount and position of wildcards introduced in the reformulated query - are used. These factors are combined in a weighted fashion, to produce a ranking value for the results of executing the reformulated query on a specific information source. Results for an original query, obtained by executing reformulated versions of this query over different information sources can in this way be combined. Results considered to be more accurate are presented higher in the result list than results considered to be less reliable. Extensive evaluations performed over real world datasets show the effectiveness and efficiency of the introduced ranking function. They also help to decide which weights and strategies perform better for the task at hand.

7.2 Contributions

This work presents a Lightweight Information Integration approach for heterogeneous RDF data sources in a pay-as-you-go fashion. For this, several aspects of the generally known information integration task were tackled:

- An automatic approach for computing mappings between ontologies by considering the *modeling perspective* (see Chapter 4): This approach is based on a combination lexical and structure-based techniques integrated with a SAT-based approach, to detect matches and to find the relation holding between elements in two ontologies under consideration of the context of these elements. The evaluation performed using a well established evaluation framework show in which cases the proposed approach presents advantages and in which cases other approaches are more suitable for this task. The approach goes beyond existing approaches by employing the *modeling perspective* as an additional criteria for detecting matching elements. In this way, only elements having the same intent, this is, modeled in a similar fashion will be selected as matching. Obtained results show good precision for the cases targeted by this approach.

- A lightweight approach to integrate heterogeneous semi-structured data sources at query time, based on partial mappings used to rewrite queries and a controlled relaxation of query elements for which there is no known mapping: Advantage is taken out of the unknown results obtained through the relaxation of the queries in order to learn new mappings between the elements employed in the query and the ontologies describing the data sources (see Chapter 5). In this way future rewritten and relaxed queries give better results, improving the quality of answers over time. Evaluations showed the good precision obtained in few iterations, as well as the strategies being more suited to minimize the cold-start problem. The main advantages of this approach are its flexibility, its on-the-fly cheap adaptability and its ability of providing highly precise results with low initial efforts, improving further the results quality over time.

- A ranking strategy for these reformulated queries, based on the introduced degree of relaxation and on the confidence of the mappings employed in the rewriting process (see Chapter 6). This ranking approach enables to rank the results coming from a given source depending on the confidence to obtain right results out of it, since more relaxation might add more incorrect results. The same holds for less confident mappings employed in the rewriting process. The evaluations performed showed the feasibility of the proposed approach to rank the results based on the query reformulation details, helping in obtaining the expected results in the first places of the result list. This approach goes beyond related approaches by combining these two elements, the degree of relaxation and the confidence of employed mappings, for ranking reformulated queries, giving good precision values for top-ranked results.

7.3 Future Directions

Since the presented results are promising, an interesting area for future work to the presented query reformulation approach is extending the complexity of the considered query structures, going beyond the conjunctive combination of selection predicates as used in the current system. In addition, the impact of more complex mapping relationships - so far only one-to-one mappings between properties and concepts were considered - is a further direction for extending the query reformulation approach. Another future work could encompass the extension of query processors to better support the recognition of successful property and concept couplings (see Section 5.4), which are needed for automatic mapping refinement as well as for the suggestion of new mappings to the user. While at present this is done as a post-processing step, better integration with query processing should further improve efficiency.

The usage of the query ranking value for deciding on executing a query (or not) on a given information source is also an interesting idea to be explored. This could serve in two directions: 1) to avoid query execution of those queries having a higher relaxation degree, increasing thus query execution performance, and 2) for finding a trade-of between result precision and recall of unknown but relevant information.

The inclusion of an entity resolution approach would allow to retrieve only partial results from different sources, and to assemble these parts in order to answer a given query as good as possible. The ranking approach should then also take into account not only the query reformulation process, but also how much information is obtained in the results from a given source for a given query, and the confidence of each of these results, in order to compute an accurate ranking of the results (opposite to ranking the queries as presented in Chapter 6).

This chapter presented a summary and the main contributions of the work described in this thesis, followed by some future work directions which were detected while performing this work.

Appendix A: Short Curriculum Vitae

Rodolfo Enrique Stecher, born on June 14th 1972, in Asunción, Paraguay.

Degrees and Seminars:

1994	Project Management Seminar Universidad Católica Nuestra Señora de la Asunción Asunción, Paraguay.
1999	Ingeniero Informático degree at Universidad Católica Nuestra Señora de la Asunción Asunción, Paraguay.
2004	Project Management (I) Seminar Fraunhofer Gesellschaft, Kaufering, Germany.
2007	Project Management Seminar Leibniz Universität Hannover, Germany.
2008	Leadership Skills: Leading Teams Seminar Leibniz Universität Hannover, Germany.
2008	Leadership Skills: Conflict Resolution Seminar Leibniz Universität Hannover, Germany.

Work Experience:

Nov. 2006 -	Project management and development at Forschungszentrum L3S, Leibniz Universität Hannover.
Feb. 2001 - Oct. 2006	Project development at Fraunhofer Gesellschaft, Darmstadt, Germany.
Nov. 1998 - Jan. 2001	Consultant at Soluziona (today `www.indra.es`) OPEN SGC - ANDE Project, Asunción, Paraguay.
Oct. 1997 - Oct. 1998	GIS programmer at Digitopar S.A. Cadastral maps of Ciudad del Este city project Asunción, Paraguay.

