K-MORPH: KNOWLEDGE MORPHING VIA RECONCILIATION OF CONTEXTUALIZED SUB-ONTOLOGIES

by

Sajjad Hussain

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
March 2011

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "*K-MORPH*: KNOWLEDGE MORPHING VIA RECONCILIATION OF CONTEXTUALIZED SUB-ONTOLOGIES" by Sajjad Hussain in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated: March 29, 2011

External Examiner: _____

Research Supervisor: _____

Examining Committee: _____

_____

_____

Departmental Representative: _____

DALHOUSIE UNIVERSITY

DATE: March 29, 2011

AUTHOR: Sajjad Hussain

TITLE: *K-MORPH*: KNOWLEDGE MORPHING VIA RECONCILIATION OF CONTEXTUALIZED SUB-ONTOLOGIES

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: Ph.D. CONVOCATION: May YEAR: 2011

_____

Signature of Author

Table of Contents

List of Tables . . . . . **viii**

List of Figures . . . . . **x**

Abstract . . . . . **xii**

List of Abbreviations and Symbols Used . . . . . **xiii**

Acknowledgements . . . . . **xvi**

Chapter 1 Introduction **1**

1.1 Knowledge Sharing and Integration: Open Issues 1

1.2 Knowledge Morphing 3

1.3 Knowledge Representation and Morphing using Ontologies 4

1.4 The Role of Context in Knowledge Morphing 5
1.4.1 What is Context? 6
1.4.2 Problem-context in Knowledge Morphing 6

1.5 Our Solution Approach 7

1.6 $\mathcal{K}$-$\mathcal{MORPH}$: Technical Challenges and Research Contributions . . . 9
1.6.1 Extracting Contextualized Sub-ontologies 10
1.6.2 Merging Contextualized Sub-ontologies 10
1.6.3 Detecting and Resolving Inconsistencies 11

1.7 Outline 11

Chapter 2 Related Work: Comparing $\mathcal{K}$-$\mathcal{MORPH}$ with State-of-the-Art 13

2.1 Support from The Semantic Web Framework Towards Knowledge Integration 14

2.2 State-of-the-Art Approaches for Knowledge Integration 16
2.2.1 ECOIN 16
2.2.2 OpenKnowledge 17
2.2.3 Dynamic Sub-Ontology Evolution for Collaborative Problem Solving 18
2.2.4 Extracting Sub-ontology from Multiple Ontologies 20
2.2.5 Knowledge Selection Using Magpie 20

2.3 Comparing $\mathcal{K}$-$\mathcal{MORPH}$ with other State-of-the-Art Approaches . . 22

Chapter 3 $\mathcal{K}$-$\mathcal{MORPH}$: A Knowledge Morphing Framework 24

3.1 Knowledge Morphing: Research Challenges 24
3.1.1 Context Awareness 25
3.1.2 Semantic Interoperability 25
3.1.3 Dealing with Inconsistencies 26

3.2 $\mathcal{K}$-$\mathcal{MORPH}$: Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies 27
3.2.1 Preliminaries 28
3.2.2 Task # 1: Representing the Problem-context in $\mathcal{K}$-$\mathcal{MORPH}$. 29
3.2.3 Task # 2: Extracting Contextualized Sub-ontologies 30
3.2.4 Task # 3: Merging Contextualized Sub-ontologies 31
3.2.5 Task # 4: Detecting and Resolving Inconsistencies 32
3.2.6 $\mathcal{K}$-$\mathcal{MORPH}$ Process 33

3.3 $\mathcal{K}$-$\mathcal{MORPH}$ Applications: Morphing Prostate Cancer Clinical Pathways for Therapeutic Decision Support 33
3.3.1 Task # 1: Defining Problem-context: *therapeutic decision support* 35
3.3.2 Task # 2: Extracting Contextualized PC Sub-ontologies 35
3.3.3 Task # 3: Merging Contextualized PC Sub-ontologies 36
3.3.4 Task # 4: Resolving Inconsistencies in PC ontologies 37

Chapter 4 Extracting Contextualized Sub-ontologies 38

4.1 Structure-based Sub-ontology Extraction: Overview and Approaches 39

4.2 Preliminaries 41

4.3 Structure-based Approach for Extracting Contextualized Sub-ontologies 42
4.3.1 Concept Selection 43
4.3.2 Identification of Selected Concepts, Sub-concepts, and their Properties 44
4.3.3 Inheriting Properties from Super-concepts 45
4.3.4 Extraction of Complex Concepts 45
4.3.5 Dealing with Concept Axioms for Extracted Concepts 47
4.3.6 Dealing with Property Axioms for Extracted Properties 48
4.3.7 Extracting RDF-Graphs for Extracted Concepts 48
4.3.8 Extracting RDF-Graphs for Extracted Properties 50
4.3.9 Extraction with Variable Boundedness 51

4.4 Evaluation 52
4.4.1 Sub-ontology Extraction by MOVE 52
4.4.2 Sub-ontology Extraction by Seidenberg et. al. 53

4.4.3 Sub-ontology Extraction by Noy et. al. . . . 53
4.4.4 Sub-ontology Extraction by Our Approach . . . 55

4.5 Experiment and Results: Extracting Contextualized Sub-ontologies for Prostate Cancer Management . . . 56
4.5.1 Prostate Cancer Pathway Ontologies . . . 56
4.5.2 Extracting Contextualized Sub-ontologies . . . 57

4.6 Summary . . . 58

Chapter 5 Aligning and Merging Contextualized Sub-ontologies . . . 60

5.1 Ontology Matching, Alignment & Merging Systems . . . 61

5.2 Matching Complex Entities . . . 69
5.2.1 Matching Complex Entities: State-of-the-Art . . . 70

5.3 Preliminaries . . . 73

5.4 Ontology Alignment using Triple-based Ontology Matching (TOM) . 75

5.5 Ontology Alignment using Proof-based Ontology Matching (POM) . 78

5.6 Evaluation . . . 80
5.6.1 Ritze et. al. Matching Results . . . 81
5.6.2 TOM Matching Results . . . 82
5.6.3 POM Matching Results . . . 84

5.7 Merging Contextualized Sub-ontologies . . . 87
5.7.1 Merging Conference Ontologies . . . 91

5.8 Experiment and Results: Merging Contextualized Sub-ontologies for Prostate Cancer Management . . . 92
5.8.1 Extracted Contextualized Sub-ontologies . . . 92
5.8.2 Merging Contextualized Sub-ontologies . . . 93

5.9 Summary . . . 95

Chapter 6 Detecting and Resolving Inconsistencies in Ontologies Using Contradiction Derivations . . . 96

6.1 Dealing with Inconsistencies: State-of-the-Art . . . 97
6.1.1 Reasoning with an Inconsistent Ontology . . . 97
6.1.2 Detecting and Resolving Inconsistencies . . . 98

6.2 Preliminaries . . . 99

6.3 Inconsistency Detection via Contradiction Derivations . . . 103

6.4 Resolving Inconsistencies: Generating Minimal Inconsistent Resolve Candidates . . . 104
6.4.1 Computing MCTSs and AATSs in Euler . . . 105
6.4.2 Generating MIRCs and its Descendants . . . 106

6.5 Evaluation: Detecting and Resolving Inconsistencies in Prostate Cancer Ontologies . . . 109

6.6 Summary . . . 112

Chapter 7 $\mathcal{K}$-$\mathcal{MORPH}$ Evaluation and Results: Morphing Healthcare Ontologies for Generating Therapeutic Knowledge about Urinary Tract Infections . . . 114

7.1 Source Ontologies . . . 116
7.1.1 Biological Top Level Ontology . . . 116
7.1.2 DebugIT Core Ontology . . . 117
7.1.3 AGFA Ontologies . . . 117

7.2 Task #1: Defining Problem-context: Therapeutic Knowledge about UTI . . . 119

7.3 Task # 2: Extracting Contextualized UTI Sub-ontologies . . . 120

7.4 Task # 3: Aligning and Merging Contextualized Sub-ontologies . . . 121

7.5 Task # 4: Detecting and Resolving Inconsistencies . . . 125

7.6 Results: Morphed Therapeutic Knowledge Generated by $\mathcal{K}$-$\mathcal{MORPH}$ 130

7.7 Summary . . . 139

Chapter 8 Conclusion and Future Work . . . 140

8.1 Assessment of Research Contributions . . . 141
8.1.1 Extracting Contextualized Sub-ontologies . . . 142
8.1.2 Aligning and Merging Ontologies . . . 143
8.1.3 Detecting and Resolving Inconsistencies in Ontologies . . . 144

8.2 $\mathcal{K}$-$\mathcal{MORPH}$ Applications: Clinical Decision Making . . . 144

8.3 Future Work . . . 146

8.4 Final Words . . . 147

Bibliography . . . 151

List of Tables

Table 2.1 Support from Semantic Web framework towards Knowledge Integration 15

Table 2.2 State-of-the-Art Knowledge Integration Approaches 22

Table 4.1 Comparison Between Extracted Sub-ontologies 54

Table 5.1 List of Ontology Matching and Alignment Tools 62

Table 5.2 Ritze et. al. Matching Results: Complex Correspondences . . . 81

Table 5.3 TOM Matching Results: Atomic Correspondences 83

Table 5.4 TOM Matching Results: Complex Correspondences 84

Table 5.5 POM Matching Results: Atomic Correspondences 85

Table 5.6 POM Matching Results: Complex Correspondences 86

Table 5.7 Merged Conference Ontology: Inferred Results 91

Table 7.1 Concepts and Alignments for Therapeutic Knowledge about UTI 119

Table 7.2 TOM Matching Results: Correspondence Accuracy 122

Table 7.3 POM Matching Results: Correspondence Accuracy 122

Table 7.4 TOM Matching Results: Atomic Correspondences 123

Table 7.5 POM Matching Results: Atomic Correspondences 124

Table 7.6 TOM Matching Results: Complex Correspondences 126

Table 7.7 POM Matching Results: Complex Correspondences 127

Table 7.8 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 1 131

Table 7.9 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 2 131

Table 7.10 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 3 132

Table 7.11 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 4 133

Table 7.12 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 5 133

Table 7.13 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 6 134

Table 7.14 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 7 135
Table 7.15 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 8 136
Table 7.16 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 9 137
Table 7.17 $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 10 138

List of Figures

Figure 1.1 Knowledge Morphing at Different Representation Levels . . . 5

Figure 1.2 Problem-Context in Knowledge Morphing 7

Figure 1.3 $\mathcal{K}$-$\mathcal{MORPH}$ Solution Approach 8

Figure 2.1 Dynamic Sub-Ontology Evolution for Collaborative Problem Solving 18

Figure 2.2 Extracting Sub-ontology from Multiple Ontologies 19

Figure 2.3 Knowledge selection process and semantic browsing with Magpie 21

Figure 3.1 $\mathcal{K}$-$\mathcal{MORPH}$ Solution Approach 28

Figure 3.2 Representing Problem-context in $\mathcal{K}$-$\mathcal{MORPH}$ 30

Figure 3.3 $\mathcal{K}$-$\mathcal{MORPH}$: Generating Therapeutic Workflow Knowledge by Morphing Prostate Cancer Clinical Pathways 34

Figure 3.4 Contextualized Sub-ontology from PC Pathway Ontology . . 36

Figure 3.5 Merged Knowledge about PC-Halifax:ActiveSurveillance . . . 37

Figure 4.1 Sub-ontology Extraction with Boundedness 52

Figure 4.2 Contextualized Sub-ontology from PC Pathway Ontology . . 58

Figure 5.1 Ontology Reconciliation in $\mathcal{K}$-$\mathcal{MORPH}$: Matching, Aligning & Merging Ontology Modules 60

Figure 5.2 Alignments for Contextualized PC Sub-ontologies 93

Figure 5.3 Merged Knowledge about PC-Halifax:ActiveSurveillance . . . 94

Figure 5.4 Merged knowledge about PC-Winnipeg:Brachytherapy 94

Figure 6.1 Structure of Reason Ontology 105

Figure 6.2 Computing Minimal Inconsistent Resolve Candidates 108

Figure 7.1 $\mathcal{K}$-$\mathcal{MORPH}$ Evaluation: Therapeutic Knowledge about Urinary Tract Infections 115

Figure 7.2 Contextualized UTI Sub-ontology Extraction 121

Abstract

Knowledge-driven problem solving demands 'complete' knowledge about the domain and its interpretation under different contexts. Knowledge Morphing aims at a context-driven integration of heterogeneous knowledge sources—in order to provide a comprehensive and networked view of all knowledge about a domain-specific problem, pertaining to the context at hand. In this PhD thesis, we have proposed a Semantic Web based framework, $\mathcal{K}$-$\mathcal{MORPH}$, for *Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies*. In order to realize our $\mathcal{K}$-$\mathcal{MORPH}$ framework, we have developed: (i) a sub-ontology extraction method for generating *contextualized sub-ontologies* from the source ontologies pertinent to the problem-context at hand; (ii) two ontology matching approaches: *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM) for finding both atomic and complex correspondences between two extracted contextualized sub-ontologies; and (iii) our approach for resolving inconsistencies in ontologies by generating *minimal inconsistent resolve candidates* (MIRCs), where removing any of the MIRCs from the inconsistent ontology results in a maximal consistent sub-ontology. Thus, $\mathcal{K}$-$\mathcal{MORPH}$ performs knowledge morphing among ontology-modelled knowledge sources and generates a comprehensive knowledge-base pertinent to the problem at hand by (a) extracting problem-specific knowledge components from ontology-modelled knowledge sources using our sub-ontology extraction method; (b) aligning and merging the extracted knowledge components using our matching approaches; and (c) repairing inconsistencies in the morphed knowledge by applying our approach for detecting and resolving inconsistencies. We demonstrated the application of our $\mathcal{K}$-$\mathcal{MORPH}$ framework in the healthcare domain, where $\mathcal{K}$-$\mathcal{MORPH}$ generated a merged ontology for providing a comprehensive therapeutic knowledge-base for Urinary Tract Infections (UTI) by first (i) extracting 20 contextualized sub-ontologies from various UTI ontologies, (ii) aligning and merging the extracted UTI sub-ontologies, and (iii) detecting and resolving inconsistencies in the merged UTI ontology.

List of Abbreviations and Symbols Used

$M_{\mathcal{O}}$	Set of matchable ontology-entities
Π	Set of context-specific alignments
$\mathcal{PC}$	Problem-context
$\mathcal{A}$	Set of assertional axioms
$\mathcal{C}$	Set of ontology concepts
$\mathcal{I}$	Set of ontology individuals
$\mathcal{L}$	Set of ontology literals
$\mathcal{M}$	Set of minimal inconsistent resolve candidates
$\mathcal{O}$	An ontology
$\mathcal{P}$	Logic program
$\mathcal{R}$	Set of ontology properties
$\mathcal{T}$	Set of terminology axioms
$\mathcal{V}$	Vocabulary of concepts, properties and individuals
$\emptyset$	Empty set
$\mathbb{A}$	Alignment between two ontologies
$\mathbb{A}_x$	Context-specific alignments between two ontologies
$\mathbb{E}$	Set of minimal contradictory triple sets
$\mathbb{O}$	Set of ontologies
$\mathbb{O}_I$	Set of initial source ontologies
$\mathbb{PC}$	Set of problem-contexts
$\mathbb{S}$	Set of asserted ancestor triple sets
$\mathbb{T}_{\mathcal{P}}(\mathcal{O})$	Set of all asserted and inferred ontology triples
$\mathcal{C}_x$	Set of context-specific concepts
$\mathcal{R}_x$	Set of context-specific properties
$\mathcal{C}_e$	Set of extracted concepts
$\mathcal{C}_s$	Set of user-selected concepts
$\mathcal{C}_{x1}$	An example problem-context

$\mathcal{I}_e$	Set of extracted individuals
$\mathcal{M}_c$	Set of meta-classes
$\mathcal{M}_p$	Set of meta-properties
$\mathcal{P}_x$	Context-specific constraints
$\mathcal{R}_e$	Set of extracted properties
AATS	Asserted Ancestor Triple Set
BioTop	Biological Top Level ontology
C-OWL	Contextualizing OWL
CL	Coherent Logic
CTS	Contradictory Triple Set
DCO	DebugIT Core Ontology
DDL	Distributed Description Logics
DeLP	Defeasible Logic Programs
DL	Description Logics
IDDL	Integrated Distributed Description Logics
IRC	Inconsistent Resolve Candidate
KM	Knowledge Management
LP	Logic Program
MCTS	Minimal Contradictory Triple Set
MIRC	Minimal Inconsistent Resolve Candidates
MOVE	Materialized Ontology View Extraction

OAEI Ontology Alignment Evaluation Initiative
OBO Open Biomedical Ontologies
OWL Web Ontology Language

P-DL Package-based Description Logics
PC Prostate Cancer
POM Proof-based Ontology Matching

RCOS Requirements Consistency Optimization Scheme
RDF Resource Description Framework
RDF-S RDF Schema

SCOS Semantic Completeness Optimization Scheme
SW Semantic Web

TOM Triple-based Ontology Matching

UTI Urinary Tract Infections

Acknowledgements

First of all, I would like to thank Dr. Raza Abidi who not only introduced me to the field of Knowledge Management, but also supervised me with his timely guidance and constructive discussions. I am also thankful to my supervisory committee for reviewing my PhD thesis, and providing their valuable feedback.

I would also like to take this opportunity to acknowledge the efforts of my previous supervisors: Dr. Jörg Siekmann who supervised my Master's thesis, Dr. Harald Ganzinger who supervised my Master's project, and Dr. Abbas K. Zaidi who supervised my Bachelor's project. Without their supervision, I might not have been able to perform and produce valuable research.

I would like to extend my thanks to AGFA Healthcare, for funding my PhD studies, providing test-cases and giving their feedback. In particular, I would like to thank Jos De Roo, who has been actively involved in fostering technical discussions, and incorporating technical advances in reasoning engines.

I am also thankful to my colleagues, especially Ali Daniyal, Faisal Abbas and Imran Rauf for giving their technical insight, Dr. Samina Abidi for proving test-cases, and Katie Boudreau and Paul Hardman (Dalhousie Writing Centre) for proof-reading my PhD thesis.

I am grateful to my parents, Sadiq Hussain and Farhat Jahan; without their resolute patronage I would not have reached this standing, and to my wife, Midhat, for her invaluable love and support during both good and not so good days, and also for giving us the little one, Zafir.

Chapter 1

Introduction

Knowledge-driven problem solving demands 'complete' knowledge about the domain and its interpretation under different contexts [1]. However, complete knowledge, especially as one holistic knowledge object, is rarely available in practice. Therefore, problem solvers resort to and integrate relevant knowledge from multiple sources to formulate a knowledge object that is just sufficient for the given problem context [2]. An example scenario in the healthcare domain, is where a practitioner's advice on a disease-specific scenario is carried out via different reasoning strategies, consulting both strong and weak evidence-based medical knowledge [3]. The importance of integrating multiple medical knowledge sources can be realized in cases where a clinical solution from one medical knowledge source is lacking, or another knowledge source can play a role in deriving alternative solutions. For instance, in the absence of explicit algorithms described in a clinical practice guideline [3, 4], practitioners may need to consult with other sources of relevant knowledge, such as previously recommended cases and/or the expertise of domain experts recorded in problem solving scenarios [5, 6]. Hence a context-driven integration of heterogeneous knowledge sources aims to provide a comprehensive and networked view of all knowledge pertaining to a domain-specific problem at hand.

1.1 Knowledge Sharing and Integration: Open Issues

The desirability of *Knowledge Sharing* and *Knowledge Integration* has been identified in the field of Knowledge Management (KM) [2, 7, 8] , where context-specific knowledge integration was highlighted as one of the milestones to be achieved. This desirability was initially acknowledge by Nonaka (1994, 1995) [9, 10], through exploring the dynamics of the inter-relationship between tacit and explicit knowledge between individuals and groups. Nonaka focused on the relationships of

communication between such knowledge types (via processes of *socialization*, *externalization*, *combination*, and *internalization*) and viewed all such knowledge as objects—in order to operationalize and combine tacit forms of knowledge [9].

At the same time, the demand for knowledge sharing and integration in KM is also realized in various application domains, such as Life Sciences [11–13], E-Business [14–17], Telecommunications [18], Government Security Services [19,20], Web Portals [21–24] and Marine Sciences [25]. In general, practitioners and domain experts seek an intelligent medium for sharing their behavioural and operational knowledge among other domain experts, in order to evolve a shared understanding between groups, and also to adapt and update their local policies based on the shared knowledge [2,8,26]. On the other hand, knowledge integration becomes crucial when experts are aiming to build a comprehensive knowledge-base for various domain-specific and context-sensitive applications [2,12,13,17]. While aiming towards integrating or sharing knowledge for different purposes and applications, selecting relevant knowledge from large knowledge-bases [27,28] can play an important role. By selecting relevant knowledge pertinent to the problem at hand, integration or sharing of knowledge can be restricted only to the relevant knowledge, as opposed to integrating large knowledge-bases and dealing with the undesired complexity of the integrated knowledge-base [28]. However, knowledge selection from large knowledge bases is a non-trivial task [27,28]. When selecting knowledge from a knowledge source, the selected knowledge should not compromised the overall consistency and completeness of the original knowledge source. It is deemed required to make sure that the selected knowledge component must also comply with all the integrity constraints defined in the original knowledge source [28].

There have been various attempts in proposing frameworks and approaches towards knowledge sharing and integration [12–26] (see Chapter 2 for further details). However, aiming towards a context-driven knowledge integration, the following challenges remain outstanding:

1. How to model the user-defined problem-context, in order to enable context-driven knowledge sharing and integration?

2. How to identify and extract knowledge components from available knowledge sources pertinent to the given problem-context?

3. How to integrate heterogeneous knowledge components that are separately modeled in terms of their individual structures, origins and functionalities?

4. How to ensure that an integration of extracted knowledge components is carried out within a particular context?

5. How to detect and resolve inconsistencies in the integrated knowledge?

1.2 Knowledge Morphing

Knowledge morphing aims to formulate a comprehensive knowledge object, specific to a given context, through "the intelligent and autonomous fusion/integration of contextually, conceptually and functionally related knowledge objects that may exist in different representation modalities and formalisms, in order to establish a comprehensive, multi-faceted and networked view of all knowledge pertaining to a domain-specific problem" Abidi 2005 [29].

The need for knowledge morphing is motivated by the realization that integration of knowledge sources should be driven by the *problem-context*—i.e. select and integrate only those knowledge fragments (within a knowledge source) that are relevant to the problem [27, 28], as opposed to integrating the entire knowledge source. A well-defined problem-context, therefore, determines the scope of knowledge that is pertinent to the problem. For instance, in the domain of healthcare, clinical guidelines incorporate broad knowledge about the diagnosis, treatment, prognosis and follow-up care for a particular disease [30]. For the context of *therapeutic decision support* one needs only therapeutic knowledge, which should be selected from multiple guidelines to formulate a comprehensive therapeutic knowledge-base [31]. We argue that the integration of the entire knowledge source exacerbates the complexity of establishing knowledge interoperability between multiple knowledge sources [28]. Hence, knowledge morphing does not deal with the complete integration of different knowledge sources; rather, it first identifies relevant knowledge components among knowledge sources based on

the problem-context, and then it integrates those components to build a comprehensive knowledge-base that provides a network-view of all knowledge applied under the given problem-context.

1.3 Knowledge Representation and Morphing using Ontologies

A necessary step for knowledge morphing is to pursue knowledge formalization in order to support domain-specific inferencing [32]. We believe Semantic Web (SW) languages and standards can provide a framework for the semantic modeling and markup of knowledge, using its properties, its relations and its underlying semantics [33]. In a SW framework, heterogeneous knowledge sources can be represented and integrated as ontologies [32]. Knowledge representation via ontologies allows: (i) formalization of domain-specific knowledge [32]; (ii) conceptualization of the knowledge along declarative and procedural dimensions [34]; (iii) annotation of the knowledge based on an ontological model [35]; (iv) re-use and evolution of the knowledge [36–38]; (v) use of standard terms and concepts [39,40]; and (vi) identification of similar knowledge components [41] that can potentially be aligned [42] to achieve knowledge morphing.

Ontologies and contexts are used to model a domain with different views. Ontologies define a shared model that provides a global perspective, whereas contexts are used to realize a local aspect of a domain [43–45]. A contextualized ontology deals with an adaptation of the ontology model to support a local view, and provides: (i) a specific interpretation of the ontology concepts, and (ii) an implementation of the procedural knowledge that can be applied in a particular context [46].

The Web Ontology Language (OWL) [47] is a language for defining and instantiating Web ontologies. The OWL ontology may include descriptions of RDF classes, properties and their instances [48]. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) [48], by providing additional vocabulary along with a formal semantics [49]. The OWL language provides three increasingly expressive sub-languages, designed for modelling domain-specific ontologies [50,51]. Given an OWL ontology, the OWL formal semantics specifies how to derive its logical consequences [49].

In order to enable knowledge morphing between knowledge sources, we argue

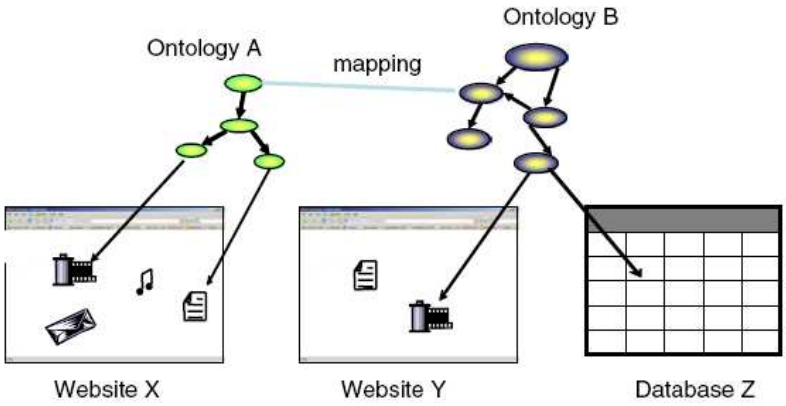

Figure 1.1: Knowledge Morphing at Different Representation Levels

that the knowledge sources are required to be modelled as ontologies [32]. For example, two different knowledge sources are represented in two different formats, namely ontology and relational database (schema), respectively (as shown in Figure 1.1). In order to perform knowledge morphing among these two knowledge sources, we argue that by first modeling the database schema in terms of ontologies will lead to a better morphed knowledge-base. After transforming the database schema into an ontology schema (*Ontology B* in Figure 1.1), more domain-specific knowledge can be inferred using ontology-specific formal semantics [49]. Hence, in order to enable knowledge morphing between the initially given *Ontology A* and the transformed schema *Ontology B*, various ontology alignment and merging approaches [41, 42] can play an effective role in merging these two ontologies and generating a desired morphed knowledge-base.

1.4 The Role of Context in Knowledge Morphing

The term "context" is frequently employed in computer science literature, but its meaning is mostly left to the reader's understanding and its usage is considered as implicit and intuitive [52, 53]. In order to discuss the role of context in the process of knowledge morphing, we first need to have a clear understanding of the notion of context. In the following subsections we will first present some informal definitions for context from a number of perspectives, and then discuss an operational definition of context and its role in knowledge morphing.

1.4.1 What is Context?

When dealing with the notion of context, the focus in various interpretations [45, 52, 54, 55] remained on the acquisition and processing of contextual information; while a clear definition of context was given secondary importance [56–58]. The major drawback of this approach is the lack of generality, flexibility and extensibility [59], which could be overcome by adopting a more generic model and a semantic representation of context. Addressing the quite limited notions and early definitions of context, Dey (2001) provided the following general definition, which is probably the most widely accepted: "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between the user and the application, including the user and the applications themselves" Dey 2001 [60].

1.4.2 Problem-context in Knowledge Morphing

In our proposed knowledge morphing problem, the role of context is of major importance, and makes our problem both equally challenging and interesting. For knowledge morphing, we adopted the *operational definition of context* [61] to represent the user-intended context, i.e. *problem-context*. According to Zimmermann et. al. [61], a knowledge source may provide five fundamental categories of contextual information: *individuality*, *activity*, *location*, *time*, and *relations* (as shown in the right-side of Figure 1.2). *Individuality* deals with the problem-specific concepts and properties described within the knowledge source itself. *Activity* covers all the tasks in which the knowledge source may be involved in a particular context. Other contextual information, such as *location* and *time* provide the context-specific spatio-temporal axioms and constraints over the knowledge source. Finally, the *relations* category represents any possible contextual compatibility between two (or more) knowledge sources. Hence, for each knowledge source, the five main contextual categories may vary based on the different contexts at hand.

For knowledge morphing, a problem-context (as shown in the middle of Figure 1.2) is provided by the user, where the user (i) identifies problem-specific knowledge that affects the individuality, activity and relations aspects of a knowledge source; (ii) defines context-specific axioms and context-specific constraints to be

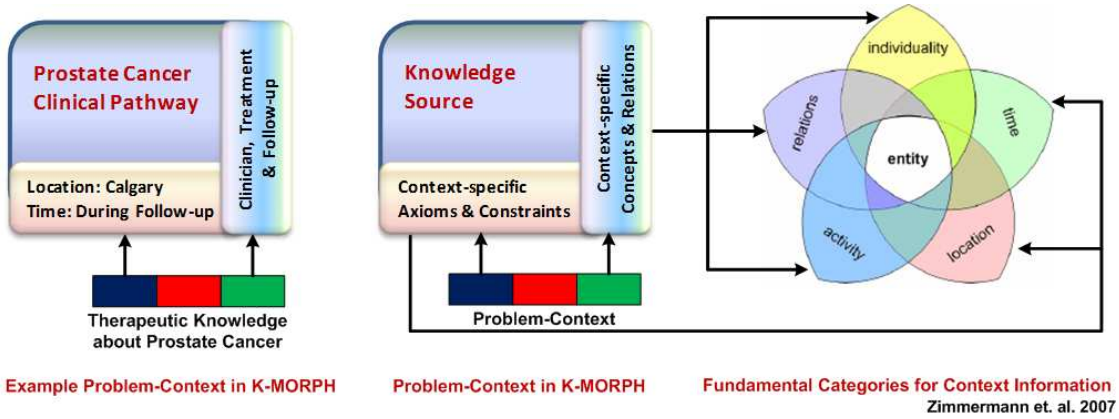

Figure 1.2: Problem-Context in Knowledge Morphing

applied on a knowledge source for the user's context at hand; and (iii) provides contextual relation between two (or more) knowledge sources which allows morphing of contextually relevant knowledge components. An example problem-context for knowledge morphing, *Therapeutic Knowledge about Prostate Cancer* [31, 62], is shown in the left-side of Figure 1.2. For the given problem-context, problem-specific concepts and relations are selected by the user from the Prostate Cancer pathway, and spatio-temporal constraints are also defined. The defined constraints are limiting the therapeutic knowledge component to only those treatments that can be performed in Calgary during the follow-up visits.

1.5 Our Solution Approach

We adopt a Semantic Web (SW) architecture to address the above-presented research challenges (see Section 1.1). SW offers a logic-based framework [33] to (a) semantically model various knowledge sources as ontologies; (b) capture and represent the underlying domain concepts, and the semantic relationships that are inherent within a problem-context; (c) ensure interoperability between multiple ontologically defined knowledge sources together with their 'trust'; and (d) maintaining a consistent change, evolution and management of ontologies. We propose our solution approach for Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies ($\mathcal{K}$-$\mathcal{MORPH}$) as shown in Figure 1.3. $\mathcal{K}$-$\mathcal{MORPH}$ is realized through the following main tasks:

- *Task # 1*: Representing the user-defined problem-context.
- *Task # 2*: Extracting contextualized sub-ontologies from the source ontologies based on the given problem-context.
- *Task # 3*: Merging contextualized sub-ontologies based on identified/context-specific alignments to generate a merged ontology.
- *Task # 4*: Detecting and resolving inconsistencies in the merged ontology.

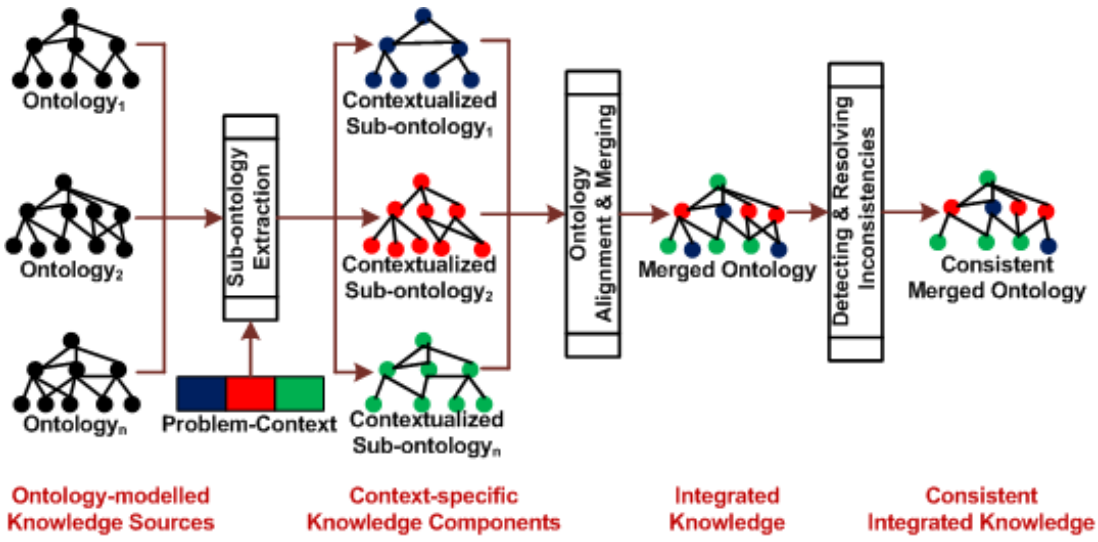

Figure 1.3: $\mathcal{K}$-$\mathcal{MORPH}$ Solution Approach

Our $\mathcal{K}$-$\mathcal{MORPH}$ solution approach is realized in a Semantic Web framework (as shown in Figure 1.3), where the available knowledge sources are required to be modelled as ontologies. These ontology-modelled knowledge sources serve as inputs to $\mathcal{K}$-$\mathcal{MORPH}$. Based on the given knowledge sources (modelled as ontologies) and a user-defined problem-context, $\mathcal{K}$-$\mathcal{MORPH}$ performs a context-driven integration of knowledge sources by: (a) extracting knowledge components from the available ontology-modelled knowledge sources pertinent to the given problem-context by extracting contextualized sub-ontologies; (b) integrating the extracted knowledge components by merging contextualized sub-ontologies; and (c) repairing inconsistencies in the morphed knowledge by detecting and resolving

inconsistencies in the merged ontology. In this way, $\mathcal{K}$-$\mathcal{MORPH}$ pursues highly-specific ontology alignment guided by the problem-context—i.e. a single knowledge morphing context forms the basis of the process. This also means that as the problem-context changes, a new merged ontology can be readily generated.

1.6 $\mathcal{K}$-$\mathcal{MORPH}$: Technical Challenges and Research Contributions

$\mathcal{K}$-$\mathcal{MORPH}$ provides a framework for context-specific knowledge integration of ontology-modelled knowledge sources by a context-driven reconciliation of ontologies. The $\mathcal{K}$-$\mathcal{MORPH}$ framework is realized by an active interplay of three major research areas in Semantic Web: (i) extracting contextualized sub-ontologies, (ii) aligning and merging ontologies, and (iii) detecting and resolving inconsistencies in ontologies. Although, there have been various attempts in these areas, whereby (a) various extraction methods were proposed to extract sub-ontologies from source ontologies [63–68]; (b) various ontology matching and alignment methods can align and merge ontologies [41,42]; and (c) available approaches can detect and resolve inconsistencies in the merged ontology [69–72]. However, yet there are challenges and limitations in the above-mentioned fields [13, 73]. Hence, aiming towards the knowledge morphing problem, we also made some scientific contributions in all three research areas.

In $\mathcal{K}$-$\mathcal{MORPH}$, we make use of an interplay of the above-mentioned areas to support our knowledge morphing problem, and propose a context-driven knowledge integration framework, whereby we (i) extract context-specific knowledge components from available (ontology-modelled) knowledge sources using our sub-ontology extraction method (see Chapter 4); (ii) integrate the extracted knowledge components using our alignment and merging approach (see Chapter 5); and (iii) repair inconsistencies in the morphed knowledge by applying our approach for detecting and resolving inconsistencies (see Chapter 6). The required technical challenges and research contributions made in realizing the $\mathcal{K}$-$\mathcal{MORPH}$ framework are discussed below:

1.6.1 Extracting Contextualized Sub-ontologies

Sub-ontology extraction, from a large ontology, leads to the generation of a specialized knowledge model that is pertinent to solve specific problems. Existing sub-ontology extraction methods tend to render either a too generalized or a too restricted sub-ontology [64–67]. In $\mathcal{K}$-$\mathcal{MORPH}$, we propose a structured-based sub-ontology extraction method for extracting contextualized sub-ontologies based on the user-selected concepts from the ontologies that are pertinent to the problem-context at hand (see Chapter 4). Our sub-ontology extraction method extracts a contextualized sub-ontology, whilst extending the semantics of the extracted concepts and their relationships in the sub-ontology. Our approach features the following tenets: (i) identifying the user-selected concepts that are pertinent to the problem-context at hand; (ii) extracting the user-selected concepts, their properties and their individuals; and (iii) extracting other concepts, properties and individuals that are structurally connected with the user-selected concepts.

1.6.2 Merging Contextualized Sub-ontologies

Matching and alignment of ontologies have been carried out based on their lexical, conceptual and structural similarities [73]. When dealing with structural similarities, similarity scores between ontology-entities can be further improved based on the similarities between their structurally connected entities [73]. We believe alignments can become more 'trustworthy' by finding similarities among entities that are driven from the underlying ontology axioms or assertions. In order to achieve knowledge morphing using $\mathcal{K}$-$\mathcal{MORPH}$, we have developed two ontology matching approaches *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM) (see Chapter 5). Both of our matchers (TOM and POM) can find an alignment between ontologies, not only based on structural similarities over structurally connected entities, but also takes into account similarities between other deductively-connected complex entity-structures—under the rules describing both domain-specific and ontology-language semantics [49].

1.6.3 Detecting and Resolving Inconsistencies

Inconsistencies—whether they occur during the process of ontology evolution or appear during the ontology reconciliation process—result in potential harm to an ontology structure, and decrease credibility in representing a consistent and shared vocabulary of an underlying domain. When an inconsistency occurs, there are mainly two ways to deal with it: either resolve it, or reason with the inconsistent ontology [69, 70, 74, 75]. In $\mathcal{K}$-$\mathcal{MORPH}$, we propose our approach for detecting and resolving inconsistencies in ontologies (see Chapter 6). Our inconsistency detection deals with the identification of *contradiction derivations* under the *integrity constraint rules*, which are given in a logic program [49]. To resolve detected inconsistencies, we generate all possible Minimal Inconsistent Resolve Candidates (MIRCs). Removing an MIRC from the inconsistent ontology results in a maximal consistent sub-ontology w.r.t. the given logic program. To inform the user about the consequences of removing an MIRC, we also provide a list of all its derived triples.

1.7 Outline

This PhD thesis is outlined as follows:

Chapter 2 provides an overview of existing knowledge integration approaches, and also presents a comparison of these approaches with $\mathcal{K}$-$\mathcal{MORPH}$. Chapter 3 presents the overall functionality of the $\mathcal{K}$-$\mathcal{MORPH}$ framework by discussing the interactions between each of its modules. In Chapter 4, we describe our sub-ontology extraction approach for extracting contextualized sub-ontologies based on the given problem-context. We present a comparison of our approach with a few existing sub-ontology extraction approaches, and also demonstrate the use of our approach by extracting contextualized sub-ontologies from three prostate cancer ontologies for the problem-context *therapeutic decision support*. In Chapter 5, we describe our ontology matching approaches, *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM), for aligning and merging contextualized sub-ontologies. We present a comparison of our matchers TOM and POM with

other ontology matching approaches, and also demonstrate the use of our matchers in aligning and merging the contextualized sub-ontologies for the problem-context *therapeutic decision support* and generated a merged prostate cancer ontology. Chapter 6 presents our approach for detecting and resolving inconsistencies in (merged) ontologies. We evaluated our approach on the merged prostate cancer ontology, where we detected all the inconsistencies in this ontology and generated all possible *minimal inconsistent resolve candidates* (MIRCs) for extracting a maximal consistent sub-ontology. Chapter 7 demonstrates the application of our $\mathcal{K}$-$\mathcal{MORPH}$ framework in the healthcare domain, where $\mathcal{K}$-$\mathcal{MORPH}$ generated a merged ontology to provide a comprehensive and networked knowledge about *therapeutic treatment plan for urinary tract infections* by: (i) extracting 20 contextualized sub-ontologies from various high-level medical ontologies of different healthcare institutions; (ii) aligning and merging the extracted sub-ontologies; and (iii) detecting and resolving inconsistencies in the merged ontology. Chapter 8 summarizes the technical contributions made in this thesis, and also highlights the applications of the $\mathcal{K}$-$\mathcal{MORPH}$ framework in other domains.