7. APPENDIX A: SHORT CURRICULUM VITAE

Selected Publications:

2010	*Query Ranking in Information Integration* Rodolfo Stecher, Stefania Costache, Claudia Niederée, Wolfgang Nejdl. To appear in Proceedings of the 22nd International Conference on Advanced Information Systems Engineering (CAiSE'10), June 9-11, 2010, Hammamet, Tunisia.
2008	*Social Recommendations of Content and Metadata* Rodolfo Stecher, Gianluca Demartini, Claudia Niedeée. In Proceedings of the 10th International Conference on Information Integration and Web-based Applications and Services (iiWAS), November 24-26, 2008, Linz, Austria.
2008	*Query rewriting for Lightweight Information Integration* Rodolfo Stecher, Claudia Niedeée, Wolfgang Nejdl. In Proceedings of the Third IEEE International Conference on Digital Information Management (ICDIM), November 13-16, 2008, London, UK.
2008	*Wildcards for Lightweight Information Integration in Virtual Desktops* Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl. In Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM), October 26-30, 2008, Napa Valley, California, USA.
2008	*Adaptive ontology re-use: finding and re-using sub-ontologies* Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl, Paolo Bouquet. In the International Journal of Web Information Systems (IJWIS) 4(2), pp. 198-214, 2008.
2007	*Sub-Ontology Discovery for Adaptive Re-use* Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl, Paolo Bouquet. In Kotsis, G., Taniar, D., Pardede, E., Ibrahim, I.K., eds.: Proceedings of The Ninth International Conference on Information Integration and Web-based Applications and Services (iiWAS). Volume 229 of books@ocg.at., Austrian Computer Society (2007).

References

[1] Foto N. Afrati, Manolis Gergatsoulis, and Theodoros G. Kavalieros. **Answering Queries Using Materialized Views with Disjunctions**. In *ICDT '99: Proceedings of the 7th International Conference on Database Theory*, pages 435–452, London, UK, 1999. Springer-Verlag.

[2] Jose-Luis Ambite, Vinay K. Chaudhri, Richard Fikes, Jessica Jenkins, Sunil Mishra, Maria Muslea, Tomas Uribe, and Guizhen Yang. **Design and Implementation of the CALO Query Manager**. In *Innovative Applications of Artificial Intelligence*. AAAI, July 2006.

[3] José Luis Ambite, Vinay K. Chaudhri, Richard Fikes, Jessica Jenkins, Sunil Mishra, Maria Muslea, Tomás E. Uribe, and Guizhen Yang. **Integration of Heterogeneous Knowledge Sources in the CALO Query Manager**. In Robert Meersman, Zahir Tari, Pilar Herrero, Gonzalo Méndez, Lawrence Cavedon, David Martin, Annika Hinze, George Buchanan, María S. Pérez, Víctor Robles, Jan Humble, Antonia Albani, Jan L. G. Dietz, Hervé Panetto, Monica Scannapieco, Terry A. Halpin, Peter Spyns, Johannes Maria Zaha, Esteban Zimányi, Emmanuel Stefanakis, Tharam S. Dillon, Ling Feng, Mustafa Jarrar, Jos Lehmann, Aldo de Moor, Erik Duval, and Lora Aroyo, editors, *OTM Workshops*, **3762** of *Lecture Notes in Computer Science*, pages 30–32. Springer, 2005.

[4] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. **A Survey of Peer-to-Peer Content Distribution Technologies**. *ACM Comput. Surv.*, **36**(4):335–371, 2004.

[5] Aperture. **Aduna Aperture**. `http://aperture.sourceforge.net/`.

[6] Yigal Arens, Craig A. Knoblock, and Wei-Min Shen. **Query Reformulation for Dynamic Information Integration**. *J. Intell. Inf. Syst.*, **6**(2-3):99–130, 1996.

[7] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[8] Qingyuan Bai, Jun Hong, and Michael F. McTear. **Query Rewriting Using Views in the Presence of Inclusion Dependencies**. In *WIDM '03: Proceedings of the 5th ACM international workshop on Web information and data management*, pages 134–138, New York, NY, USA, 2003. ACM.

[9] Qingyuan Bai, Jun Hong, Michael F. McTear, and Hui Wang. **Bucket-Based Query Rewriting with Disjunctive Data Source**. In *WI '04: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 566–569, Washington, DC, USA, 2004. IEEE Computer Society.

[10] Mark Baillie, Leif Azzopardi, and Fabio Crestani. **Towards Better Measures: Evaluation of Estimated Resource Description Quality for Distributed IR**. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, page 41, New York, NY, USA, 2006. ACM.

[11] Massimo Benerecetti, Paolo Bouquet, and Chiara Ghidini. **On the Dimensions of Context Dependence: Partiality, Approximation, and Perspective**. In V. Akman, P. Bouquet, R. Thomason, and R.A. Young, editors, *Proceedings of the Third International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT'01)*, **2116** of *LNCS*, pages 59–72. Springer, 2001.

[12] Omar Benjelloun, Hector Garcia-Molina, David Menestrina, Qi Su, Steven Euijong Whang, and Jennifer Widom. **Swoosh: A Generic Approach to Entity Resolution**. *The VLDB Journal*, **18**(1):255–276, 2009.

[13] Philip A. Bernstein, Fausto Giunchiglia, Anastasios Kementsietsidis, John Mylopoulos, Luciano Serafini, and Ilya Zaihrayeu. **Data Management for Peer-to-Peer Computing : A Vision**. In *WebDB*, pages 89–94, 2002.

[14] Pim Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, Universiteit Twente, September 1997.

[15] P. Bouquet, B. Magnini, L. Serafini, and S. Zanobini. **A SAT-based Algorithm for Context Matching**. In *In CONTEXT*, pages 66–79, 2003.

[16] Paolo Bouquet, Luciano Serafini, and Stefano Zanobini. **Semantic Coordination: A New Approach and an Application**. In Fensel et al. (51), pages 130–145.

[17] Paolo Bouquet, Luciano Serafini, and Stefano Zanobini. **Peer-to-Peer Semantic Coordination**. *Journal of Web Semantics*, **2**(1), 2005.

[18] Paolo Bouquet, Luciano Serafini, and Stefano Zanobini. **Semantic Coordination of Heterogeneous Classification Schemas**. In S. Staab and H. Stuckenschmidt, editors, *Peer-to-peer and Semantic Web*. Springer, 2005.

[19] E. Bozsak, Marc Ehrig, Siegfried Handschuh, Andreas Hotho, Alexander Maedche, Boris Motik, Daniel Oberle, Christoph Schmitz, Steffen Staab, Ljiljana Stojanovic, Nenad Stojanovic, Rudi Studer, Gerd Stumme, York Sure, Julien Tane, Raphael Volz, and Valentin Zacharias. **KAON - Towards a Large Scale Semantic Web**. In Kurt Bauknecht, A. Min Tjoa, and Gerald Quirchmayr, editors, *E-Commerce and Web Technologies, Third International Conference, EC-Web 2002, Aix-en-Provence, France, September 2-6, 2002, Proceedings*, **2455** of *LNCS*, pages 304–313. Springer, 2002.

[20] I. Brunkhorst, P. A. Chirita, S. Costache, J. Gaugaz, E. Ioannou, T. Iofciu, E. Minack, W. Nejdl, and R. Paiu. **The Beagle++ Toolbox: Towards an Extendable Desktop Search Architecture**. In *Proceedings of Semantic Desktop and Social Semantic Collaboration Workshop, ISWC*, 2006.

[21] Meru Brunn, Yllias Chali, and Christopher Pinchak. **Text Summarization Using Lexical Chains**. In *in Document Understanding Conference (DUC*, pages 135–140, 2001.

[22] Jamie Callan. **Distributed Information Retrieval**. In *In: Advances in Information Retrieval*, pages 127–150. Kluwer Academic Publishers, 2000.

[23] Jamie Callan, James Allan, Charles L. A. Clarke, Susan Dumais, David A. Evans, Mark Sanderson, and ChengXiang Zhai. **Meeting of the MINDS: An Information Retrieval Research Agenda**. *SIGIR Forum*, **41**(2):25–34, 2007.

[24] Diego Calvanese and Giuseppe De Giacomo. **Data Integration: A Logic-based Perspective**. *AI Mag.*, **26**(1):59–70, 2005.

[25] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. **A Framework for Ontology Integration**. In Isabel F. Cruz, Stefan Decker, Jérôme Euzenat, and Deborah L. McGuinness, editors, *SWWS*, pages 303–316, 2001.

[26] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. **Description Logics for Information Integration**. In *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, pages 41–60, London, UK, 2002. Springer-Verlag.

[27] M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. W. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, and E. L. Wimmers. **Towards Heterogeneous Multimedia Information Systems: The Garlic Approach**. In *RIDE '95: Proceedings of the 5th International Workshop on Research Issues in Data Engineering-Distributed Object Management (RIDE-DOM'95)*, page 124, Washington, DC, USA, 1995. IEEE Computer Society.

[28] Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. **Jena: Implementing the Semantic Web Recommendations**. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 74–83, New York, NY, USA, 2004. ACM. `http://jena.sourceforge.net/`.

[29] Stefano Ceri and Jennifer Widom. **Deriving Production Rules for Incremental View Maintenance**. In *VLDB '91: Proceedings of the 17th International Conference on Very Large Data Bases*, pages 577–589, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

[30] Hans Chalupsky. **OntoMorph: A Translation System for Symbolic Knowledge**. In *Principles of Knowledge Representation and Reasoning*, pages 471–482, 2000.

[31] Laura Chiticariu, Phokion G. Kolaitis, and Lucian Popa. **Interactive Generation of Integrated Schemas**. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 833–846, New York, NY, USA, 2008. ACM.

[32] Oscar Corcho, Mariano Fernández-López, and Asunción Gómez-Pérez. **Methodologies, Tools and Languages for Building Ontologies: Where is their Meeting Point?** *Data Knowl. Eng.*, **46**(1):41–64, 2003.