Chapter 2

Related Work: Comparing $\mathcal{K}$-$\mathcal{MORPH}$ with State-of-the-Art

The generation of a comprehensive knowledge object has already been pursued as a knowledge integration problem, and achieved via semantic interoperability between knowledge sources [76–84]. By modeling knowledge sources as ontologies, semantic interoperability among knowledge sources can be achieved via *ontology reconciliation* [41,85,86]. However, aiming towards a context-driven knowledge integration of ontology-modelled knowledge sources, ontology reconciliation under different contexts is still an outstanding challenge [73]. It may be noted that the literature suggests other approaches towards context-driven knowledge integration from different perspectives [12–26]; some of the prominent approaches will be discussed in this chapter.

As knowledge morphing demands a context-driven integration of heterogeneous knowledge sources, the main research challenges for knowledge morphing are (i) dealing with the heterogeneity and semantic interoperability between available knowledge sources, and (ii) ensuring that the knowledge integration is carried out within a particular context. Hence, aiming towards knowledge morphing, the key issues to be solved are dealing with context, semantic interoperability and heterogeneity [13]. For enabling a context-driven knowledge integration, Zimmermann et. al. [13] highlighted a list of features that a knowledge integration system should provide:

C1 $\Rightarrow$ *Context-awareness*: Identifying context-specific components from a knowledge source relevant to the context at hand.

C2 $\Rightarrow$ *Modularity*: Reusing context-specific fragments from multiple knowledge sources.

C3 $\Rightarrow$ *Profile and Policy Management*: Treating internal policies or profiles distinctively.

C4 ⇒ *Correspondence Expressiveness*: Relating heterogeneous knowledge, either within or between contexts.

C5 ⇒ *Dealing with Inconsistencies*: Repairing or tolerating incompatibilities or inconsistencies, while solving heterogeneity within or between contexts [13].

In this chapter, we will first perform a 'technology-check' of the Semantic Web (SW) framework by evaluating the formal approaches offered in SW, based on the above-mentioned criteria, and will present their strengths and short-comings. Next, we shall also present a comparison of available knowledge integration approaches based on the same criteria. We will conclude this chapter by comparing $\mathcal{K}$-$\mathcal{MORPH}$ with other state-of-the-art approaches, and highlighting the unique differences in our $\mathcal{K}$-$\mathcal{MORPH}$ approach.

2.1 Support from The Semantic Web Framework Towards Knowledge Integration

The Semantic Web (SW) framework [33] allows knowledge representation via SW ontologies [32] by providing constructs that represent shared vocabularies from multiple domains, such as Medicine and Healthcare [3, 87]. Hence, these days, various domain experts are developing domain-specific ontologies for maintaining a standardized and shareable knowledge-base that can be used among different parties for implementing different applications [32, 87]. However, when it comes to satisfying various domain-specific applications, a context-driven reconciliation/networking of domain-specific ontologies is required to provide a network-view of vocabularies pertaining to the context at hand [88]. Aiming towards context-driven knowledge integration, there have been various approaches and standards proposed in the SW community [13] that offer their unique features in resolving (parts of) the mentioned problem.

In this section, we will highlight the support and limitations of the available SW standards and languages in solving the knowledge integration problem based on the above-mentioned criteria [13]. Some of the initial and foundational attempts towards knowledge representation were Description Logics (DL) [89] and OWL [49]. Given that Description Logics (DL) [89] and OWL [49] appear incapable

Table 2.1: Support from Semantic Web framework towards Knowledge Integration

Formal Approaches	**C1**	**C2**	**C3**	**C4**	**C5**
DL/OWL	No	Very limited	No	Good	Very weak
DDL/C-OWL	Yes	Yes	No	Very Good	Good
P-DL	Yes	Yes	Very limited	Very limited	Weak
DDL Revisited	Yes	Yes	No	as DDL	Medium
$\mathcal{E}$-connections	Yes	Yes	No	Good	Excellent
IDDL	Yes	Yes	No	Good	Very good
DeLP/Paraconsistent	No	No	Limited	as DL	Good
Modular Rule Bases	Yes	Yes	Limited	Limited	Weak

C1: *Context-awareness*; C2: *Modularity*; C3: *Profile and Policy Management*; C4: *Correspondence Expressiveness*; C5: *Dealing with Inconsistencies*.

of dealing with context-awareness [43] and are intolerant to inconsistent ontologies, further extensions, such as Distributed Description Logics (DDL) [90] and C-OWL [91], were proposed to overcome such aspects. On the other hand, Package-based Description Logics (P-DL) [92] is a formalism that was essentially designed for the modularity of Web ontologies, and lacks expressivity for ontology alignment and has weak tolerance towards inconsistent ontologies. $\mathcal{E}$-connections [93] is another formalism for reasoning with heterogeneous ontologies, focusing on context-awareness and modularity aspects, and is fully tolerant to inconsistent ontologies. $\mathcal{E}$-connections also supports ontology alignment between ontologies (from different domain) through *links*. Integrated Distributed Description Logics (IDDL) [94], when compared to DDL, $\mathcal{E}$-connections and P-DL, allows users to assert ontology alignments from a 'third party"s point of view, by which correspondences can be manipulated and reasoned independent of the ontologies. One effective way of tolerating inconsistencies consist of using Paraconsistent Logics [95], by which reasoning can be done in the presence of inconsistency. Alternatively, *defeasible argumentation* [96], such as Defeasible Logic Programs (DeLP) [97], have been introduced to reason and resolve inconsistencies using provenance. In this case, the terminology axioms [89] is separated into 2 subsets: (a) one being strict (i.e. it must always be used in reasoning), and (b) the other being defeatable (i.e. an argumentation process may defeat them and nullify them for a particular reasoning task). A modular web rule bases framework proposed in Analyti et. al. [98] makes the distinction between global knowledge, local knowledge and internal

knowledge. The framework is based on a rule-based language rather than DL, and allow the user to express and reason with modularity on top of Semantic Web. In this framework, reasoning is possible during inconsistency of knowledge bases. Based on the findings reported in Zimmermann et. al. [13], possible ways in which some of the prominent formal approaches can play a role towards context-driven knowledge integration are highlighted in Table 2.1.

2.2 State-of-the-Art Approaches for Knowledge Integration

It may be noted that the literature suggests other approaches towards context-driven knowledge integration from different perspectives [12–26]; some of the prominent approaches are discussed and compared based on the above-mentioned criteria in the following sub-sections.

2.2.1 ECOIN

ECOIN is one notable framework that performs semantic reconciliation of independent data sources under a defined context [99]. Semantic reconciliation is performed at the context level by defining *conversion functions* between contexts as a network. The ECOIN approach believes in the *single ontology, multiple views* notion [99], and introduces the notion of *modifiers* to explicitly describe the multiple specializations/views of the concepts used in different data sources. It exploits the modifiers and conversion functions, to enable context mediation between data sources, and reconciles and integrates source schemas with respect to their conceptual specializations. ECOIN can be evaluated based on the above-mentioned criteria as follows:

C1 $\Rightarrow$ *Context-awareness*: Based on the context at hand, ECOIN allows users to define *conversion functions* and *modifiers* on data objects to support context-driven mediation of data sources.

C2 $\Rightarrow$ *Modularity*: ECOIN does not filter context-specific components from given data sources. It attempts to reuse available data sources as whole, but does not have the ability to reuse suitable parts of data sources.

C3 ⇒ *Profile and Policy Management*: ECOIN does not make any distinction between internal and external policies.

C4 ⇒ *Correspondence Expressiveness*: ECOIN uses *conversion functions* as pre-defined correspondences. Applying ontology matching and alignment techniques is one of their future work, but no progress has yet been reported. It performs a complete integration and mediation among data sources supported by the conversion functions.

C5 ⇒ *Dealing with Inconsistencies*: ECOIN provides no framework for dealing with inconsistencies or tolerating inconsistencies during data integration and mediation processes.

2.2.2 OpenKnowledge

The OpenKnowledge framework [100] supports knowledge sharing among different knowledge sources, not by sharing their asserted statements, instead by sharing their *interaction models*. An interaction model provides a context in which knowledge can be transmitted between two (or more) knowledge sources (i.e. peers). This approach has a closer relevance with semantic service composition [101], where each interaction model (stands for a knowledge source) can be seen as a service that interacts with other services based on their service descriptions and business logics. The OpenKnowledge project can be evaluated as follows:

C1 ⇒ *Context-awareness*: OpenKnowledge uses the *interaction model* as context for sharing certain pre-defined knowledge components among knowledge sources.

C2 ⇒ *Modularity*: OpenKnowledge offers limited modularity. It only provides a medium (via *interaction model*) by which two knowledge sources can exchange knowledge based on their pre-defined protocols.

C3 ⇒ *Profile and Policy Management*: OpenKnowledge allows knowledge sharing only through *interaction models*, and keeps and applies the local and foreign policies distinctively.

C4 ⇒ *Correspondence Expressiveness*: OpenKnowledge does not provide (or use) any language for defining correspondences between knowledge sources. It rather relies on the interaction models defined by the users.

C5 ⇒ *Dealing with Inconsistencies*: It provides no framework for dealing with inconsistencies or tolerating inconsistencies during knowledge sharing process.

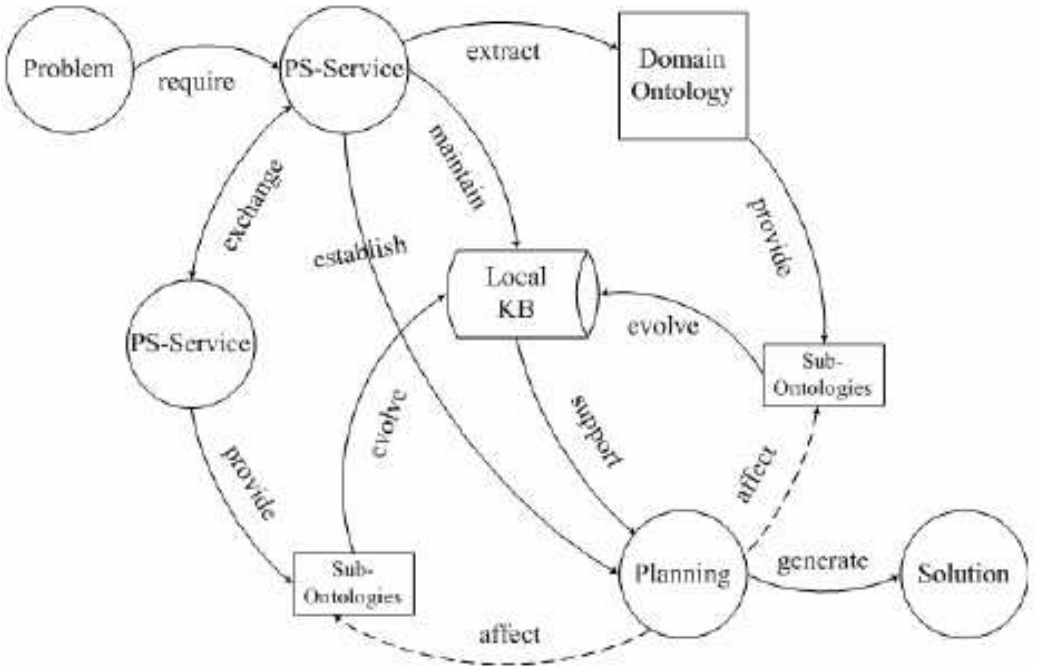

Figure 2.1: Dynamic Sub-Ontology Evolution for Collaborative Problem Solving—taken from [102]

2.2.3 Dynamic Sub-Ontology Evolution for Collaborative Problem Solving

Mao et. al. [102] use a Semantic Web framework, and propose an agent-oriented architecture, which adopts a local sub-ontology evolution mechanism for dynamic self-organization of domain knowledge to support intelligent and efficient planning for problem solving in a distributed environment like the Web/Grid. Mao et. al. approach is shown in Figure 2.1. This approach can be evaluated as follows:

C1 ⇒ *Context-awareness*: Mao et. al. framework [102] offers limited context-awareness

for its proposed *collaborative problem solving* task. It allows to extract sub-ontologies based on the input/output specifications (represented as concepts and properties in domain ontologies) for the problem solving services.

C2 ⇒ *Modularity*: It supports modularity by extracting and reusing sub-ontologies that are relevant for solving the given problem.

C3 ⇒ *Profile and Policy Management*: Mao et. al. do not make any distinction between internal and external policies.

C4 ⇒ *Correspondence Expressiveness*: Mao et. al. do not aim to merge (sub-)ontologies, therefore correspondence expressiveness is yet not supported in their framework.

C5 ⇒ *Dealing with Inconsistencies*: It provides no framework for dealing with inconsistencies or tolerating inconsistencies during its sub-ontology evolution process.

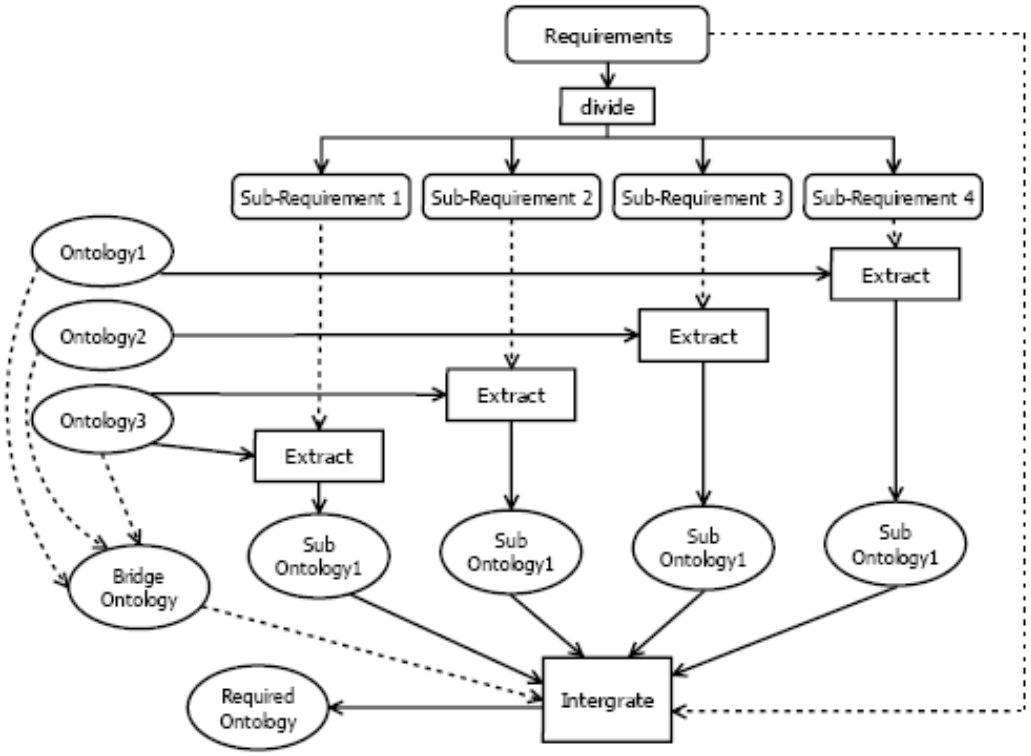

Figure 2.2: Extracting Sub-ontology from Multiple Ontologies—taken from [103]

2.2.4 Extracting Sub-ontology from Multiple Ontologies

Kang et. al. [103] propose a Semantic Web based framework for (i) extracting sub-ontologies based on the user requirements, and (ii) integrating extracted sub-ontologies into a required ontology that satisfies the user demands. Kang et. al. approach is shown in Figure 2.2. This approach is very much similar to our $\mathcal{K}$-$\mathcal{MORPH}$ framework. In Figure 2.2, (i) *Requirements* correspond to the *Problem-context*, (ii) *Bridge Ontology* correspond to the *Context-specific Alignments*, and (iii) *Required Ontology* correspond to the *Merged Ontology* in the $\mathcal{K}$-$\mathcal{MORPH}$ approach. However it is still a proposal, and no concrete results have been presented so far. This approach can be evaluated as follows:

C1 $\Rightarrow$ *Context-awareness*: Kang et. al. framework [103] allows context-awareness for integrating context-specific sub-ontologies to obtain the *required ontology*. It extracts sub-ontologies based on the given user-specific requirements, and then merges the extracted sub-ontologies.

C2 $\Rightarrow$ *Modularity*: It supports modularity by extracting and merging sub-ontologies that are relevant for the given requirements.

C3 $\Rightarrow$ *Profile and Policy Management*: Kang et. al. framework does not make any distinction between internal and external policies.

C4 $\Rightarrow$ *Correspondence Expressiveness*: The proposed framework uses the bridge ontology, based on the Distributed Description Logics (DDL) [90], which allows complex correspondences between ontologies.

C5 $\Rightarrow$ *Dealing with Inconsistencies*: It provides no framework for dealing with inconsistencies or tolerating inconsistencies during its ontology integration process.

2.2.5 Knowledge Selection Using Magpie

Similar to Kang et. al. [103], Aquin et. al. [104] plan to extend the knowledge selection process of Magpie [105] (see Figure 2.3) by (i) extracting relevant and useful fragments from source ontologies using available ontology modularization

methods, and (ii) combining the extracted ontology fragments to provide relevant knowledge for the semantic browsing with Magpie [105]. This approach can be evaluated as follows:

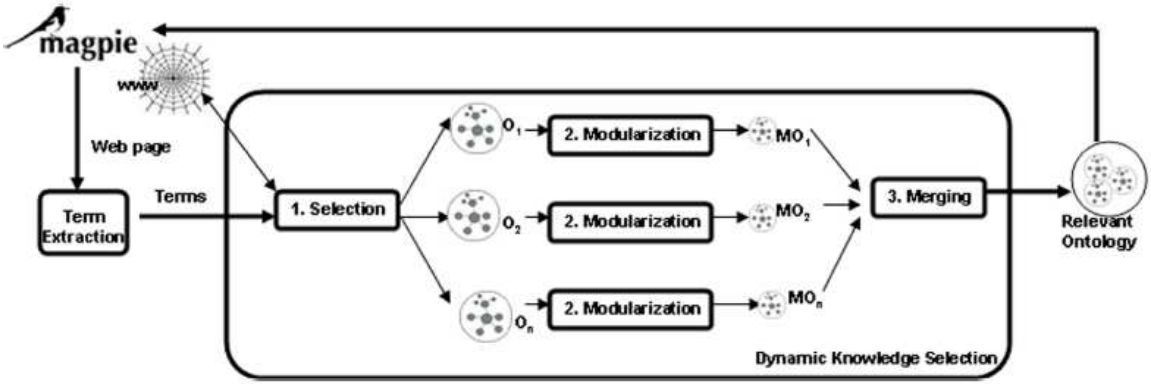

Figure 2.3: Knowledge selection process and semantic browsing with Magpie—taken from [104]

C1 ⇒ *Context-awareness*: Magpie framework [104] offers limited context-awareness for selecting and integrating context-specific sub-ontologies to obtain the *relevant ontology*. It extracts sub-ontologies based on the terms (as concepts and properties) generated by the term extraction module of the Magpie browser.

C2 ⇒ *Modularity*: It supports modularity by extracting and merging sub-ontologies that are relevant for the Magpie generated terms.

C3 ⇒ *Profile and Policy Management*: It does not make any distinction between internal and external policies.

C4 ⇒ *Correspondence Expressiveness*: The proposed framework aims to integrate relevant fragments from source ontologies, however correspondence expressiveness issue is not yet dealt in this framework.

C5 ⇒ *Dealing with Inconsistencies*: It provides no framework for dealing with inconsistencies or tolerating inconsistencies during its ontology integration process.

A comparison of the above-mentioned approaches in terms of the mentioned criteria is shown in Table 2.2.

Table 2.2: State-of-the-Art Knowledge Integration Approaches

Systems & Frameworks	**Input Format**	**C1**	**C2**	**C3**	**C4**	**C5**	**Output Knowledge**
ECOIN	Schemas	Limited	No	No	Limited	No	Integrated Knowledge
Open-Knowledge	Interaction Models	Limited	Limited	Yes	Limited	No	Shared Knowledge
Mao et. al.	Ontologies	Limited	Yes	No	No	No	Collaborative Knowledge
Kang et. al.	Ontologies	Limited	Good	No	Good	No	Integrated Knowledge
Aquin et. al.	Ontologies	Limited	Yes	No	Limited	No	Integrated Knowledge
$\mathcal{K}$-$\mathcal{MORPH}$	Ontologies	Limited	Yes	Limited	Good	Good	Integrated Knowledge

C1: *Context-awareness*; C2: *Modularity*; C3: *Profile and Policy Management*; C4: *Correspondence Expressiveness*; C5: *Dealing with Inconsistencies*.

2.3 Comparing $\mathcal{K}$-$\mathcal{MORPH}$ with other State-of-the-Art Approaches

Our $\mathcal{K}$-$\mathcal{MORPH}$ approach is in-line with the above-mentioned approaches. $\mathcal{K}$-$\mathcal{MORPH}$ performs a context-driven knowledge integration of ontology-modelled knowledge sources by (i) extracting context-specific knowledge components from available (ontology-modelled) knowledge sources using our sub-ontology extraction method (see Chapter 4); (ii) integrating the extracted knowledge components using our alignment and merging approach (see Chapter 5); and (iii) repair inconsistencies in the morphed knowledge by applying our approach for detecting and resolving inconsistencies (see Chapter 6). Based on the above-mentioned criteria, our $\mathcal{K}$-$\mathcal{MORPH}$ approach can be evaluated as follows:

C1 $\Rightarrow$ *Context-awareness*: $\mathcal{K}$-$\mathcal{MORPH}$ supports context-awareness and allows users to define a *problem-context*, whereby the user can identify problem-specific concepts, relations and context-specific constraints for extracting *contextualized sub-ontologies* from available source ontologies.

C2 $\Rightarrow$ *Modularity*: It supports modularity by extracting and merging the contextualized sub-ontologies based on the given problem-context.

C3 ⇒ *Profile and Policy Management*: $\mathcal{K}$-$\mathcal{MORPH}$ provides limited support for profile and policy management. Since a contextualized sub-ontology is represented as an RDF-Graph [106], which allows $\mathcal{K}$-$\mathcal{MORPH}$ to identify and deal with the internal profiles and policies within each sub-ontology distinctively.

C4 ⇒ *Correspondence Expressiveness*: $\mathcal{K}$-$\mathcal{MORPH}$ provides good support for correspondence expressiveness by finding and representing complex correspondences between ontology-entities. $\mathcal{K}$-$\mathcal{MORPH}$ allows users to define context-specific correspondences between ontologies. These context-specific correspondences are then used in finding complex correspondences—using our proposed matching and alignment approaches—and merging sub-ontologies.

C5 ⇒ *Dealing with Inconsistencies*: $\mathcal{K}$-$\mathcal{MORPH}$ repairs inconsistencies in the morphed knowledge using our approach for detecting and resolving inconsistencies in the merged ontology.

$\mathcal{K}$-$\mathcal{MORPH}$ provides considerable advancements towards knowledge integration, compared to other state-of-the-art approaches. Compared to ECOIN [99], $\mathcal{K}$-$\mathcal{MORPH}$ enforces modularity, which scopes the integration to only the context-specific knowledge components. Furthermore, based on the complex correspondence expressiveness and context-specific alignments provided in $\mathcal{K}$-$\mathcal{MORPH}$, the semantic reconciliation supported by ECOIN can also be achieved in $\mathcal{K}$-$\mathcal{MORPH}$. Compared to OpenKnowledge [100], Mao et. al. [102] and Aquin e. al. [104], $\mathcal{K}$-$\mathcal{MORPH}$ not only offers richer context-awareness, but also provides stronger correspondence expressiveness and policy management. Similar to Kang et. al. [103], complex and context-specific correspondences can also be expressed in $\mathcal{K}$-$\mathcal{MORPH}$. In contrast to all the above-mentioned approaches, $\mathcal{K}$-$\mathcal{MORPH}$ can detect and repair inconsistencies in the merged ontology. This feature is not yet supported by any of the above-mentioned approaches. In the following chapter, we will describe the $\mathcal{K}$-$\mathcal{MORPH}$ framework by demonstrating the active interplay of its modules using a motivating example.

Chapter 3

K-MORPH: A Knowledge Morphing Framework

Knowledge morphing aims to formulate a comprehensive knowledge-base, specific to a given context, through "the intelligent and autonomous fusion/integration of contextually, conceptually and functionally related knowledge objects that may exist in different representation modalities and formalisms, in order to establish a comprehensive, multi-faceted and networked view of all knowledge pertaining to a domain-specific problem" Abidi 2005 [29]. The need for knowledge morphing can be realized in cases when a solution from one knowledge source is lacking, or another knowledge source can play a role in deriving alternative solutions [3,9,10]. Although, knowledge integration aims to generate a comprehensive knowledge-base [2, 12, 13, 17], we argue that the integration of the entire knowledge source exacerbates the complexity of establishing knowledge interoperability between multiple knowledge sources [28]. Therefore, knowledge morphing does not deal with the complete integration of different knowledge sources; rather, it first identifies relevant knowledge components among knowledge sources based on the context at hand, and then it integrates those components to build a comprehensive knowledge-base that provides a network-view of all knowledge described/applied under the given problem-context. Knowledge morphing extends the traditional notion of knowledge integration by providing the ability to reason over the morphed knowledge to (a) infer context-specific knowledge fragments, and (b) suggest recommendations and actions for solving domain-specific problems.

3.1 Knowledge Morphing: Research Challenges

Knowledge morphing aims towards a context-driven knowledge integration of domain-related knowledge sources [13]. As discussed in the previous chapter, research challenges for building a knowledge morpher are based on the criteria

mentioned in Zimmermann et. al. [13]. Aiming towards knowledge morphing leads to three main research challenges, discussed in the following sub-sections.

3.1.1 Context Awareness

In order to obtain a comprehensive knowledge-base from available domain-related knowledge sources specific to a context at hand, context-awareness becomes crucial for both representing and identifying context-specific knowledge fragments within available knowledge sources [107]. Context-awareness over the Web has already been an active research area, and various attempts have been made in the Semantic Web community by proposing frameworks and approaches, such as Contextualizing Ontologies (C-OWL) [91], Distributed Description Logics (DDL) [90], Package-based Description Logics (P-DL) [92] and Integrated Distributed Description Logics (IDDL) [94].

3.1.2 Semantic Interoperability

Semantic Interoperability aims to resolve heterogeneity between knowledge sources. Heterogeneities can occur at different levels [76–85, 108–111]. Heterogeneities can be classified into the following main types:

1. *Syntactic Heterogeneity* occurs when two knowledge sources are expressed in two different knowledge representation languages. This kind of mismatch is generally resolved by establishing equivalences between constructs of different languages [77].

2. *Terminological Heterogeneity* occurs due to variations in vocabulary (i.e. local names) when referring to the same entities in different knowledge sources. This can be caused by a number of terminological differences due to (i) the use of different natural languages, e.g., Paper vs. Articulo, (ii) the use of different sub-domains, e.g., Paper vs. Memo, or (iii) the use of synonyms, e.g., Paper vs. Article.

3. *Conceptual Heterogeneity* stands for the differences in modelling the same domain of interest. This can happen due to differences in concept hierarchies

(i.e. *conceptualization mismatch*) [83, 112], whereby the same (or similar) concepts are arranged using different (and, sometimes, equivalent) axioms, or due to the use of similar concepts in expressing totally different concepts (i.e. *explicitation mismatch*) [83, 112].

4. *Semiotic Heterogeneity*, also called pragmatic heterogeneity [113], is concerned with how entities are interpreted. Entities that may have exactly the same semantic meaning, are often interpreted differently under different contexts. The intended use of entities, under different contexts, has a great impact on their interpretation. For example, although `<Person,Chair>` may have no obvious relevance, however under the context *conferences*, `<Person,Chair>` indeed represents a context-specific similarity between two domain-related concepts.

Ontology Reconciliation [85] addresses the problem of heterogeneity in ontologies, which allows an interchange of knowledge that is modeled in various (domain-related) ontologies. Ontology reconciliation among ontologies is normally performed by (i) identifying conceptual similarities among two source ontologies [41]; (ii) aligning and mapping sources ontologies based on identified similarities [42]; (iii) merging and integrating source ontologies based on found mappings/alignments [114]; and (iv) finding and resolving inconsistencies in reconciled ontologies [36].

3.1.3 Dealing with Inconsistencies

As knowledge engineering and sharing require a sharable and convincing consent from a group of domain experts, the evolution and sharing process mostly faces issues, such as (i) differences in conceptualization, (ii) logical contradictions, (iii) errors in conceptualization, and (iv) different realizations of the concepts and properties in different contexts and applications. Most of the above issues affect the axioms and assertions within a knowledge source, and may also turn a consistent knowledge-base into an incoherent or inconsistent one [69–72].

Regardless of whether an inconsistency occurs during knowledge engineering [9, 10] or knowledge sharing [2, 7, 8], there are mainly two ways to deal with it:

either resolve it, or reason with the inconsistent knowledge-base. When reasoning with an inconsistent knowledge-base, querying is applied on a consistent fragment of the inconsistent knowledge-base [74]. Identifying a consistent fragment is based on a selection function, which can be defined by some syntactic or semantic relevance [75]. Ontology evolution and management aims at timely monitoring of knowledge consistency within ontology-modelled knowledge sources [36,37,115]. Ontology management deals with detecting and resolving inconsistencies in both being developed and reconciled ontologies [69–72].

3.2 $\mathcal{K}$-$\mathcal{MORPH}$: Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies

By modeling knowledge sources as ontologies, semantic interoperability among knowledge sources can be achieved via ontology reconciliation [41]. However, ontology reconciliation under different contexts is still a challenge that has not been undertaken [73]. In this PhD thesis, we propose our solution approach for Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies ($\mathcal{K}$-$\mathcal{MORPH}$) as shown in Figure 3.1. In $\mathcal{K}$-$\mathcal{MORPH}$, available knowledge sources are required to be modelled as ontologies. These ontology-modelled knowledge sources serve as inputs to $\mathcal{K}$-$\mathcal{MORPH}$. In addition to the source ontologies (i.e. initially modelled knowledge sources), $\mathcal{K}$-$\mathcal{MORPH}$ also requires the *problem-context* from the user as an input. To represent the user-intended context, a problem-context is a data-structure in which the user provides (i) the user-selected concepts (properties and individuals) from the source ontologies that are pertinent to the context at hand; (ii) context-specific axioms and constraints to be applied on the source ontologies; and (iii) context-specific alignments between the source ontologies. Based on the given source ontologies and the problem-context from the user, $\mathcal{K}$-$\mathcal{MORPH}$ performs a context-driven reconciliation of source ontologies by: (a) extracting contextualized sub-ontologies from the source ontologies. The contextualized sub-ontologies get validated for conceptual/contextual consistency and completeness; (b) aligning and then merging the contextualized sub-ontologies to generate a merged ontology; and (c) detecting and resolving inconsistencies in the merged ontology. In this way, $\mathcal{K}$-$\mathcal{MORPH}$ pursues knowledge morphing by (i)

extracting knowledge components from the available ontology-modelled knowledge sources pertinent to the given problem-context by extracting contextualized sub-ontologies; (ii) integrating the extracted knowledge components by merging contextualized sub-ontologies; and (iii) repairing inconsistencies in the morphed knowledge by detecting and resolving inconsistencies in the merged ontology. In this chapter, we will describe the overall $\mathcal{K}$-$\mathcal{MORPH}$ process by demonstrating the active inter-play of the above-mentioned $\mathcal{K}$-$\mathcal{MORPH}$ tasks.

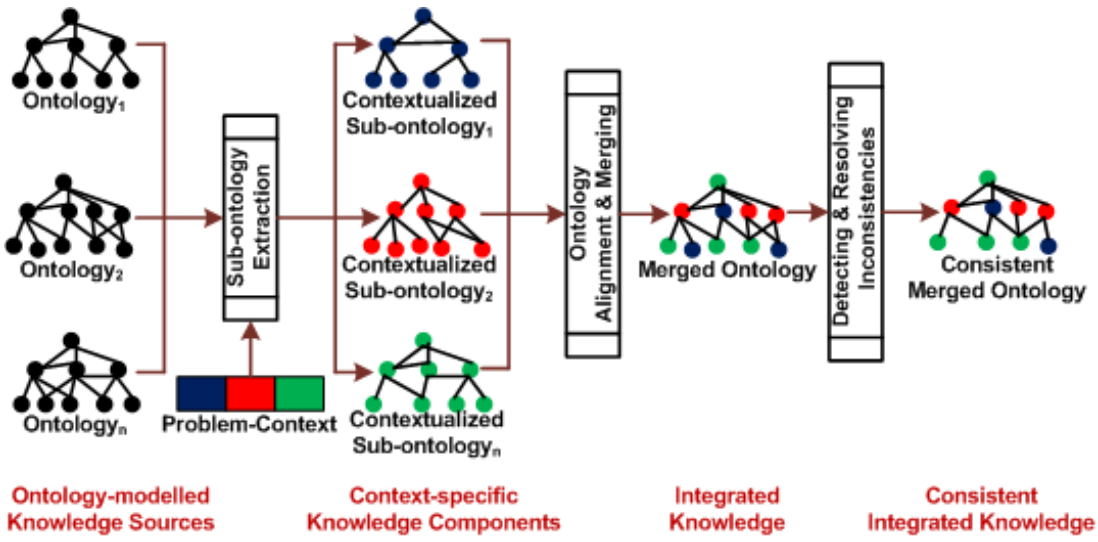

Figure 3.1: $\mathcal{K}$-$\mathcal{MORPH}$ Solution Approach

3.2.1 Preliminaries

In $\mathcal{K}$-$\mathcal{MORPH}$, for knowledge representation via ontologies, we consider RDF/OWL ontologies that are defined based on a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, comprised of concepts $\mathcal{C}$, properties $\mathcal{R}$, individuals $\mathcal{I}$, literals $\mathcal{L}$, and RDF/OWL constructs representing meta-classes $\mathcal{M}_c$ and meta-properties $\mathcal{M}_p$. An RDF/OWL ontology $\mathcal{O}$ can be expressed as triples of the form $\langle s, p, o \rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c)$. In a triple $\langle s, p, o \rangle$, s is called subject, p predicate, and o object. Triples allow to define Terminology and Assertional axioms in $\mathcal{O}$ [89]. Terminology axioms (concept axioms and property axioms) $\mathcal{T}$ are of the form $C \sqsubseteq D$ ($R \sqsubseteq S$) or $C \equiv D$ ($R \equiv S$) such that $C, D \in \mathcal{C}$ and $R, S \in \mathcal{R}$. Assertional axioms (concept assertions and property assertions) $\mathcal{A}$ are of the form $C(a)$ or $R(b, c)$ such

that $C \in \mathcal{C}$, $R \in \mathcal{R}$, $a, b, c \in \mathcal{I}$. The set of matchable entities $M_{\mathcal{O}}$ of $\mathcal{O}$ is defined as $M_{\mathcal{O}} = \mathcal{C} \cup \mathcal{R} \cup \mathcal{I}$. The semantics of an ontology is defined by an *interpretation* that provides mapping from (i) ontology individuals, (ii) ontology concepts and (iii) ontology properties to (a) elements of the domain, (b) collections of the domain-elements and (c) binary relations between the domain-elements, respectively. A *model* of an ontology is such an interpretation, under which all ontology-axioms are satisfied. An ontology is called *consistent*, iff there exists a model for it. An ontology that has no model is called an *inconsistent ontology* [116]. The set of ontologies is denoted by $\mathbb{O}$.

Definition 1 (Logic Program) *A logic program $\mathcal{P}$, over a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I} \rangle$ and a set of variables $\mathcal{X}$, is a set of Horn Logic rules of the form $X_1, \ldots, X_n \Rightarrow Y$, where $X_i, Y \in \{\langle s, p, o \rangle\} \cup \{\top, \bot\}$ such that $\langle s, p, o \rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{X}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p \cup \mathcal{X}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c \cup \mathcal{X})$. A rule of the form $X_1, \ldots, X_n \Rightarrow \bot$ is called an integrity constraint rule. $\mathbb{T}_{\mathcal{P}}(\mathcal{O})$ is the set of all asserted and inferred ontology triples of $\mathcal{O}$ under $\mathcal{P}$.*

3.2.2 Task # 1: Representing the Problem-context in $\mathcal{K}$-$\mathcal{MORPH}$

In $\mathcal{K}$-$\mathcal{MORPH}$, a problem-context is given by the user. The user-defined problem-context serves as a declarative knowledge for (i) extracting contextualized sub-ontologies from the initial source ontologies, based on the user-selected concepts (properties or individuals) defined in the problem-context; (ii) applying context-specific axioms and constraints defined in the problem-context; and (iii) establishing context-dependent interoperability between extracted sub-ontologies based on the context-specific alignments given in the problem-context. $\mathcal{K}$-$\mathcal{MORPH}$ provides the following data-structure for representing the user-defined problem-context (shown in Figure 3.2).

Definition 2 (Problem Context) *Given a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I} \rangle$, a logic program $\mathcal{P}$ and an alignment $\mathbb{A}$ between source ontologies, a problem-context $\mathcal{PC} = \langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$ is a tuple comprised of a context label l, problem-specific concepts $\mathcal{C}_x \subseteq \mathcal{C}$, problem-specific properties $\mathcal{R}_x \subseteq \mathcal{R}$, context-specific alignments $\mathbb{A}_x \subseteq \mathbb{A}$ between ontologies, and context-specific constraints and axioms $\mathcal{P}_x \subseteq \mathcal{P}$ to be applied to generate a contextualized sub-ontology.*

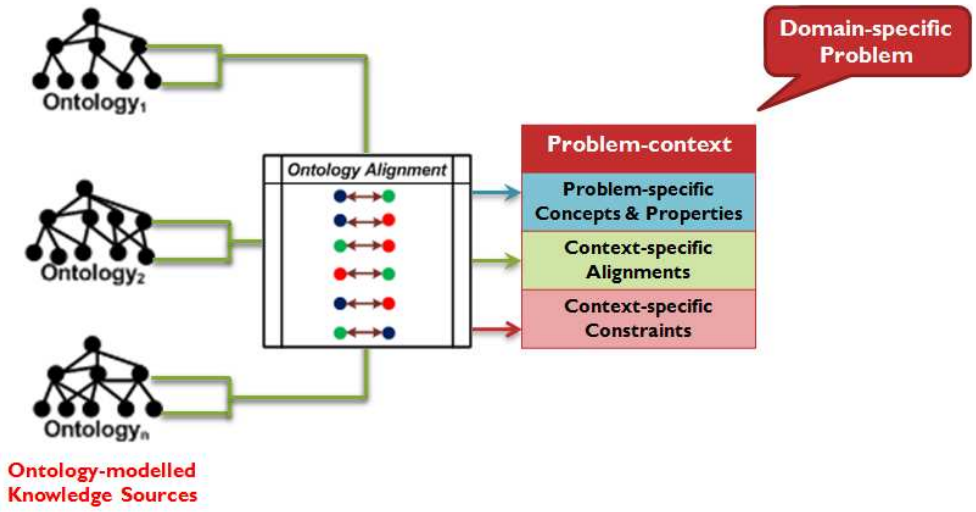

Figure 3.2: Representing Problem-context in $\mathcal{K}$-$\mathcal{MORPH}$

3.2.3 Task # 2: Extracting Contextualized Sub-ontologies

In $\mathcal{K}$-$\mathcal{MORPH}$, the user identifies problem-specific concepts and properties from the given source ontologies and declares them in the problem-context. These user-selected concepts and properties serve as inputs to our sub-ontology extraction method to extract such a contextualized sub-ontology (from the source ontology) that is pertinent to the user-intended context at hand. Based on the user-selected concepts (and properties), we apply our structure-based extraction method to extract a contextualized sub-ontology (i.e. an RDF-Sub-Graph) comprised of triples that correspond to the axioms and assertions for (i) the user-selected concept C, (ii) individuals for C, (iii) properties of C, (iv) range-concepts for the properties of C, (v) sub-concepts of C, (vi) equivalent-concepts for C, (vii) restrictions on C, (viii) complex concepts that are composed of C, (ix) only the properties of the super-concepts that are also associated with C, and (x) super-concepts of C as RDF Blank-Nodes. The contextualized sub-ontologies are validated for conceptual/contextual consistency and completeness. We also apply the context-specific constraints and context-specific ontology alignments on the extracted sub-ontologies. Our approach for extracting contextualized sub-ontologies is described in detail in Chapter 4. An abstract process for extracting contextualized sub-ontologies can be defined as follows:

Definition 3 (Contextualized Sub-ontology) *Given an RDF/OWL ontology $\mathcal{O}$ having vocabulary vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$ and a problem-context $\mathcal{PC} = \langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$. An ontology $\mathcal{O}'$ is a contextualized sub-ontology of $\mathcal{O}$ (denoted as $\mathcal{O}' \prec_c \mathcal{O}$) is constructed on a limited vocabulary $\mathcal{V} = \langle \mathcal{C}', \mathcal{R}', \mathcal{I}', \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, where ontology triples $\mathbb{T}_{\mathcal{P}}(\mathcal{O}')$ of the sub-ontology $\mathcal{O}'$ define axioms and assertions for the concepts $\mathcal{C}'$ such that $\mathcal{C}_x \subseteq \mathcal{C}' \subseteq \mathcal{C}$, properties $\mathcal{R}'$ such that $\mathcal{R}_x \subseteq \mathcal{R}' \subseteq \mathcal{R}$ and individuals $\mathcal{I}' \subseteq \mathcal{I}$; and having context-specific constraints and axioms $\mathcal{P}_x$ applied on $\mathcal{O}'$.*

Definition 4 (Extracting Contextualized Sub-ontologies) *Let $\mathbb{O}_I \subseteq \mathbb{O}$ be the set of initial source ontologies and $\mathbb{PC}$ be the set of problem-contexts, where each problem-context $\mathcal{PC} \in \mathbb{PC}$ is of the form $\langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$ representing the user-defined label l to the problem-context $\mathcal{PC}$, user-selected concepts $\mathcal{C}_x$ and properties $\mathcal{R}_x$ from $\mathbb{O}_I$, context-specific axioms and constraints $\mathcal{P}_x$ and context-specific alignments $\mathbb{A}_x$ between ontologies in $\mathbb{O}_I$. Our sub-ontology extraction method extracts a contextualized sub-ontology $\mathcal{O}' \prec_c \mathcal{O}$ from a source ontology $\mathcal{O} \in \mathbb{O}_I$, and is defined as a function:*
$extract\_sub\_onto : 2^{\mathbb{O}_I} \times \mathbb{PC} \longrightarrow 2^{\mathbb{O}}$

3.2.4 Task # 3: Merging Contextualized Sub-ontologies

Matching and alignment of ontologies have been carried out based on their lexical, conceptual and structural similarities [73]. When dealing with structural similarities, similarity scores between ontology-entities can be further improved based on the similarities between their structurally connected entities [73]. We believe that alignments between two entities e_1 and e_2 can become more 'trustworthy' by finding similarities in their justifications under a logic program $\mathcal{P}$—in which both ontology-language and domain-specific rules are defined (see Definition 1). Therefore we propose our ontology matching approaches *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM). TOM and POM can find alignments not only based on structural similarities but also take into account similarities between other deductively connected complex entity-structures—under the rules in $\mathcal{P}$ describing both domain-specific and ontology-language semantics. Our TOM and POM methods are described in detail in Chapter 5.

In $\mathcal{K}$-$\mathcal{MORPH}$, we apply our matchers TOM and POM for (i) finding alignments between the extracted contextualized sub-ontologies; and (ii) based on found

alignments, merge these sub-ontologies. TOM and POM can find new alignments between contextualized sub-ontologies based on the set of pre-defined context-specific alignments $\mathbb{A}_x$ in the given problem-context $\mathcal{PC} = \langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$. Based on both pre-defined context-specific alignments $\mathbb{A}_x$ and new alignments found by our matchers, we merge the extracted sub-ontologies and generate a merged ontology. An abstract process for merging contextualized sub-ontologies can be defined as follows:

Definition 5 (Merging Contextualized Sub-ontologies) *Let $\mathbb{O}$ be the set of ontologies, $\mathbb{P}$ be the set of logic programs, and Π be the set of pre-defined context-specific alignments. Our ontology merging method is defined as a function:*
$merge\_sub\_onto : 2^{\mathbb{O}} \times \mathbb{P} \times \Pi \longrightarrow \mathbb{O}$

3.2.5 Task # 4: Detecting and Resolving Inconsistencies

Inconsistencies—whether they occur during the process of ontology evolution or appear during the ontology reconciliation process—result in potential harm to an ontology structure, and decrease credibility in representing a consistent and shared vocabulary of an underlying domain. When an inconsistency occurs, there are mainly two ways to deal with it: either resolve it, or reason with the inconsistent ontology [69, 70, 74, 75]. In $\mathcal{K}$-$\mathcal{MORPH}$, we propose our approach for detecting and resolving inconsistencies in (merged) ontologies. Our inconsistency detection deals with the identification of contradiction derivations under the integrity constraint rules, which are given in a logic program $\mathcal{P}$. To resolve detected inconsistencies, we aim to generate all possible Minimal Inconsistent Resolve Candidates (MIRCs) (see Definition 6). Removing an MIRC from the inconsistent ontology will result in a maximal consistent sub-ontology w.r.t. the given logic program. To inform the user about the consequences of removing an MIRC, we also provide a list of all its derived triples. Our method for detecting and resolving inconsistencies is described in detail in Chapter 6. An abstract process for detecting and resolving inconsistencies can be defined as follows:

Definition 6 (Minimal Inconsistent Resolve Candidate) *Given an inconsistent ontology $\mathcal{O}$ w.r.t. a logic program $\mathcal{P}$, an inconsistent resolve candidate (IRC) is a set of triples*

$M \subseteq \mathbb{T}_{\mathcal{P}}(\mathcal{O})$ such that $\mathcal{O} - M = \mathcal{O}'$ becomes consistent w.r.t. $\mathcal{P}$. A minimal inconsistent resolve candidate (MIRC) is such an IRC M such that for any $M' \subset M$, $\mathcal{O} - M' = \mathcal{O}'$ remains inconsistent w.r.t. $\mathcal{P}$. The set of MIRCs is denoted by $\mathcal{M}$.

Definition 7 (Detecting and Resolving Inconsistencies) *Let $\mathbb{O}$ be the set of ontologies, $\mathbb{P}$ be the set of logic programs, and $\mathcal{M}$ be the set of MIRCs $\mathbb{M}$. Our method for detecting and resolving inconsistencies is defined as a function:*
$resolve\_inconsis : \mathbb{O} \times \mathbb{P} \longrightarrow \mathbb{O} \times \mathcal{M}$

3.2.6 $\mathcal{K}$-$\mathcal{MORPH}$ Process

$\mathcal{K}$-$\mathcal{MORPH}$ pursues context-driven ontology reconciliation guided by the problem-context. Based on the above defined tasks, an abstract process of $\mathcal{K}$-$\mathcal{MORPH}$ can be defined as follows:

Definition 8 ($\mathcal{K}$-$\mathcal{MORPH}$ Process) *Let $\mathbb{O}_I \subseteq \mathbb{O}$ be the set of initial source ontologies, $\mathbb{PC}$ be the set of problem-contexts, Π be the set of pre-defined context-specific alignments, $\mathbb{P}$ be the set of logic programs, and $\mathbb{M}$ be the set of MIRCs $\mathcal{M}$. Our $\mathcal{K}$-$\mathcal{MORPH}$ process is defined as a function:*
$\mathcal{K}$-$\mathcal{MORPH} : 2^{\mathbb{O}_I} \times \mathbb{PC} \longrightarrow \mathbb{O} \times \mathbb{M}$, and implemented by the above defined methods:

$$\mathcal{K}\text{-}\mathcal{MORPH}(\mathbb{O}_I, \mathcal{PC}) = resolve\_inconsis(merge\_sub\_onto(extract\_sub\_onto(\mathbb{O}_I, \mathcal{PC}), \mathcal{P}, \mathbb{A}_x), \mathcal{P})$$

3.3 $\mathcal{K}$-$\mathcal{MORPH}$ Applications: Morphing Prostate Cancer Clinical Pathways for Therapeutic Decision Support

In order to show the support of knowledge morphing for developing different domain-specific applications, we demonstrate the use of our knowledge morphing framework $\mathcal{K}$-$\mathcal{MORPH}$ in the healthcare domain, by generating a comprehensive and context-sensitive knowledge-base for Prostate Cancer (PC) management. For this experiment, we use three location-specific PC clinical pathways [62]. All three PC pathways are modelled as ontologies [62], entailing their institution-specific knowledge about diagnosis, treatments, tests and follow-up care for PC at three different locations: Halifax, Winnipeg and Calgary.

Consider a healthcare practitioner, who is in need of a comprehensive therapeutic knowledge-base that can provide and compare different consensus for treating PC patients. Thus in this scenario, only the therapeutic (and other related) knowledge fragments should be extracted from the PC pathways, and then merged to obtain a comprehensive therapeutic knowledge about prostate cancer (as shown in Figure 3.3).

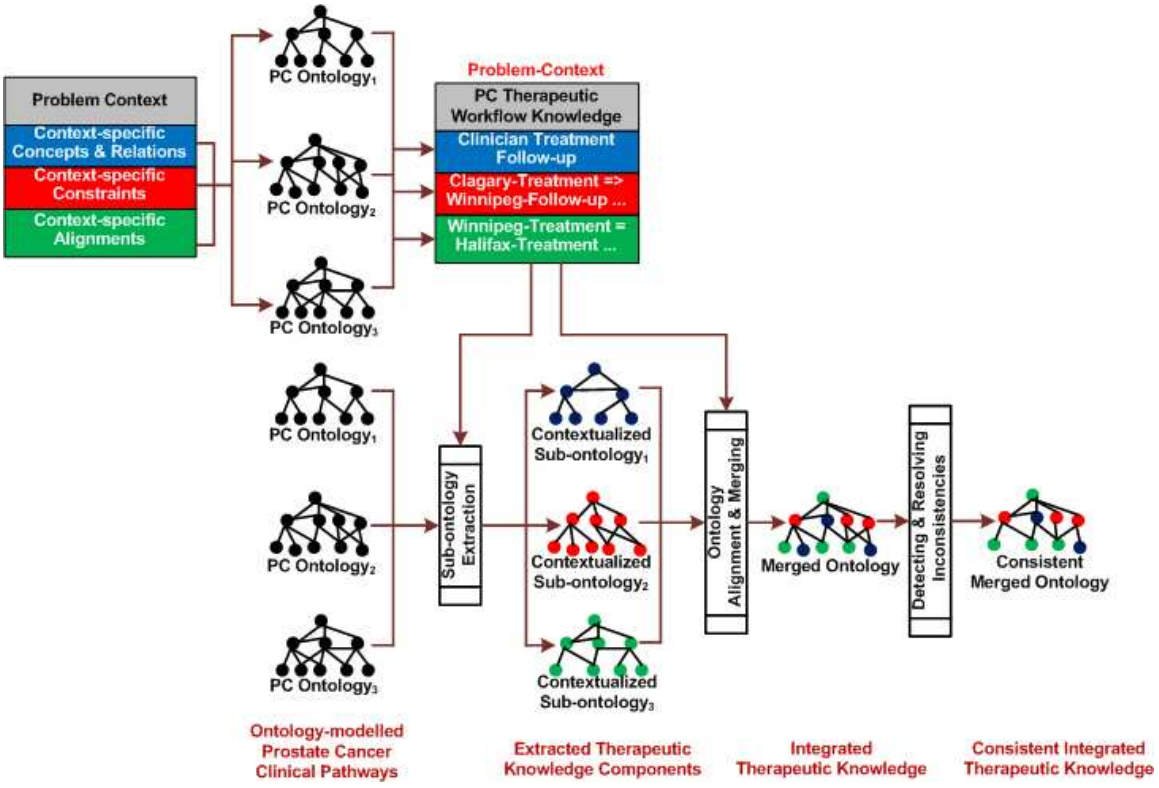

Figure 3.3: $\mathcal{K}$-$\mathcal{MORPH}$: Generating Therapeutic Workflow Knowledge by Morphing Prostate Cancer Clinical Pathways

The objective of this experiment is to generate a more comprehensive PC ontology, whereby one can suggest alternative treatments or extend interventions at one location based on knowledge contained in other PC ontologies. For instance, if location L_1 is prescribing intervention I_1 for condition C, and location L_2 is prescribing intervention I_2 for condition C', where conditions C and C' are similar to each other, then one can imply, after satisfying clinical pragmatics, that condition C (or C') can be treated by both interventions I_1 and I_2. Likewise, if location L_3 has no information about how to handle condition C (or C'), then the same inference

can be applied to suggest interventions I_1 and I_2.

In order to fulfill the desired objective (i.e. *therapeutic decision support*) using $\mathcal{K}$-$\mathcal{MORPH}$, we will first define the given scenario in terms of the problem-context in $\mathcal{K}$-$\mathcal{MORPH}$, and then briefly demonstrate the main tasks achieved in $\mathcal{K}$-$\mathcal{MORPH}$ for generating a comprehensive therapeutic knowledge about PC.

3.3.1 Task # 1: Defining Problem-context: *therapeutic decision support*

We define the given scenario/application *therapeutic decision support* as a problem-context in $\mathcal{K}$-$\mathcal{MORPH}$:

```
1. Cx1 = ⟨ therapeutic-decision-support,
2.     {pc-calgary:Clinician, pc-halifax:Clinician, pc-winnipeg:Clinician,
3.     pc-calgary:Treatment, pc-halifax:Treatment, pc-winnipeg:Treatment},
4.     ∅,
5.     {<winnipeg:Clinician, pc-halifax:Clinician, owl:equivalentClass, 1.0>,
6.     <winnipeg:Treatment, pc-halifax:Treatment, owl:equivalentClass, 1.0>}
7.     {{?X a pc-calgary:Clinician} => {?X a pc-halifax:Clinician}.
8.      {?X a pc-calgary:Treatment} => {?X a pc-winnipeg:Followup}}
9.   ⟩
```

$\mathcal{C}_{x1}$ is annotated by its unique label `therapeutic-decision-support` and described by its *context-axioms*. For $\mathcal{C}_{x1}$, the list of concepts (or properties) identified by the user from each of the three source PC ontologies are `Clinician` and `Treatment` (see line 2-3). Moreover in $\mathcal{C}_{x1}$, the user has given pre-defined alignments between PC ontologies (see line 5-6). In $\mathcal{C}_{x1}$, context-axioms declare context-specific knowledge, such as (i) *Calgary-Clinicians* are to be realized as *Halifax-Clinicians*; and (ii) *Calgary-Treatments* can be viewed as *Winnipeg-Followups* (see line 7-8).

3.3.2 Task # 2: Extracting Contextualized PC Sub-ontologies

For *therapeutic decision support* context (denoted as $\mathcal{C}_{x1}$), the user is interested in such extracted sub-ontologies that describe only (i) the treatments, (ii) their durations, (iii) their follow-ups, (iv) their care-settings, and (v) the practitioners involved for them. We applied our sub-ontology extraction approach (see Chapter 4) and extracted contextualized ontologies from three PC pathway ontologies. Chapter 4 will present our sub-ontology extraction approach and also demonstrate the

extraction of contextualized PC sub-ontologies in detail. Figure 3.4 shows a contextualized fragment of the PC-Halifax sub-ontology.

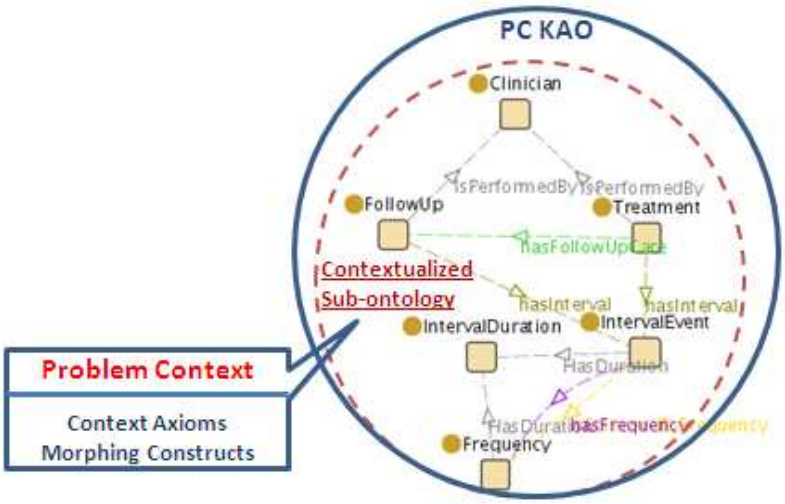

Figure 3.4: Contextualized Sub-ontology from PC Pathway Ontology

3.3.3 Task # 3: Merging Contextualized PC Sub-ontologies

The next step in $\mathcal{K}$-$\mathcal{MORPH}$ is to merge the extracted contextualized PC sub-ontologies. We applied our ontology alignment approaches, TOM and POM (see Chapter 5), and generated a merged ontology that provides a comprehensive therapeutic workflow for PC management. Chapter 5 will present our ontology alignment and merging approaches, and also demonstrate the merging of the extracted PC sub-ontologies in detail.

Figure 3.5 shows one of the exemplar results generated after merging PC sub-ontologies. In figure 3.5, the merged knowledge has determined that the treatment *Active Surveillance* in Halifax (represented by the instance `PC-Halifax:ActiveSurveillance`) can be conducted by a *Primary Urologist*. In the actual pathway, this information was not available for Halifax; but due to the ontology alignments, this task was found to be similar to one in Calgary, and the actor performing this task in Calgary was extended to Halifax.

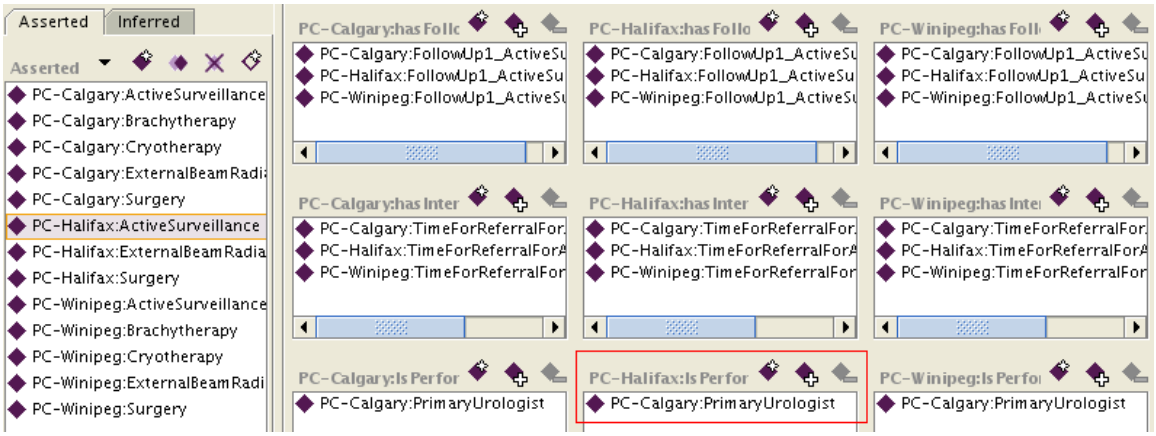

Figure 3.5: Merged Knowledge about PC-Halifax:ActiveSurveillance (highlighted in the left panel)

3.3.4 Task # 4: Detecting and Resolving Inconsistencies in PC ontologies

During the reconciliation and merging of these three extracted PC contextualized sub-ontologies, $\mathcal{K}$-$\mathcal{MORPH}$ found inconsistencies in the merged PC ontology. To resolve the detected inconsistencies, we generated all possible MIRCs for extracting a maximal consistent sub-ontology from the merged PC ontology. Chapter 6 will present our approach for detecting and resolving inconsistencies, and will also demonstrate the detection and resolution of inconsistencies in the merged PC ontology.

Chapter 4

Extracting Contextualized Sub-ontologies

A key aspect of Semantic Web research is knowledge modeling that is pursued by developing a domain-specific ontology [32]. Ontologies [89] serve as the backbone for the Semantic Web as they provide formal constructs to model the knowledge of a domain based on domain-specific concepts, their relationships and constraints. A domain-specific ontology, then, serves as the foundational representation formalism for a variety of activities, such as web-based information sharing, retrieval, mediation, collaboration and decision support [6, 32]. Knowledge modelers, or ontology engineers, develop domain ontologies in a variety of ways—i.e. developing a new ontology by abstracting domain concepts and relations, adding new concepts to an existing ontology [36], or even aligning/merging multiple existing ontologies to realize a specialized ontology [41, 42, 85]. For mature domains, such as medicine, genetics and business [11–13], the common practice is to leverage large-scale domain ontologies in order to systematically select a sub-set of concepts or a specialized sub-ontology that is representative of the task at hand [28, 39]. A sub-ontology presents a focused representation, at a desired level of abstraction, of selected aspects of the source ontology, whilst offering the operational processing and knowledge constraints pertinent to the problem-context at hand. The rationale for extracting contextualized sub-ontologies is that working with the larger ontologies leads to the introduction of irrelevant concepts to the required knowledge model for a specialized problem, which in turn not only extends the deductive closure of the model to undesired interpretations, but also compromises the efficiency, complexity and granularity of the knowledge model [27, 28]. Therefore, for solving specialized problems in large domains there is a need to extract a sub-ontology that offers a consistent and relevant fragment of the source ontology.

Structure-based sub-ontology extraction [64–68] is usually guided by a particular problem-context that may lead to the selection, adaptation, reconciliation, or

merging of concepts from multiple extracted sub-ontologies [41]. However, the challenge is to ensure that a merged sub-ontology does not lead to (i) differences in domain conceptualization, (ii) logical contradictions, or (iii) inaccurate realizations of the concepts and their properties as per context, as these issues render the sub-ontology inconsistent [116]. In this case, it is important that the extracted sub-ontology is semantically consistent.

Ontologies and contexts are used to model a domain with different views. Ontologies define a shared model that provides a global perspective, whereas contexts are used to realize a local aspect of a domain. A contextualized (sub-)ontology deals with an adaptation of the ontology model to support a local view, and provides (i) specific interpretation of the ontology concepts; and (ii) implementation of the procedural knowledge that can be applied in a particular context [46].

To achieve knowledge morphing via the $\mathcal{K}$-$\mathcal{MORPH}$ framework, we have developed a structure-based method for extracting contextualized sub-ontologies via N3 rules [117]. Our approach is to exploit the complete inference rules for RDF(S) and OWL defined in [49], and to extract a sub-ontology that consists of the axioms and assertions for user-selected concepts and their structurally connected concepts, properties and individuals—from the source ontology. Although our extraction rules can be used by various Semantic Web reasoning engines, we use the Euler inference engine [118] to infer interpretations beyond basic ontological structures to extract a semantically-rich sub-ontology. In this way, we argue that our approach extracts a more semantically extensive sub-ontology compared to the semantics offered by existing sub-ontology extraction approaches [64–67] (see Section 4.4). We demonstrate the working of our sub-ontology extraction approach by extracting contextualized sub-ontologies from three prostate cancer pathway ontologies [62]. Furthermore, we present a comparison of our approach with a few existing sub-ontology extraction approaches.

4.1 Structure-based Sub-ontology Extraction: Overview and Approaches

For structure-based sub-ontology extraction, a source ontology is considered as an un-directed graph, and an extracted sub-ontology is a sub-graph that describes the knowledge about the selected concepts pertinent to a specified context [64–68].

One of the initial attempts at sub-ontology extraction is the Materialized Ontology View Extraction (MOVE) process [64], where a sub-ontology is extracted based on the *requirements consistency optimization scheme* (RCOS) and *semantic completeness optimization scheme* (SCOS). Given a source ontology, MOVE allows the user to label both concepts and properties as *selected* or *un-selected*. For a selected property R, based on RCOS criteria, MOVE extracts (i) concept axioms for R; (ii) property assertions for R and the concept axioms for concepts that are associated with R; and so on. SCOS is applied to preserve the defined semantic completeness in MOVE. For a selected concept C, based on SCOS criteria (a) the super-concepts of C and their associated properties must be selected; (b) the sub-concepts of C and their associated properties must be selected; (c) the properties of C that require a minimum cardinality greater than zero, and their property assertions must be selected. It may be noted that when the MOVE approach is applied to a connected graph (of an ontology), where there exists a path between every two nodes, due to the SCOS criterion the extracted sub-graph can potentially result in extracting of the entire graph itself (i.e. the whole ontology). A more restricted approach compared to MOVE is reported in Seidenberg et. al. [65], where (i) (similar to SCOS criteria (a)-(c)) all super-concepts, sub-concepts and restrictions of C are extracted; (ii) their properties and individual are extracted; however (iii) sibling of C and sub-property axioms of the extracted properties must not be extracted. It also restricts the depth-limit of the extracted concept hierarchy up to the defined *boundary concepts*. Similarly, Noy et. al. [66] propose an extraction of sub-ontologies by traversing ontology structure, and also restrict unnecessary expansion of the extracted graph by defining boundary concepts. Further restricted extraction is described by Miao et. al. [67] that extracts sub-ontologies from multiple RDF(S) ontologies. It only finds a restricted sub-graph that is induced by the set of selected nodes (concepts)—it extracts only the selected nodes (concepts, properties, and individuals), and does not extract any further concepts, properties and individuals associated to the selected nodes.

Compared to the existing sub-ontology extraction approaches, in our extraction approach, we focus on two main aspects:

1. **Controlled Extraction:** We deal with the SCOS criterion differently than the existing MOVE and other approaches [64–66]. The key feature of our approach is that for a given target concept C, we restrict the recursive selection of the super-concepts of C in order to avoid the unnecessary selection of the siblings and super-concepts of C that results in the expansion of the sub-graph to include concepts that are not relevant. However for the semantic completeness of C and its sub-concepts, the super-concepts of C are replaced by *blank-nodes*, and the properties of the super-concepts of C are now described with C (via extracted blank-nodes). In this way, we avoid a situation where the sub-ontology is too-generalized by the undesired inclusion of higher levels of concepts that may even extend all the way to `owl:Thing`.

2. **Extraction for both RDF(S) and OWL Ontologies:** We improve on the rather restricted sub-ontology extraction approach by Miao et. al. [67], by extending our approach to both RDF(S) and OWL ontologies, where our sub-ontology extraction method is applied on the closure of RDF(S) and OWL ontologies under their complete inference rules [49]. For example, for the concept axiom `:C3 owl:intersectionOf (:C1 :C2)`, based on the OWL inference rules [49], additional axioms `:C3 rdfs:subClassOf :C1` and `:C3 rdfs:subClassOf :C2` can be inferred—that provide additional RDF(S) semantics for OWL concepts `:C1`, `:C2` and `:C3`.

4.2 Preliminaries

For our purpose, we consider RDF/OWL ontologies that are defined based on a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, comprised of concepts $\mathcal{C}$, properties $\mathcal{R}$, individuals $\mathcal{I}$, literals $\mathcal{L}$, and RDF/OWL constructs representing meta-classes $\mathcal{M}_c$ and meta-properties $\mathcal{M}_p$. An RDF/OWL ontology $\mathcal{O}$ can be expressed as triples of the form $\langle s, p, o \rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c)$. In a triple $\langle s, p, o \rangle$, s is called subject, p predicate, and o object. Triples allow to define Terminology and Assertional axioms in $\mathcal{O}$ [89]. Terminology axioms (concept axioms and property axioms) $\mathcal{T}$ are of the form $C \sqsubseteq D$ ($R \sqsubseteq S$) or $C \equiv D$ ($R \equiv S$) such that $C, D \in \mathcal{C}$ and $R, S \in \mathcal{R}$. Assertional axioms (concept assertions and property

assertions) $\mathcal{A}$ are of the form $C(a)$ or $R(b, c)$ such that $C \in \mathcal{C}$, $R \in \mathcal{R}$, $a, b, c \in \mathcal{I}$. The semantics of an ontology is defined by an *interpretation* that provides mapping from (i) ontology individuals, (ii) ontology concepts and (iii) ontology properties to (a) elements of the domain, (b) collections of the domain-elements and (c) binary relations between the domain-elements, respectively. A *model* of an ontology is such an interpretation, under which all ontology-axioms are satisfied. An ontology is called *consistent*, iff there exists a model for it. An ontology that has no model is called an *inconsistent ontology* [116]. The set of ontologies is denoted by $\mathbb{O}$.

Definition 9 (Contextualized Sub-ontology) *Given an RDF/OWL ontology $\mathcal{O}$ having vocabulary vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$ and a problem-context $\mathcal{PC} = \langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$ (see Definition 2). An ontology $\mathcal{O}'$ is a contextualized sub-ontology of $\mathcal{O}$ (denoted as $\mathcal{O}' \prec_c \mathcal{O}$) is constructed on a limited vocabulary $\mathcal{V} = \langle \mathcal{C}', \mathcal{R}', \mathcal{I}', \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, where ontology triples $\mathbb{T}_{\mathcal{P}}(\mathcal{O}')$ of the sub-ontology $\mathcal{O}'$ define axioms and assertions for the concepts $\mathcal{C}'$ such that $\mathcal{C}_x \subseteq \mathcal{C}' \subseteq \mathcal{C}$, properties $\mathcal{R}'$ such that $\mathcal{R}_x \subseteq \mathcal{R}' \subseteq \mathcal{R}$ and individuals $\mathcal{I}' \subseteq \mathcal{I}$; and having context-specific constraints and axioms $\mathcal{P}_x$ applied on $\mathcal{O}'$.*

One of the Euler built-ins used in our work is `(?SCOPE ?SPAN) e:findall (?SELECT ?WHERE ?ANSWER)`, which unifies `?ANSWER` with a list that contains all the instantiations of `?SELECT` (represented as a N3 formula) satisfying the `?WHERE` clause (represented as a N3 formula) in the `?SCOPE ?SPAN` of all asserted N3 formulae and their `log:conclusion` [118].

4.3 Structure-based Approach for Extracting Contextualized Sub-ontologies

In $\mathcal{K}$-$\mathcal{MORPH}$, the user identifies a set of pertinent concepts (in the problem-context) from each of the source ontologies to generate contextualized sub-ontologies (see Section 3.2.2). Based on the user-selected concepts $\mathcal{C}_s$ from an ontology $\mathcal{O}$, we apply our structure-based extraction method to extract a contextualized sub-ontology $\mathcal{O}' \prec_c \mathcal{O}$ (i.e. an RDF-Sub-Graph) comprised of triples that correspond to the axioms and assertions of the following forms:

1. Selected Concept: The user-selected concept $C \in \mathcal{C}_s$.

2. Individuals: Individuals for the selected concept C.

3. Properties: Properties of C; and their sub-properties.

4. Property Ranges: Range-concepts for the properties of C.

5. Sub-concepts: Sub-concepts of C.

6. Equivalent Concepts: equivalent-concepts for C.

7. Restrictions: Property, cardinality and value restrictions on C.

8. Complex Concepts: Complex concepts, such as union or intersection, that are composed of C.

9. Associated Properties: Only such properties of the super-concepts that are also associated with C.

10. Restricted Super-concepts: Super-concepts of C as RDF blank-nodes.

An abstract process for extracting contextualized sub-ontologies can be defined as follows:

Definition 10 (Extracting Contextualized Sub-ontologies) *Let $\mathbb{O}_I \subseteq \mathbb{O}$ be the set of initial source ontologies and $\mathbb{PC}$ be the set of problem-contexts (see Definition 2), where each problem-context $\mathcal{PC} \in \mathbb{PC}$ is of the form $\langle l, \mathcal{C}_x, \mathcal{R}_x, \mathbb{A}_x, \mathcal{P}_x \rangle$ representing the user-defined label l to the problem-context $\mathcal{PC}$, user-selected concepts $\mathcal{C}_x$ and properties $\mathcal{R}_x$ from $\mathbb{O}_I$, context-specific axioms and constraints $\mathcal{P}_x$ and context-specific alignments $\mathbb{A}_x$ between ontologies in $\mathbb{O}_I$. Our sub-ontology extraction method extracts a contextualized sub-ontology $\mathcal{O}' \prec_c \mathcal{O}$ from a source ontology $\mathcal{O} \in \mathbb{O}_I$, and is defined as a function:*
$extract\_sub\_onto : 2^{\mathbb{O}_I} \times \mathbb{PC} \longrightarrow 2^{\mathbb{O}}$

In the following sub-sections, we will first introduce the preliminary concepts and definitions, and then describe our sub-ontology extraction approach in detail.

4.3.1 Concept Selection

Let $\mathcal{C}_s$ be the set of user-selected concepts, and $\mathcal{C}_e$ be the set of extracted concepts, $\mathcal{R}_e$ be the set of extracted properties, $\mathcal{I}_e$ be the set of individuals to be extracted from an ontology $\mathcal{O}$ for given $\mathcal{C}_s$. In our method, user-selected concepts are labelled

as triples of the form $\mathcal{O}$ `usr:selected` C, where $C \in \mathcal{C}_s$ is an user-selected concept from an ontology $\mathcal{O}$ for extracting a sub-ontology $\mathcal{O}' \prec_c \mathcal{O}$. Any selected concept is also extracted ($\mathcal{O}$ `sbont:extract` C) in $\mathcal{O}'$ (by the rule: `{?O usr:selected ?C} => {?O sbont:extract ?C}`) denoted as $C \in \mathcal{C}_e$. We also check whether a concept (or property) is extracted (i.e. `sbont:extract`) in $\mathcal{O}'$ or not, using the Euler built-in `e:findall`. Any concept (or property) that is not extracted in $\mathcal{O}'$ is inferred as triples of the form $\mathcal{O}$ `sbont:noValue` C (denoted as $C \notin \mathcal{C}_e$)—under the following N3 rules:

(i) `{?C a rdfs:Class. (?scp 3) e:findall (? {?O sbont:extract ?C} ())} => {?O sbont:noValue ?C}.`

(ii) `{?P a owl:ObjectProperty. (?scp 3) e:findall (? {?O sbont:extract ?P} ())} => {?O sbont:noValue ?P}.`

(iii) `{?P a owl:DatatypeProperty. (?scp 3) e:findall (? {?O sbont:extract ?P} ())} => {?O sbont:noValue ?P}.`

4.3.2 Identification of Selected Concepts, Sub-concepts, and their Properties

Concepts are usually defined in terms of other (sub-)concepts and properties via concept axioms and property axioms, respectively. Hence when a concept C is extracted in a sub-ontology $\mathcal{O}'$, it is already deemed necessary to extract all the sub-concepts and properties of C in $\mathcal{O}'$ so that the complete semantics of C is still preserved in $\mathcal{O}'$. For this purpose, for any selected concept C, we also select sub-concepts and properties of C to be extracted in $\mathcal{O}'$ by the following N3 rules:

Sub-concept: All sub-concepts of C are also selected (i.e. `usr:selected`):
`{?O usr:selected ?C. ?C a rdfs:Class. ?S rdfs:subClassOf ?C} => {?O usr:selected ?S}`, where `rdfs:subClassOf a owl:TransitiveProperty.`

Properties: All properties of C should also be extracted (i.e. `sbont:extract`) in $\mathcal{O}'$:
(i) `{?O usr:selected ?C. ?C a rdfs:Class. ?P rdfs:domain ?C} => {?O sbont:extract ?P}.`

All properties of the super-concepts of C should also be extracted in $\mathcal{O}'$:
(ii) {?O usr:selected ?C. ?C a rdfs:Class. ?C rdfs:subClassOf ?C1. ?P rdfs:domain ?C1} => {?O sbont:extract ?P}.

Property-ranges: All property-range concepts of extracted properties should also be extracted in $\mathcal{O}'$:
(i) {?O usr:selected ?C. ?C a rdfs:Class. ?P rdfs:domain ?C. ?P rdfs:range ?C1} => {?O sbont:extract ?C1};
(ii) {?O usr:selected ?C. ?C a rdfs:Class. ?C rdfs:subClassOf ?D. ?P rdfs:domain ?D. ?P rdfs:range ?C1} => {?O sbont:extract ?C1}.

4.3.3 Inheriting Properties from Super-concepts

For a selected concept $C \in \mathcal{C}_s$, our method does not extract the axioms and assertions of the super-concepts of C; rather it extracts the properties of the super-concepts that are inherited in C. In order to discuss the rationale behind this step, consider the extracted sub-ontology shown in Example 4.1. The user is interested in a specialized concept :C2, but not its super-concept :C1—that is more general to :C2. Since :C2 rdfs:subClassOf :C1, therefore :P1 is an inherited property in :C2. Although :P1 is structurally connected with :C2 via its individuals (e.g. :i2 :P1 :i4.). However for :P1, including its domain concept :C1 would result in an over-generalized extracted sub-ontology. For this, we replace the super-concepts (e.g. :C1) of the selected concepts (e.g. :C2) by unique RDF blank-nodes (e.g. _:t8 e:tuple (:C1)), using the Euler built-in e:tuple [118]. Section 4.3.8 and 4.3.7 describe the construction for such RDF blank-nodes. In Example 4.1, _:t8 is the unique RDF blank-node for :C1, and is structurally connected with :C2 by :C2 rdfs:subClassOf _:t8 and :P1 rdfs:domain _:t8. Hence, due to our extraction approach for super-concepts, unnecessary expansion of the extracted sub-ontology through siblings concepts is restricted [65].

4.3.4 Extraction of Complex Concepts

OWL constructs, such as owl:intersectionOf and owl:unionOf, allow users to define complex concepts using already defined (atomic or complex) concepts. For

our purposes, when complex concepts (or the concepts that define complex concepts) are required to be extracted, sub-ontology extraction must be scoped to the required concepts only, not to their more-generalized concepts (see Section 4.3.3). In the Example 4.1, `:C5` is defined as union of the concepts `:C2` and `:C3`, where `:C5` is a more-generalized concept of `:C2`, and `:C5` is not of interest to the user. Although, for the selected concept `:C2`, it is important to deal with `:C5` as it is structurally connected (under the concept axiom `:C5 owl:unionOf (:C2 :C3)`) with `:C2`. However, extracting `:C5` and its axioms would unnecessarily extend the scope of extraction to other undesired concepts. In order to allow a scoped extraction of complex concepts, the semantics of such complex concepts (defined by `owl:intersectionOf` and `owl:unionOf` constructs) is described as concepts in RDF(S) semantics by the following rules:

(a) {`?C owl:intersectionOf ?L. ?L a rdf:List.  ?X list:in ?L`} =>
{`?C rdfs:subClassOf ?X`}.

(b) {`?C owl:unionOf ?L. ?L a rdf:List.  ?X list:in ?L`} =>
{`?X rdfs:subClassOf ?C`}.

Using the above rules, complex OWL concepts (that can be nested unions or intersections of concepts) are defined using the `rdfs:subClassOf` construct. In the example ontology $\mathcal{O}_1$ (see left-side of Example 4.1), `:C4` is defined as an intersection of the concepts `:C2` and `:C3`. Therefore, by rule (a), `:C4 rdfs:subClassOf :C2` and `:C4 rdfs:subClassOf :C3` are inferred. Similarly, by rule (b), `:C2 rdfs:subClassOf :C5` and `:C3 rdfs:subClassOf :C5` are inferred. Given `:C2` as a selected concept, the extracted sub-ontology from $\mathcal{O}_1$ is shown (on the right-side) in Example 4.1. Given `:C4 rdfs:subClassOf :C2` in $\mathcal{O}_1$, by *sub-concept* rule (in Section 4.3.2), `:C4` is also considered as a selected concept. For the selected concept `:C2`, the super-concepts are `:C1` and `:C5`. For `:C4`, the super-concepts are `:C1`, `:C2`, `:C3` and `:C5`. Based on the earlier described rationale (see Section 4.3.3), all the non-selected super-concepts of the selected concepts `:C2` and `:C4` are described as RDF blank-nodes (see bottom-part of Example 4.1).