[33] Philippe Cudré-Mauroux, Suchit Agarwal, Adriana Budura, Parisa Haghani, and Karl Aberer. **Self-organizing Schema Mappings in the GridVine Peer Data Management System**. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 1334–1337. VLDB Endowment, 2007.

[34] DBPEDIA. `http://wiki.dbpedia.org/`.

[35] Barry Devlin. *Data Warehouse: From Architecture to Implementation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.

[36] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal Doshi, and Joel Sachs. **Swoogle: A Search and Metadata Engine for the Semantic Web**. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 652–659, New York, NY, USA, 2004. ACM.

[37] Anhai Doan, Pedro Domingos, and Alon Halevy. **Learning to Match the Schemas of Data Sources: A Multistrategy Approach**. *Mach. Learn.*, **50**(3):279–301, 2003.

[38] AnHai Doan and Alon Y. Halevy. **Semantic-integration Research in the Database Community**. *AI Mag.*, **26**(1):83–94, 2005.

[39] Xin Dong, Alon Halevy, and Jayant Madhavan. **Reference Reconciliation in Complex Information Spaces**. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 85–96, New York, NY, USA, 2005. ACM.

[40] Xin Dong and Alon Y. Halevy. **Malleable Schemas: A Preliminary Report**. In AnHai Doan, Frank Neven, Robert McCann, and Geert Jan Bex, editors, *WebDB*, pages 139–144, 2005.

[41] Xin Luna Dong, Alon Halevy, and Cong Yu. **Data Integration With Uncertainty**. *The VLDB Journal*, **18**(2):469–500, 2009.

[42] Ebiquity Group, UMBC, 2006. `http://swoogle.umbc.edu/`.

[43] Johann Eder, Christian Koncilia, and Tadeusz Morzy. **The COMET Metamodel for Temporal Data Warehouses**. In *CAiSE '02: Proceedings of the 14th International Conference on Advanced Information Systems Engineering*, pages 83–99, London, UK, 2002. Springer-Verlag.

[44] Marc Ehrig and Steffen Staab. **QOM - Quick Ontology Mapping**. In *International Semantic Web Conference*, pages 683–697, 2004.

[45] Marc Ehrig, Steffen Staab, and York Sure. **Bootstrapping ontology alignment methods with APFEL**. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1148–1149, New York, NY, USA, 2005. ACM.

[46] Jérôme Euzenat. **Evaluating Ontology Alignment Methods**. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[47] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007.

[48] Jérôme Euzenat, Heiner Stuckenschmid, and Mikalai Yatskevich. **Introduction to the Ontology Alignment Evaluation 2005**. In Benjamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors, *Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies*, **156** of *CEUR Workshop Proceedings*, Banff, Canada, October 2005. CEUR.

[49] Ronald Fagin, Phokion G. Kolaitis, Alan Nash, and Lucian Popa. **Towards a Theory of Schema-Mapping Optimization**. In *PODS '08: Pro-*

ceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 33–42, New York, NY, USA, 2008. ACM.

[50] C. Fellbaum. *Wordnet: An Electronic Lexical Database.* MIT Press, 1998.

[51] Dieter Fensel, Katia P. Sycara, and John Mylopoulos, editors. *The Semantic Web - ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003, Proceedings*, **2870** of *Lecture Notes in Computer Science*. Springer, 2003.

[52] S. Flesca and S. Greco. **Rewriting Queries Using Views**. *IEEE Trans. on Knowl. and Data Eng.*, **13**(6):980–995, 2001.

[53] Flickr. `http://www.flickr.com`.

[54] Michael Franklin, Alon Halevy, and David Maier. **From Databases to Dataspaces: A New Abstraction for Information Management**. *SIGMOD Rec.*, **34**(4):27–33, 2005.

[55] Freebase. `http://www.freebase.com`.

[56] Marc Friedman, Alon Levy, and Todd Millstein. **Navigational Plans for Data Integration**. In *AAAI '99/IAAI '99: Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, pages 67–73, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence.

[57] Terry Gaasterland. **Cooperative Answering through Controlled Query Relaxation**. *IEEE Expert: Intelligent Systems and Their Applications*, **12**(5):48–59, 1997.

[58] Michel Galley and Kathleen McKeown. **Improving Word Sense Disambiguation in Lexical Chaining.** In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 1486–1488. Morgan Kaufmann, 2003.

[59] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. **Sweetening WORDNET with DOLCE**. *AI Mag.*, **24**(3):13–24, 2003.

[60] Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey Ullman, and Jennifer Widom. **Integrating**

and Accessing Heterogeneous Information Sources in TSIMMIS. In *In Proceedings of the AAAI Symposium on Information Gathering*, pages 61–64, 1995.

[61] Hector Garcia-Molina, Yannis Papakonstantinou, Dallan Quass, Anand Rajaraman, Yehoshua Sagiv, Jeffrey Ullman, Vasilis Vassalos, and Jennifer Widom. **The TSIMMIS Approach to Mediation: Data Models and Languages**. *J. Intell. Inf. Syst.*, **8**(2):117–132, 1997.