Example 4.1: Sub-Ontology extraction with complex concepts

~~Strikeout~~ **Triples: Not included in the extracted sub-ontology.**

OWL Ontology ($\mathcal{O}_1$):	**Extracted Sub-ontology (`:C2` is user-selected):**
`:C1 a owl:Class; rdfs:subClassOf owl:Thing`	`_:t8 a owl:Class; rdfs:subClassOf owl:Thing`
`:C2 a owl:Class; rdfs:subClassOf :C1.`	`:C2 a owl:Class; rdfs:subClassOf _:t8.`
`:C3 a owl:Class; rdfs:subClassOf :C1.`	`:C3 a owl:Class; rdfs:subClassOf _:t8.`
`:C4 a owl:Class;`	`:C4 a owl:Class.`
`owl:intersectionOf (:C2 :C3).`	`:C4 rdfs:subClassOf :C2, :C3, _:t8, _:t9.`
`:C5 a owl:Class; owl:unionOf (:C2 :C3).`	`_:t9 a owl:Class. :C2 rdfs:subClassOf _:t9.`
	`:C3 rdfs:subClassOf _:t9.`
`:C6 a owl:Class; rdfs:subClassOf :C1.`	~~`:C6 a owl:Class; rdfs:subClassOf :C1.`~~
`:P1 a owl:ObjectProperty;`	`:P1 a owl:ObjectProperty;`
`rdfs:domain :C1; rdfs:range :C2.`	`rdfs:domain _:t8; rdfs:range :C2.`
`:P2 a owl:ObjectProperty;`	`:P2 a owl:ObjectProperty;`
`rdfs:domain :C2; rdfs:range :C3.`	`rdfs:domain :C2; rdfs:range :C3.`
`:i1 a :C1. :i11 a :C1.`	~~`:i1 a :C1.`~~ ~~`:i11 a :C1.`~~
`:i2 a :C2. :i22 a :C2.`	`:i2 a :C2. :i22 a :C2.`
`:i3 a :C3. :i33 a :C3.`	`:i3 a :C3.` ~~`:i33 a :C3.`~~
`:i4 a :C4. :i5 a :C5. :i6 a :C6.`	`:i4 a :C4.` ~~`:i5 a :C5.`~~ ~~`:i6 a :C6.`~~
`:i2 :P1 :i4. :i2 :P2 :i3.`	`:i2 :P1 :i4. :i2 :P2 :i3.`
`:i3 :P1 :i4. :i4 :P1 :i2.`	`:i3 :P1 :i4. :i4 :P1 :i2.`
`:i6 :P1 :i2. :i6 :P1 :i4.`	~~`:i6 :P1 :i2.`~~ ~~`:i6 :P1 :i4.`~~
	RDF Blank-Nodes: `_:t8 e:tuple (:C1).`
	RDF Blank-Nodes: `_:t9 e:tuple (:C5).`

4.3.5 Dealing with Concept Axioms for Extracted Concepts

For an extracted concept C in a sub-ontology $\mathcal{O}'$, it is deemed required to extract all the other concepts that are structurally connected through the axioms of C—so that the complete semantics of C is still preserved in $\mathcal{O}'$. Concept axioms by which C can be structurally connected with other concepts are: *Sub-concept*, *Equivalent-class*, *Complement-of* and *Restriction*. Extraction of concepts that are structurally connected with C, through its concept axioms, is done by the following N3 rules:

Sub-concept: The sub-concepts of a C are also extracted in $\mathcal{O}'$:

`{?O sbont:extract ?C. ?C a rdfs:Class. ?S rdfs:subClassOf ?C.} =>`

`{?O sbont:extract ?S}`, where `rdfs:subClassOf a owl:TransitiveProperty.`

Equivalent-class: The equivalent concepts C are also extracted in $\mathcal{O}'$:

`{?O sbont:extract ?C. ?C owl:equivalentClass ?S.} => {?O sbont:extract ?S}`,

where `owl:equivalentClass a owl:SymmetricProperty, owl:TransitiveProperty.`

Complement-of: The concepts that are complement of C are also extracted in $\mathcal{O}'$:

```
{?O sbont:extract ?C. ?C owl:complementOf ?S.} => {?O sbont:extract ?S}.
```

Restriction: The concepts that impose restrictions on C are also extracted in $\mathcal{O}'$:

```
{?O sbont:extract ?C. ?C a rdfs:Class.  ?C rdfs:subClassOf ?R.
?R a owl:Restriction} => {?O sbont:extract ?R}.
```

4.3.6 Dealing with Property Axioms for Extracted Properties

Property axioms by which an extracted property P can be structurally connected with other properties are: *Sub-property*, *Equivalent-property* and *Inverse-property*. Similarly to the approach for concept axioms of C, extraction of properties that are structurally connected with P, through its property axioms, is done by the following N3 rules:

Sub-property: Implicitly done by the rule *Sub-concept* (see Section 4.3.2 and 4.3.5).

Equivalent-property: The equivalent properties of P are also extracted in $\mathcal{O}'$:

```
{?O sbont:extract ?P. ?P owl:equivalentProperty ?P1} =>
{?O usr:selected ?P1}.
```

Inverse-property: The inverse properties of P are also extracted in $\mathcal{O}'$:

```
{?O sbont:extract ?P. ?P owl:inverseOf ?P1} => {?O sbont:extract ?P1}.
```

4.3.7 Extracting RDF-Graphs for Extracted Concepts

For an extracted concept `?C` $\in \mathcal{C}_e$, we extract an RDF-Sub-Graph that consists of three types of axioms and assertions of `?C`: (i) concept axioms and concept assertions that need to be extracted, (ii) concept axioms that need to be replaced, and (iii) concept axioms and concept assertions that need to be removed. Although extraction of an RDF-Sub-Graph for `?C` is implemented via N3 rules, however we present an abstract implementation of the extraction method as follows:

Let: $\mathcal{C}_s$ be the set of user-selected concepts, and $\mathcal{C}_e$ be the set of extracted concepts, $\mathcal{R}_e$ be the set of extracted properties and $\mathcal{I}_e$ be the set of individuals to be extracted from an ontology $\mathcal{O}$.

Given: `?O sbont:extract ?C` (i.e. `?C` $\in \mathcal{C}_e$).

Method: Extract triples of the form:

1. `?C ?P ?Q` in sub-graph `?G1`, where `?P` $\in \mathcal{R}_e$, `?Q` $\in \mathcal{C}_e$.

2. `?i a ?C` in sub-graph `?G2`, where `?C` $\in \mathcal{C}_s$ (i.e. also user-selected) and `?i` $\in \mathcal{I}_e$.

3. `?i1 a ?C` in sub-graph `?G3`, where `?P1` $\in \mathcal{R}_e$, `?C2` $\in \mathcal{C}_e$, `?i1, ?i2` $\in \mathcal{I}_e$ such that `(?i1 a ?C)` $\wedge$ `(?i2 a ?C2)` $\wedge$ `(?i1 ?P1 ?i2)`.

4. `?i2 a ?C` in sub-graph `?G4`, where `?P1` $\in \mathcal{R}_e$, `?C1` $\in \mathcal{C}_e$, `?i1, ?i2` $\in \mathcal{I}_e$ such that `(?i1 a ?C1)` $\wedge$ `(?i2 a ?C)` $\wedge$ `(?i1 ?P1 ?i2)`.

5. `?C rdfs:subClassOf ?B` and `?B a ?Q` in sub-graph `?G5`, where `?D` $\notin \mathcal{C}_e$, `?Q` $\in \mathcal{C}$ such that `(?C rdfs:subClassOf ?D)` $\wedge$ `(?D a ?Q)` $\wedge$ `(?B e:tuple (?D))`.

6. `?C rdfs:subClassOf ?D` in sub-graph `?G6`, where `?D` $\notin \mathcal{C}_e$.

7. `?X1 usr:selected ?Y1, ?X2 sbont:extract ?Y2, ?X3 sbont:noValue ?Y3, ?X4 owl:unionOf ?Y4, ?X5 owl:intersectionOf ?Y5` in sub-graph `?G7`.

Result: Return `?GRAPH` $=$ `(?G1` $\cup$ `?G2` $\cup$ `?G3` $\cup$ `?G4` $\cup$ `?G5)` $-$ `(?G6` $\cup$ `?G7)`

The first four RDF-Sub-Graphs (`?G1`, `?G2`, `?G3`, `?G4`) consist of the concept axioms and concept assertions that need to be extracted for `?C`, where (i) `?G1` consists of the concept axioms for `?C`; (ii) `?G2` consists of all the concept assertions, if `?C` is also `usr:selected` (i.e. `?C` $\in \mathcal{C}_s$); (iii) `?G3` and `?G4` consist of only those concept assertions of `?C` that are also defined via property assertions using other extracted properties and concepts. In the Example 4.1, since `:C2` is a selected concept, therefore all the concept assertions (e.g. `:i2 a :C2.` `:i22 a :C2`) for `:C2` are also extracted in the sub-ontology (i.e. concept assertions that are collected

in ?G2). Since :C3 is a sbont:extract concept, therefore the only concept assertion extracted for :C3 is :i3 a :C3 (and not :i33 a :C3); because :i3 is defined via the already extracted property assertion :i2 :P2 :i3 (i.e. concept assertions that are collected in ?G4). The concept assertion :i33 a :C3 is not extracted in $\mathcal{O}'$, because :i33 does not appear in any of the extracted properties assertions. The RDF-Sub-Graph ?G5 consists of the concept axioms that need to be replaced. Such concept axioms are of the form ?C rdfs:subClassOf ?D such that ?C $\in \mathcal{C}_e$, ?D $\notin \mathcal{C}_e$. Based on the described motivation in Section 4.3.3, these concepts axioms are replaced by the axioms of the following form ?C rdfs:subClassOf ?B, where ?B e:tuple ?D. Axioms of the form ?C rdfs:subClassOf ?D such that ?C $\in \mathcal{C}_e$, ?D $\notin \mathcal{C}_e$ need to be removed (collected in ?G6). Some other non-related assertions are collected in ?G7. Thus, for the concept ?C, the final RDF-Graph ?GRAPH $= (?G1 \cup ?G2 \cup ?G3 \cup ?G4 \cup ?G5) - (?G6 \cup ?G7)$ is extracted in $\mathcal{O}'$.

4.3.8 Extracting RDF-Graphs for Extracted Properties

Similar to the extraction approach for concepts (see Section 4.3.7), we extract an RDF-Sub-Graph for an extracted property ?P $\in \mathcal{R}_e$ that consists of three types of axioms and assertions of ?P: (i) property axioms and property assertions that need to be extracted, (ii) property axioms that need to be replaced, and (iii) property axioms and property assertions that need to be removed. An abstract implementation for extracting an RDF-Sub-Graph for ?P is shown as follows:

Given: ?O sbont:extract ?P (i.e. ?P $\in \mathcal{R}_e$).
Method: Extract triples of the form:

1. ?P ?S ?Q in sub-graph ?G1, where ?S $\in \mathcal{R}_e$, ?Q $\in \mathcal{C}_e$.
2. ?S ?Q ?P in sub-graph ?G2, where ?S $\in \mathcal{R}_e$, ?Q $\in \mathcal{R}_e$.
3. ?i1 ?P ?i2 in sub-graph ?G3, where ?P1 $\in \mathcal{R}_e$, ?C1 $\in \mathcal{C}_e$, ?C2 $\in \mathcal{C}_e$, ?i1, ?i2 $\in \mathcal{I}_e$ such that (?i1 a ?C1) $\wedge$ (?i2 a ?C2).
4. ?P rdfs:domain ?B, ?C rdfs:subClassOf ?B and ?B a ?Q in sub-graph ?G4, where ?C $\in \mathcal{C}_e$, ?D $\notin \mathcal{C}_e$, ?Q $\in \mathcal{C}$ such that (?C rdfs:subClassOf ?D) $\wedge$ (?D a ?Q) $\wedge$ (?B e:tuple (?D)).

5. `?P rdfs:domain ?D` and `?P rdfs:range ?R` in sub-graph `?G5`, where $D, R \notin \mathcal{C}_e$.

6. `?X1 usr:selected ?Y1, ?X2 sbont:extract ?Y2, ?X3 sbont:noValue ?Y3` in sub-graph `?G6`.

Result: Return `?GRAPH` $= (\texttt{?G1} \cup \texttt{?G2} \cup \texttt{?G3} \cup \texttt{?G4}) - (\texttt{?G5} \cup \texttt{?G6})$

The first three RDF-Sub-Graphs (`?G1`, `?G2`, `?G3`) consist of the property axioms and property assertions that need to be extracted for `?P`, where (i) `?G1` and `?G2` consists of the property axioms for `?P`; and (ii) `?G3` consists of only those property assertions for `?P` that are defined between the individuals of extracted concepts. In the Example 4.1, since `:i2 :P1 :i4` is a property assertion defined between extracted concepts `:C2` and `:C4`, therefore `:i2 :P1 :i4` is extracted in the sub-ontology (i.e. property assertions that are collected in `?G3`). Whereas, `:i6 :P1 :i2` and `:i6 :P1 :i4` are not extracted; because `:i6 a :C6`, `:C6` $\notin \mathcal{C}_e$. Based on the described motivation in Section 4.3.3, given `?P rdfs:domain ?D` and `?C rdfs:subClassOf ?D` such that `?C` $\in \mathcal{C}_e$, `?D` $\notin \mathcal{C}_e$, the property axiom for the domain of `?P` is replaced by `?P rdfs:domain ?B` and `?C rdfs:subClassOf ?B`, where `?B e:tuple (?D)` (collected in `?G4`). Axioms of the form `?P rdfs:domain ?D` and `?P rdfs:range ?R` such that `?D`, `?R` $\notin \mathcal{C}_e$ need to be removed (collected in `?G5`). Some other non-related assertions are collected in `?G6`. Thus, for an extracted property `?P`, the final RDF-Graph `?GRAPH` $= (\texttt{?G1} \cup \texttt{?G2} \cup \texttt{?G3} \cup \texttt{?G4}) - (\texttt{?G5} \cup \texttt{?G6})$ is extracted in the sub-ontology $\mathcal{O}'$.

4.3.9 Extraction with Variable Boundedness

In our approach, we also allow sub-ontology extraction with *variable boundedness* [65, 66]. Based on the user-selected concepts and user-defined boundedness score $\mathcal{B}$, we extract all the structurally connected concepts that accumulate a boundedness score $b \leq \mathcal{B}$. A concept with a boundedness score $b = \mathcal{B}$ is known as *boundary concept* (similar to [65,66]). For a selected concept C with boundedness score b: (i) a sub-concept of C is assigned the same boundedness score b; (ii) a super-concept of C is assigned -1 (meaning, super-concepts need to be replaced by blank-nodes); (iii) a range-concept C' for a property P of C is assigned $b + 1$; (iv) a range-concept

C'' for a property P' of C' is assigned $b + 2$; and so on. In our extraction method, any concept C with a boundedness score $b \leq \mathcal{B}$ is treated as a `usr:selected` concept, and an extracted RDF-Graph for C is obtained using the above extraction rules. Figure 4.1 shows an extracted sub-ontology for the selected concept `C2` with a boundedness score = 1.

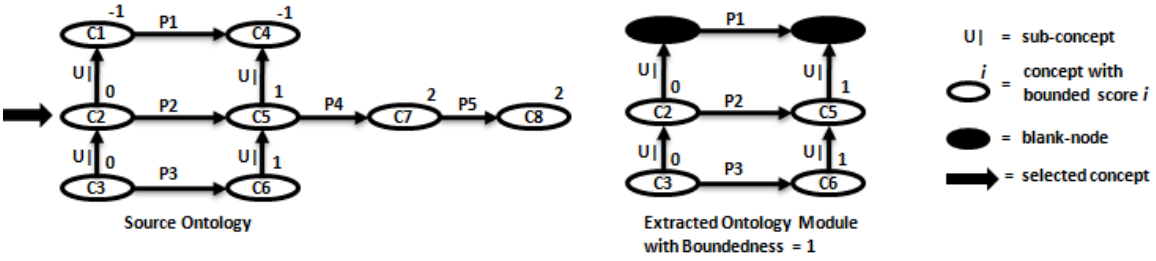

Figure 4.1: Sub-ontology Extraction with Boundedness

4.4 Evaluation

In this section, we present a theoretical comparison[1] of our extraction method with other mentioned approaches, such as MOVE [64], Seidenberg et. al. [65], Noy et. al. [66] and Miao et. al. [67]—as follows:

4.4.1 Sub-ontology Extraction by MOVE

For a given user-selected concept `:C2`, Table 4.1 shows a comparison between the sub-ontologies extracted by existing methods. All methods were applied on the source ontology $\mathcal{O}_2$ (shown in upper-left column of Table 4.1). Sub-ontology extraction using MOVE is performed via the following main criteria:

1. Due to the 1st SCOS criterion in MOVE (see Section 4.1), concepts `:C1` and `:C5` (and their properties) also got selected; because `:C1` and `:C5` are super-concepts of `:C2`.

2. Since `:C1` and `:C5` are selected, therefore by the 2nd SCOS criterion (see Section 4.1), all its sub-concepts `:C2`, `:C3`, `:C4` and `:C6` (and their properties) are also selected.

[1]Since implementations of the mentioned methods are not available, experimental comparison is not presented in this thesis.

3. Hence all the concepts and their properties in $\mathcal{O}_2$ are selected, therefore the extracted sub-ontology results in the same source ontology (shown in upper-left column of Table 4.1).

It was due to the inclusion of sibling concepts of the user-selected concepts, which causes unnecessary expansion of the extracted sub-ontology.

4.4.2 Sub-ontology Extraction by Seidenberg et. al.

In order to restrict the recursive expansion of extracted sub-ontology, Seidenberg et. al. [65] do not extract the (axioms and assertions for) siblings concepts of the user-selected concepts. Extracted sub-ontology by Seidenberg et. al. [65] is shown in the lower-left column of Table 4.1, where axioms and assertions of the sibling concept `:C6` are not extracted.

4.4.3 Sub-ontology Extraction by Noy et. al.

The extraction approach presented by Noy et. al. [66] is flexible to a number of user-defined parameters, such as (i) `direct-superclasses`, (ii) `direct-subclasses`, (iii) `every-property`, and (iv) `instances`. For the user-specifications: (i) `direct-superclasses = YES`, (ii) `direct-subclasses = YES`, (iii) `every-property = YES`, and (iv) `instances = YES`, Noy et. al. [66] extracts the same sub-ontology that was extracted by Seidenberg et. al. [65] (shown in lower-left column).

However, for a modified user-specification `direct-superclasses = NO`, the extracted sub-ontology by Noy et. al. [66] is shown in the upper-right column of Table 4.1. In this setting, Noy et. al. extraction approach is demonstrated as follows:

1. Super-concepts of user-selected concepts are not extracted in the extracted sub-ontology (shown in upper-right column).

2. For an user-selected concept `:C2`, since `:C1` (where `:C2 rdfs:subClassOf :C1`) is not extracted, the property `:P1` of `:C1` is also not extracted.

3. As a result, the properties assertions of `:P1` (such as `:i2 :P1 :i4`, `: i3 :P1 :i4`, `:i4 :P1 :i2`) are also not extracted.

Table 4.1: Comparison Between Extracted Sub-ontologies

~~Strikeout~~ Triples: Not included in the extracted sub-ontology.

Source Ontology $\mathcal{O}_2$: **:C2 is user-selected for Sub-ontology Extraction:**	**Sub-ontology Extracted by Noy et. al. [66]:** *Extract* `direct-superclasses: NO` *Extract* `direct-subclasses: YES` *Extract* `every-property: YES` *Extract* `instances: YES`
:C1 a owl:Class; rdfs:subClassOf owl:Thing. :C2 a owl:Class; rdfs:subClassOf :C1. :C3 a owl:Class; rdfs:subClassOf :C1. :C4 a owl:Class. :C4 rdfs:subClassOf :C2, :C3, :C1, :C5. :C5 a owl:Class. :C2 rdfs:subClassOf :C5. :C3 rdfs:subClassOf :C5. :C6 a owl:Class; rdfs:subClassOf :C1. :P1 a owl:ObjectProperty; rdfs:domain :C1; rdfs:range :C2. :P2 a owl:ObjectProperty; rdfs:domain :C2; rdfs:range :C3. :i1 a :C1. :i11 a :C1. :i2 a :C2. :i22 a :C2. :i3 a :C3. :i33 a :C3. :i4 a :C4. :i5 a :C5. :i6 a :C6. :i2 :P1 :i4. :i2 :P2 :i3. :i3 :P1 :i4. :i4 :P1 :i2. :i6 :P1 :i2. :i6 :P1 :i4.	~~:C1 a owl:Class; rdfs:subClassOf owl:Thing.~~ :C2 a owl:Class; ~~rdfs:subClassOf :C1.~~ :C3 a owl:Class; ~~rdfs:subClassOf :C1.~~ :C4 a owl:Class. :C4 rdfs:subClassOf :C2, :C3, ~~:C1, :C5.~~ ~~:C5 a owl:Class. :C2 rdfs:subClassOf :C5.~~ ~~:C3 rdfs:subClassOf :C5.~~ ~~:C6 a owl:Class; rdfs:subClassOf :C1.~~ ~~:P1 a owl:ObjectProperty;~~ ~~rdfs:domain :C1; rdfs:range :C2.~~ :P2 a owl:ObjectProperty; rdfs:domain :C2; rdfs:range :C3. ~~:i1 a :C1.~~ ~~:i11 a :C1.~~ :i2 a :C2. :i22 a :C2. :i3 a :C3. :i33 a :C3. :i4 a :C4. ~~:i5 a :C5.~~ ~~:i6 a :C6.~~ ~~:i2 :P1 :i4.~~ :i2 :P2 :i3. ~~:i3 :P1 :i4.~~ ~~:i4 :P1 :i2.~~ ~~:i6 :P1 :i2.~~ ~~:i6 :P1 :i4.~~
Sub-ontology Extracted by Noy et. al. [66]: *Extract* `direct-superclasses: YES` *Extract* `direct-subclasses: YES` *Extract* `every-property: YES` *Extract* `instances: YES` **Sub-ontology Extracted by Seidenberg et. al. [65]):**	**Our Extracted Sub-ontology:** *RDF Blank-Node:* `_:t8 e:tuple (:C1).` *RDF Blank-Node:* `_:t9 e:tuple (:C5).`
:C1 a owl:Class; rdfs:subClassOf owl:Thing. :C2 a owl:Class; rdfs:subClassOf :C1. :C3 a owl:Class; rdfs:subClassOf :C1. :C4 a owl:Class. :C4 rdfs:subClassOf :C2, :C3, :C1, :C5. :C5 a owl:Class. :C2 rdfs:subClassOf :C5. :C3 rdfs:subClassOf :C5. ~~:C6 a owl:Class; rdfs:subClassOf :C1.~~ :P1 a owl:ObjectProperty; rdfs:domain :C1; rdfs:range :C2. :P2 a owl:ObjectProperty; rdfs:domain :C2; rdfs:range :C3. :i1 a :C1. :i11 a :C1. :i2 a :C2. :i22 a :C2. :i3 a :C3. :i33 a :C3. :i4 a :C4. :i5 a :C5. ~~:i6 a :C6.~~ :i2 :P1 :i4. :i2 :P2 :i3. :i3 :P1 :i4. :i4 :P1 :i2. ~~:i6 :P1 :i2.~~ ~~:i6 :P1 :i4.~~	_:t8 a owl:Class; rdfs:subClassOf owl:Thing. :C2 a owl:Class; rdfs:subClassOf _:t8. :C3 a owl:Class; rdfs:subClassOf _:t8. :C4 a owl:Class. :C4 rdfs:subClassOf :C2, :C3, _:t8, _:t9. _:t9 a owl:Class. :C2 rdfs:subClassOf _:t9. :C3 rdfs:subClassOf _:t9. ~~:C6 a owl:Class; rdfs:subClassOf :C1.~~ :P1 a owl:ObjectProperty; rdfs:domain _:t8; rdfs:range :C2. :P2 a owl:ObjectProperty; rdfs:domain :C2; rdfs:range :C3. ~~:i1 a :C1.~~ ~~:i11 a :C1.~~ :i2 a :C2. :i22 a :C2. :i3 a :C3. ~~:i33 a :C3.~~ :i4 a :C4. ~~:i5 a :C5.~~ ~~:i6 a :C6.~~ :i2 :P1 :i4. :i2 :P2 :i3. :i3 :P1 :i4. :i4 :P1 :i2. ~~:i6 :P1 :i2.~~ ~~:i6 :P1 :i4.~~

4. Although, `:P1` provides additional property assertions between instances of extracted concepts `:C2, :C3, :C4`. However, the properties assertions of `:P1` are not extracted.

This shows that for `direct-superclasses = NO`, Noy et. al. [66] extracts a restricted sub-ontology, where some of the (inherited) properties (and properties assertions) of user-selected concepts are omitted from the extracted sub-ontology. The extracted sub-ontology by Miao et. al. [67] is too restricted, and consists of the single axioms `:C2 a owl:Class`.

4.4.4 Sub-ontology Extraction by Our Approach

Similar to Seidenberg et. al. [65], our extraction method also restricts the unnecessary expansion of extracted sub-ontologies towards recursive inclusion of siblings of the user-selected concepts. The extracted sub-ontology by our method is shown in the lower-right column of Table 4.1, and the extraction process is demonstrated as follows:

1. For the user-selected concept `:C2`, the sub-concept `:C4` and its properties are also selected.

2. Since `:C1` and `:C5` are the super-concepts of `:C2` and `:C4`, therefore `:C1` and `:C5` are extracted as RDF blank-nodes `_:t8` and `_:t9`, respectively.

3. However, the property `:P1` (and its property assertions) in `:C1` and the property `:P2` (and its property assertions) in `:C2` are also extracted (see Section 4.3.3).

4. Axioms and assertions of the sibling concept `:C6` (`:C6 a owl:Class, :C6 rdfs:subClassOf :C1, :i6 a :C6`) are not extracted.

5. Since `:C1` and `:C5` are not extracted in their original form, therefore their individuals (`:i1 a :C1, :i11 a :C1, :i5 a :C5`) are also not extracted.

6. In addition, any property assertions that are described through non-extracted individuals (e.g. `:i6 :P1 :i2`, `:i6 :P1 :i4`) are also not extracted in the sub-ontology.

7. Moreover, concept assertions of the extracted concept (`:i33 a :C3`) that are not described through the property assertions for the extracted concepts (`:C2`, `:C3`, `:C4`) are also not extracted.

Hence compared to MOVE, the sub-ontology extracted by our method restricts the unnecessary expansion towards sibling concepts. Furthermore, compared to Seidenberg et. al. [65] and Noy et. al. [66], we treat the super-concepts differently, and provide all the pertinent axioms and assertions specific to the selected concept `:C2`; without describing extra and irrelevant triples for `:C2`.

4.5 Experiment and Results: Extracting Contextualized Sub-ontologies for Prostate Cancer Management

For the above discussed prostate cancer (PC) scenario (see Section 3.3), we demonstrate the extraction of contextualized sub-ontologies from three location-specific PC clinical pathway ontologies in order to extract therapeutic PC work-flow knowledge for the problem-context *therapeutic decision support*. Each PC ontology stipulates its care processes based on tasks, treatments, actors and resources pertinent to local conditions. The objective of this experiment is to generate a more comprehensive PC ontology, whereby one can suggest alternative treatments or extend interventions at one location based on knowledge contained in other PC ontologies.

In order to fulfill the desired objective (i.e. *therapeutic decision support*), we demonstrate the extraction of therapeutic PC work-flow knowledge from three location-specific PC ontologies by extracting contextualized PC sub-ontologies based on the given problem-context *therapeutic decision support*.

4.5.1 Prostate Cancer Pathway Ontologies

We used three PC ontologies that describe PC clinical pathways, namely (i) PC Halifax Pathway (denoted as $\mathcal{O}_{PC-H}$); (ii) PC Calgary Pathway (denoted as $\mathcal{O}_{PC-C}$); and (iii) PC Winnipeg Pathway (denoted as $\mathcal{O}_{PC-W}$) [62]. Each PC ontology is defined using RDF(S) and OWL constructs, and deals with four major types of clinical tasks, namely (a) *Consultation Task*; (b) *Non-consulation Task*; (c) *Referral Task*;

and (d) *Followup Task* (represented as concepts). Such tasks are supported by other concepts, such as *Clinician*, *Decision Criteria*, *Frequency*, *Interval Duration*, *Investigation*, *Patient Condition Severity*, *Test Result* and *Treatment*. There are 90 concepts, 69 properties and 288 individuals in total in those three ontologies.

4.5.2 Extracting Contextualized Sub-ontologies

To establish integrated knowledge about the treatments and clinician availability from the three PC pathways, we consider a problem-context *therapeutic decision support*:

$\mathcal{C}_{x1} = \langle$ `therapeutic-decision-support,`
`{pc-calgary:Clinician, pc-halifax:Clinician, pc-winnipeg:Clinician,`
`pc-calgary:Treatment, pc-halifax:Treatment, pc-winnipeg:Treatment},`
$\emptyset$,
`{<winnipeg:Clinician, pc-halifax:Clinician, owl:equivalentClass, 1.0>,`
`<winnipeg:Treatment, pc-halifax:Treatment, owl:equivalentClass, 1.0>}`
`{{?X a pc-calgary:Clinician} => {?X a pc-halifax:Clinician}.`
`{?X a pc-calgary:Treatment} => {?X a pc-winnipeg:Followup}}`
$\rangle$

$\mathcal{C}_{x1}$ is annotated by its unique label `therapeutic-decision-support` and described by its *context-axioms*. For $\mathcal{C}_{x1}$, the list of concepts (or properties) identified by the user, from each of the three source PC ontologies, includes `Clinician` and `Treatment`. In $\mathcal{C}_{x1}$, context-axioms declare context-specific knowledge, such as (i) *Calgary-Clinicians* are to be realized as *Halifax-Clinicians*; and (ii) *Calgary-Treatments* can be viewed as *Winnipeg-Followups*. For the *therapeutic decision support* context $\mathcal{C}_{x1}$, the user is interested in such extracted sub-ontologies that describe only (i) the treatments, (ii) their durations, (iii) their follow-ups, (iv) their care-settings, and (v) the practitioners involved in them.

Based on user interest, two concepts *Treatment* and *Clinician* were labelled as `usr:selected` concepts. Given the user-selected concepts, their structurally connected concepts, properties and individuals also were extracted. $\mathcal{K}$-$\mathcal{MORPH}$ extracted a contextualized sub-ontology from each of the three PC ontologies. The

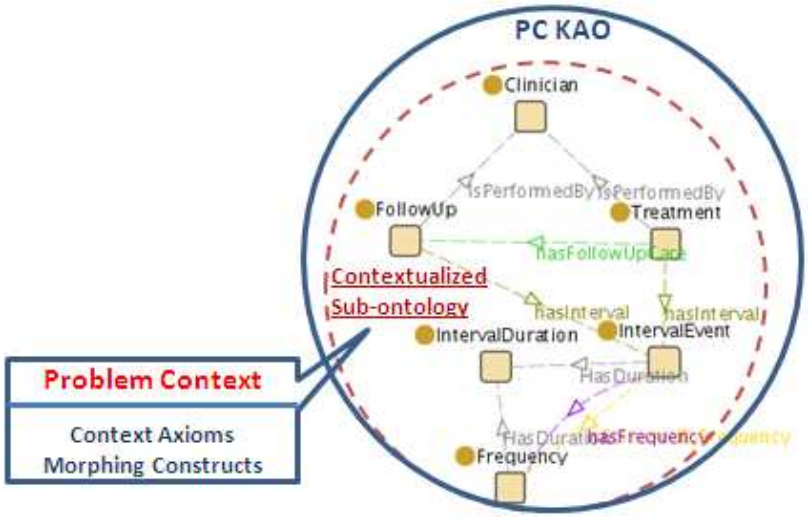

Figure 4.2: Contextualized Sub-ontology from PC Pathway Ontology

extracted sub-ontologies were validated for conceptual consistency and completeness. Concepts (including their axioms and assertions) that were extracted in the sub-ontologies are *Treatment*, *Followup*, *Frequency*, *Interval Duration*, and *Clinician*. Figure 4.2 shows a fragment of the extracted PC-Halifax sub-ontology. Concepts and properties (and their associated axioms) that have contextual relevance were extracted in the PC-Halifax sub-ontology (shown in red-dotted circle), and the context-axioms were applied on it.

4.6 Summary

Extracting sub-ontologies from source ontologies is viable for supporting specialized utilization of knowledge. We have presented our approach for sub-ontology extraction, where the concepts (and properties) that are pertinent to a target ontology are extracted, and then can either be reused or reconciled with other extracted concepts and relationships. In our sub-ontology extraction approach, we restrict the recursive selection of the super-concepts of a selected concept C in order to avoid the unnecessary selection of the siblings and super-concepts of C. In this way, we avoid a situation where the sub-ontology is too-generalized by undesired inclusion of higher levels of concepts that may even extend all the way

to `owl:Thing`. We demonstrated the application of our approach in the healthcare domain, where we extracted contextualized sub-ontologies from three different prostate cancer pathway ontologies based on the given problem-context *therapeutic-decision-support*.

Chapter 5

Aligning and Merging Contextualized Sub-ontologies

Ontologies are developed by different individuals, institutions and organizations with different intentions and origins. It is seemingly impossible to develop an ontology that can model the complete domain knowledge, and then treat it as a global and standardizes knowledge among various parties. Especially, in open or evolving systems, such as the Semantic Web, different parties adopt and develop different ontologies with unique level of domain-related specifications, expressivity, conceptual coverage and contextual determination. Thus, merely using ontologies, does not reduce heterogeneity; it rather raises heterogeneity problems to a higher level [76–84].

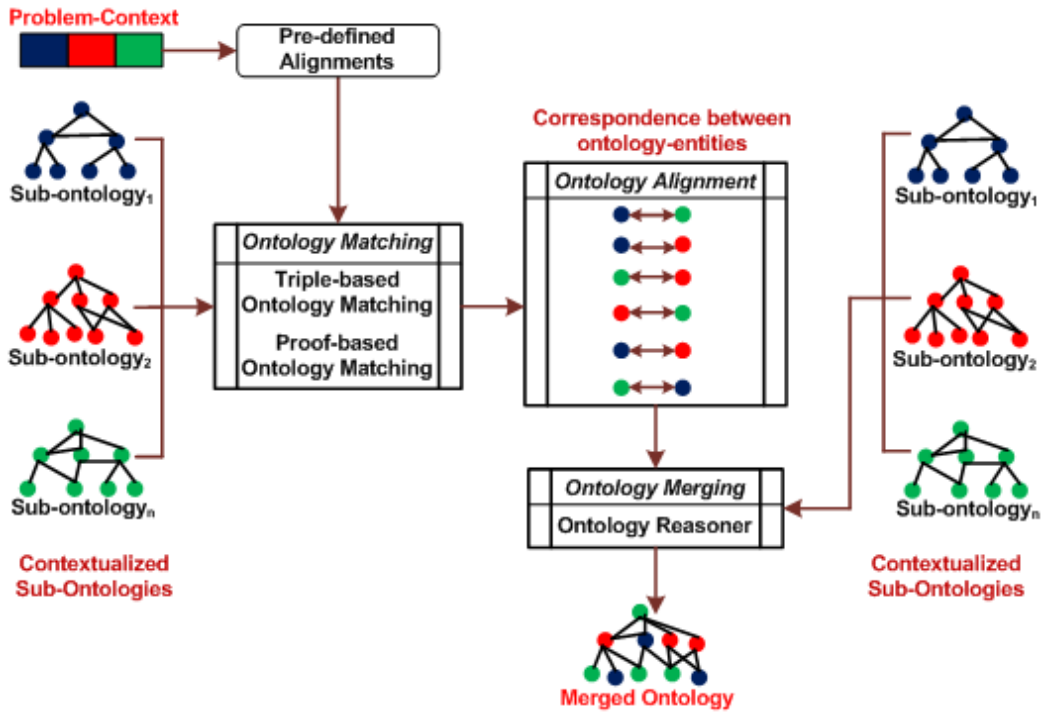

Figure 5.1: Ontology Reconciliation in $\mathcal{K}$-$\mathcal{MORPH}$: Matching, Aligning & Merging Ontology Modules

Ontology reconciliation addresses the problem of heterogeneity in ontologies [85]. It allows an interchange of knowledge that is modeled in various (domain-related) ontologies. In $\mathcal{K}$-$\mathcal{MORPH}$, ontology reconciliation among extracted sub-ontologies (see Figure 5.1) is performed by the following main tasks:

1. *Ontology Matching*: Identifying similarities between two ontology-entities from the extracted sub-ontologies.

2. *Ontology Alignment*: Identifying or defining a correspondence between two entities from the extracted sub-ontologies.

3. *Ontology Merging*: Merging the extracted sub-ontologies based on the pre-defined/found alignment.

In this chapter, we will discuss some of the existing ontology matching methods and famous alignment approaches, and also present our matching approaches for aligning and merging ontologies in $\mathcal{K}$-$\mathcal{MORPH}$.

5.1 Ontology Matching, Alignment & Merging Systems

This section presents an overview of ontology alignment and merging systems that have emerged during the last decade. There have already been some comparisons of such systems [119–126]. Our purpose here is not to compare them in full detail, but rather to show their variety, in order to demonstrate different ways of matching and aligning ontologies. We present a list of (≈ 30) ontology alignment and merging systems [41] in Table 5.1[1]. These systems can be distinguished based on their (i) input formats, (ii) matching methods, (iii) user interactions and needs, (iv) output format and (v) matching and alignment approaches. They also vary in terms of the need of any external assistance by the user/agent, or finding alignments in an automatic fashion. Most of the systems output an alignment between input source ontologies; but some of them go a step beyond, and generate merged ontologies.

[1]This table summarizes the list of matching systems discussed in Euzenat et. al. [41].

Table 5.1: List of Ontology Matching and Alignment Tools

System	Input	Matching Method(s)	Needs	Output	Approach
TranScm [127]	SGML, OO	String-based, Thesaurus, Taxonomic structure, Matching of neighbourhood	Semi	Translator	By using rules, it produces an alignment in a semi-automatic way. Then, this alignment is used to translate data instances of the source schema to instances of the target schema. Matching is performed between nodes of the graphs in a top-down and one-to-one fashion. Matchers are viewed as rules. For example, according to the identical rule, two nodes match if their labels are found to be synonyms based on the built-in thesaurus.
DIKE [128]	ER	String-based, WordNet, Domain compatibility,	Semi	Merge	DIKE can find matching based on (i) terminological properties (such as synonyms, homonyms among objects, namely entities and relationships); (ii) structural properties (such as object inclusion); (iii) subschema similarities (such as similarities between schema fragments). It works by computing the above-mentioned properties in a sequential order.
SKAT & ONION [129]	RDF	String-based, Corpus-based, Taxonomic structure	Semi	Bridge rules	It is a rule-based system that semi-automatically discovers mappings between two ontologies. Rules are provided by domain experts, and are encoded in First-order Logic. In particular, experts specify initially desired matches and mismatches. Experts are required to approve or reject the automatically suggested matches, thereby producing the resulting alignment.
Artemis [130]	Relational schema, OO, ER	Language-based, Domain compatibility, Common thesaurus, Matching of neighbours, Clustering	Auto	Views	It performs affinity-based analysis and hierarchical clustering of database schema elements. Affinity-based analysis represents the matching step in a sequential manner, which calculates the name, structural and global affinity coefficients exploiting a common thesaurus. Based on global affinity coefficients, a hierarchical clustering technique categorizes classes into groups at different levels of affinity. For each cluster it creates a set of global attributes and the global class. Logical correspondence between the attributes of a global class and source schema attributes is determined through a mapping table.
H-Match [131]	OWL	Language-based, Domain compatibility, Thesaurus, Matching of neighbours, Domains & ranges Relations	Auto	Alignment	The approach is based on a similarity analysis through affinity metrics, e.g., term to term affinity, datatype compatibility, and thresholds. H-Match computes two types of affinities, namely linguistic and contextual affinity. These are then combined by using weighting schemas, thus yielding a final measure, called semantic affinity. Linguistic affinity builds on top of a thesaurus-based approach of the Artemis system [130]. Contextual affinity requires consideration of the neighbour concepts, e.g., linked via taxonomical or mereological relations, of the actual concept. By applying thresholds, correspondences with semantic (final) affinity higher than the cut-off threshold value are returned in the final alignment.

Table 5.1: List of Ontology Matching and Alignment Tools (continued)

System	Input	Matching Method(s)	Needs	Output	Approach
Anchor-Prompt [132]	OWL, RDF	String-based, Domains and ranges, Bounded paths matching, arbitrary links, Taxonomic structure	User	Axioms (OWL/ RDF)	It is a sequential matching algorithm that takes as input two ontologies, internally represented as graphs and a set of anchors-pairs of related terms, which are identified with the help of string-based techniques, such as edit-distance, user-defined distance or another matcher computing linguistic similarity. Then the algorithm refines them by analyzing the paths of the input ontologies limited by the anchors; in order to determine terms frequently appearing in similar positions on similar paths. Finally, based on the frequencies and user feedback, the algorithm determines matching candidates.
OntoBuilder [133]	Web form, XML schema	String-based, Language-based, Thesaurus look up	User	Mediator	The matching algorithm works on terms; sequentially executing various matchers. Some examples of the matchers used are: removing noisy characters and stop terms, substring matching. If all else fail, thesaurus look-up is performed. Finally, mismatched terms are presented to users for manual matching.
Cupid [134]	XML schema, Relational schema	String-based, Language-based, Datatypes, Key properties, Auxiliary thesauri, Tree matching weighted by leaves	Auto	Alignment	The matching algorithm consists of three phases. The first phase (linguistic matching) computes linguistic similarity coefficients between schema element names (labels) based on morphological normalization, categorization, string-based techniques, such as common prefix, suffix tests, and thesauri look-up. The second phase (structural matching) computes structural similarity coefficients weighted by leaves which measure the similarity between contexts in which elementary schema elements occur. The third phase (mapping elements generation) aggregates the results of the linguistic and structural matching through a weighted sum and generates a final alignment, which are higher than a threshold.
COMA & COMA++ [135]	Relational schema, XML schema, OWL	String-based, Language-based, Datatypes, Thesauri, Alignment reuse, Repository of structures, DAG matching	User	Alignment	COMA contains 6 elementary matchers, 5 hybrid matchers, and one reuse-oriented matcher. Most of them implement string-based techniques, such as affix, n-gram, edit distance; others share techniques with Cupid [134], e.g., thesauri look-up. An original component, called *reuse-oriented* matcher, tries to reuse previously obtained results for entire new schemas or for their fragments. Distinct features of the COMA tool with respect to Cupid are (a) a more flexible architecture, and (b) the possibility of performing iterations in the matching process. COMA++ is built on top of COMA by elaborating in more detail the alignment reuse operation. Also it provides a more efficient implementation of the COMA algorithms and a graphical user interface.
Similarity flooding [136]	XML schema, Relational schema	String-based, Datatypes, Key properties, Iterative fixed point computation	User	Alignment	This approach is based on the ideas of similarity propagation. The technique starts from string-based comparison, such as common prefix, suffix tests, of the vertices labels to obtain an initial alignment which is refined through iterative computation. From iteration to iteration, the similarity measure is spread to the (adjacent/neighbouring nodes of) graph until a fixed point is reached or the computation is stopped.

Table 5.1: List of Ontology Matching and Alignment Tools (continued)

System	Input	Matching Method(s)	Needs	Output	Approach
MapOnto [137]	Relational schema, XML schema, OWL	External alignments, Structure comparison	Alignment	Rules	The system produces (in a semi-automatic way) a set of complex mapping formulae expressed in a subset of First-order Logic (Horn clauses). The list of logical formula's is also ordered by the tool, thereby suggesting the most reasonable mappings.
OntoMerge [138]	OWL	External alignments,	Alignment	Ontology	It performs ontology translation by ontology merging and automated reasoning. Merging two ontologies is performed by taking the union of the axioms defining them. Bridge axioms or bridge rules are then added to relate the terms in one ontology to the terms in the other. Once the merged ontology is constructed, the ontology translation tasks can be performed fully automatically by mechanized reasoning.
CtxMatch/ Ctx-Match2 [139]	Classification, OWL	String-based, Language-based, WordNet, Description Logics	User	Alignment	CtxMatch translates the ontology matching problem into the logical validity problem and computes logical relations, such as equivalence, subsumption between concepts and properties. CtxMatch2 improves on CtxMatch by handling properties. At the structure level, it exploits description logic reasoners, such as Pellet [140] or FaCT [141] to compute the final alignment
S-Match [142]	Classification, XML Schema, OWL	String-based, Language-based, WordNet, Propositional SAT	Auto	Alignment	It takes as input two graph-like structures, e.g., classifications, XML schemas, ontologies, and returns as output logic relations, e.g., equivalence, subsumption, which are supposed to hold between the nodes of the graphs. The relations are determined by (i) expressing the entities of the ontologies as logical formulae, and (ii) reducing the matching problem to a propositional validity problem.
HCONE [143]	OWL	Language-based (LSI) WordNet	Auto, Semi, User	Ontology	It is an approach for domain ontology matching and merging by exploiting different levels of interaction with users. First, an alignment between two input ontologies is computed with the help of WordNet. Then, the alignment is processed straightforwardly by using some merging rules, to generate a new merged ontology.
MoA [144]	OWL	Language-based WordNet	Auto	Axiom OWL	The matching approach is based on concept (dis)similarity derived from linguistic clues. In this approach, (i) names of classes and properties are tokenized; (ii) tokens of entities are associated with their meaning by using WordNet senses; (iii) meanings of tokens of ancestors of the entity under consideration are also taken into account, thereby extending the local meanings.
ASCO [145]	RDF(S), OWL	String-based, Language-based, WordNet Iterative similarity propagation	Auto	Alignment	The matching is organized sequentially in three phases. During the first phase (linguistic matching) the system normalizes terms and expressions, e.g., by punctuation, upper cases, special symbols. The second phase (structure matching), computes similarities between classes and relations by propagating the input of linguistic similarities. In the third phase, the linguistic and structural similarity are aggregated through a weighted sum. If the similarities between matching candidates exceed a threshold, they are selected for the resulting alignment.
BayesOWL [146]	Classification, OWL	Text classifier, Google thesauri, Bayesian inference	Auto	Alignment	First, two input ontologies are translated into two Bayesian networks. Matching candidates are generated between two Bayesian networks, by learning joint probabilities from the web data. A similarity between two concepts is determined with the help of the Jaccard coefficient computed from the joint probabilities. These are used to construct the conditional probability tables. By performing Bayesian inference, the final alignment is produced.

Table 5.1: List of Ontology Matching and Alignment Tools (continued)

System	Input	Matching Method(s)	Needs	Output	Approach
OMEN [147]	OWL	External alignment, Bayesian inference, Meta-rules	Auto, Alignment	Alignment	The approach can be summarized in four logical steps. First, it creates a Bayesian network, where a node stands for a mapping between pairs of classes or properties of the input ontologies. During the second step, OMEN uses a set of meta-rules that capture the influence of the structure of input ontologies in the neighbourhood of the input mappings in order to generate conditional probability tables for the given network. During the third step, inferences are made by Bayesian Network tools in Java (BNJ) to generate a posteriori probabilities for each node. Finally, a posteriori probabilities, which are higher than a threshold, are selected to generate the resulting alignment.
DCM [148]	Web form	Correlation mining, Statistics	Auto	Alignment	Schema matching is viewed as *correlation mining* problem, where co-occurrence patterns often suggest complex matches. Technically, this means that those attributes are positively correlated. Contrary, attribute names which are synonyms, e.g., quantity and amount, rarely co-occur, thus representing an example of negative correlation between them.
T-tree [149]	Ontology	Correlation mining	Auto, Instances	Alignment	It infers dependencies between classes, called *bridges*, of different ontologies sharing the same set of instances based only on the extension of classes.
CAIMAN [150]	Classification	String-based (Rocchio classifier)	Semi	Alignment	It calculates a probability measure between the concepts of two ontologies, by applying machine learning techniques for text classification, e.g., the Rocchio classifier. In particular, based on the documents, a representative feature vector is created for each concept in an ontology. Then, the cosine measure is computed for two of those class vectors. Finally, with the help of a threshold, the resulting alignment is produced.
FCA-merge [151]	Ontology	Formal concept analysis	User	Ontology	The overall process of merging two ontologies consists of three steps, namely (i) instance extraction, (ii) concept lattice computation, (iii) interactive generation of the final merged ontology.
LSD/ GLUE [152]/ iMAP [153]	Relational schema, XML schema, Taxonomy	WHIRL, Naive Bayes, Domain constraints, Hierarchical structure	Auto	Alignment	LSD approach works in two phases. During the first (training) phase, using useful objects and manually created alignments between them, the system trains multiple basic matchers. During the second (matching) phase, by applying the trained basic matchers and the meta-matcher on the new objects (the classification operation), LSD obtains a prediction list of matching candidates. Finally, by taking into account integrity constraints and applying some thresholds, the final alignment is extracted. GLUE, a successor of LSD, follows a multi-strategy learning approach, involving several basic matchers and a meta-matcher. iMAP uses several matchers, called *searchers*, are run in parallel. They provide candidate matches that can be complex. These candidates are further selected by applying the similarity estimator, and then, the final alignment is extracted.
Automatch [154]	Relational schema	Naive Bayes, Internal structure Statistics	User	Alignment	Automatch acquires probabilistic knowledge from the manually matched schemas, and creates the *attribute dictionary*, which accumulates the knowledge about each attribute by means of its possible values and the probability estimates of these values. The system first matches each attribute of the input schemas against the attribute dictionary, thereby producing individual match scores. Then, these individual scores are further combined by taking their sum to produce the scores between the attributes of the input schemas.
Wang & al. [155]	Web form	Language-based, Mutual information	Instances	Alignment	The approach works in two phases: (i) query probing and (ii) instance-based matching.

Table 5.1: List of Ontology Matching and Alignment Tools (continued)

System	Input	Matching Method(s)	Needs	Output	Approach
sPLMap [156]	Database schema	Naive Bayes, kNN classifier, String-based	Auto	Rules	The system operates in three main steps. First, it selects the set of correspondences that maximizes probability on the basis of instance data. Second, for each correspondence, matchers are used as quality estimators: they provide a measure of the plausibility of the correspondence. Finally, the computed probabilities are transformed in correspondence weights by using the Bayes theorem.
SEMINT [157]	Relational schema	Neural network, Datatypes, Value patterns	Auto	Alignment	The approach works on four main tasks. First, it extracts all the necessary information useful for matching. Second, by using a neural network as a classifier with the self-organizing map algorithm, it groups the attributes based on similarity of the features for the first database. Third, it uses a back-propagation neural network for learning and recognition. Finally, using a trained neural network on the first database features and clusters, the system recognizes and computes similarities between the categories and between the attributes among two databases, and generates a list of match candidates.
Clio [158]	Relational schema	String-based, Language-based, Naive Bayes Structure comparison	Semi	Query	It combines, in a sequential manner, instance-based attribute classification via a variation of a Naive Bayes classifier and string matching between elements names, e.g., by using an edit distance. Then, taking the n-m value correspondences (the alignment) together with constraints coming from the input schemas, Clio compiles these into an internal query graph representation.
NOM [159] & QOM [160]	RDF, OWL	String-based, Domains and ranges, Matching of neighbours, Taxonomic structure	Auto	Alignment	In NOM, the similarity measures produced by basic matchers are refined by using a sigmoid function—thereby emphasizing high individual similarities and de-emphasizing low individual similarities. They are then aggregated through weighted average. Finally, with the help of thresholds, the final alignment is produced. QOM is a variation of the NOM system dedicated to improve the efficiency of the system. It avoids the complete pairwise comparison of trees in favour of an incomplete top-down strategy, thereby focusing only on promising matching candidates.
oMap [161]	OWL	Naive Bayes, String-based, Similarity propagation	Auto	Alignment	It uses several matchers, which includes: (i) a classifier based on classic string similarity measure over normalized entity names; (ii) a Naive Bayes classifier used on instance data, and (iii) a semantic matcher which propagates initial weights through the ontology constructors used in the definitions of ontology-entities.
Xu & al. [162]	XML schema, Taxonomy	String-based, Language-based, WordNet, Domain ontology, Decision trees	Auto	Alignment	It is performed by a combination of multiple matchers and with the help of external knowledge resources, such as domain ontologies. The basic element level matchers used in the approach include *name matcher* and *value-characteristic matcher*. Structure level matchers are used to suggest new correspondences as well as to confirm correspondences identified by element level matchers.
OLA [163]	RDF, OWL	String-based, Language-based, Datatypes, WordNet, Fixed point computation, Matching of neighbours, Taxonomic structure	Auto	Alignment	It computes similarity between entities based on two principles: (i) it depends on the category of entity considered, e.g., class, property, and (ii) it takes into account all the features of this category, e.g., superclasses, properties. The similarity distance between entities are expressed as a system of equations based on string-based, language-based and structure-based similarities. For computing these distances, the algorithm starts with base distance measures computed from labels and concrete datatypes. Then, it iterates a fixed point algorithm until no improvement is produced. From that solution, an alignment is generated which satisfies some additional criterion on the obtained alignment and the distance between matched entities.

Table 5.1: List of Ontology Matching and Alignment Tools (continued)

System	Input	Matching Method(s)	Needs	Output	Approach
IF-Map [164]	XML schema	String-based, Formal concept analysis	Instances	Transformation	It matches two local ontologies by looking at how these are related to a common reference ontology. It generates candidate pairs of mappings (called *infomorphism* in information flow theory) and artificial instances by heuristics based on string-based and structure-based methods.
Falcon-AO [165]	RDF, OWL	String-based, WordNet, Structural affinity	Auto	Mediator	Matching is done via two components, namely, (i) LMO: linguistic matching, and (ii) GMO: structure matching. First, LMO is used for assessing the similarity between ontology-entities on the basis of their name and text annotations. If the result has a high confidence, then it is directly returned for extracting an alignment. Otherwise, the result is used as input for the GMO matcher which tries to find an alignment on the basis of the relationships between entities.
RiMOM [166]	OWL	String-based, WordNet, Taxonomic structure, Similarity propagation	Auto	Alignment	Its matching process is organized into 6 tasks: (i) select the matcher to be used, (ii) execute multiple independent matchers, (iii) combine the results by aggregating the similarities evaluated by individual matchers, (iv) similarity propagation, (v) extract alignment for a pair of ontologies based on thresholds, (vi) iterate the above tasks with extracted alignments, until no new correspondences are produced.
Detecting Complex Correspondences [176]	RDF, OWL	Linguistic analysis, Linguistic matching conditions	Alignment	Alignment	Ritze et. al. [176] proposed an approach for finding complex correspondences in absence of matchable instances, and proposed further improvements by employing *linguistic analysis* in their matching process to produce an enriched set of correspondence patterns leveraging linguistic matching conditions [190].
TOM	RDF, OWL	Structure-based similarity, Triple-set similarity	Alignment	Alignment	Triple-base Ontology Matching (TOM) (see Section 5.4) finds structural similarities between two entities e_1 and e_2 based on the similarities between the set of triples that describe e_1 and the set of triples that describe e_2. TOM requires source ontologies and some initial alignments between those ontologies to produce new alignments between them.
POM	RDF, OWL	Proof-based similarity, Ancestor-set similarity	Alignment	Alignment	Proof-based Ontology Matching (POM) (see Section 5.5) finds alignments not only based on structural similarities but also takes into account (similarities between) other deductively connected complex entity-structures—under the rules in $\mathcal{P}$ describing both domain-specific and ontology-language semantics. POM requires (i) source ontologies (ii) individuals from those ontologies, (iii) some initial alignments between those ontologies, and (iv) a logic program $\mathcal{P}$ to produce new alignments. However compared to TOM, POM is designed to find more complex and non-trivial alignments between source ontologies.

The series of alignment approaches considered in this section has utilized a diversity of lexical and structure-based techniques [41, 42]. Usually, each individual system employs a particular strategy. However, there are several common features that are shared by the majority of systems. Here, we present our general observation about such features based on the evaluation carried in Euzenat et. al. [41]:

1. Schema-based matching solutions have been so far investigated more intensively than the instance-based solutions.

2. Most of the systems focus on specific application domains, such as books and music, and are limited to deal with particular ontology types, such as DTDs, relational schemas and OWL ontologies. Only few systems aim to be general in (a) suiting with various application domains, and (b) dealing with multiple types of ontologies. Some examples of the latter include: Cupid, COMA and COMA++, Similarity flooding and S-Match.

3. Most of the approaches take as input a pair of ontologies, while only few systems take as input multiple ontologies. Some examples of the latter include: DCM and Wise-Integrator.

4. Most of the approaches handle only tree-like structures, while only few systems handle graphs. Some examples of the latter include: Cupid, COMA and COMA++, and OLA.

5. Most of the systems focus on discovery of one-to-one alignments, while only few systems have tried to address the problem of discovering more complex correspondences, such as one-to-many and many-to-many, e.g., iMAP, DCM, Ritze et. al. [176, 190], TOM (see Section 5.4) and POM (see Section 5.5).

6. Most of the systems focus on computing confidence measures in the [0 1] range, most often standing for the fact that the equivalence relation holds between ontology-entities. Only few systems compute logical relations between ontology-entities, such as equivalence, subsumption, etc. Some examples of the latter include: CtxMatch and S-Match.

5.2 Matching Complex Entities

Matching and alignment of ontologies have been carried out based on their lexical, conceptual and structural similarities [73]. When dealing with structural similarities, similarity scores between ontology-entities can be further improved based on the similarities between their structurally connected entities [73]. We believe that alignments between two entities e_1 and e_2 can become more 'trustworthy' by finding similarities in their justifications under a logic program $\mathcal{P}$—in which both ontology-language and domain-specific rules are defined.

In our knowledge morphing framework $\mathcal{K}$-$\mathcal{MORPH}$, in order to align and merge the extracted contextualized sub-ontologies, we have developed two ontology matching approaches—that aim to find both structure-based and proof-based similarities between source ontologies—are listed as follows:

1. **Triple-based Ontology Matching (TOM):** TOM finds structural similarities between two entities e_1 and e_2 based on the similarities between the set of triples that describe e_1 and the set of triples that describe e_2. TOM requires source ontologies and some initial alignments between those ontologies to produce new alignments between them.

2. **Proof-based Ontology Matching (POM):** POM finds alignments not only based on structural similarities but also takes into account (similarities between) other deductively connected complex entity-structures—under the rules in $\mathcal{P}$ describing both domain-specific and ontology-language semantics. POM requires (i) source ontologies (ii) individuals from those ontologies, (iii) some initial alignments between those ontologies, and (iv) a logic program $\mathcal{P}$ to produce new alignments. However compared to TOM, POM is designed to find more complex and non-trivial alignments between source ontologies.

In $\mathcal{K}$-$\mathcal{MORPH}$, we apply both of our matchers, POM and TOM, for (i) finding alignments between the extracted contextualized sub-ontologies; and (ii) based on found alignments, merge these sub-ontologies. Our matchers find new alignments

between contextualized sub-ontologies, based on the set of pre-defined context-specific alignments in the given problem-context. Based on both pre-defined context-specific alignments and new alignments found by our matchers, we merge the extracted sub-ontologies and generate a merged ontology (see Figure 5.1).

We will demonstrate the working of POM and TOM on data-sets of Ontology Alignment Evaluation Initiative (OAEI) [167] and compare their results with existing ontology matching systems. Furthermore, we will show the significance of our matchers in aligning and merging the contextualized sub-ontologies for the problem-context *therapeutic decision support*.

5.2.1 Matching Complex Entities: State-of-the-Art

Matching complex entities is a well known problem in database schema matching [168], where complex attributes are first identified and then matched, e.g. a name is equivalent with concatenation of a first-name and a last-name. On the other hand, matching complex entities is one of the crucial challenges in the Ontology Matching field [73]. Most of the state-of-the-art matchers just find (simple) correspondences between two atomic entities (see Section 5.1). However, pragmatic issues demand for complex matching. Hence, only the simple correspondences between atomic entities are too limited to capture all meaningful relations between concepts and properties of two related ontologies.

There are three diverse aspects of complex correspondences: designing (or defining), finding and representing them. In Scharffe et. al. [169], complex correspondences are analyzed from the design and representation aspects. It provides a representation format to define complex correspondences as *correspondence patterns*. Such complex patterns can be defined by domain experts, who can take advantage of diverse templates for capturing complex and correct matchings. Moreover, as suggested and shown in Scharffe et. al. [169], complex correspondences can also be exploited by an automated matching approach.

An initial attempt on finding complex correspondences is reported in Zamazal et. al. [170] using pattern-based detection of different semantic structures in ontologies. The most refined pattern is concerned with 'N-ary' relation detection. Once such a pattern-based semantic structure is detected (using query language

and some string-based similarity methods), additional conditions (such as string-based comparisons) over the detected entities yields correspondences between complex entities. The authors showed some experiments with pattern detection, but experiments to demonstrate matching tasks were missing.

Another pattern-based approach is reported in Zamazal et. al. [171], where it enables finding originally missed correspondences. First, ontologies are transformed according to transformation patterns, and then any matcher can be applied to find (complex) correspondences. Authors also emphasized that matchers can work with certain structures better than with others. This approach uses the *expressive alignment language* [172], which extends the original INRIA alignment format [173]. This language allows users to express a correspondence between two complex entities/structures (such as set operators, restrictions applied to entities and relations).

In addition, there have been some attempts towards finding complex correspondences using machine learning approaches [174]. Although such systems can detect correspondences between two complex entities, but these systems also require the ontologies to include matchable instances. However, ontologies often contain disjoint sets of instances, and it is also not trivial to find matchable instances [175]. This issue is addressed in Ritze et. al. [176], where the authors proposed an approach for finding complex correspondences in absence of matchable instances [175]. This approach also takes reference alignments (if available) into account for detecting complex correspondences based on the given reference alignment composed of simple correspondences.

In $\mathcal{K}$-$\mathcal{MORPH}$, we have proposed our approach for finding both atomic and complex correspondences, and developed two matchers, TOM and POM, that detects correspondences between two atomic/complex entities. Such entities can be of the following forms:

1. *Complex Concept*: A complex concept can be defined as a union/intersection of other (complex) concepts. A complex correspondence can be detected either between two complex concepts, or between a complex and an atomic concept. An example correspondence is listed below:

```
[ a :Cell;
  :entity1 {o1:Writer a rdfs:Class};
  :entity2 {[ a rdfs:Class; owl:unionOf (o2:Author o2:Person)]};
  :measure 0.663913043478261;
  :relation rdfs:subClassOf].
```

The above correspondence represents a subsumption relation (i.e. `rdfs:subClassOf`) between a atomic concept `o1:Writer` and a complex concept defined as a union of two concepts `o2:Author` and `o2:Person`.

2. *Property-restricted Concept*: A property-restricted concept is a complex concept, where a property-specific constraint is defined on the concept. Such constraints are of two types: `owl:allValuesFrom` and `owl:someValuesFrom`. The property constraint `owl:allValuesFrom` on a property P specifies that all the property-values of P must belong to a certain concept. Whereas, `owl:someValuesFrom` ensures that there exists at least one property-value of P that belongs to a certain concept. A property-restricted concept C is a collection of all such instances that satisfy all the property restrictions on C. An example correspondence dealing with a property-restricted concept is listed below:

```
[ a :Cell;
  :entity1 {o1:ConferenceChair a rdfs:Class};
   :entity2 {[ a rdfs:Class; a owl:Restriction;
               owl:onProperty o2:was_a_member_of;
               owl:someValuesFrom o2:Committee]};
  :measure 0.233319397993311;
  :relation rdfs:subClassOf].
```

The above correspondence represents a subsumption relation (i.e. `rdfs:subClassOf`) between `o1:ConferenceChair` and a property-restricted concept `[ owl:onProperty o2:was_a_member_of; owl:someValuesFrom o2:Committee ...]`.

3. *Cardinality-restricted Concept*: A cardinality-restricted concept is a complex concept C, where a property of C is constraint by fixed (interval of) cardinality values. Cardinality value constraints can be defined on properties using two OWL constructs: `owl:minCardinality` and `owl:maxCardinality`. The property constraint `owl:minCardinality` on a property P specifies the minimal cardinality for the property-values of P. Whereas, `owl:maxCardinality` specifies the maximal cardinality for the property-values of P. A cardinality-restricted concept C is a collection of all such instances that satisfy all the cardinality restrictions on the properties of C. An example correspondence dealing with a cardinality-restricted concept is listed below:

```
[ a :Cell;
  :entity1 {o1:Paper a rdfs:Class};
  :entity2 {[ a rdfs:Class; a owl:Restriction;
              owl:onProperty o2:readByReviewer;
              owl:minCardinality "1"^^xsd:int] };
  :measure 0.254826254826255;
  :relation rdfs:subClassOf].
```

   The above correspondence represents a subsumption relation (i.e. `rdfs:subClassOf`) between `o1:Paper` and a cardinality-restricted concept `[ owl:onProperty o2:readByReviewer; owl:minCardinality "1"^^xsd:int ...]`. In this way, there is a correspondence between the concept `o1:Paper` and a collection of literature that is reviewed by at least one reviewer.

Compared to Ritze et. al. [176], we also take the reference alignments into an account, and find complex correspondences based on the given simple correspondences. Both TOM and POM can work without matchable instances, however matchable instances can improve their matching process.

5.3 Preliminaries

For our purpose, we consider RDF/OWL ontologies that are defined based on a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, comprised of concepts $\mathcal{C}$, properties $\mathcal{R}$,

individuals $\mathcal{I}$, literals $\mathcal{L}$ and RDF/OWL constructs representing meta-classes $\mathcal{M}_c$ and meta-properties $\mathcal{M}_p$. An RDF/OWL ontology $\mathcal{O}$ can be expressed as triples of the form $\langle s, p, o\rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c)$. In a triple $\langle s, p, o\rangle$, s is called subject, p predicate, and o object. Triples allow to define *terminology* and *assertional* axioms in $\mathcal{O}$ [89]. Terminology axioms (concept axioms and property axioms) $\mathcal{T}$ are of the form $C \sqsubseteq D$ ($R \sqsubseteq S$) or $C \equiv D$ ($R \equiv S$) such that $C, D \in \mathcal{C}$ and $R, S \in \mathcal{R}$. Assertional axioms (concept assertions and property assertions) $\mathcal{A}$ are of the form $C(a)$ or $R(b, c)$ such that $C \in \mathcal{C}$, $R \in \mathcal{R}$, $a, b, c \in \mathcal{I}$. The set of matchable entities $M_\mathcal{O}$ of $\mathcal{O}$ is defined as $M_\mathcal{O} = \mathcal{C} \cup \mathcal{R} \cup \mathcal{I}$. The semantics of an ontology is defined by an *interpretation* that provides mapping from (i) ontology individuals, (ii) ontology concepts and (iii) ontology properties to (a) elements of the domain, (b) collections of the domain-elements and (c) binary relations between the domain-elements, respectively. A *model* of an ontology is such an interpretation, under which all ontology-axioms are satisfied. An ontology is called *consistent*, iff there exists a model for it. An ontology that has no model is called an *inconsistent ontology* [116]. The set of ontologies is denoted by $\mathbb{O}$. $\mathbb{T}_\mathcal{P}(\mathcal{O})$ is the set of all asserted and inferred ontology triples of $\mathcal{O}$ under a given logic program $\mathcal{P}$ (see Definition 1).

Definition 11 (Correspondence) *Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ together with their set of matchable entities $M_{\mathcal{O}_1}$ and $M_{\mathcal{O}_2}$ respectively, and a confidence interval $[0, 1]$, a correspondence $\Phi = \langle e, e', r, n\rangle$ states that the two matchable entities $e \in M_{\mathcal{O}_1}$ and $e' \in M_{\mathcal{O}_2}$ are related via a relation r with confidence $n \in [0, 1]$.*

Definition 12 (Alignment) *Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ together with their set of matchable entities $M_{\mathcal{O}_1}$ and $M_{\mathcal{O}_2}$ respectively, an alignment $\mathbb{A}$ is a set of correspondences between pairs of entities belonging to $M_{\mathcal{O}_1}$ and $M_{\mathcal{O}_2}$.*

Definition 13 (Concept Triples) *For a concept $C \in \mathcal{C}$ in an ontology $\mathcal{O}$, the concept triples of C, denoted by $\mathbb{T}_C$, is a set of ontology triples of the form $\langle C, p, o\rangle$ or $\langle s, p, C\rangle$.*

Definition 14 (Property Triples) *For a property $R \in \mathcal{R}$ in an ontology $\mathcal{O}$, the property triples of R, denoted by $\mathbb{T}_R$, is a set of ontology triples of the form $\langle s, R, o\rangle$, $\langle R, p, o\rangle$ or $\langle s, p, R\rangle$.*

Definition 15 (Individual Triples) *For an individual $I \in \mathcal{I}$ in an ontology $\mathcal{O}$, the individual triples of I, denoted by $\mathbb{T}_I$, is a set of ontology triples of the form $\langle I, p, o\rangle$ or $\langle s, p, I\rangle$.*

Definition 16 (Triple Ancestors) *Given an ontology triple $t \in \mathbb{T}_{\mathcal{P}}(\mathcal{O})$, triple ancestors of t, denoted by $\mathcal{A}nces_t$, is the set of all the triples corresponding to the internal nodes and the leaves of the proof-tree $\mathcal{TR}_t$ (i.e. the set of all the ontology triples from which t is derived under P). For an asserted ontology triple t', $\mathcal{A}nces_{t'} = \emptyset$.*

Definition 17 (Concept, Property and Individual Triples Ancestors) *Given concept triples $\mathbb{T}_C$ for a concept $C \in \mathcal{C}$, property triples $\mathbb{T}_P$ for a property $P \in \mathcal{R}$, and individual triples $\mathbb{T}_I$ for an individual $I \in \mathcal{I}$,*

concept triples ancestors $\mathcal{A}nces_C$ for C is defined as $\mathcal{A}nces_C = \bigcup_{t \in \mathbb{T}_C} \mathcal{A}nces_t$

property triples ancestors $\mathcal{A}nces_P$ for P is defined as $\mathcal{A}nces_P = \bigcup_{t \in \mathbb{T}_P} \mathcal{A}nces_t$

individual triples ancestors $\mathcal{A}nces_I$ for I is defined as $\mathcal{A}nces_I = \bigcup_{t \in \mathbb{T}_I} \mathcal{A}nces_t$

5.4 Ontology Alignment using Triple-based Ontology Matching (TOM)

We propose a structure-based ontology matching approach *triple-based ontology matching* (TOM). TOM also tries to maximize the similarity between ontology-entities based on the similarities between their structurally connected entities. TOM requires two source ontologies $\mathcal{O}, \mathcal{O}'$, together with their set of matchable entities $M_{\mathcal{O}}$ and $M_{\mathcal{O}'}$ as inputs. TOM also requires some initial correspondences of the form $\Phi_{ij} = \langle e_i, e_j, r, n\rangle$. In TOM, for any two entities $e_k \in M_{\mathcal{O}}$ and $e_l \in M_{\mathcal{O}'}$, we find a structure-based similarity between e_k and e_l by first collecting all triples of e_k (denoted by $\mathbb{T}_{e_k}$) and all triples of e_l (denoted by $\mathbb{T}_{e_l}$), and then we calculate a similarity score between e_k and e_l based on the similarities between the entities $e_i \in M_{\mathcal{O}}$ and $e_j \in M_{\mathcal{O}'}$ that are used to describe axioms and assertions of e_k and e_l stored in $\mathbb{T}_{e_k}$ and $\mathbb{T}_{e_l}$, respectively. An abstract algorithm of our TOM method is shown in Algorithm 1. Algorithm 1 is explained as follows:

Algorithm 1 Triple-based Ontology Matching (TOM)

Require: Source ontologies $\mathcal{O}, \mathcal{O}'$ Similarity threshold H

Require: Pre-defined correspondences $\Phi_{ij} = \langle e_i, e_j, r, n \rangle$

Ensure: Triple-based Alignments $\mathbb{A}$

for all $e_k \in M_{\mathcal{O}}$ **do**
  $\mathbb{T}_{e_k}$ = Set of triples where e_k appears;
end for
for all $e_l \in M_{\mathcal{O}'}$ **do**
  $\mathbb{T}_{e_l}$ = Set of triples where e_l appears;
end for
for all e_k, e_l **do**
  $sim_{e_k} = 0$; $sim_{\mathbb{T}_{e_k}} = 0$; $sim_{e_l} = 0$; $sim_{\mathbb{T}_{e_l}} = 0$;
  for all $t \in \mathbb{T}_{e_k}$ **do**
    $sim_t = 0$;
    for all $t' \in \mathbb{T}_{e_l}$ **do**
      for all $\langle e_i, e_j, r, n \rangle$ **do**
        $sim_t = sim_t + s$; such that $\underset{s}{\operatorname{argmax}}(\langle e_i, e_j, r, n \rangle)$, where $t \in \mathbb{T}_{e_i}$ and $t' \in \mathbb{T}_{e_j}$
      end for
    end for
    $sim_{\mathbb{T}_{e_k}} = sim_{\mathbb{T}_{e_k}} + sim_t$;
  end for
  for all $t' \in \mathbb{T}_{e_l}$ **do**
    $sim_{t'} = 0$;
    for all $t \in \mathbb{T}_{e_k}$ **do**
      for all $\langle e_j, e_i, r, n \rangle$ **do**
        $sim_{t'} = sim_{t'} + s$; such that $\underset{s}{\operatorname{argmax}}(\langle e_j, e_i, r, n \rangle)$, where $t \in \mathbb{T}_{e_i}$ and $t' \in \mathbb{T}_{e_j}$
      end for
    end for
    $sim_{\mathbb{T}_{e_l}} = sim_{\mathbb{T}_{e_l}} + sim_{t'}$;
  end for
  $sim_{e_k} = \frac{sim_{\mathbb{T}_{e_k}}}{|\mathbb{T}_{e_k}|}$; $sim_{e_l} = \frac{sim_{\mathbb{T}_{e_l}}}{|\mathbb{T}_{e_l}|}$;
  if $sim_{e_k} \geq H$ & $sim_{e_l} \geq H$ **then**
    $score = \frac{sim_{e_k} + sim_{e_l}}{2}$;
    $\mathbb{A}.add(\langle e_k, e_l, =, score \rangle)$; $\mathbb{A}.add(\langle e_l, e_k, =, score \rangle)$;
  end if
end for
return $\mathbb{A}$;

line 1-3: For each entity $e_k \in M_{\mathcal{O}}$ (i.e. concept, property or individual) from an ontology $\mathcal{O}$, compute the set of triples $\mathbb{T}_{e_k}$ where e_k appears (see Definition 13, 14, 15).

line 4-6: For each entity $e_l \in M_{\mathcal{O}'}$ (i.e. concept, property or individual) from an ontology $\mathcal{O}'$, compute the set of triples $\mathbb{T}_{e_l}$ where e_l appears (see Definition 13, 14, 15).

line 7-32: For any two entities $e_k \in M_{\mathcal{O}}$ and $e_l \in M_{\mathcal{O}'}$ from the source ontologies $\mathcal{O}$ and $\mathcal{O}'$, compute the triple-set similarity between e_k and e_l as follows.

line 9-17: Calculating the triple-set similarity of e_k with respect to triple-set of e_l by comparing each triple $t \in \mathbb{T}_{e_k}$ with an arbitrary triple $t' \in \mathbb{T}_{e_l}$ (see line 11-15), and aggregating the highest of the similarities between $t \in \mathbb{T}_{e_k}$ and any other triple $t' \in \mathbb{T}_{e_l}$ (see line 12-14). A triple-set similarity of e_k triple-set $\mathbb{T}_{e_k}$ is stored in $sim_{\mathbb{T}_{e_k}}$ by aggregating the highest similarity score for each triple $t \in \mathbb{T}_{e_k}$ with respect to the triple-set $\mathbb{T}_{e_l}$ (see line 16).

line 18-26: Calculating the triple-set similarity of e_l with respect to triple-set of e_k by comparing each triple $t' \in \mathbb{T}_{e_l}$ with an arbitrary triple $t \in \mathbb{T}_{e_k}$ (see line 20-24), and aggregating the highest of the similarities between $t' \in \mathbb{T}_{e_l}$ and any other triple $t \in \mathbb{T}_{e_k}$ (see line 21-23). A triple-set similarity of e_l triple-set $\mathbb{T}_{e_l}$ is stored in $sim_{\mathbb{T}_{e_l}}$ by aggregating the highest similarity score for each triple $t' \in \mathbb{T}_{e_l}$ with respect to the triple-set $\mathbb{T}_{e_k}$ (see line 25).

line 27-31: Calculating the triple-set similarity ratios for both entities e_k and e_l. These ratios get stored in sim_{e_k} and sim_{e_l}, which are computed by dividing the triple-set similarities $sim_{\mathbb{T}_{e_k}}$ and $sim_{\mathbb{T}_{e_l}}$ with the cardinality of their triple-sets $\mathbb{T}_{e_k}$ and $\mathbb{T}_{e_l}$, respectively (see line 27). If the computed ratios sim_{e_k} and sim_{e_l} are higher than the given threshold H, then their average gets stored in $score$ (see line 29) and two correspondences $\langle e_k, e_l, =, score\rangle$ and $\langle e_k, e_l, =, score\rangle$ get added into the new alignment set $\mathbb{A}$ (see line 30).

line 33: Return the new alignment set $\mathbb{A}$.

5.5 Ontology Alignment using Proof-based Ontology Matching (POM)

In addition to TOM, we propose another ontology matching method *proof-based ontology matching* (POM) for ontology alignment. POM requires source ontologies at the proof-level. A *proof-level ontology* is an ontology $\mathcal{O}$, where for each inferred triple t' there exists a unique proof-tree $\mathcal{TR}_{t'}$ that describes the derivation of t' under a logic program $\mathcal{P}$ (see Definition 1). Each internal node in $\mathcal{TR}_{t'}$ is derived from its children nodes using a single rule in $\mathcal{P}$, whereas the leaf nodes corresponds to the asserted ontology triples. For an asserted ontology triple t, $\mathcal{TR}_t = \emptyset$. An ontology at the proof-level can provide the justifications (see Definition 16) behind inferred triples based on the rules and constraints defined in $\mathcal{P}$.

POM requires two source ontologies ($\mathcal{O}$ and $\mathcal{O}'$) at proof-level, together with their set of matchable entities $M_{\mathcal{O}}$ and $M_{\mathcal{O}'}$ as inputs. POM also needs a set of pre-defined correspondences of the form $\Phi = \langle e_i, e_j, r, n \rangle$, where n is a similarity score to construct a relation r between any two matchable entities $e_i \in M_{\mathcal{O}}$ and $e_j \in M_{\mathcal{O}'}$. Given pre-defined correspondences $\Phi = \langle e_i, e_j, r, n \rangle$, we calculate a similarity score between two matchable entities $e_k \in M_{\mathcal{O}}$ and $e_l \in M_{\mathcal{O}'}$, based on the similarity score $\langle e_i, e_j, r, n \rangle$ of entities e_i and e_j such that $e_i \in \mathcal{A}nces_{e_k}$ and $e_j \in \mathcal{A}nces_{e_l}$. Therefore, POM finds alignments not only based on structural similarities but also takes into account (similarities between) other deductively connected complex entity-structures—under the rules in $\mathcal{P}$ describing both domain-specific and ontology-language semantics. An abstract algorithm of our POM method is shown in Algorithm 2.

line 1-3: For each entity $e_k \in M_{\mathcal{O}}$ (i.e. concept, property or individual) from an ontology $\mathcal{O}$, compute the set of triples $\mathcal{A}nces_{e_k}$ that derived triples $\mathbb{T}_{e_k}$ where e_k appears (see Definition 17).

line 4-6: For each entity $e_l \in M_{\mathcal{O}'}$ (i.e. concept, property or individual) from an ontology $\mathcal{O}'$, compute the set of triples $\mathcal{A}nces_{e_l}$ that derived triples $\mathbb{T}_{e_l}$ where e_l appears (see Definition 17).

line 7-32: For any two entities $e_k \in M_{\mathcal{O}}$ and $e_l \in M_{\mathcal{O}'}$ from the source ontologies $\mathcal{O}$ and $\mathcal{O}'$, compute the triple-set similarity between e_k and e_l as follows.