[62] Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors. *The Semantic Web - ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, Proceedings*, **3729** of *Lecture Notes in Computer Science*. Springer, 2005.

[63] Fausto Giunchiglia and Mikalai Yatskevich. **Element Level Semantic Matching**. In *In Proceedings of Meaning Coordination and Negotiation workshop at ISWC'04*, 2004.

[64] Ashish Gupta and Inderpal Singh Mumick. **Maintenance of Materialized Views: Problems, Techniques, and Applications**. *Materialized views: techniques, implementations, and applications*, pages 145–157, 1999.

[65] Marc Gyssens and Laks V.S. Lakshmanan. **A Foundation for Multi-Dimensional Databases**. In M. Jarke, editor, *International Conference on Very Large Data Bases*, pages 106–115. ACM Press, 1997.

[66] Peter Haase, Frank van Harmelen, Zhisheng Huang, Heiner Stuckenschmidt, and York Sure. **A Framework for Handling Inconsistency in Changing Ontologies**. In Gil et al. (62), pages 353–367.

[67] Alon Halevy, Michael Franklin, and David Maier. **Principles of Dataspace Systems**. In *PODS '06: Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–9, New York, NY, USA, 2006. ACM.

[68] Alon Y. Halevy. **Answering Queries Using Views: A Survey**. *The VLDB Journal*, **10**(4):270–294, 2001.

[69] Alon Y. Halevy, Zachary G. Ives, Peter Mork, and Igor Tatarinov. **Piazza: Data Management Infrastructure for Semantic Web Applica-**

tions. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 556–567, New York, NY, USA, 2003. ACM.

[70] Y. Halevy, G. Ives, Dan Suciu, and Igor Tatarinov. **Schema Mediation for Large-Scale Semantic Data Sharing**. *The VLDB Journal*, **14**(1):68–83, 2005.

[71] Birgit Hamp and Helmut Feldweg. **GermaNet - a Lexical-Semantic Net for German**. In *In Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15, 1997.

[72] Graeme Hirst and David St-Onge. **Lexical Chains as Representation of Context for the Detection and Correction Malapropisms**. WordNet: An electronic lexical database and some of its applications. Cambrige, MA: The MIT Press, 1997.

[73] Markus Holi and Eeru Hyvönen. **A Method for Modeling Uncertainty in Semantic Web Taxonomies**. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 296–297, New York, NY, USA, 2004. ACM.

[74] Eduard Hovy, Chin-Yew Lin, and Liang Zhou. **A BE-based Multi-document Summarizer with Sentence Compression**. In *Proceedings of Multilingual Summarization Evaluation (ACL 2005 workshop)*, Ann Arbor, MI, 2005.

[75] Ekaterini Ioannou, Claudia Niederée, and Wolfgang Nejdl. **Probabilistic Entity Linkage for Heterogeneous Information Spaces**. In Zohra Bellahsene and Michel Léonard, editors, *CAiSE*, **5074** of *Lecture Notes in Computer Science*, pages 556–570. Springer, 2008.

[76] Shawn R. Jeffery, Michael J. Franklin, and Alon Y. Halevy. **Pay-as-you-go User Feedback for Dataspace Systems**. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 847–860, New York, NY, USA, 2008. ACM.

[77] Ricardo Kawase, Enrico Minack, Wolfgang Nejdl, Samur Arajo, and Daniel Schwabe. **Incremental End-user Query Construction for the Semantic Desktop.** In Joaquim Filipe and Jos Cordeiro, editors, *WEBIST*, pages 270–275. INSTICC Press, 2009.

[78] Jaehong Kim, Minsu Jang, Young-Guk Ha, Joo-Chan Sohn, and Sang Jo Lee. **MoA: OWL Ontology Merging and Alignment Tool for the Semantic Web**. In *Innovations in Applied Artificial Intelligence: 18th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2005, Bari, Italy, June 22-24, 2005*, page 722. Springer-Verlag, 2005.

[79] Thomas Kirk, Alon Y. Levy, Yehoshua Sagiv, and Divesh Srivastava. **The Information Manifold**. In *In Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Enviroments*, pages 85–91, 1995.

[80] Karoline Kirmse. *Eine Methode zur Berechnung der Ähnlichkeit zwischen Konzepten unter Berücksichtigung der Modellierungsperspektive.* Master's thesis, Universität Konstanz, Universitätsstr. 10, 78457 Konstanz, 2006.

[81] Graham Klyne and Jeremy J. Carroll. *Resource Description Framework (RDF): Concepts and Abstract Syntax W3C Recommendation*, February 2004. `http://www.w3.org/TR/rdf-concepts/`.

[82] Cornelis H. A. Koster. **Head/modifier Pairs for Everyone**. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 466–466, New York, NY, USA, 2003. ACM Press.

[83] Cornelis H.A. Koster. **Head-Modifier Frames for Everyone**. Demo at SIGIR 2003, Toronto, July 2003. Proceedings SIGIR 2003 pg. 466.

[84] Nick Koudas, Chen Li, Anthony K. H. Tung, and Rares Vernica. **Relaxing Join and Selection Queries**. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 199–210. VLDB Endowment, 2006.