Algorithm 2 Proof-based Ontology Matching (POM)

Require: Source ontologies $\mathcal{O}, \mathcal{O}'$ Similarity threshold H

Require: Pre-defined correspondences $\Phi_{ij} = \langle e_i, e_j, r, n \rangle$

Ensure: Proof-based Alignments $\mathbb{A}$

for all $e_k \in M_{\mathcal{O}}$ **do**
  $\mathcal{A}nces_{e_k}$ = Set of triples that derived triples $\mathbb{T}_{e_k}$;
end for
for all $e_l \in M_{\mathcal{O}'}$ **do**
  $\mathcal{A}nces_{e_l}$ = Set of triples that derived triples $\mathbb{T}_{e_l}$;
end for
for all e_k, e_l **do**
  $sim_{e_k} = 0$; $sim_{\mathcal{A}nces_{e_k}} = 0$; $sim_{e_l} = 0$; $sim_{\mathcal{A}nces_{e_l}} = 0$;
  for all $t \in \mathcal{A}nces_{e_k}$ **do**
    $sim_t = 0$;
    for all $t' \in \mathcal{A}nces_{e_l}$ **do**
      for all $\langle e_i, e_j, r, n \rangle$ **do**
        $sim_t = sim_t + s$; such that $\underset{s}{\operatorname{argmax}}(\langle e_i, e_j, r, n \rangle)$, where $t \in \mathbb{T}_{e_i}$ and $t' \in \mathbb{T}_{e_j}$
      end for
    end for
    $sim_{\mathcal{A}nces_{e_k}} = sim_{\mathcal{A}nces_{e_k}} + sim_t$;
  end for
  for all $t' \in \mathcal{A}nces_{e_l}$ **do**
    $sim_{t'} = 0$;
    for all $t \in \mathcal{A}nces_{e_k}$ **do**
      for all $\langle e_j, e_i, r, n \rangle$ **do**
        $sim_{t'} = sim_{t'} + s$; such that $\underset{s}{\operatorname{argmax}}(\langle e_j, e_i, r, n \rangle)$, where $t \in \mathbb{T}_{e_i}$ and $t' \in \mathbb{T}_{e_j}$
      end for
    end for
    $sim_{\mathcal{A}nces_{e_l}} = sim_{\mathcal{A}nces_{e_l}} + sim_{t'}$;
  end for
  $sim_{e_k} = \frac{sim_{\mathcal{A}ncese_k}}{|\mathcal{A}nces_{e_k}|}$; $sim_{e_l} = \frac{sim_{\mathcal{A}ncese_l}}{|\mathcal{A}nces_{e_l}|}$;
  if $sim_{e_k} \geq H$ & $sim_{e_l} \geq H$ **then**
    $score = \frac{sim_{e_k} + sim_{e_l}}{2}$;
    $\mathbb{A}.add(\langle e_k, e_l, =, score \rangle)$; $\mathbb{A}.add(\langle e_l, e_k, =, score \rangle)$;
  end if
end for
return $\mathbb{A}$;

line 9-17: Calculating the ancestor-set similarity of e_k with respect to ancestor-set of e_l by comparing each triple $t \in \mathcal{A}nces_{e_k}$ with an arbitrary triple $t' \in \mathcal{A}nces_{e_l}$ (see line 11-15), and aggregating the highest of the similarities between $t \in \mathcal{A}nces_{e_k}$ and any other triple $t' \in \mathcal{A}nces_{e_l}$ (see line 12-14). An ancestor-set similarity of e_k ancestor-set $\mathcal{A}nces_{e_k}$ is stored in $sim_{\mathcal{A}nces_{e_k}}$ by aggregating the highest similarity score for each triple $t \in \mathcal{A}nces_{e_k}$ with respect to the ancestor-set $\mathcal{A}nces_{e_l}$ (see line 16).

line 18-26: Calculating the ancestor-set similarity of e_l with respect to ancestor-set of e_k by comparing each triple $t' \in \mathcal{A}nces_{e_l}$ with an arbitrary triple $t \in \mathcal{A}nces_{e_k}$ (see line 20-24), and aggregating the highest of the similarities between $t' \in \mathcal{A}nces_{e_l}$ and any other triple $t \in \mathcal{A}nces_{e_k}$ (see line 21-23). An ancestor-set similarity of e_l ancestor-set $\mathcal{A}nces_{e_l}$ is stored in $sim_{\mathcal{A}nces_{e_l}}$ by aggregating the highest similarity score for each triple $t' \in \mathcal{A}nces_{e_l}$ with respect to the ancestor-set $\mathcal{A}nces_{e_k}$ (see line 25).

line 27-31: Calculating the ancestor-set similarity ratios for both entities e_k and e_l. These ratios get stored in sim_{e_k} and sim_{e_l}, which are computed by dividing the ancestor-set similarities $sim_{\mathcal{A}nces_{e_k}}$ and $sim_{\mathcal{A}nces_{e_l}}$ with the cardinality of their ancestor-set $\mathcal{A}nces_{e_k}$ and $\mathcal{A}nces_{e_l}$, respectively (see line 27). If the computed ratios sim_{e_k} and sim_{e_l} are higher than the given threshold H, then their average gets stored in $score$ (see line 29) and two correspondences $\langle e_k, e_l, =, score\rangle$ and $\langle e_k, e_l, =, score\rangle$ get added into the new alignment set $\mathbb{A}$ (see line 30).

line 33: Return the new alignment set $\mathbb{A}$.

5.6 Evaluation

We found the matching approach mentioned in Ritze et. al. [176] competitive to our matchers. So, we compared our matching results with the results of Ritze et. al. [176] on the *Conference* data-set of the Ontology Alignment Evaluation Initiative (OAEI) [167]. The *Conference* data-set is comprised of 16 ontologies from the same domain (conference organization). These ontologies are quite suitable for ontology

alignment task because of their heterogeneous character of origin. Based on the following input ontologies from the conference data-set, we compare the results generated by our matchers with the results generated by Ritze et. al. [176]:

1. **Source Ontology # 1** (`cmt.owl`): 36 Concepts, 10 Datatype Properties, 49 Object Properties, 0 Individuals.

2. **Source Ontology # 2** (`conference.owl`): 60 Concepts, 18 Datatype Properties, 46 Object Properties, 0 Individuals.

3. **Initial Alignments** (`cmt-Conference.owl`): Total of 15 Correspondences between atomic entities (concepts and properties) from the source ontologies.

4. **Similarity Threshold** (0.3): Considering all the correspondences that have a similarity score above 0.3.

5.6.1 Ritze et. al. Matching Results

Based on the above-mentioned input ontologies (`cmt.owl` and `conference.owl`) and initial alignments between them (`cmt-Conference.owl`), results generated by Ritze et. al. [176] approach are shown in Table 5.2.

Table 5.2: Ritze et. al. Matching Results: Complex Correspondences

	CMT Ontology	Relation	Conference Ontology	Score
1	Meta-Review	⇔	[owl:onProperty reviews; owl:someValuesFrom Camera_ready_contribution]	1.0
2	Meta-Reviewer	⇔	[owl:onProperty contributes; owl:someValuesFrom Camera_ready_contribution]	1.0
3	ExternalReviewer	⇔	[owl:onProperty contributes; owl:someValuesFrom Extended_abstract]	1.0
4	ConferenceChair	⇔	[owl:onProperty contributes; owl:someValuesFrom Poster]	1.0
5	AssociatedChair	⇔	[owl:onProperty contributes; owl:someValuesFrom Abstract]	1.0
6	Reviewer	⇔	[owl:onProperty contributes; owl:someValuesFrom Reviewed_contribution]	1.0
7	AuthorNotReviewer	⇔	[owl:onProperty contributes; owl:someValuesFrom Reviewed_contribution]	1.0
8	ConferenceMember	⇔	[owl:onProperty contributes; owl:someValuesFrom Poster]	1.0
9	User	⇔	[owl:onProperty contributes; owl:someValuesFrom Poster]	1.0
10	ProgramCommitteeChair	⇔	[owl:onProperty was_a_committee_chair_of; owl:someValuesFrom Program_committee]	1.0

Ritze et. al. approach identifies only boolean correspondences for establishing an equivalence relation between two complex entities (i.e. equivalence relation with 100% confidence). Hence, the detected correspondences are either correct or incorrect with 100% confidence. On the *Conference* data-set, from the Ritze et. al. results (shown in Table 5.2), only two of the complex correspondences were found correct. First correct correspondence (see line 6 in Table 5.2) provides an equivalence relation between the concept `Reviewer` (from `cmt.owl`) and a property-restricted concept (from `Conference.owl`), where its property `contributes` must have some property values from the concept `Reviewed_contribution`. Similarly, the second one (see line 7) defines a correspondence between the concept `Author-NotReviewer` (from `cmt.owl`) and same the property-restricted concept (defined in the first correspondence; see line 6).

5.6.2 TOM Matching Results

Our matching approach in TOM identified correspondences between both atomic and complex entities. Detected atomic correspondences are listed in Table 5.3; whereas complex correspondences are shown in Table 5.4.

TOM found very promising matching results between the two source ontologies. Most of the correspondences listed in Table 5.3 were found correct. The correspondences, which were found to be incorrect are `<Document,Conference_announcement>`, `<Document,Conference_www>`, `<Document,Review>`, `<Meta-Review, Conference_document>` and `<Review,Conference_document>`. Although, for some of the of the above correspondences (such as `<Meta-Review,Conference_document>` and `<Review, Conference_document>`), neither an equivalence nor a subsumption relation can be established among their entities. However, such correspondences can still be realized by other domain-specific relations. For example a `Review` or a `Meta-Review` can be related with `Conference_document` by a domain-specific relation `is_review_for`.

One of the most interesting correspondences, that really showed the strength of our structure-based approach TOM, is `<Person,Chair>` (see line 8 in Table 5.3). If

Table 5.3: TOM Matching Results: Atomic Correspondences

	CMT Ontology	Conference Ontology	Score
1	Author	ns1:Contribution_co-author	1.0
2	Author	Person	1.0
3	Co-author	Person	1.0
4	Co-author	Regular_author	1.0
5	Person	ns1:Contribution_co-author	1.0
6	Person	Regular_author	1.0
7	Chairman	Person	1.0
8	Person	Chair	1.0
9	Document	Abstract	1.0
10	PaperAbstract	Conference_document	1.0
11	Document	Review	1.0
12	Review	Conference_document	1.0
13	assignedByReviewer	invites_co-reviewers	1.0
14	assignExternalReviewer	invited_by	1.0
15	Meta-Review	Conference_document	0.75
16	Meta-Review	ns1:Review	0.75
17	Author	Contribution_1th-author	0.6818
18	Co-author	Contribution_1th-author	0.6818
19	Person	Contribution_1th-author	0.6818
20	Paper	Conference_document	0.6818
21	Document	Call_for_paper	0.6666
22	Document	Call_for_participation	0.6666
23	Document	Conference_announcement	0.6666
24	Document	Conference_www	0.6666
25	Document	Information_for_participants	0.6666
26	Author	Conference_contributor	0.65
27	Co-author	Conference_contributor	0.65
28	Person	Conference_contributor	0.65
29	User	Contribution_co-author	0.65
30	User	Person	0.65
31	User	Regular_author	0.65
32	User	ns1:Contribution_1th-author	0.3318
33	User	Conference_contributor	0.3

we consider entities from the source ontologies without any underlying ontology-structure or domain application in mind, it will not be possible to find a correspondence between two concepts `Person` and `Chair`. First of all, there is no lexical or string-based similarity between these two concepts [41]. Also other thesauri-based approaches might not find any similarities between these concepts [41]. It is only the ontology structure that can provide the domain semantics to identify the domain-specific similarities between ontology-entities and can find domain-dependent correspondences between concepts, such as `<Person, Chair>`.

In addition to atomic correspondences (shown in Table 5.3), TOM also found correspondences between complex entities. Complex correspondences found by TOM are listed in Table 5.4. Compare to the results generated by Ritze et. al. approach (see Table 5.2), TOM not only found complex correspondences between restricted concepts (see lines 3, 8, 9), but also found correspondences between

Table 5.4: TOM Matching Results: Complex Correspondences

	CMT Ontology	Conference Ontology	Score
1	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	Chair	0.6666
2	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	Person	0.6666
3	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.6666
4	Author	[owl:intersectionOf (_:e24 _:e25)] _:e24 owl:unionOf (Contribution_1th-author Contribution_co-author) _:e25 owl:onProperty ns1:contributes _:e25 owl:someValuesFrom ns1:Conference_contribution	0.6875
5	Co-author	[owl:intersectionOf (_:e24 _:e25)]	0.6875
6	Person	[owl:intersectionOf (_:e24 _:e25)]	0.6875
7	User	[owl:intersectionOf (_:e24 _:e25)]	0.3375
8	Chairman	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.6666
9	Person	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.6666

concepts that are union or intersection of other complex concepts (see lines 1-8). Examples of such correspondences are listed in lines 1-3 in Table 5.4. First correspondence (see line 1) presents a similarity between the concept `Chair` and a complex concept, which is union of concepts `AssociatedChair`, `ConferenceChair` and `ProgramCommitteeChair`. Similarly the second correspondence (see line 2) presents a similarity between the same unioned concept and the concept `Person`. The third correspondence (see line 3) presents a similarity between two complex concepts: (i) a collection of the concepts: `AssociateChair`, `ConferenceChair` and `ProgramCommitteeChair`; and (ii) a restricted concept, whereby any `Associate-Chair`, `ConferenceChair`, or `ProgramCommitteeChair` should have been assigned as a committee chair of some previous `Committee`. All other correspondences (see line 4-9 in Table 5.4) were found correct as well.

5.6.3 POM Matching Results

POM also identified correspondences between both atomic and complex entities. Detected atomic correspondences are listed in Table 5.5; whereas complex correspondences are shown in Table 5.6. All the atomic correspondences detected by POM (shown in Table 5.5) were found correct. Similar to TOM, POM also found domain-dependent correspondences, such as `<AssociatedChair, Person>`,

Table 5.5: POM Matching Results: Atomic Correspondences

	CMT Ontology	Conference Ontology	Score
1	Co-author	Contribution_co-author	0.5647
2	AuthorNotReviewer	Contribution_1th-author	0.5019
3	AuthorNotReviewer	Contribution_co-author	0.5019
4	Co-author	Contribution_1th-author	0.5019
5	Co-author	Regular_author	0.5
6	Meta-Review	Review	0.5
7	AuthorNotReviewer	Regular_author	0.5
8	Person	Contribution_co-author	0.4919
9	User	Contribution_co-author	0.4919
10	Co-author	Conference_contributor	0.4833
11	Co-author	Person	0.4792
12	Person	Chair	0.4772
13	User	Contribution_1th-author	0.4625
14	Person	Contribution_1th-author	0.4625
15	Person	Regular_author	0.4606
16	ConferenceMember	Contribution_co-author	0.4570
17	AuthorNotReviewer	Conference_contributor	0.45
18	ConferenceChair	Chair	0.45
19	AssociatedChair	Chair	0.45
20	AuthorNotReviewer	Person	0.4459
21	ProgramCommitteeChair	Chair	0.4404
22	Author	Contribution_1th-author	0.4352
23	Author	Contribution_co-author	0.4352
24	Author	Regular_author	0.4333
25	ConferenceMember	Contribution_1th-author	0.4276
26	ConferenceMember	Regular_author	0.4256
27	User	Person	0.4065
28	User	Regular_author	0.4606
29	Author	Person	0.3792
30	ConferenceMember	Person	0.3715
31	AssociatedChair	Person	0.3603
32	ConferenceChair	Person	0.3603
33	ProgramCommitteeChair	Person	0.3508
34	Chairman	Person	0.3445

<Person,Chair>, and so on.

Compared to TOM, POM found more correspondences between complex entities. Complex correspondences detected by POM are listed in Table 5.6. POM also gave promising results on the *Conference* data-set. For example, a correspondence (shown in line 13 in Table 5.6) presents a similarity between concepts Associate-Chair and a restricted concept, whereby any AssociateChair should have been assigned as a committee chair of some previous Committee. Despite the good results on the *Conference* data-set, POM also generated some incorrect correspondences, e.g. line 1 in Table 5.6 presents a correspondence between Person and a union concept (Conference⊔Person). Two concepts, Person and (Conference⊔Person), found similar (see line 1 in Table 5.6) due to the following criteria:

1. The concept *Person* was present in both ontologies and hence found similar.

Table 5.6: POM Matching Results: Complex Correspondences

	CMT Ontology	Conference Ontology	Score
1	[owl:unionOf (Conference Person)]	Person	0.4139
2	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	Chair	0.4444
3	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	Person	0.3548
4	[owl:unionOf (AssociatedChair ConferenceChair ProgramCommitteeChair)]	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.3694
5	Author	[owl:intersectionOf (_:e24 _:e25)] _:e24 owl:unionOf (Contribution_1th-author Contribution_co-author) _:e25 owl:onProperty ns1:contributes _:e25 owl:someValuesFrom ns1:Conference_contribution	0.4777
6	AuthorNotReviewer	[owl:unionOf (Invited_speaker Regular_author)]	0.4
7	AuthorNotReviewer	[owl:intersectionOf (_:e24 _:e25)]	0.5444
8	Co-author	[owl:unionOf (Invited_speaker Regular_author)]	0.4333
9	Co-author	[owl:intersectionOf (_:e24 _:e25)]	0.5777
10	ConferenceMember	[owl:intersectionOf (_:e24 _:e25)]	0.4700
11	Person	[owl:intersectionOf (_:e24 _:e25)]	0.5050
12	User	[owl:intersectionOf (_:e24 _:e25)]	0.5050
13	AssociatedChair	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.375
14	Chairman	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.3592
15	ConferenceChair	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.375
16	Person	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.4022
17	ProgramCommitteeChair	[owl:onProperty was_a_committee_chair_of owl:someValuesFrom Committee]	0.3654

2. The concept *Person* was also appearing in the union concept (`Conference` ∪ `Person`).

3. Since the concept *Person* was appearing in both `Person` and (`Conference` ∪ `Person`), their ancestors were found similar, where triples in both ancestor-sets were deriving new inferred triples about *Person*.

Another observation is that POM generated lower similarity scores between ontology-entities, as compared to TOM. The reason behind the lower similarity scores generated by POM was the lack of ontology individuals. POM operates on the proof-steps that are generated from the ontology model under a given logic program $\mathcal{P}$. In the given logic program $\mathcal{P}$ defining RDF/OWL semantics, most of the rules were defined on *assertional* axioms $\mathcal{A}$ that require individuals; where as the source *Conference* ontologies lack in individuals. Hence there were lesser proof-steps generated, and therefore generated similarity scores were compromised.

5.7 Merging Contextualized Sub-ontologies

Ontology merging is achieved by (i) asserting the source ontologies and an alignment between them in an ontology reasoner [118, 140, 141, 177], and then (ii) inferring new relations in the merged ontology (see Figure 5.1). An ontology alignment consists of correspondences that define a relation between two entities from different source ontologies (see Definition 12). Our matchers, TOM and POM, based on their defined similarity measures (see Sections 5.4 and 5.5), generate only a similarity score between (both atomic and complex) entities, but not a relation between those entities. Thus, based on the found similarity between two ontology-entities e_1 and e_2, a correspondence relation between e_1 and e_2 (see Definition 11) is then defined by the user. OWL and RDF(S) offer constructs that allow users to define correspondences (see Definition 11) by providing a semantic relation between two different ontology-entities (concepts, properties or individuals). Such OWL and RDF(S) constructs are listed as follows:

1. `owl:equivalentClass`: declares an equivalence relation between two ontology concepts. Semantics of `owl:equivalentClass` is defined via the following N3 rules:
   (a) `{?A owl:equivalentClass ?B} => {?B owl:equivalentClass ?A}`: `owl:equivalentClass` is a symmetric relation.
   (b) `{?A owl:equivalentClass ?B. ?B owl:equivalentClass ?C} => {?A owl:equivalentClass ?C}`: `owl:equivalentClass` is a transitive relation.
   (c) `{?A owl:equivalentClass ?B} => {?A rdfs:subClassOf ?B}`: If two concepts have an equivalence relation, then both concepts are subsumed by each other.

2. `rdfs:subClassOf`: declares a subsumption relation between two ontology concepts. Semantics of `rdfs:subClassOf` is defined via the following rules:
   (a) `{?C rdfs:subClassOf ?D. ?X a ?C} => {?X a ?D}`: Given concept C is a sub-concept of concept D, then any individual of C is also an individual of D.

(b) {?C rdfs:subClassOf ?D. ?D rdfs:subClassOf ?E} => {?C rdfs:subClassOf ?E}: rdfs:subClassOf is a transitive relation.

3. owl:disjointWith: declares a disjointness relation between two ontology concepts. Semantics of owl:disjointWith is defined via the following rules:

   (a) {?C owl:disjointWith ?D. ?X a ?C. ?Y a ?D } => {?X owl:differentFrom ?Y}: Given two concepts C and D are disjoint with each other, then any individual of C is different from any individual of D.

   (b) {?C owl:disjointWith ?D. ?X a ?C. ?X a ?D} => false: Given two concepts C and D are disjoint with each other, then any individual belonging to both C and D makes an ontology inconsistent.

4. owl:equivalentProperty: declares an equivalence relation between two ontology properties. Semantics of owl:equivalentProperty is defined via the following N3 rules:

   (a) {?P owl:equivalentProperty ?Q. ?S ?P ?O} => {?S ?Q ?O}: Given two properties P and Q have an equivalence relation between them, then all extensions of P are also the extensions of Q.

   (b) {?P owl:equivalentProperty ?Q} => {?Q owl:equivalentProperty ?P}: owl:equivalentProperty is a symmetric relation.

   (c) {?P owl:equivalentProperty ?Q. ?Q owl:equivalentProperty ?R} => {?P owl:equivalentProperty ?R}: owl:equivalentProperty is a transitive relation.

   (d) {?P owl:equivalentProperty ?Q} => {?P rdfs:subPropertyOf ?Q. ?Q rdfs:subPropertyOf ?P}: If two properties P and Q have an equivalence relation between them, then both P and Q are subsumed by each other.

5. rdfs:subPropertyOf: declares a subsumption relation between two ontology properties. Semantics of rdfs:subPropertyOf is defined via the following N3 rules:

(a) {?P rdfs:subPropertyOf ?Q. ?S ?P ?O} => {?S ?Q ?O}: Given a property P is a sub-property of Q, then all extensions of P are also the all extensions of Q.

(b) {?P rdfs:subPropertyOf ?Q. ?Q rdfs:subPropertyOf ?R} => {?P rdfs:subPropertyOf ?R}: rdfs:subPropertyOf is a transitive relation.

(c) {?P rdfs:subPropertyOf ?Q. ?Q rdfs:domain ?C} => {?P rdfs:domain ?C}: Given a property P is a sub-property of Q, then the domain of Q is also the domain of P.

(d) {?P rdfs:subPropertyOf ?Q. ?Q rdfs:range ?C} => {?P rdfs:range ?C}: Given a property P is a sub-property of Q, then the range of Q is also the range of P.

6. owl:propertyDisjointWith: declares a disjointness relation between two ontology properties. Semantics of owl:propertyDisjointWith is defined via the following N3 rule:

   {?P owl:propertyDisjointWith ?Q. ?X ?P ?Y. ?X ?Q ?Y} => false: Given two properties P and Q are disjoint with each other, then any pair of extensions belonging to both P and Q makes an ontology inconsistent.

7. owl:sameAs: declares an equivalence relation between two (different) ontology individuals. Semantics of owl:sameAs is defined via the following N3 rules:

   (a) {?X owl:sameAs ?Y} => {?Y owl:sameAs ?X}: owl:sameAs is a symmetric relation.

   (b) {?X owl:sameAs ?Y. ?Y owl:sameAs ?Z} => {?X owl:sameAs ?Z}: owl:sameAs is a transitive relation.

   (c) {?X owl:sameAs ?Y. ?X owl:differentFrom ?Y} => false: Two ontology individuals that have both equivalence and nonequivalence relation makes an ontology inconsistent.

8. `owl:differentFrom`: declares a nonequivalence relation between two ontology individuals. Semantics of `owl:differentFrom` is defined via the following N3 rule:

   `{?A owl:differentFrom ?B} => {?B owl:differentFrom ?A}`: `owl:differentFrom` is a symmetric relation.

In $\mathcal{K}$-$\mathcal{MORPH}$, after extracting contextualized sub-ontologies (see Chapter 4), we align and merge the extracted contextualized sub-ontologies. In order to align these sub-ontologies, we use two of our developed matchers, TOM and POM (see Sections 5.4 and 5.5), that find similarities between two ontology-entities from the extracted sub-ontologies based on the given context-specific alignments $\mathbb{A}_x$ (see Definition 2). Based on the found similarities between ontology-entities e_1 and e_2, a user then defines a correspondence relation between e_1 and e_2 using the above-described OWL and RDF(S) constructs. Hence, by defining correspondences between similar ontology-entities, an alignment $\mathbb{A}$ between the extracted sub-ontologies is obtained. The defined alignment $\mathbb{A}$ serves as the basis for merging the extracted contextualized sub-ontologies, where new correspondences in $\mathbb{A}$ (see Definition 11) provides new semantic relations between different ontology-entities (see Figure 5.1). Hence, in the merged ontology, entities from two different ontologies are aligned based on the above-mentioned semantic relations. Such an alignment supports semantic interoperability between two different ontologies by relating corresponding entities; and by using an ontology reasoner [118, 140, 141, 177], inferring new knowledge from the aligned entities under the above-mentioned rules. An abstract process for merging contextualized sub-ontologies can be defined as follows:

Definition 18 (Merging Contextualized Sub-ontologies) *Let $\mathbb{O}$ be the set of ontologies, $\mathbb{P}$ be the set of logic programs (see Definition 1), and Π be the set of pre-defined context-specific alignments (see Definition 2). Our ontology merging method is defined as a function:*

$merge\_sub\_onto : 2^{\mathbb{O}} \times \mathbb{P} \times \Pi \longrightarrow \mathbb{O}$

5.7.1 Merging Conference Ontologies

As discussed in the previous section, TOM and POM can only generate a similarity score between two ontology entities; but not a correspondence relation between them. Therefore, in order to demonstrate the merging of the source conference ontologies (see Section 5.6), we defined a correspondence relation for each of the correspondences found in Tables 5.3 and 5.5. Based on the defined correspondences, Table 5.7 highlights some of the inferred results found in the final merged ontology.

Table 5.7: Merged Conference Ontology: Inferred Results

	Alignment	**Inferred Knowledge**
1	ctm:ProgramCommitteeChair ⊑ conf:Chair	ctm:ProgramCommitteeChair ⊑ conf:Committee_member. ctm:ProgramCommitteeChair ⊑ ctm:Chairman
2	conf:Contribution_co-author ⊑ ctm:Author	conf:Contribution_co-author ⊑ ctm:ConferenceMember. conf:Contribution_co-author ⊑ conf:Person
3	ctm:Author ⊑ conf:Person	ctm:AuthorNotReviewer ⊑ conf:Person. conf:Contribution_1th-author ⊑ conf:Person. conf:Contribution_co-author ⊑ conf:Person
4	conf:Conference_contributor ⊑ ctm:Person	conf:Active_conference_participant ⊑ ctm:Person. conf:Invited_speaker ⊑ ctm:Person
5	ctm:Person ≡ conf:Person	ctm:Person ≡ ctm:User. ctm:Chairman ⊑ ctm:User. ctm:ConferenceMember ⊑ conf:Person. ctm:ConferenceMember ⊑ ctm:User. ctm:ExternalReviewer ⊑ conf:Person. ctm:ExternalReviewer ⊑ ctm:User. ctm:ProgramCommitteeMember ⊑ conf:Person. ctm:ProgramCommitteeMember ⊑ ctm:User. ctm:Reviewer ⊑ conf:Person. conf:Committee_member ⊑ ctm:Person. conf:Conference_applicant ⊑ ctm:Person. conf:Conference_participant ⊑ ctm:Person. conf:Passive_conference_participant ⊑ ctm:Person. conf:Registeered_applicant ⊑ ctm:Person. conf:Reviewer ⊑ ctm:Person. conf:Track-workshop_chair ⊑ ctm:Person. conf:Chair ⊑ ctm:User. ctm:Meta-Reviewer ⊑ conf:Person. conf:Co-chair ⊑ ctm:Person. conf:Early_paid_applicant ⊑ ctm:Person. conf:Late_paid_applicant ⊑ ctm:Person
6	ctm:User ≡ conf:Person	ctm:User ≡ ctm:User. ctm:Administrator ⊑ conf:Person. ctm:Chairman ⊑ ctm:User. ctm:ConferenceMember ⊑ ctm:User. ctm:ExternalReviewer ⊑ ctm:User. ctm:ProgramCommitteeMember ⊑ ctm:User. ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ conf:Late_paid_applicant ⊑ ctm:User
7	conf:Conference_document ⊑ ctm:Document	conf:Conference_contribution ⊑ ctm:Document. conf:Poster ⊑ ctm:Document. conf:Presentation ⊑ ctm:Document. conf:Written_contribution ⊑ ctm:Document. conf:Regular_contribution ⊑ ctm:Document. conf:Invited_talk ⊑ ctm:Document. conf:Paper ⊑ ctm:Document. conf:Extended_abstract ⊑ ctm:Document. conf:Submitted_contribution ⊑ ctm:Document. conf:Accepted_contribution ⊑ ctm:Document. conf:Camera_ready_contribution ⊑ ctm:Document. conf:Rejected_contribution ⊑ ctm:Document. conf:Reviewed_contribution ⊑ ctm:Document
8	ctm:PaperAbstract ⊑ conf:Abstract	ctm:PaperAbstract ⊑ conf:Extended_abstract. ctm:PaperAbstract ⊑ conf:Regular_contribution. ctm:PaperAbstract ⊑ conf:Written_contribution. ctm:PaperAbstract ⊑ conf:Conference_contribution

≡: `owl:equivalentClass`; ⊑: `rdf:subClassOf`;

5.8 Experiment and Results: Merging Contextualized Sub-ontologies for Prostate Cancer Management

We extended our earlier experiment (see section 4.5) for the prostate cancer (PC) scenario (see Section 3.3) to demonstrate the merging of the extracted contextualized sub-ontologies from three location-specific PC clinical pathway ontologies, in order to generate comprehensive therapeutic PC work-flow knowledge for the problem-context *therapeutic decision support*. The objective of this experiment is to generate a more comprehensive PC ontology, whereby one can suggest alternative treatments or extend interventions at one location based on knowledge contained in other PC ontologies. For instance, if location L_1 is prescribing intervention I_1 for condition C, and location L_2 is prescribing intervention I_2 for condition C', where conditions C and C' are similar to each other, then one can imply, after satisfying clinical pragmatics, that condition C (or C') can be treated by both interventions I_1 and I_2. Likewise, if location L_3 has no information about how to handle condition C (or C'), then the same inference can be applied to suggest interventions I_1 and I_2.

In order to fulfill the desired objective (i.e. *therapeutic decision support*), we demonstrate the merging of therapeutic PC work-flow knowledge from three location-specific PC ontologies.

5.8.1 Extracted Contextualized Sub-ontologies

We used three PC ontologies that describes PC clinical pathways, namely: (i) PC Halifax Pathway, (ii) PC Calgary Pathway, and (iii) PC Winnipeg Pathway [62]. For *therapeutic decision support* context, the user is interested in such extracted sub-ontologies that describes only (i) the treatments, (ii) their durations, (iii) their follow-ups, (iv) their care-settings, and (v) the practitioners involved for them. $\mathcal{K}$-$\mathcal{MORPH}$ extracted a contextualized sub-ontology from each of the three PC ontologies (see Section 4.5.2 for details). Concepts (including their axioms and assertions) that were extracted in the sub-ontologies are *Treatment*, *Followup*, *Frequency*, *Interval Duration*, and *Clinician*.

```
@prefix pc1:  $<http://www.owl-ontologies.com/PC-Halifax.owl#>.$
@prefix pc2:  $<http://www.owl-ontologies.com/PC-Calgary.owl#>.$
@prefix pc3:  $<http://www.owl-ontologies.com/PC-Winipeg.owl#>.$
```

Alignments Between Classes:

```
pc1:Clinician owl:equivalentClass pc2:Clinician.
pc1:Clinician owl:equivalentClass pc3:Clinician.

pc1:Treatment owl:equivalentClass pc2:Treatment.
pc1:Treatment owl:equivalentClass pc3:Treatment.
```

Alignments Between Properties:

```
pc1:hasTask owl:equivalentProperty pc2:hasTask.
pc1:hasTask owl:equivalentProperty pc3:hasTask.

pc1:applyToCareSetting owl:equivalentProperty
pc2:applyToCareSetting.
pc1:applyToCareSetting owl:equivalentProperty
pc3:applyToCareSetting.

pc1:hasFrequency owl:equivalentProperty pc2:hasFrequency.
pc1:hasFrequency owl:equivalentProperty pc3:hasFrequency.

pc1:hasInterval owl:equivalentProperty pc2:hasInterval.
pc1:hasInterval owl:equivalentProperty pc3:hasInterval.

pc1:hasFollowUpCare owl:equivalentProperty
pc2:hasFollowUpCare.
pc1:hasFollowUpCare owl:equivalentProperty
pc3:hasFollowUpCare.

pc1:hasDecisionCriteria owl:equivalentProperty
pc2:hasDecisionCriteria.
pc1:hasDecisionCriteria owl:equivalentProperty
pc3:hasDecisionCriteria.

pc1:HasDuration owl:equivalentProperty pc2:HasDuration.
pc1:HasDuration owl:equivalentProperty pc3:HasDuration.

pc1:IsPerformedBy owl:equivalentProperty pc2:IsPerformedBy.
pc1:IsPerformedBy owl:equivalentProperty pc3:IsPerformedBy.

pc1:isFollowedByFrequency owl:equivalentProperty
pc3:isFollowedByFrequency.

pc1:applyToInvestigation owl:equivalentProperty
pc2:applyToInvestigation.

pc1:isFollowedBy owl:equivalentProperty pc3:isFollowedBy.
```

Alignments Between Individuals:

```
pc2:ActiveSurveillance owl:sameAs pc1:ActiveSurveillance.
pc2:ActiveSurveillance owl:sameAs pc3:ActiveSurveillance.

pc2:ExternalBeamRadiationTherapy owl:sameAs
pc1:ExternalBeamRadiationTherapy.
pc2:ExternalBeamRadiationTherapy owl:sameAs
pc3:ExternalBeamRadiationTherapy.

pc2:Surgery owl:sameAs pc1:Surgery.
pc2:Surgery owl:sameAs pc3:Surgery.

pc2:Brachytherapy owl:sameAs pc3:Brachytherapy.

pc2:Cryotherapy owl:sameAs pc3:Cryotherapy.

pc2:Urologist owl:sameAs pc1:Urologist.
pc2:Urologist owl:sameAs pc3:Urologist.

pc2:RadiationOncologist owl:sameAs pc1:RadiationOncologist.
pc2:RadiationOncologist owl:sameAs pc3:RadiationOncologist.

pc2:FamilyPhysician owl:sameAs pc1:FamilyPhysician.
pc2:FamilyPhysician owl:sameAs pc3:FamilyPhysician.
```

Figure 5.2: Alignments for Contextualized PC Sub-ontologies

5.8.2 Merging Contextualized Sub-ontologies

Based on the pre-defined context-specific alignments among three contextualized sub-ontologies (see section 4.5), new correspondences were found between classes, including their properties and instances, for *Treatment*, *Followup*, *Frequency*, *Interval Duration*, and *Clinician*. Correspondences obtained through TOM and POM are shown in Figure 5.2. Based on the identified correspondences, and the given context-specific axioms and alignments, extracted contextualized sub-ontologies were then merged to generate possible 'knowledge-links' between the aligned PC treatments.

Figures 5.3 and 5.4 show exemplar results based on the merging of ontology concepts: *Clinician*, *Treatment*, *Followup* and *Interval*. In figure 5.3, the merged knowledge has determined that the treatment *Active Surveillance* in Halifax (represented by the instance `PC-Halifax:ActiveSurveillance`) can be conducted by a *Primary Urologist*. In the actual pathway, this information was not available for

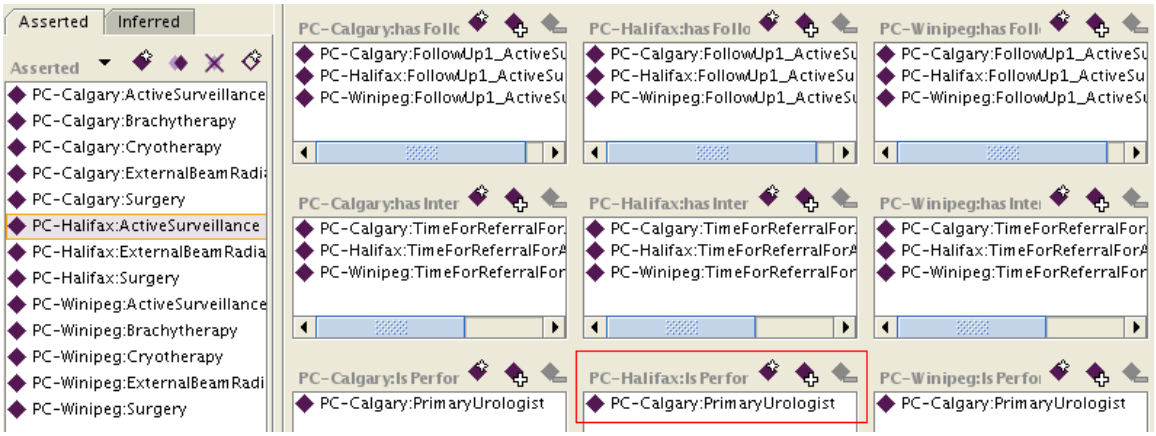

Figure 5.3: Merged Knowledge about PC-Halifax:ActiveSurveillance (highlighted in the left panel)

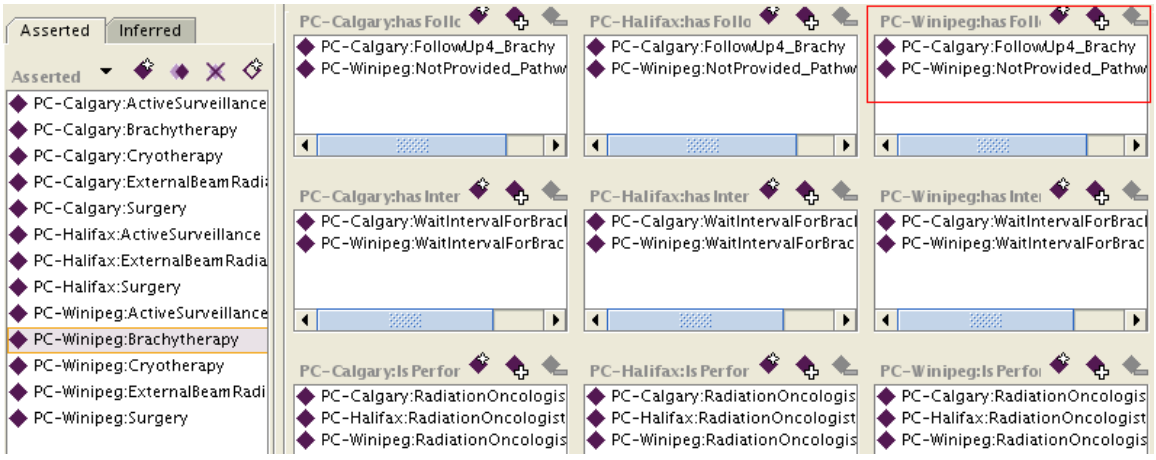

Figure 5.4: Merged knowledge about PC-Winnipeg:Brachytherapy

Halifax; but due to the ontology alignments (see Figure 5.2) this task was found to be similar to one in Calgary, and the actor performing this task in Calgary was extended to Halifax. In figure 5.4, we have inferred that in Winnipeg the treatment *Brachytherapy* (represented as `PC-Winnipeg:Brachytherapy`) can have a follow-up treatment because in Calgary a similar treatment has a follow-up treatment. In the Winnipeg PC ontology, this information was not initially present, and merging of contextualized PC sub-ontologies have extended the original knowledge about treatments.