[85] Andreas Langegger. **Virtual Data Integration on the Web: Novel Methods for Accessing Heterogeneous and Distributed Data with Rich Semantics**. In *iiWAS '08: Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*, pages 559–562, New York, NY, USA, 2008. ACM.

[86] Douglas Lenat and R.V. Guha. *Building Large Knowledge Based Systems.* Addison Wesley, Reading, Massachusetts, 1990.

[87] Maurizio Lenzerini. **Data Integration: A Theoretical Perspective**. In *PODS '02: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, New York, NY, USA, 2002. ACM.

[88] Michael Lesk. **Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone**. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, New York, NY, USA, 1986. ACM Press.

[89] Leonid Libkin and Cristina Sirangelo. **Data Exchange and Schema Mappings in Open and Closed Worlds**. In *PODS '08: Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 139–148, New York, NY, USA, 2008. ACM.

[90] Robert M. MacGregor and Raymond Bates. **The Loom Knowledge Representation Language**. Technical report, Information Sciences Institute, University of Southern California, May 1987.

[91] Alexander Maedche and Steffen Staab. **Ontology Learning for the Semantic Web**. *IEEE Intelligent Systems*, **16**(2):72–79, 2001.

[92] Ioana Manolescu, Daniela Florescu, and Donald Kossmann. **Answering XML Queries on Heterogeneous Data Sources**. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, pages 241–250, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[93] Sally Mcclean, Bryan Scotney, Philip Morrow, and Kieran Greer. **Integrating Semantically Heterogeneous Aggregate Views of Distributed Databases**. *Distrib. Parallel Databases*, **24**(1-3):73–94, 2008.

[94] Martin Michalowski, Jose Luis Ambite, Snehal Thakkar, and Rattapoom Tuchinda. **Retrieving and Semantically Integrating Heterogeneous Data from the Web**. *IEEE Intelligent Systems*, **19**(3):72–79, 2004.

[95] Peter Mika. **Ontologies Are Us: A Unified Model of Social Networks and Semantics**. In *International Semantic Web Conference*, **3729** of *Lecture Notes in Computer Science*, pages 522–536. International Semantic Web Conference 2005, Springer, November 2005.

[96] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. **Introduction to WordNet: An On-line Lexical Database**. *Journal of Lexicography*, **3**(4):235–244, 1990.

[97] Prasenjit Mitra, Natasha F. Noy, and Anuj Jaiswal. **OMEN: A Probabilistic Ontology Mapping Tool**. In Gil et al. (62), pages 537–547.

[98] Jane Morris and Graeme Hirst. **Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text**. *Comput. Linguist.*, **17**(1):21–48, 1991.

[99] Tadeusz Morzy and Robert Wrembel. **On Querying Versions of Multiversion Data Warehouse**. In *DOLAP '04: Proceedings of the 7th ACM international workshop on Data warehousing and OLAP*, pages 92–101, New York, NY, USA, 2004. ACM.

[100] Ion Muslea and Thomas J. Lee. **Online Query Relaxation via Bayesian Causal Structures Discovery**. In Manuela M. Veloso and Subbarao Kambhampati, editors, *AAAI*, pages 831–836. AAAI Press / The MIT Press, 2005.

[101] Ullas Nambiar and Subbarao Kambhampati. **Answering Imprecise Queries over Autonomous Web Databases**. In *ICDE'06: International Conference on Data Engineering*, 2006.

[102] NEPOMUK. **The Social Semantic Desktop.** `http://nepomuk.semanticdesktop.org/`.

[103] SAP Community Network. `http://www.sdn.sap.com`.

[104] Ian Niles and Adam Pease. **Towards a Standard Upper Ontology**. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA, 2001. ACM.

[105] Henrik Nottelmann and Norbert Fuhr. **Evaluating Different Methods of Estimating Retrieval Quality for Resource Selection**. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 290–297, New York, NY, USA, 2003. ACM.

[106] Natalya F. Noy and Deborah L. McGuinness. **Ontology Development 101: A Guide to Creating Your First Ontology**. Technical Report KSL-01-05, Knowledge Systems Laboratory, Stanford University, Stanford, CA, 94305, USA, March 2001.

[107] Natalya F. Noy and Mark A. Musen. **The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping**. *Int. J. Hum.-Comput. Stud.*, **59**(6):983–1024, 2003.

[108] OAEI 2009 Preliminary Benchmark Results, 2009. `http://oaei.ontologymatching.org/2009/results/benchmarks.html`.

[109] Task-Force Ontologies. **NEPOMUK Ontologies**. Technical report, Nepomuk, 2007. `http://nepomuk.semanticdesktop.org/ontologies/`.

[110] Ontology Alignment Evaluation Initiative 2009 Benchmark, 2009. `http://oaei.ontologymatching.org/2009/`.

[111] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. **The PageRank Citation Ranking: Bringing Order to the Web**. Technical report, Stanford University, 1999.

[112] Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. **Semantics and Complexity of SPARQL**. *ACM Trans. Database Syst.*, **34**(3):1–45, 2009.