5.9 Summary

Semantic Web enabled applications usually demand a networking of ontologies for providing a comprehensive knowledge-base to derive domain-specific solutions under different contexts. This involves solving heterogeneity among source ontologies under different contexts, which is an outstanding challenge in the field of Ontology Matching and Alignment [73]. Context-driven reconciliation of ontologies aims to extract contextualized sub-ontologies and then merge those sub-ontologies to generate a merged ontology for facilitating the problem-context at hand. In this chapter, we described two of our ontology matching approaches TOM and POM for finding complex correspondences between two source ontologies. We evaluated our matchers TOM and POM on data-sets of OAEI [167] and compared its results with existing ontology matching systems. Furthermore, we demonstrated the use of our matchers in finding correspondences among three prostate cancer pathway ontologies and merging those ontologies for establishing therapeutic PC work-flow knowledge for the problem-context *therapeutic decision support*.

Chapter 6

Detecting and Resolving Inconsistencies in Ontologies Using Contradiction Derivations

Knowledge representation via Semantic Web (SW) ontologies requires a careful analysis of a domain of discourse for identifying domain-related concepts, properties and constraints [32]. Ontologies [89] serve as a backbone for the Semantic Web as they provide constructs that represent shared vocabularies from multiple domains, such as Medicine and Healthcare [3, 87]. For different domain-specific applications, domain experts try to maintain a standardized and shareable knowledge-base by modelling their domain knowledge as ontologies through SW languages and standards [32, 87]. Given that domain-specific concepts and properties are modelled in an ontology, the next step is to infer new knowledge based on the domain-specific semantics and ontology-language constructs. A Logic Program (LP) is comprised of rules that define semantics for both the ontology-language constructs and the domain-specific knowledge modelled into the ontology structure. Hence by declaring a logic program and executing it, domain experts can obtain inferred knowledge and also can define domain-specific constraints. When modelling knowledge sources in terms of ontologies, two kind of ontology constraints can be defined:

1. *Ontology Language Constraints* define such situations, based on the ontology language semantics, which should not be reasoned and inferred. Ontology language semantics and constraints are represented in well-defined rules [49] and are usually embedded in SW reasoners [140, 141, 177].

2. *Domain-specific Constraints* define such domain-specific situations, based on the domain knowledge, which should not be reasoned and inferred. These constraints are defined by the user, using ontology-modelled knowledge and ontology constructs [178].

Both types of constraints are modelled in terms of *integrity constraint rules* (in a logic program), which are used to detect logical contradictions in the ontology model. Thus when a logical contradiction is raised, the ontology is considered inconsistent [69,70]. Logical contradictions may occur due to a number of reasons including: (i) difference in conceptualization, (ii) contradictory domain knowledge, (iii) errors in conceptualization, and (iv) different realizations of the concepts and properties in different contexts and applications. Most of the above issues affect the ontology model, and may turn a consistent ontology into an incoherent or inconsistent one [70, 179].

In contrast to finding inconsistencies in evolving ontologies [36, 70], inconsistencies can also appear in a networked ontology when reconciling various source ontologies [180]. Our $\mathcal{K}$-$\mathcal{MORPH}$ framework performs a context-driven ontology reconciliation among ontology-modelled knowledge sources, and generates a networked ontology for providing a comprehensive knowledge-base pertinent to the problem-context at hand (see Chapter 3). In $\mathcal{K}$-$\mathcal{MORPH}$, during reconciliation of ontologies, based on the identified alignments between consistent source ontologies, a networked (i.e. merged or integrated or mediated) ontology may also have inconsistencies [116, 180]. Thus, in order to ensure consistency in the $\mathcal{K}$-$\mathcal{MORPH}$ generated ontology, we have developed a method for detecting and resolving inconsistencies. In this chapter, we will describe our approach for detecting and resolving inconsistencies in networked ontologies.

6.1 Dealing with Inconsistencies: State-of-the-Art

Regardless whether an inconsistency occurs during ontology evolution [36] or ontology reconciliation [180], there are mainly two ways to deal with it: either resolve it, or reason with the inconsistent ontology.

6.1.1 Reasoning with an Inconsistent Ontology

When reasoning with an inconsistent ontology, querying is applied on a consistent fragment of the inconsistent ontology [74]. Identifying a consistent fragment is based on a selection function, which can be defined by some syntactic or semantic

relevance [75]. One of the recent attempts was reported in [74], where ontologies are expressed as *defeasible logic programs* (DeLP). For a given query, a dialectical analysis is performed on the corresponding DeLP program, where all arguments in favor and against the query results will be taken into account [74].

6.1.2 Detecting and Resolving Inconsistencies

In contrast to reasoning with inconsistent ontologies, there have been various attempts towards measuring and resolving inconsistencies [116, 179, 181]. Radon is one of these approaches that generates *minimal inconsistent subsets* for resolving inconsistencies [180]. The proposed work in Haase et. al. [36, 70] deals with *consistent evolution of ontologies*. The evolution process consists of two main phases: (i) *inconsistency detection* and (ii) *change generation*. During inconsistency detection, based on the defined consistency conditions, certain parts in the ontology that do not meet consistency conditions are identified and presented as inconsistencies. Identifying relevant axioms that contribute to an inconsistency are based on a selection function, which determines how the ontology axioms are structurally connected with the inconsistency. Change generation ensures the consistency of the ontology by removing identified axioms that resolve detected inconsistencies. This approach provides methodologies for extracting *minimal inconsistent sub-ontologies* and *maximal consistent sub-ontologies*. One of the recent attempts was reported in Scharrenbach et. al. [71] that uses *default logics* for relaxing the axioms that cause incoherency [69, 70], and shows how *probabilistic description logics* can be used to resolve conflicts and retrieve a consistent knowledge base.

Both (i) *reasoning with inconsistencies* and (ii) *resolving inconsistencies* share a common goal: extracting a (maximal) consistent sub-ontology from an inconsistent one. The former reasons only on the identified consistent sub-ontology; whereas the latter first removes the inconsistent part, and then applies the reasoner on the resultant consistent sub-ontology to find satisfiable deductions. Nowadays, ontologies are usually evolved by either re-using or reconciling other ontologies [41]. During ontology development, re-using or reconciling an inconsistent source ontology into a native ontology will also make the native ontology inconsistent [180].

Thus, we argue that compared to reasoning with inconsistencies, resolving inconsistencies first shall be more beneficial for a consistent ontology evolution [36,70]. When resolving inconsistencies in an inconsistent ontology, the target is to extract a consistent sub-ontology [36]. Thus, by resolving all the detected inconsistencies, a consistent fragment of an inconsistent ontology can still be re-used or reconciled with other ontologies, without importing its inconsistencies.

In our approach, we focus on detecting and resolving inconsistencies in ontologies when combined with a logic program (LP) [72]. A logic program consists of rules of the form $X_1, \ldots, X_n \Rightarrow Y$ to be applied on an ontology, from which inferences can be drawn [49]. Our approach for detecting inconsistencies in an ontology deals with the identification of *contradiction derivations* under the *integrity constraint rules* defined in an LP. For resolving detected inconsistencies, we generate all possible *minimal inconsistent resolve candidates* (MIRCs). An inconsistent resolve candidate of an inconsistent ontology is a set of asserted ontology triples whose removal results in a consistent sub-ontology. Removing an MIRC from the inconsistent ontology will result in a maximal consistent sub-ontology w.r.t. the given logic program. To inform the user about the consequences of removing an MIRC, we also provide a list of all its derived triples. We evaluated our approach on the prostate cancer ontology [62], where we detected all the inconsistencies in this ontology and generated all possible MIRCs for extracting a maximal consistent sub-ontology.

6.2 Preliminaries

For our purpose, we consider RDF/OWL ontologies that are defined based on a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I}, \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, comprised of concepts $\mathcal{C}$, properties $\mathcal{R}$, individuals $\mathcal{I}$, literals $\mathcal{L}$ and RDF/OWL constructs representing meta-classes $\mathcal{M}_c$ and meta-properties $\mathcal{M}_p$. An RDF/OWL ontology $\mathcal{O}$ can be expressed as triples of the form $\langle s, p, o \rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c)$. In a triple $\langle s, p, o \rangle$, s is called subject, p predicate, and o object. Triples allow to define *terminology* and *assertional* axioms in $\mathcal{O}$ [89]. Terminology axioms (concept axioms and property axioms) $\mathcal{T}$ are of the form $C \sqsubseteq D$ ($R \sqsubseteq S$) or $C \equiv D$ ($R \equiv S$) such that $C, D \in \mathcal{C}$ and $R, S \in \mathcal{R}$. Assertional axioms (concept assertions and property

assertions) $\mathcal{A}$ are of the form $C(a)$ or $R(b,c)$ such that $C \in \mathcal{C}$, $R \in \mathcal{R}$, $a,b,c \in \mathcal{I}$. The semantics of an ontology is defined by an *interpretation* that provides mapping from (i) ontology individuals, (ii) ontology concepts and (iii) ontology properties to (a) elements of the domain, (b) collections of the domain-elements and (c) binary relations between the domain-elements, respectively. A *model* of an ontology is such an interpretation, under which all ontology-axioms are satisfied. An ontology is called *consistent*, iff there exists a model for it. An ontology that has no model is called an *inconsistent ontology* [116]. The set of ontologies is denoted by $\mathbb{O}$.

Definition 19 (Logic Program) *A logic program $\mathcal{P}$, over a vocabulary $\mathcal{V} = \langle \mathcal{C}, \mathcal{R}, \mathcal{I} \rangle$ and a set of variables $\mathcal{X}$, is a set of Horn Logic rules of the form $X_1, \ldots, X_n \Rightarrow Y$, where $X_i, Y \in \{\langle s,p,o \rangle\} \cup \{\top, \bot\}$ such that $\langle s,p,o \rangle \in (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{X}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{M}_p \cup \mathcal{X}) \times (\mathcal{C} \cup \mathcal{R} \cup \mathcal{I} \cup \mathcal{L} \cup \mathcal{M}_c \cup \mathcal{X})$. A rule of the form $X_1, \ldots, X_n \Rightarrow \bot$ is called an integrity constraint rule.*

A triple t' is an inferred ontology triple, if t' is derived from a set of ontology triples under a logic program $\mathcal{P}$. For each inferred triple t' there exists a unique proof-tree $\mathcal{TR}_{t'}$ that describes the derivation of t' under $\mathcal{P}$. Each internal node in $\mathcal{TR}_{t'}$ is derived from its children nodes using a single rule in $\mathcal{P}$, whereas the leaf nodes correspond to the asserted ontology triples. $\mathbb{T}_{\mathcal{P}}(\mathcal{O})$ is the set of all asserted and inferred ontology triples of $\mathcal{O}$ under $\mathcal{P}$. A derivation of $\bot$ under $\mathcal{P}$ over the ontology triples $\mathbb{T}_{\mathcal{P}}(\mathcal{O})$ is called a *contradiction derivation* under $\mathcal{P}$.

Definition 20 (Minimal Contradictory Triple Set) *Given an ontology $\mathcal{O}$ and a logic program $\mathcal{P}$, a contradictory triple set (CTS) $E \subseteq \mathbb{T}_{\mathcal{P}}(\mathcal{O})$ is a set of triples used to derive $\bot$ in a single derivation step under $\mathcal{P}$. A CTS E is a minimal contradictory triple set (MCTS), if there exists no CTS E' such that $E' \subset E$. The set of MCTSs is denoted by $\mathbb{E}$.*

Definition 21 (Asserted Ancestor Triple Set) *Given a minimal contradictory triple set $E \subseteq \mathbb{T}_{\mathcal{P}}(\mathcal{O})$, an asserted ancestor triple set S for E is the set of all asserted ontology triples used to derive all the triples in E under $\mathcal{P}$.*

Definition 22 (Inconsistent Ontology) *If a contradiction derivation is possible in an ontology $\mathcal{O}$ under a logic program $\mathcal{P}$, then $\mathcal{O}$ is called inconsistent w.r.t. $\mathcal{P}$.*

Definition 23 (Sub-Ontology) *Given an ontology $\mathcal{O}$, an ontology $\mathcal{O}'$ is a sub-ontology of $\mathcal{O}$ (denoted as $\mathcal{O}' \prec \mathcal{O}$) is constructed on a limited vocabulary $\mathcal{V} = \langle \mathcal{C}', \mathcal{R}', \mathcal{I}', \mathcal{L}, \mathcal{M}_c, \mathcal{M}_p \rangle$, where ontology triples $\mathbb{T}_{\mathcal{P}}(\mathcal{O}')$ of the sub-ontology $\mathcal{O}'$ define axioms and assertions for the concepts $\mathcal{C}' \subseteq \mathcal{C}$, properties $\mathcal{R}' \subseteq \mathcal{R}$ and individuals $\mathcal{I}' \subseteq \mathcal{I}$.*

Definition 24 (Maximal Consistent Sub-Ontology) *Given an inconsistent ontology $\mathcal{O}$ w.r.t. a logic program $\mathcal{P}$, an ontology $\mathcal{O}' \prec \mathcal{O}$ is a maximal consistent sub-ontology, iff $\mathcal{O}'$ is consistent w.r.t. $\mathcal{P}$; and any ontology $\mathcal{O}''$ such that $\mathcal{O}' \prec \mathcal{O}'' \prec \mathcal{O}$, $\mathcal{O}''$ is inconsistent w.r.t. $\mathcal{P}$.*

Definition 25 (Minimal Inconsistent Resolve Candidate) *Given an inconsistent ontology $\mathcal{O}$ w.r.t. a logic program $\mathcal{P}$, an inconsistent resolve candidate (IRC) is a set of triples $M \subseteq \mathbb{T}_{\mathcal{P}}(\mathcal{O})$ such that $\mathcal{O} - M = \mathcal{O}'$ becomes consistent w.r.t. $\mathcal{P}$. A minimal inconsistent resolve candidate (MIRC) is such an IRC $M' \subset M$ such that $\mathcal{O} - M' = \mathcal{O}''$ is a maximal consistent sub-ontology of $\mathcal{O}$ w.r.t. $\mathcal{P}$. The set of MIRCs is denoted by $\mathbb{M}$.*

An abstract process for detecting and resolving inconsistencies can be defined as follows:

Definition 26 (Detecting and Resolving Inconsistencies) *Let $\mathbb{O}$ be the set of ontologies, $\mathbb{P}$ be the set of logic programs, and $\mathcal{M}$ be the set of MIRCs $\mathbb{M}$. Our method for detecting and resolving inconsistencies is defined as a function:*
$resolve\_inconsis : \mathbb{O} \times \mathbb{P} \longrightarrow \mathbb{O} \times \mathcal{M}$

Below we present an example that illustrates the concepts and definitions presented above. Consider an example logic program $\mathcal{P}'$ applied on the ontology $\mathcal{O}_1$ as shown in Example 6.1. Based on the RDF(S) subsumption rule in the logic program $\mathcal{P}'$ [49], two additional concept assertions `:Whale a :Fish` and `:Whale a :Mammal` are inferred. The ontology $\mathcal{O}_1$ is found inconsistent because two contradiction derivations were generated under the logic program $\mathcal{P}'$ (shown in the *Contradiction Derivations* Section of Example 6.1). The first contradiction derivation identified a situation where `Whale` has been classified both as a `Fish` and `Mammal`; whereas `Fish` and `Mammal` are defined as disjoint concepts. The second contradiction derivation described an error in date and time consistency (i.e. start-tracking

time is later than end-tracking time) over tracking intervals of marine animals. Thus, minimal contradictory triple sets (MCTSs) were generated (shown in the *Minimal Contradictory Triple Sets* Section). Based on the generated MCTSs, their asserted ancestor triple sets (AATSs) were computed (shown in the *Asserted Ancestor Triple Sets* Section).

Example 6.1 (Detecting and Resolving Inconsistencies):

Ontology $\mathcal{O}_1$:	***Logic Program $\mathcal{P}'$:***	***Inferred Triples:***
:Fish a owl:Class.	{?I a ?C1. ?C1 rdfs:subClassOf ?C2} ⇒ {?I a ?C2}	:Whale a :Fish.
:Mammal a owl:Class.	{?I a ?C1. ?I a ?C2. ?C1 owl:disjointWith ?C2} ⇒ ⊥	:Whale a :Mammal.
:BigFish a owl:Class.	{?F :trackingStart ?S. ?F :trackingEnd ?E.	
:Fish owl:disjointWith :Mammal.	?S math:greaterThan ?E } ⇒ ⊥	
:BigFish rdfs:subClassOf :Fish.		
:BigFish rdfs:subClassOf :Mammal.		
:Whale a :BigFish.		
:Whale :trackingStart "2006-04-10T00:00:00"^^xsd:dateTime.		
:Whale :trackingEnd "2003-04-21T00:00:00"^^xsd:dateTime.		

Contradiction Derivations:

(i)

:Whale a :BigFish, :BigFish rdfs:subClassOf :Fish $\vdash_{\mathcal{P}'}$:Whale a :Fish

:Fish owl:disjointWith :Mammal $\vdash_{\mathcal{P}'}$ ⊥

:Whale a :BigFish, :BigFish rdfs:subClassOf :Mammal $\vdash_{\mathcal{P}'}$:Whale a :Mammal

(ii)

:Whale :trackingStart "2006-04-10T00:00:00"
:Whale :trackingEnd "2003-04-21T00:00:00" $\vdash_{\mathcal{P}'}$ ⊥
"2006-04-10T00:00:00" math:greaterThan "2003-04-21T00:00:00"

Minimal Contradictory Triple Sets (MCTSs):

E_1 = {:Whale a :Fish. :Fish owl:disjointWith :Mammal. :Whale a :Mammal}

E_2 = {:Whale :trackingStart "2006-04-10T00:00:00". :Whale :trackingEnd "2003-04-21T00:00:00". "2006-04-10T00:00:00" math:greaterThan "2003-04-21T00:00:00"}

Asserted Ancestor Triple Sets:

S_1 = {:Whale a :BigFish. :BigFish rdfs:subClassOf :Fish. :BigFish rdfs:subClassOf :Mammal. :Fish owl:disjointWith :Mammal}

S_2 = {:Whale :trackingStart "2006-04-10T00:00:00". :Whale :trackingEnd "2003-04-21T00:00:00"}

Minimal Inconsistent Resolve Candidates:

M_1 = {:Whale a :BigFish. :Whale :trackingStart "2006-04-10T00:00:00"}

M_2 = {:BigFish rdfs:subClassOf :Fish. :Whale :trackingStart "2006-04-10T00:00:00"}

M_3 = {:BigFish rdfs:subClassOf :Mammal. :Whale :trackingStart "2006-04-10T00:00:00"}

M_4 = {:Fish owl:disjointWith :Mammal. :Whale :trackingStart "2006-04-10T00:00:00"}
M_5 = {:Whale a :BigFish. :Whale :trackingEnd "2003-04-21T00:00:00"}
M_6 = {:BigFish rdfs:subClassOf :Fish. :Whale :trackingEnd "2003-04-21T00:00:00"}
M_7 = {:BigFish rdfs:subClassOf :Mammal. :Whale :trackingEnd "2003-04-21T00:00:00"}
M_8 = {:Fish owl:disjointWith :Mammal. :Whale :trackingEnd "2003-04-21T00:00:00"}

For resolving all detected inconsistencies in $\mathcal{O}_1$ w.r.t. $\mathcal{P}'$, we generated all the possible MIRCs $M_1, \ldots, M_8$. Each MIRC M_i consists of a minimal set of asserted ontology triples used in generating contradiction derivations under $\mathcal{P}'$. Removing any MIRC M_i from $\mathcal{O}_1$ will result in a maximal consistent sub-ontology $\mathcal{O}'_1 \prec \mathcal{O}_1$ w.r.t. $\mathcal{P}'$; i.e. $\mathcal{O}'_1$ will not produce any contradiction derivations under $\mathcal{P}'$. In the following sections, we will describe our approach for (i) detecting inconsistencies based on found contradiction derivations; and (ii) generating MIRCs for resolving all detected inconsistencies.

6.3 Inconsistency Detection via Contradiction Derivations

In addition to the ontology language semantics rules (such as RDF(S) and OWL-DL rules) [49], a logic program $\mathcal{P}$ can also incorporate domain-specific rules. Although available reasoners, such as Pellet [140], RACER [177] and FaCT [141], can detect inconsistencies in an ontology w.r.t. the ontology language semantics (such as RDF(S) and OWL-DL) [49], none of these reasoners can detect and resolve inconsistencies under the semantics of the domain, captured as a logic program. Therefore, we still require techniques for detecting and resolving inconsistencies in an ontology $\mathcal{O}$ under a logic program $\mathcal{P}$—that can incorporate both ontology language semantics and domain-specific constraints—in order to extract a maximal consistent sub-ontology $\mathcal{O}' \prec \mathcal{O}$ w.r.t. $\mathcal{P}$. Thus in our work, we check the consistency of an ontology $\mathcal{O}$ under a logic program $\mathcal{P}$ by detecting any produced contradiction derivations under $\mathcal{P}$.

Considering the Example 6.1, the ontology $\mathcal{O}_1$ is found inconsistent because two contradiction derivations were generated under the logic program $\mathcal{P}'$. The first contradiction derivation is due to the ontology language semantic rules of RDF(S) subsumption and OWL disjointness over the asserted concept axioms in $\mathcal{O}_1$. The second contradiction derivation is due to the domain-specific rule `{?F`

`:trackingStart ?S. ?F :trackingEnd ?E. ?S math:greaterThan ?E } ⇒ ⊥`, incorporating date and time consistency over tracking intervals of marine animals. Thus, our inconsistency detection approach is scalable to both ontology-language and domain-specific constraints, and can detect inconsistencies that violate both types of constraints.

6.4 Resolving Inconsistencies: Generating Minimal Inconsistent Resolve Candidates

In our approach, we resolve all the detected inconsistencies by the following stepwise process:

Step # 1 *(Identifying Minimal Contradictory Triple Sets)*: We first identify *minimal contradictory triple sets* (MCTSs) from all produced *contradiction derivations* under a logic program $\mathcal{P}$ (see Definition 20) .

Step # 2 *(Computing Asserted Ancestor Triple Sets)*: For each MCTS E_i, we compute its *asserted ancestor triple set* (AATS) S_i (see Definition 21) .

Step # 3 *(Computing Minimal Inconsistent Resolve Candidates)*: We generate all possible MIRCs $M_1, \ldots, M_n$ (see Definition 25), where removing any MIRC M_i from $\mathcal{O}$ will result in a maximal consistent sub-ontology $\mathcal{O}' \prec \mathcal{O}$ w.r.t. $\mathcal{P}$ (see Definition 24).

Step # 4 *(Generating MIRCs and its Descendants)*: To inform the user about the consequences of removing M_i from $\mathcal{O}$, we also provide a list of all its derived triples (see Definition 25).

In the following sub-sections, we will first describe the extraction of MCTSs and AATSs (i.e. Step # 1 and Step # 2) from the produced contradiction derivations under $\mathcal{P}$ in Euler generated proofs. Later, we will focus on Step # 3 and Step # 4 of the above-mentioned process, and describe our approach for generating all possible MIRCs from the AATSs of the produced contradiction derivations under $\mathcal{P}$.

6.4.1 Computing MCTSs and AATSs in Euler

We have implemented our inconsistency detection and resolution approach using N3 rules [117] that are reasoned in Euler [118]. Euler is an inference engine supporting logic based proofs based on Coherent Logic (CL) [182]. Due to CL expressivity, disjunctions are possible in a conclusion (i.e. head of a CL clause). In order to satisfy such disjunctions, Euler generates all *possible models* that hold under given set of Prolog CL formulae. It can also generate all the (possible-)models that leads to $\perp$, denoted as *false models* (i.e. contradiction derivations).

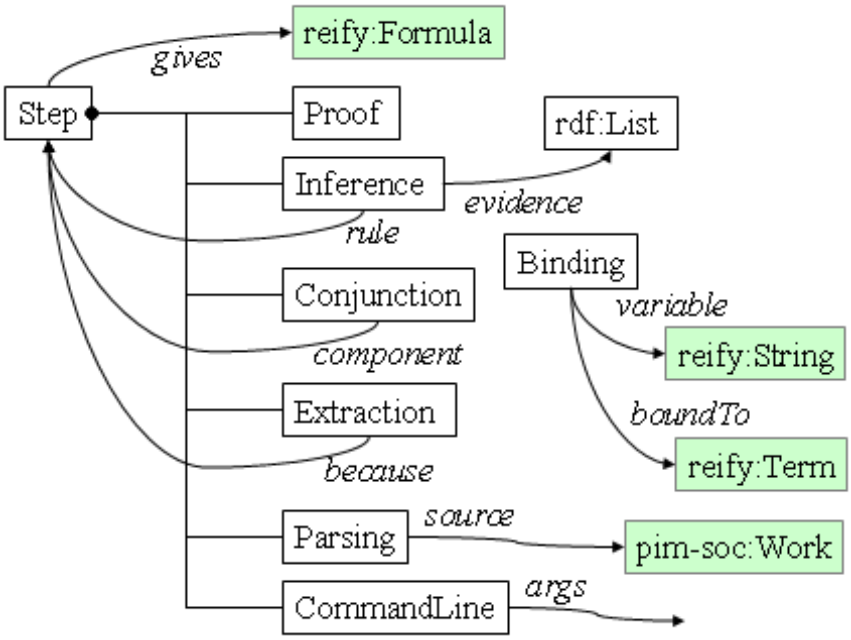

Figure 6.1: Structure of Reason Ontology—adopted from [183]

Euler uses the Cwm's reason ontology [183] to represent the proofs generated by Euler. An abstract structure of the reason ontology is shown in Figure 6.1. In order to compute the *asserted ancestor triple sets* (AATSs), we first parse the Euler-generated proof to extract all the *minimal contradictory triple sets* (MCTSs). Once all the MCTSs are extracted, the next step is to extract the AATS for each of the MCTSs. An AATS S_i for a MCTS E_i is extracted from the generated proof through a recursive parsing from all triples $t_j \in E_i$, appearing as instances of the concept `Inference` (in reason ontology; see Figure 6.1), to the triple sets that are connected with t_j by the property `evidence` and appearing as instances of the concept

`Extraction`. We apply the same parsing for all MCTSs and compute their AATSs.

6.4.2 Generating MIRCs and its Descendants

Our algorithm for computing MIRCs is shown in Algorithm 3, and an example trace of this algorithm is shown in Figure 6.2. Computing MIRCs requires the set of AATSs $\mathbb{S} = \{S_1, \ldots, S_n\}$ and a (indexed) set of triples $\mathbb{I} = \bigcup_{i=1}^{n} S_i = \{t_1, \ldots, t_m\}$ as inputs. For a candidate MIRC $M_i^j \subseteq \mathbb{I}$, the function $compute\_triple\_existence(M_i^j)$ calculates the set L_i^j of those S_i's that contain any $t \in M_i^j$, and implemented as follows:

$compute\_triple\_existence(M_i^j)\{$
$\mathbb{S}' = \emptyset;$
for all $t \in M_i^j$ **do**
  for all $S_i \in \mathbb{S}$ **do**
    if $t \in S_i$ **then**
      $\mathbb{S}'.add(S_i);$
    end if
  end for
end for
return $\mathbb{S}';$
$\}$

line 1: Declare $i = 1$, set of MIRCs $\mathbb{M} = \emptyset$, and non-MIRC sets $\mathbb{N}_1 = \emptyset \ldots \mathbb{N}_n = \emptyset$.

line 2-11: Construct singleton triple sets $M_1^1, \ldots, M_{m_1}^1$ from all triples in $\mathbb{I}$. For each singleton triple set M_i^j (where i denotes the index and j denotes the cardinality of M_i^j; for singletons $j = 1$):

line 3: Compute a candidate MIRC $M_i^j = \{t_i\}$, and also store the highest index of the triples appearing in M_i^j, $\eta_i^j = i$.

line 4: For M_i^j, compute $L_i^j = compute\_triple\_existence(M_i^j)$

line 5-6: In case $|L_i^j| = |\mathbb{S}|$ (i.e. M_i^j is an MIRC), therefore M_i^j is added into the set of MIRCs $\mathbb{M}$.

line 7-9: Otherwise, M_i^j is added into a non-MIRC set of singletons $\mathbb{N}_j$.

Algorithm 3 Computing Minimal Inconsistent Resolve Candidates (MIRCs)

Require: AATSs $\mathbb{S} = \{S_1, \ldots, S_n\}$ and $\mathbb{I} = \bigcup_{i=1}^{n} S_i = \{t_1, \ldots, t_m\}$

Ensure: Minimal Inconsistent Resolve Candidates $\mathbb{M}$

1: $\mathbb{M} = \emptyset$; $i = 1$; $\mathbb{N}_1 = \emptyset$; $\ldots$ $\mathbb{N}_n = \emptyset$;
2: **for** $i = 1$ to m **do**
3: $M_i^1 = \{t_i\}$; $\eta_i^1 = i$;
4: $L_i^1 = compute\_triple\_existence(M_i^1)$;
5: **if** $|L_i^1| = |\mathbb{S}|$ **then**
6: $\mathbb{M}.add(M_i^1)$;
7: **else**
8: $\mathbb{N}_1.add(M_i^1)$;
9: **end if**
10: $i = i + 1$;
11: **end for**
12: **for** $j = 2$ to n **do**
13: $\mathbb{N}_j = \emptyset$; $i = 1$;
14: **for all** $M_l^{j-1} \in \mathbb{N}_{j-1}$ **do**
15: **for all** $M_x^1 \in \mathbb{N}_1$ **do**
16: **if** $\eta_x^1 > \eta_l^{j-1}$ **then**
17: $M_i^j = M_l^{j-1} \cup M_x^1$; $\eta_i^j = \eta_x^1$;
18: $L_i^j = compute\_triple\_existence(M_i^j)$;
19: **if** $|L_i^j| > |L_i^{j-1}|$ **then**
20: **if** $|L_i^j| = |\mathbb{S}|$ **then**
21: $\mathbb{M}.add(M_i^j)$;
22: **else**
23: $\mathbb{N}_j.add(M_i^j)$;
24: **end if**
25: **end if**
26: $i = i + 1$;
27: **end if**
28: **end for**
29: **end for**
30: **end for**
31: **return** $\mathbb{M}$;

In the example trace (shown in Figure 6.2), singleton triple sets M_1^1 and M_3^1 were found to be MIRCs (i.e. they appear in all the AATSs) and added into $\mathbb{M}$. Whereas all other singleton triple sets $M_2^1, M_4^1, \ldots, M_{m_1}^1$ were added into

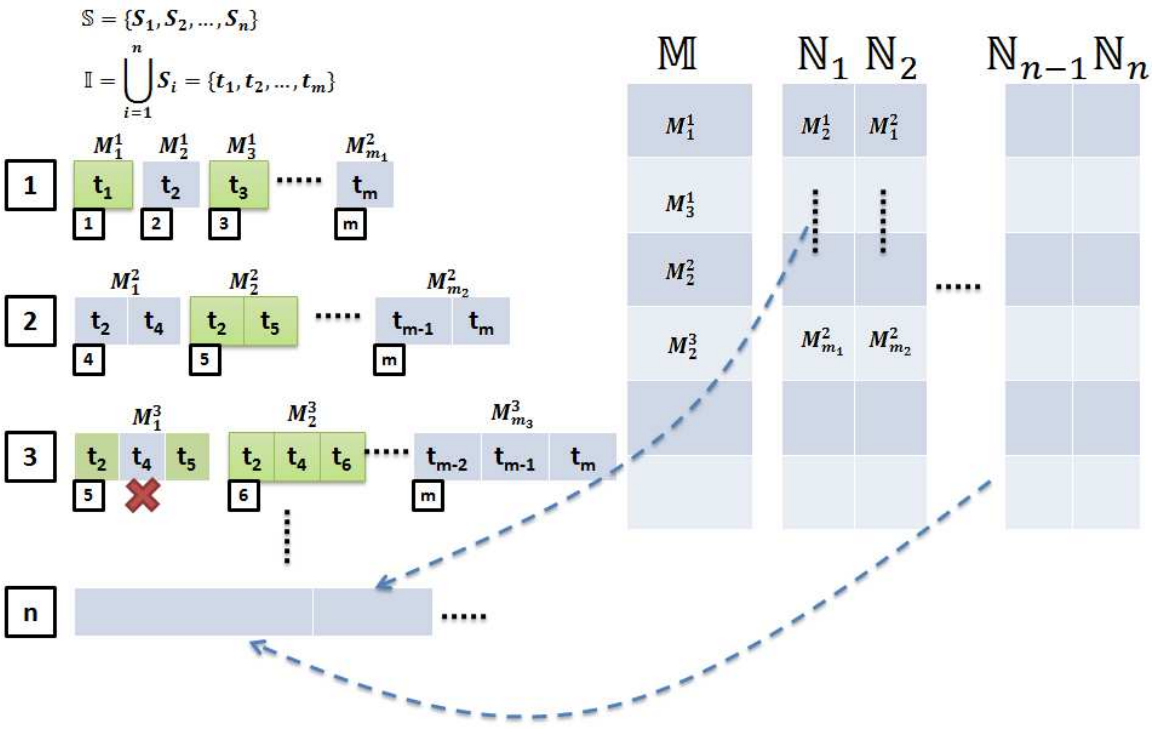

Figure 6.2: Computing Minimal Inconsistent Resolve Candidates

a non-MIRC singletons set $\mathbb{N}_1$.

line 12-30: Construct triple sets M_i^j of cardinality $j = 2$ to n, by successively adding singleton triple sets $M_x^1 \in \mathbb{N}_1$ (line 15-28) into each M_i^{j-1} (line 14-29).

line 14-29: For all $M_l^{j-1} \in \mathbb{N}_{j-1}$

line 15-28: For all $M_x^1 \in \mathbb{N}_1$

line 16-17: Compute a candidate MIRC $M_i^j = M_l^{j-1} \cup M_x^1$, and also store the highest index of the triples appearing in M_i^j, $\eta_i^j = \eta_x^1$

line 18: For M_i^j, compute $L_i^j = compute\_triple\_existence(M_i^j)$

line 19-21: In case $|L_i^j| = |\mathbb{S}|$, M_i^j is added into the set of MIRCs $\mathbb{M}$

line 22-25: Otherwise, M_i^j is added into an non-MIRCs set $\mathbb{N}_j$

In the example trace (shown in Figure 6.2), $M_2^2 = \{t_2, t_5\}$ is found to be an MIRC and added into $\mathbb{M}$; whereas all other triple sets (of cardinality 2) were added into a non-MIRC set $\mathbb{N}_2$. Similarly a triple set $M_2^3 = \{t_2, t_4, t_6\}$ (of cardinality 3) is computed by combining two non-MIRCs $M_1^2 = \{t_2, t_4\}$ and

$M_6^1 = \{t_6\}$. The triple set M_1^3 was found to be an MIRC and added into $\mathbb{M}$. The construction of M_i^j (where $j > 1$) using $\mathbb{N}_{j-1}$ ensures that IRCs containing smaller MIRCs are not computed; e.g. $M_1^3 = \{t_2, t_4, t_5\}$ is not computed since there exists an MIRC $M_2^2 \subset M_1^3$.

line 31: Return the computed MIRCs $\mathbb{M}$

6.5 Evaluation: Detecting and Resolving Inconsistencies in Prostate Cancer Ontologies

Here, we extended our prostate cancer (PC) test-case (see Section 3.3). During the reconciliation and merging of three extracted PC contextualized sub-ontologies, $\mathcal{K}$-$\mathcal{MORPH}$ found inconsistencies in the merged PC ontology $\mathcal{O}_{PC-Merge}$. In this section, we will demonstrate the detection and resolution of inconsistencies in the merged PC ontology.

Based on the pre-defined context-specific correspondences among three contextualized sub-ontologies (see Section 4.5), alignments were found between classes, including their properties and instances, for *Treatment*, *Followup*, *Frequency*, *Interval Duration*, and *Clinician* (using our TOM and POM; see Sections 5.4 and 5.5). Based on the found alignments shown in Figure 5.2 and also the context-specific alignments (defined in $\mathcal{C}_{x1}$; see Section 4.5), three extracted PC contextualized sub-ontologies were then merged to generate possible 'knowledge-links' between the aligned PC treatments. One of such context-specific correspondence given by the user (in $\mathcal{C}_{x1}$; see Section 4.5) is `calgary:Treatment rdfs:subClassOf winnipeg:FollowUp`, stating that treatments modelled in the Calgary PC ontology can be realized as Follow-ups in the Winnipeg PC ontology. However, on the other hand in the Winnipeg PC ontology, `winnipeg:Treatment` and `winnipeg:FollowUp` are defined as disjoint concepts. Also, at the same time, the `Treatment` concept is found to be equivalent across all three PC ontologies (see Figure 5.2). Hence in this case, the combination of the given context-specific alignments and found alignments caused inconsistencies in the merged PC ontology $\mathcal{O}_{PC-Merge}$, due to the following kinds of contradictions:

Case # 1:

Align # 1: Halifax `Treatment` is equivalent to Calgary `Treatment`

Therefore: Halifax treatments are realized as Calgary treatments.

Align # 3: Calgary `Treatment` is sub-class of Winnipeg `winnipeg:FollowUp`

Therefore: Halifax treatments are also realized as Winnipeg follow-ups.

Case # 2:

Align # 2: Halifax `Treatment` is equivalent to Winnipeg `Treatment`

Therefore: Halifax treatments are realized as Winnipeg treatments.

Inconsistencies: Case #1 $\wedge$ Case # 2 $\vdash_{\mathcal{P}_{\mathcal{PC}}} \bot$.
Contradiction: Halifax treatments can not be realized as both Winnipeg treatments and Winnipeg follow-ups; because Winnipeg treatments and Winnipeg follow-ups are defined as disjoint concepts!

Example 6.2 demonstrates the working of our approach for detecting and resolving inconsistencies in the merged PC ontology $\mathcal{O}_{PC-Merge}$ by (i) finding contradiction derivations of the above discussed forms, and then (ii) generating MIRCs for resolving all the detected inconsistencies in $\mathcal{O}_{PC-Merge}$. A relevant fragment of alignments is shown in the *Alignments* section of Example 6.2; whereas the logic program $\mathcal{P}_{PC}$—under which inferences were made—is shown in the *Logic Program* section. Based on the alignments and the source ontologies, found MCTSs are shown in the *Minimal Contradictory Triple Sets* section, and computed AATSs are shown in the *Asserted Ancestor Triple Sets* section.

Example 6.2 (Detecting and Resolving Inconsistencies in PC Ontologies):

Alignments:
1) halifax:Treatment owl:equivalentClass calgary:Treatment.
2) halifax:Treatment owl:equivalentClass winnipeg:Treatment.
3) calgary:Treatment rdfs:subClassOf winnipeg:FollowUp.

Winnipeg Ontology:
winnipeg:Treatment owl:disjointWith winnipeg:FollowUp.

Logic Program $\mathcal{P}_{PC}$:

{?A owl:equivalentClass ?B. ?X a ?A} ⇒ {?X a ?B}.

{?C rdfs:subClassOf ?D. ?X a ?C} ⇒ {?X a ?D}.

{?C rdfs:subClassOf ?D. ?D rdfs:subClassOf ?E} ⇒ {?C rdfs:subClassOf ?E}.

{?X a ?A. ?X a ?B. ?A owl:disjointWith ?B} ⇒ false.

Minimal Contradictory Triple Sets (MCTSs):

E_1 = {halifax:ActiveSurveillance a winnipeg:Treatment, halifax:ActiveSurveillance a winnipeg:FollowUp, winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}.

E_2 = {halifax:ExternalBeamRadiationTherapy a winnipeg:Treatment, halifax:ExternalBeamRadiationTherapy a winnipeg:FollowUp, winnipeg:Treat- ment owl:disjointWith winnipeg:FollowUp}.