[113] Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. **MultiWordNet: Developing an Aligned Multilingual Database**. In *Proceedings of the First International Conference on Global WordNet.*, January 2002.

[114] Neoklis Polyzotis, Minos Garofalakis, and Yannis Ioannidis. **Approximate XML Query Answers**. In *SIGMOD '04: International Conference on Management of Data*, 2004.

[115] The Huffington Post. `http://www.huffingtonpost.com`.

[116] Racer. **Racer**, 2004. Available at: `http://www.sts.tu-harburg.de/~r.f.moeller/racer`.

[117] Erhard Rahm and Philip A. Bernstein. **A Survey of Approaches to Automatic Schema Matching**. *The VLDB Journal*, **10**(4):334–350, 2001.

[118] Anand Rajaraman, Yehoshua Sagiv, and Jeffrey D. Ullman. **Answering Queries Using Templates With Binding Patterns (extended abstract)**. In *PODS '95: Proceedings of the fourteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 105–112, New York, NY, USA, 1995. ACM.

[119] Elke A. Rundensteiner, Andreas Koeller, and Xin Zhang. **Maintaining Data Warehouses Over Changing Information Sources**. *Commun. ACM*, **43**(6):57–62, 2000.

[120] Stuart J. Russell, Peter Norvig, John F. Candy, Jitendra M. Malik, and Douglas D. Edwards. *Artificial intelligence: a modern approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[121] Sunita Sarawagi and Anuradha Bhamidipaty. **Interactive Deduplication Using Active Learning**. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–278, New York, NY, USA, 2002. ACM.

[122] Anish Das Sarma, Xin Dong, and Alon Halevy. **Bootstrapping Pay-as-you-go Data Integration Systems**. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 861–874, New York, NY, USA, 2008. ACM.

[123] Milad Shokouhi, Justin Zobel, and Yaniv Bernstein. **Distributed Text Retrieval from Overlapping Collections**. In *ADC '07: Proceedings of the eighteenth conference on Australasian database*, pages 141–150, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc.

[124] Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Natalya Fridman Noy, and Arnon Rosenthal, editors. *Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009*, **551** of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[125] Luo Si and Jamie Callan. **Unified Utility Maximization Framework for Resource Selection**. In *CIKM '04: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, pages 32–41, New York, NY, USA, 2004. ACM.

[126] Luo Si and Jamie Callan. **Modeling Search Engine Effectiveness for Federated Search**. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–90, New York, NY, USA, 2005. ACM.

[127] H. Grogory Silber and Kathleen F. McCoy. **Efficiently Computed Lexical Chains as an Intermediate Representation for Automatic Text Summarization**. *Computational Linguistics*, **28**(4):487–496, 2002.

[128] Fon Silvers. *Building and Maintaining a Data Warehouse*. CRC Press, 2008.

[129] Peter Spyns, Robert Meersman, and Mustafa Jarrar. **Data Modelling Versus Ontology Engineering**. *SIGMOD Rec.*, **31**(4):12–17, 2002.

[130] Rodolfo Stecher, Stefania Costache, Claudia Niederée, and Wolfgang Nejdl. **Query Ranking in Information Integration**. In *Accepted for publication at The 22nd International Conference on Advanced Information Systems Engineering (CAiSE'10)*, 2010.

[131] Rodolfo Stecher and Claudia Niederée. **Ontology Fitness - Supporting Ontology Quality beyond Logical Consistency**. In *Formal Ontologies Meet Industry Workshop (FOMI), June 9-10, Italy*, Nune 2005.

[132] Rodolfo Stecher, Claudia Niederée, and Wolfgang Nejdl. **Query Rewriting for Lightweight Information Integration**. In Pit Pichappan and Ajith Abraham, editors, *Proceedings of the Third IEEE International Conference on Digital Information Management (ICDIM)*, pages 375–380. IEEE, 2008.

[133] Rodolfo Stecher, Claudia Niederée, and Wolfgang Nejdl. **Wildcards for Lightweight Information Integration in Virtual Desktops**. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM'08)*, 2008.

[134] Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl, and Paolo Bouquet. **Sub-Ontology Discovery for Adaptive Re-use**. In Gabriele Kotsis, David Taniar, Eric Pardede, and Ismail Khalil Ibrahim, editors, *Proceedings of The Ninth International Conference on Information Integration and Web-based Applications and Services(iiWAS).* , **229** of *books@ocg.at*, pages 11–20. Austrian Computer Society, 2007.

[135] Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl, and Paolo Bouquet. **Adaptive Ontology Re-Use: Finding and Re-Using Sub-Ontologies**. *International Journal of Web Information Systems*, **4**(2):198–214, 2008.

[136] Nenad Stojanovic, Rudi Studer, and Ljiljana Stojanovic. **An Approach for the Ranking of Query Results in the Semantic Web**. In Fensel et al. (51), pages 500–516.

[137] Umberto Straccia and Raphaël Troncy. **Towards Distributed Information Retrieval in the Semantic Web: Query Reformulation Using the oMAP Framework.** In York Sure and John Domingue, editors, *ESWC*, **4011** of *Lecture Notes in Computer Science*, pages 378–392. Springer, 2006.

[138] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. **Knowledge Engineering: Principles and Methods**. *Data Knowledge Engineering*, **25**(1-2):161–197, 1998.

[139] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. **Yago: A Core of Semantic Knowledge**. In *International World Wide Web Conference*, New York, NY, USA, 2007.

[140] Elke Teich and Peter Fankhauser. **WordNet for Lexical Cohesion Analysis**. In P. Sojka, K. Pala, P. Smrz, C. Fellbaum, and P. Vossen, editors, *Proceedings of the 2nd Global WordNet Conference*, Brno, Czech Republic, 2004.

[141] Dimitri Theodoratos, Joachim Hammer, Manfred A. Jeusfeld, and Martin Staudt, editors. *Proceedings of the 3rd Intl. Workshop on Design and Management of Data Warehouses, DMDW'2001, Interlaken, Switzerland, June 4, 2001*, **39** of *CEUR Workshop Proceedings*. CEUR-WS.org, 2001.

[142] TreeTagger. **- a language independent part-of-speech tagger**. `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html`.

[143] Jeffrey D. Ullman. **Information Integration Using Logical Views**. In *ICDT '97: Proceedings of the 6th International Conference on Database Theory*, pages 19–40, London, UK, 1997. Springer-Verlag.

[144] UMBEL. `http://www.umbel.org/`.

[145] LUDGER VAN ELST AND MALTE KIESEL. **Generating and Integrating Evidence for Ontology Mappings**. In *Engineering Knowledge in the Age of the Semantic Web: Proceedings of the 14th International Conference, EKAW 2004*, **3257** of *LNAI*, pages 15–29, Heidelberg, 2004. Springer.

[146] GERTJAN VAN HEIJST, A. TH. (GUUS) SCHREIBER, AND BOB J. WIELINGA. **Using Explicit Ontologies in KBS Development**. *International Journal of Human-Computer Studies*, **46**(2-3):183–292, 1997.

[147] VIKEF. **Virtual Information and Knowledge Environment Framework.** `http://www.l3s.de/web/page54g.do?changedAlts=alt4g&sp=page22g&alt4g=page55g&kcond15g.att178a=150`.

[148] W3C. **OWL Web Ontology Language Semantics and Abstract Syntax**.

[149] W3C. **RDF Primer**, February 2004. `http://www.w3.org/TR/rdf-primer/`.

[150] W3C. **XQuery 1.0: An XML Query Language**, January 2007. `http://www.w3.org/TR/xquery/`.

[151] W3C. **SPARQL Query Language for RDF**, January 2008. `http://www.w3.org/TR/rdf-sparql-query/`.

[152] WIKIPEDIA. `http://www.wikipedia.org`.

[153] SOPHIA KATRENKO WILLEM ROBERT VON HAGE AND GUUS SCHREIBER. **A Method to Combine Linguistic Ontology-Mapping Techniques**. In Gil et al. (62), pages 732–744.

[154] WILLIAM E. WINKLER. **Overview of Record Linkage and Current Research Directions**. Technical report, U.S. Census Bureau, Washington, 2006.

[155] ROBERT WINTER AND ANKE GERICKE. **Teradata University Network - Ein Portal zur Unterstützung der Lehre in den Bereichen Business Intelligence, Data Warehousing und Datenbanken**. *Wirtschaftsinformatik*, **48**(4):276–281, 2006.

[156] ROBERT WREMBEL AND CHRISTIAN KONCILIA. *Data Warehouses And Olap: Concepts, Architectures And Solutions.* IRM Press, 2007.

[157] YOUTUBE. `http://www.youtube.com`.

[158] Cong Yu and Lucian Popa. **Constraint-based XML query rewriting for data integration**. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 371–382, New York, NY, USA, 2004. ACM.

[159] Stefano Zanobini. *Semantic Coordination - The Model and an Application to Schema Matching*. PhD thesis, DIT - University of Trento, March 2006.

[160] Xuan Zhou, Julien Gaugaz, Wolf-Tilo Balke, and Wolfgang Nejdl. **Query relaxation using malleable schemas**. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 545–556, New York, NY, USA, 2007. ACM.

[161] Yue Zhuge, Hector Garcia-Molina, and Janet L. Wiener. **The Strobe algorithms for multi-source warehouse consistency**. In *DIS '96: Proceedings of the fourth international conference on on Parallel and distributed information systems*, pages 146–157, Washington, DC, USA, 1996. IEEE Computer Society.

[162] Yue Zhuge, Hector Garcia-Molina, and Janet L. Wiener. **Multiple View Consistency for Data Warehousing**. In *ICDE '97: Proceedings of the Thirteenth International Conference on Data Engineering*, pages 289–300, Washington, DC, USA, 1997. IEEE Computer Society.

[163] Patrick Ziegler and Klaus R. Dittrich. **Three Decades of Data Integration - All Problems Solved?** In Ren Jacquart, editor, *18th IFIP World Computer Congress (WCC 2004), Volume 12, Building the Information Society*, **156** of *IFIP International Federation for Information Processing*, pages 3–12, Toulouse, France, August 22-27, 2004. Kluwer.