E_3 = {halifax:Surgery a winnipeg:Treatment, halifax:Surgery a winnipeg:FollowUp, winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}.

Asserted Ancestor Triple Sets (AATSs):

$\mathbb{S}_1$ = {:Treatment owl:equivalentClass winnipeg:Treatment, :ActiveSurveillance a halifax:Treatment, calgary:Treatment rdfs:sub-
ClassOf winnipeg:FollowUp, :Treatment owl:equivalentClass calgary:Treatment, winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}.

$\mathbb{S}_2$ = {:Treatment owl:equivalentClass winnipeg:Treatment, :ExternalBeamRadiationTherapy a halifax:Treatment, calgary:Treatment rdfs:subClassOf winnipeg:FollowUp, :Treatment owl:equivalentClass calgary:Treatment, winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}.

$\mathbb{S}_3$ = {:Treatment owl:equivalentClass winnipeg:Treatment, :Surgery a halifax:Treatment, calgary:Treatment rdfs:subClassOf winnipeg:FollowUp, :Treatment owl:equivalentClass calgary:Treatment, winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}.

Generated MIRCs ($M_1, \ldots, M_5 << M_1, \ldots, \ldots, M_{57}$):

Generated MIRC: M_1 = {halifax:Treatment owl:equivalentClass winnipeg:Treatment}.

MIRC-Descendants (M_1) = {halifax:ActiveSurveillance a winnipeg:Treatment, halifax:ExternalBeamRadiationTherapy a winnipeg:Treatment, halifax:Surgery a winnipeg:Treatment}

numberofTriplesRemove = 4

Generated MIRC: M_2 = {calgary:Treatment rdfs:subClassOf winnipeg:FollowUp}.

MIRC-Descendants (M_2) = {halifax:ActiveSurveillance a winnipeg:FollowUp, halifax:ExternalBeamRadiationTherapy a winnipeg:FollowUp, halifax:Surgery a winnipeg:FollowUp}

numberofTriplesRemove = 4

Generated MIRC: M_3 = {halifax:Treatment owl:equivalentClass calgary:Treatment}

MIRC-Descendants (M_3) = {halifax:ActiveSurveillance a winnipeg:FollowUp, halifax:ActiveSurveillance a calgary:Treatment, halifax:ExternalBeamRadiationTherapy a winnipeg:FollowUp, halifax:ExternalBeamRadiationTherapy a calgary:Treatment, halifax:Surgery a winnipeg:FollowUp, halifax:Surgery a calgary:Treatment}

numberofTriplesRemove = 7

Generated MIRC: M_4 = {winnipeg:Treatment owl:disjointWith winnipeg:FollowUp}

MIRC-Descendants (M_4) = $\emptyset$

numberofTriplesRemove = 1

Generated MIRC: M_5 = {halifax:Surgery a halifax:Treatment, halifax:ExternalBeamRadiationTherapy a halifax:Treatment, halifax:ActiveSurveillance a halifax:Treatment}
MIRC-Descendants (M_5) = {halifax:Surgery a winnipeg:Treatment, halifax:Surgery a winnipeg:FollowUp, halifax:Surgery a calgary:Treatment, halifax:ExternalBeamRadiationTherapy a winnipeg:Treatment, halifax:ExternalBeamRadiationTherapy a winnipeg:FollowUp, halifax:ExternalBeamRadiationTherapy a calgary:Treatment, halifax:ActiveSurveillance a winnipeg:Treatment, halifax:ActiveSurveillance a winnipeg:FollowUp, halifax:ActiveSurveillance a calgary:Treatment}
numberofTriplesRemove = 12

Three unique AATSs were computed (shown in the *Asserted Ancestor Triple Sets* section) from the contradiction derivations under $\mathcal{P}_{PC}$, where each AATS consists of 5 asserted triples from the merged PC ontology $\mathcal{O}_{PC-Merge}$. There are in total 57 possible *inconsistent resolve candidates*, where removing any of these candidates from $\mathcal{O}_{PC-Merge}$ results in a consistent sub-ontology; but not necessarily a maximal consistent sub-ontology. However using our algorithm (see Algorithm 3), we computed only 5 MIRCs (shown in the *Minimal Inconsistent Resolve Candidates* section in Example 6.2). These are in fact all possible MIRCs, where removing any of these minimal candidates from the merged PC ontology $\mathcal{O}_{PC-Merge}$ results in a maximal consistent sub-ontology of $\mathcal{O}_{PC-Merge}$ w.r.t. $\mathcal{P}_{PC}$. For each extracted MIRC M_i, we generated the list of its derived triples and also provided the total number of (asserted and their derived) triples, which will be removed in the maximal consistent sub-ontology by removing M_i from $\mathcal{O}_{PC-Merge}$ (as shown in Example 6.2). Based on the cardinality of M_i, the cardinality of its derived triples set and the total number of triples removed for each MIRC M_i, the computed MIRCs can be ordered and shown to the users. Such an ordering can be helpful for the users in selecting an MIRC suitable for user's needs. Thus, the user-selected MIRC can then be removed from an inconsistent ontology to extract a maximal consistent sub-ontology.

6.6 Summary

Detecting and resolving inconsistencies is a non-trivial task, as it requires an in-depth understanding of the ontology axioms—in order to select axioms (either manually or automatically), which are either to be removed or repaired for resolving the identified inconsistencies. We have presented our approach for detecting and resolving inconsistencies in ontologies that (i) detects inconsistencies by finding *contradiction derivations* produced under a given logic program; and

(ii) generates *minimal inconsistent resolve candidates* MIRCs, where removing any of the MIRCs from the inconsistent ontology results in a maximal consistent sub-ontology w.r.t. the logic program. We evaluated our approach on the prostate cancer ontology [62], where during our knowledge morphing process we detected all the inconsistencies in this ontology and generated all possible MIRCs for extracting a maximal consistent sub-ontology.

Chapter 7

K-MORPH Evaluation and Results: Morphing Healthcare Ontologies for Generating Therapeutic Knowledge about Urinary Tract Infections

Clinical decision making involves an active interplay between various medical knowledge sources, and enables various reasoning strategies to achieve solutions for a clinical problem [3]. Medical knowledge sources can be categorized as (a) the tacit knowledge of a practitioner in terms of problem solving skills, judgment and intuition [5]; (b) clinical experiences (both recorded and observed) and lessons learnt [5, 6]; (c) collaborative problem solving discussions or consultations between practitioners; (d) published medical literature and clinical practice guidelines [4, 30]; (e) operational knowledge in terms of clinical protocols and pathways [62, 184]; (f) medical education content for practitioners and patients; (g) social networks eliciting members of a community of practice and their communication patterns, interests and maybe even expertise; and (h) data-mediated knowledge based on data of clinical observations, diagnostic tests and therapeutic treatments, recorded in medical records and stored in clinical data-warehouses [87].

A practitioner's decision for providing his/her clinical advice on a disease-specific scenario is carried out via different reasoning strategies using both strong and weak evidence-based medical knowledge [3]. The importance of integrating multiple medical knowledge sources can be realized in cases when a clinical solution from one medical knowledge source is lacking, or another knowledge source can play a role in deriving alternative solutions. For instance, in the absence of explicit algorithms described in a clinical practice guideline [3, 4], practitioners may need to refer to other modalities of knowledge, such as previously recommended cases and/or the expertise of domain experts recorded in problem solving scenarios [5]. For efficient and flexible decision making, we argue that the morphing of heterogeneous knowledge sources may provide an overall view of all knowledge

pertaining to a specific problem for identifying which solution will work, why it will work, and how to make it work.

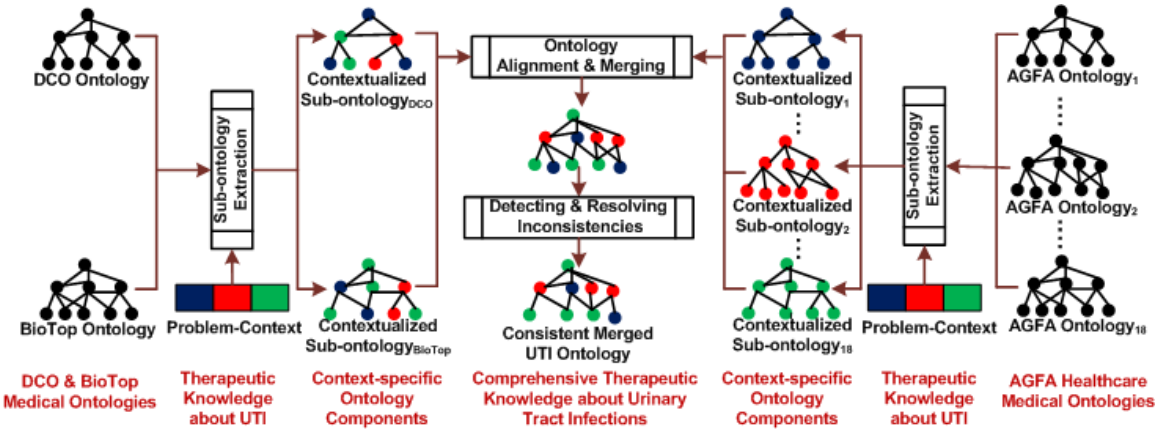

Figure 7.1: $\mathcal{K}$-$\mathcal{MORPH}$ Evaluation: Therapeutic Knowledge about Urinary Tract Infections

In this chapter, we will demonstrate the use of our $\mathcal{K}$-$\mathcal{MORPH}$ framework in the healthcare domain by showing the morphing of various high-level medical ontologies to obtain a comprehensive and networked knowledge-base for generating treatment plans for urinary tract infections (UTI). The overall morphing activity is shown in Figure 7.1. In order to realize this test-case, within our $\mathcal{K}$-$\mathcal{MORPH}$ framework, we consider 20 medical ontologies developed by three groups of medical experts from three different institutions. Hence, these ontologies can be classified into three categories: (i) BioTop Ontology, (ii) DCO Ontology, and (iii) Agfa Ontologies (18 ontologies). The desired objectives of this experiment are listed as follows:

1. Identifying and extracting context-specific fragments (i.e. knowledge dealing with UTI treatments) from the DCO, BioTop and Agfa ontologies.

2. Establishing semantic interoperability between context-specific fragments from the DCO/BioTop and Agfa ontologies.

3. Generating a comprehensive knowledge-base for providing therapeutic knowledge about UTIs.

In order to fulfill the above listed objectives in $\mathcal{K}$-$\mathcal{MORPH}$, we first extracted 20 contextualized sub-ontologies from the initial source ontologies based on the

user-defined problem-context *Therapeutic Knowledge about Urinary Tract Infections*. $\mathcal{K}$-$\mathcal{MORPH}$ also validated the conceptual/contextual consistency and completeness of the extracted contextualized sub-ontologies. Based on the provided context-specific alignments and constraints (in the problem-context), we applied our matchers TOM and POM, and found new correspondence between the extracted DCO/BioTop and Agfa sub-ontologies. Based on the identified correspondence, $\mathcal{K}$-$\mathcal{MORPH}$ aligned the extracted sub-ontologies and generated a merged ontology to provide a networked medical knowledge-base for generating treatment plans for UTIs. This test-case is explained in further details in the following sections.

7.1 Source Ontologies

In this experiment, we consider 20 medical ontologies developed by three groups of medical experts from three different institutions. Hence, these ontologies can be classified into three categories: (i) BioTop Ontology, (ii) DCO Ontology, and (iii) Agfa Ontologies (18 ontologies). These 20 ontologies provide knowledge about prognosis, diagnosis, treatment and follow-up for various infectious diseases. These ontologies are briefly described as follows:

7.1.1 Biological Top Level Ontology

The Biological Top Level (BioTop) ontology [185] is an upper domain ontology for molecular biology. It provides an interface to a selected set of Open Biomedical Ontologies (OBO)[1], which contains more detailed terminological knowledge about specific areas of molecular biology; e.g. cell types, molecular functions, biological processes and chemical compounds.

BioTop has extended the original structure of the GENIA ontology[2] for providing an ontologically sound layer for linking and integrating various domain-specific ontologies from the Life Sciences domain. In BioTop, some of the GENIA classes were removed and new BioTop classes were introduced, which significantly extended the scope of the original ontology. Instead of reusing GENIA's

[1] http://obo.sourceforge.net/

[2] http://www-tsujii.is.s.u-tokyo.ac.jp/~genia/topics/Corpus/genia-ontology.html

top-level distinction between *source* and *substance*, the general top-level ontology BFO [186] was set on top of BioTop. The BioTop ontology contains a total of 175 classes arranged in class hierarchies, 171 relations between classes arranged in 9 semantic hierarchies, and 171 restrictions.

7.1.2 DebugIT Core Ontology

The DebugIT Core Ontology (DCO) is an application ontology that enables data miners to query distributed clinical information systems in a semantically-rich and content-driven manner [187]. DCO serves as the core component of the interoperability platform for the DebugIT project [188] and covers the complete conceptual space of the domain of interest in the DebugIT project. The ontology provides a semantic model to formally represent all basic kinds of entities in the domain of interest, together with their invariant and context-independent properties. DCO ontology provides a *semantic glue* function, in order to standardize and formally describe meaning identifiers across the whole project—hence, solving interoperability issues among health institutions.

7.1.3 AGFA Ontologies

Researchers at Advanced Clinical Applications (ACA) Group, Agfa Healthcare are contributing actively in the DebugIT project [188] and have developed a series of healthcare ontologies. In order to fulfill the desired objectives of this experiment, the following Agfa ontologies are used as source ontologies in $\mathcal{K}$-$\mathcal{MORPH}$:

1. *Organism Ontology* models living entities, states they can be in, and roles they can play; e.g. the state 'pregnant'.

2. *Human Ontology* describes aspects of a human; e.g. gender, age related categories.

3. *HumanBody Ontology* represents general aspects of a human body; e.g. structure, weight.

4. *Agent Ontology* models agents and their roles in different actions; e.g. general property 'plays role' e.g. of a patient.

5. *HealthCare Ontology* provides general knowledge about the healthcare environment; e.g. patient, physician.
6. *ClinicalObservation Ontology* describes general clinical findings; e.g. pain, fever, etc.
7. *HumanDisorder Ontology* provides high-level description of human disorders; e.g. 'Lower Urinary Tract Disorder'
8. *InfectiousDisorder Ontology* describes different types of infections; e.g. viral, bacterial, vaginal, etc.
9. *UrinaryTractInfection Ontology* focuses on detailed descriptions of Urinary Tract Infections.
10. *ClinicalEvaluation Ontology* describes general clinical evaluative concepts; e.g. diagnosis, prognosis, etc.
11. *Therapy Ontology* represents general aspects of clinical therapy; e.g. intention 'curative', property 'treatedBy'.
12. *DrugTherapy Ontology* provides a general description about drug therapy; e.g. class 'dosage' and property 'hasDosage'.
13. *Antibiotics Ontology* models various kinds antibiotics applied in human medicine.
14. *AdministrationRoutes Ontology* describes routes/workflows for drug administration to a patient.
15. *ClinicalMorphology Ontology* provides descriptions about altered human morphology; e.g. 'Inflammation'.
16. *Anatomy Ontology* represents a high-level description of human anatomy; e.g. 'Lower Urinary Tract Structure'.
17. *ClinicalProcedure Ontology* models general clinical procedures; e.g. 'preventive' being an intention of a procedure.
18. *Event Ontology* models events, actions, and temporal relations; e.g. to describe the start and end of a UTI.

Table 7.1: Concepts and Alignments for Therapeutic Knowledge about UTI

BioTop Ontology	AGFA Ontologies	DebugIT Core Ontology
	Inflammation <==========>	InflammatoryMorphology
AcquiredAbnormalStructure <======>	MorphologicallyAbnormalStructure	
	Infection <==========>	InfectiousDisease
	Infector <==========>	InfectorRole
PathologicalState <==========>	Finding	
AcquiredPathologicalState <=======>	Disorder	
	UrinarySystemDisorder <======>	DisorderOfUrinaryTract
	LowerUrinaryTractDisorder	
	LowerUrinaryTractInfection	
OrganismPart <==========>	AnatomicalStructure	
	Cystitis <==========>	Cystitis
	UrinaryTractInfection <=======>	UrinaryTractInfection
	AutoimmuneDisorder <=======>	AutoimmuneDisorder
	ImmuneSystemDisorder <======>	DisorderOfImmuneFunction
	Patient <==========>	PatientRole
Human <==========>	Human	
TaxonQuality <==========>	Taxon	
SpeciesHomoSapiensRegion <======>	subspeciesHomoSapiensSapiens	
	Therapy <==========>	AntibioticTherapy
	AdministrationRoute <=======>	RouteOfAdministration
	Antibiotic <==========>	Antibiotic
	Amoxicillin <==========>	PortionOfMixture
	Fluoroquinolone <==========>	PortionOfMixture
AntibioticRole <==========>	AntibioticRole	
	Drug <==========>	PharmacologicSubstance
DrugRole <==========>	DrugRole	
AmountOfSubstance <==========>	PhysicalResource	
	Therapy <==========>	TherapeuticOrPreventiveProcedure
Action <==========>	Action <==========>	HealthCareActivity
ProcessualEntity <==========>	Process	
TemporalEntity <==========>	Event	
	Diagnosis <==========>	Diagnosis
	Diagnosing <==========>	Diagnosing
	DiagnosticRole <==========>	DiagnosticRole
PhysicianRole <==========>	Physician	
Role <==========>	Role	
hasAgent <==========>	hasAgent	
agentIn <==========>	actsIn	
hasParticipant <==========>	hasAgent	
participatesIn <==========>	actsIn	
hasLocus <==========>	hasSite	
	hasSnapshot <==========>	hasDateTime
	hasDuration <==========>	hasDuration
	begins <==========>	hasStartDateTime
	ends <==========>	hasEndDateTime
	hasBirthDate <==========>	hasDateTimeOfBirth

7.2 Task #1: Defining Problem-context: Therapeutic Knowledge about UTI

Based on the desired objectives of the experiment, the user selected the problem-specific concepts and properties from the source ontologies (DCO, BioTop and Agfa). In addition to providing problem-specific concepts and properties, the user

also specified context-specific alignments between the selected concepts and properties. The problem-context data-structure (see Definition 2) represented the user-specified concepts, properties and alignments in $\mathcal{K}$-$\mathcal{MORPH}$ (see Section 3.2.2). For the given problem-context *Therapeutic Knowledge about Urinary Tract Infections*, Table 7.1 shows the user-selected concepts and the user-defined context-specific alignments between them. Context-specific alignments between Agfa and DCO/ BioTop ontologies are highlighted by <===>.

7.3 Task # 2: Extracting Contextualized UTI Sub-ontologies

In order to extract contextualized sub-ontologies, we used our structure-based extraction method (see Chapter 4). Based on the user-selected concepts (as shown in Table 7.1) defined in the problem-context, we obtained a contextualized sub-ontology from each of the 20 source ontologies. Hence $\mathcal{K}$-$\mathcal{MORPH}$ extracted 20 contextualized sub-ontologies from the initially given source ontologies. One of the extracted sub-ontologies is shown in Figure 7.2. The *Infection Ontology* $\mathcal{O}_{INF}$ (one of the AGFA ontologies) is shown on the left side of Figure 7.2. Based on the user-selected concept `LowerUrinaryTractInfection`, the concept-hierarchy from $\mathcal{O}_{INF}$ extracted in the sub-ontology for UTI $\mathcal{O}'_{UTI}$ is shown on the right side. Using our structure-based ontology extraction method (see Chapter 4), $\mathcal{K}$-$\mathcal{MORPH}$ extracted only the concept `LowerUrinaryTractInfection` and its related concepts into the generated sub-ontology $\mathcal{O}'_{UTI}$. Although, in the right side of Figure 7.2, we only show the extracted concept-hierarchy in $\mathcal{O}'_{UTI}$. However, we also extracted properties, property-hierarchies and individuals relevant to `LowerUrinaryTract-Infection` in $\mathcal{O}'_{UTI}$. All the extracted sub-ontologies were then treated as contextualized sub-ontologies in $\mathcal{K}$-$\mathcal{MORPH}$, and were reconciled to generate the final merged ontology pertinent to the problem-context *Therapeutic Knowledge about Urinary Tract Infections*.

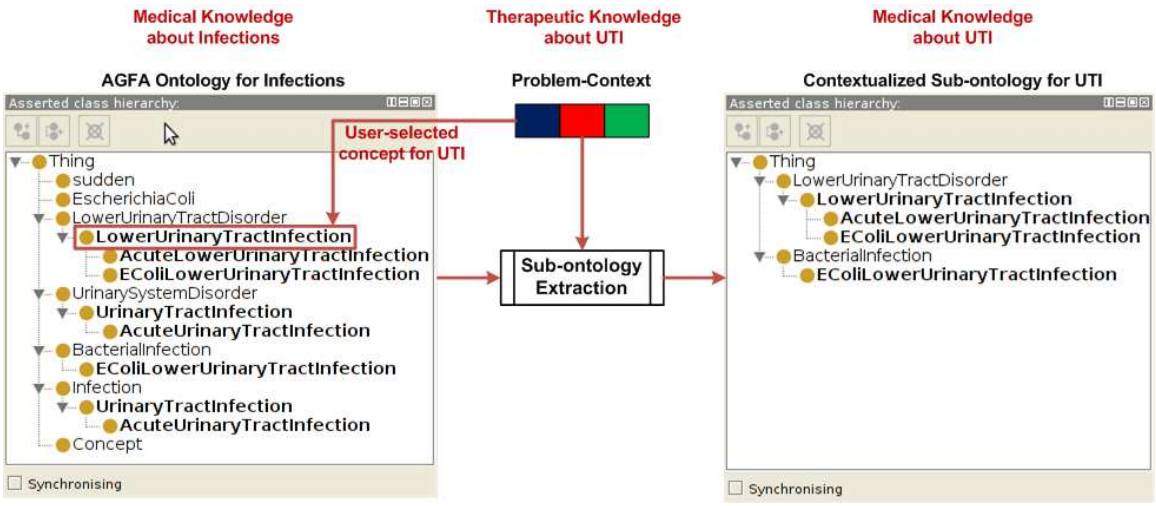

Figure 7.2: Contextualized UTI Sub-ontology Extraction

7.4 Task # 3: Aligning and Merging Contextualized Sub-ontologies

Based on the defined context-specific alignments in the given problem-context (as shown in Table 7.1), the next steps to be performed by $\mathcal{K}$-$\mathcal{MORPH}$ were: (i) finding new alignments between the extracted DCO/BioTop and Agfa contextualized sub-ontologies, and then (ii) merging those sub-ontologies to generate a merged ontology that provides a comprehensive therapeutic knowledge about UTIs. For finding new alignments between the extracted sub-ontologies, we used two of our matchers TOM and POM (see Chapter 5) and found similarities between both atomic and complex ontology-entities. Detected similarities by our matchers are shown in Tables 7.4, 7.5, 7.6 and 7.7.

We tested both of our matchers on various thresholds. In Tables 7.2 and 7.3, each row provides the number of correct and incorrect correspondences found between ontology-entities based on different similarity thresholds. In Table 7.2, the first 3 rows show the accuracy results of TOM on atomic entities, and the last 3 rows show the accuracy of complex correspondences found under different thresholds. Similarly in Table 7.3, the first 3 rows present the accuracy results of POM on atomic entities, and the last 3 rows present the accuracy of complex correspondences found under different thresholds.

Table 7.2: TOM Matching Results: Correspondence Accuracy

Entities	Threshold	Total	Evaluated	Correct	Incorrect	Accuracy
Atomic	0.5	146	146	145	1	99.31%
Atomic	0.6	51	51	51	0	100%
Atomic	0.7	4	4	4	0	100%
Complex	0.4	62	57	36	21	63.15%
Complex	0.5	59	54	36	18	66.66%
Complex	0.6	1	1	1	0	100%

As shown in Tables 7.2 and 7.3, TOM and POM were able to find non-trivial correspondences between both atomic and complex entities. Due to the fact that complex correspondences were missing in the initial alignments (see Table 7.1), complex correspondences generated by TOM and POM cannot be compared against a gold standard, resulting in missing recall values. Also the initially given alignment (see Table 7.1) provides some of the desired correspondences between the source ontologies (DCO/BioTop and Agfa ontologies); but not all possible correspondences between those ontologies. Therefore investigating the specificity and sensitivity of our matching results on both atomic and complex correspondences was not possible. Even though it might be possible to construct a complete reference alignment for a finite number of entity structures, it will be extremely laborious to construct a complete reference alignment among large and complicated ontologies that contains all non-trivial atomic and complex correspondences.

Table 7.3: POM Matching Results: Correspondence Accuracy

Entities	Threshold	Total	Evaluated	Correct	Incorrect	Accuracy
Atomic	0.2	47	47	46	1	97.87%
Atomic	0.4	31	31	31	0	100%
Atomic	0.6	18	18	18	0	100%
Complex	0.6	116	49	48	1	97.95%
Complex	0.7	65	24	23	1	95.83%
Complex	0.8	37	16	16	0	100%

Our matching results were manually evaluated by the domain experts and the original ontology developers [185, 187]. For each correspondence found between two ontology-entities, the domain experts evaluated the correspondence and assigned a correspondence relation between those entities. Moreover, since some of the found correspondences were very complex for the domain experts, they were

Table 7.4: TOM Matching Results: Atomic Correspondences

DCO & BioTop Ontologies	Relation	Agfa Ontologies	Score
DisorderOfUrinaryTract	$\sqsupseteq$	Cystitis	1.0.
Cystitis	$\sqsubseteq$	UrinarySystemDisorder	1.0.
DisorderOfImmuneFunction	$\sqsupseteq$	AutoimmuneDisorder	1.0.
AutoimmuneDisorder	$\sqsubseteq$	ImmuneSystemDisorder	1.0.
RouteOfAdministration	$\sqsupseteq$	Oral	0.666.
RouteOfAdministration	$\sqsupseteq$	Intravenous	0.666.
RouteOfAdministration	$\sqsupseteq$	Subcutaneous	0.666.
Chloramphenicol	$\sqsubseteq$	Antibiotic	0.625.
Clofazimine	$\sqsubseteq$	Antibiotic	0.625.
Colistin	$\sqsubseteq$	Antibiotic	0.625.
FusidicAcid	$\sqsubseteq$	Antibiotic	0.625.
Linezolid	$\sqsubseteq$	Antibiotic	0.625.
Nitroxoline	$\sqsubseteq$	Antibiotic	0.625.
MyastheniaGravis	$\sqsubseteq$	ImmuneSystemDisorder	0.6.
MyastheniaGravis	$\sqsubseteq$	AutoimmuneDisorder	0.6.
Amphotericin	$\sqsubseteq$	Antibiotic	0.6.
Dalfopristin	$\sqsubseteq$	Antibiotic	0.6.
Fosfomycin	$\sqsubseteq$	Antibiotic	0.6.
NitrofuranDerivative	$\sqsubseteq$	Antibiotic	0.6.
Nitroimidazole	$\sqsubseteq$	Antibiotic	0.6.
Polymyxin	$\sqsubseteq$	Antibiotic	0.6.
Quinupristin	$\sqsubseteq$	Antibiotic	0.6.
Amikacin	$\sqsubseteq$	Antibiotic	0.6.
AmphotericinB	$\sqsubseteq$	Antibiotic	0.6.
Azithromycin	$\sqsubseteq$	Antibiotic	0.6.
HealthCareActivity	$\sqsupseteq$	HealthCaring	0.6.
PresumptiveDiagnosis	$\sqsubseteq$	Diagnosis	0.6.
PrimaryDiagnosis	$\sqsubseteq$	Diagnosis	0.6.
SecondaryDiagnosis	$\sqsubseteq$	Diagnosis	0.6.
VerifiedDiagnosis	$\sqsubseteq$	Diagnosis	0.6.

$\equiv$: `owl:equivalentClass`; $\equiv_P$: `owl:equivalentProperty`; $\sqsubseteq$: `rdf:subClassOf`; `simThreshold = 0.6`.

able to evaluate only a fragment of the total correspondences. For example as shown in Table 7.2, row 4, there were total 62 correspondences found between the complex entities. Out of 62 complex correspondences, domain experts were able to evaluate 57, and found 36 correct and 21 incorrect. Although finding and evaluating complex correspondences are challenging tasks [176], our matchers TOM and POM successfully found some of the non-trivial correspondences between fairly complicated and complex entities (see Tables 7.6 and 7.7).

Tables 7.4 and 7.5 show a similarity score between atomic entities from Agfa ontologies and DCO/BioTop ontologies, calculated by matchers TOM and POM,

Table 7.5: POM Matching Results: Atomic Correspondences

Agfa Ontologies	Relation	DCO & BioTop Ontologies	Score
AcquiredStructure	$\sqsupseteq$	AcquiredAbnormalStructure	0.6
Drug	$\sqsupseteq$	Antibiotic	0.875
AntiInfectiveSubstance	$\sqsupseteq$	Drug	0.7142
Drug	$\sqsubseteq$	AmountOfSubstance	0.618
Drug	$\sqsubseteq$	AmountOfSubstanceByHomogenityPartition	0.618
Drug	$\sqsubseteq$	PortionOfMonosubstance	0.628
Action	$\equiv$	Action	0.642
Evaluation	$\sqsubseteq$	IntellectualProduct	0.75
Evaluating	$\sqsubseteq$	BiologicalAction	0.625
Action	$\sqsubseteq$	BiologicalAction	0.625
Evaluating	$\sqsubseteq$	OrganismAction	0.625
Action	$\sqsubseteq$	OrganismAction	0.625
Evaluating	$\sqsubseteq$	HumanAction	1.0
Action	$\sqsubseteq$	HumanAction	1.0
ns3:infector1	$=$	i:infectorRole2	0.6
ns3:patient1	$=$	i:patientRole2	0.611
ns3:abTherapy1	$=$	i:antibioticTherapy2	0.616
ns3:diagnosticRole1	$=$	i:diagnosticRole2	0.625

$\equiv$: `owl:equivalentClass`; $\sqsubseteq$: `rdf:subClassOf`; $=$: `owl:sameAs`
`simThreshold = 0.6`.

and also present a correspondence relation (assigned by the domain experts) between those entities. The matching results generated from TOM, based on a given similarity threshold of 0.7, are shown in Table 7.4. Similarly, the results generated from POM, based on a given similarity threshold of 0.6, are shown in Table 7.5.

Tables 7.6 and 7.7 show a similarity score between complex entities from Agfa ontologies and DCO/BioTop ontologies, calculated by matchers TOM and POM, and also present a correspondence relation (assigned by the domain experts) between those entities. Since the generated alignments consisted of fairly complex correspondences, we present only some of the prominent correspondences in Tables 7.6 and 7.7.

In Table 7.6, line 1 shows an equivalence relation between (i) the concept `Infection` (in Agfa ontologies) and (ii) any arbitrary concept that has an equivalence relation with DCO concept `InfectiousDisease` as well as a subsumption relation with another DCO concept `AcquiredPathologicalState`. Therefore, any instance of `Infection` in Agfa ontologies can be treated as an `InfectiousDisease` (or `AcquiredPathologicalState`) in the DCO/BioTop ontologies, and vice versa.

Hence we have obtained extensive knowledge about UTIs, such as `Infection` is an `InfectiousDisease` and also an `AcquiredPathologicalState`. Another correspondence on line 7 shows a subsumption relation between a DCO/BioTop concept `Cystitis` and a property restricted Agfa concept. Based on this correspondence, all instances of `Cystitis` can also be realized in the Agfa ontologies as diseases that are found in `LowerUrinaryTractStructure`. Hence due to such a correspondence, we have obtained extended knowledge about Cystitis, such as `Cystitis` is one of the diseases of `LowerUrinaryTractStructure`. Similarly, based on the correspondence on line 8, we have gathered more knowledge about Cystitis, such as `Cystitis` is one of the diseases of `Urinary_bladder`. Moreover, correspondences on lines 9 and 10 extend the knowledge about Antibiotic Therapies, such as `AntibioticTherapy` plays an `AntibioticRole`, and `AntibioticTherapy` requires `Administering`. Lines 2-6 and 11 present correspondences of a similar nature.

Table 7.7 presents some of the prominent complex correspondences found by POM, whereby POM found more complex correspondences as compared to TOM (see Table 7.6). Each of the correspondences generated by TOM (in Table 7.6) presents a relation between a complex and an atomic entity, whereas POM generated correspondences between two complex entities. For example, Table 7.7 line 1 presents an equivalence relation between complex concepts from Agfa and DCO/BioTop ontologies. Based on this correspondence, all kinds of urinary bladder diseases modelled in Agfa ontologies are also realized as urinary bladder diseases in the DCO/BioTop ontologies, and vice versa. Similarly, based on the correspondence in line 7, any antibiotic treatment that plays an `AntibioticRole`, shall also be the bearer of `PharmacologicSubstanceIntakeDose`. Lines 2-6 and 8-10 present correspondences of a similar nature.

7.5 Task # 4: Detecting and Resolving Inconsistencies

During the reconciliation of the extracted contextualized sub-ontologies (see Section 7.4), $\mathcal{K}$-$\mathcal{MORPH}$ found inconsistencies in the merged UTI ontology $\mathcal{O}_{UTI-Merge}$ using our inconsistency detection approach. In order to resolve the detected inconsistencies in $\mathcal{O}_{UTI-Merge}$, $\mathcal{K}$-$\mathcal{MORPH}$ generated all possible MIRCs (see Chapter

Table 7.6: TOM Matching Results: Complex Correspondences

	Agfa Ontologies	Rel	DCO & BioTop Ontologies	Score
1	Infection	$\equiv$	[$\sqsubseteq$ AcquiredPathologicalState $\equiv$ InfectiousDisease]	0.5230
2	Antibiotic	$\equiv$	[$\sqsubseteq$ AntiInfectiveSubstance $\equiv$ Antibiotic]	0.6111
3	Infector	$\equiv$	[On_P bearerOf V_{some} InfectorRole]	0.5833
4	[On_P rolePlayedBy V_{some} Infector]	$\equiv$	InfectorRole	0.5714
5	[On_P hasSite V_{some} UrinarySystemStructure]	$\sqsupseteq$	DisorderOfUrinaryTract	0.5833
6	[On_P hasSite V_{some} GenitourinarySystemStructure]	$\sqsupseteq$	DisorderOfUrinaryTract	0.5714
7	[On_P hasSite V_{some} LowerUrinaryTractStructure]	$\sqsupseteq$	Cystitis	0.55
8	[On_P hasSite V_{some} Urinary_bladder]	$\sqsupseteq$	Cystitis	0.5714
9	[On_P hasSubProcedure V_{some} Administering]	$\equiv$	AntibioticTherapy	0.5714
10	[On_P hasPlayedRole V_{some} AntibioticRole]	$\equiv$	AntibioticTherapy	0.5714
11	[On_P after V_{some} Evaluating]	$\sqsupseteq$	Diagnosing	0.5714

$\equiv$: `owl:equivalentClass`; $\sqsubseteq$: `rdf:subClassOf`; On_P: `owl:onProperty`; V_{some}: `owl:someValuesFrom`

6), where removing any MIRC from $\mathcal{O}_{UTI-Merge}$ will result in a maximal consistent sub-ontology.

Based on the context-specific correspondences among AGFA and DCO/BioTop ontologies (see Table 7.1), new alignments were found using our TOM and POM matching approaches (see Tables 7.4 and 7.5). Based on the found alignments and also the context-specific correspondences, the extracted contextualized sub-ontologies were then merged to generate possible 'knowledge-links' between the aligned UTI treatments. Two such context-specific correspondences given by the user are `antibio:Fluoroquinolone owl:equivalentClass dco:PortionOfMixture` and `antibio:Amoxicillin owl:equivalentClass dco:PortionOfMixture`, stating that *Fluoroquinolone* and *Amoxicillin* drugs should be realized as a *Portion of Mixture*. Also, drugs are modelled as a kind of *Portion of Monosubstances*. On the other hand, in DCO ontology, `dco:PortionOfMixture` and `PortionOfMonosubstance` are

Table 7.7: POM Matching Results: Complex Correspondences

	Agfa Ontologies	Rel	DCO & BioTop Ontologies	Score
1	[On_P hasSite V_{some} Urinary_bladder]	$\equiv$	[On_P hasLocus V_{some} UrinaryBladder]	1.0
2	[On_P hasSubProcedure V_{some} Administering]	$\sqsubseteq$	[On_P bearerOf V_{some} PharmacologicSubstanceIntakeDose]	1.0
3	Drug	$\sqsupseteq$	[$\sqsubseteq$ AntiInfectiveSubstance $\equiv$ Antibiotic]	0.9166
4	[On_P hasPlayedRole V_{some} Professional]	$\sqsubseteq$	HumanAction	1.0
5	Action	$\sqsupseteq$	[On_P precededBy V_{some} PatientAdmissionActivity]	1.0
6	[On_P hasPlayedRole V_{some} Professional]	$\sqsupseteq$	[On_P precededBy V_{some} PatientAdmissionActivity]	1.0
7	[On_P hasPlayedRole V_{some} AntibioticRole]	$\sqsubseteq$	[On_P bearerOf V_{some} PharmacologicSubstanceIntakeDose]	1.0
8	[On_P hasSubProcedure V_{some} Administering]	$\sqsubseteq$	[On_P bearerOf V_{some} RouteOfAdministration]	1.0
9	[On_P hasSubProcedure V_{some} Administering]	$\equiv$	[On_P hasParticipant V_{some} Antibiotic]	1.0
10	[On_P hasPlayedRole V_{some} AntibioticRole]	$\equiv$	[On_P hasParticipant V_{some} Antibiotic]	1.0

$\equiv$: `owl:equivalentClass`; $\sqsubseteq$: `rdf:subClassOf`; On_P: `owl:onProperty`; V_{some}: `owl:someValuesFrom`

defined as disjoint concepts. Hence in this case, the combination of the given context-specific correspondences and found alignments caused inconsistencies in the merged UTI ontology $\mathcal{O}_{UTI-Merge}$ due to the following kinds of contradictions:

Case # 1: `Fluoroquinolone`

Agfa # 1: Agfa `Fluoroquinolone` is a kind of Agfa `Antibiotic`

Therefore: Fluoroquinolone is realized as Antibiotics.

Agfa # 3: Agfa `Antibiotic` is a kind of Agfa `Drug`

Therefore: Fluoroquinolone is realized as Drugs.

Align # i: Agfa `Drug` is a kind of DCO `PortionOfMonosubstance`

Therefore: Fluoroquinolone is realized as Portion of Monosubstances.

Align # ii: Agfa `Fluoroquinolone` is equivalent to DCO `PortionOfMixture`

Therefore: Fluoroquinolone is realized as both *Portion of Monosubstances* & *Portion Of Mixture*.

DCO # 1: `PortionOfMonosubstance` & `PortionOfMixture` are disjoint DCO concepts

Contradiction # 1: Fluoroquinolone can not be realized as both *Portion of Monosubstances* & *Portion Of Mixture*!

Case # 2: `Amoxicillin`

Analogous to Case # 1:

Contradiction # 2: Amoxicillin can not be realized as both *Portion of Monosubstances* & *Portion Of Mixture*!

Inconsistencies: Merged UTI ontology is inconsistent due to contradiction # 1 and # 2.

Example 7.1 demonstrates the working of our approach for detecting and resolving inconsistencies in the merged UTI ontology $\mathcal{O}_{UTI-Merge}$ in $\mathcal{K}$-$\mathcal{MORPH}$ by (i) finding contradiction derivations of the above discussed forms, and then (ii) generating MIRCs for resolving all the detected inconsistencies in $\mathcal{O}_{UTI-Merge}$. A relevant fragment of alignments is shown in the *Alignments* section of Example 7.1, whereas the logic program $\mathcal{P}_{UTI}$—under which inferences were made—is shown in the *Logic Program* section. Based on the alignments and the source ontologies, found MCTSs are shown in the *Minimal Contradictory Triple Sets* section, and computed AATSs are shown in the *Asserted Ancestor Triple Sets* section.

Example 7.1 (Detecting and Resolving Inconsistencies in UTI Ontologies):

Agfa Ontology:

1) antibio:Fluoroquinolone rdfs:subClassOf antibio:Antibiotic.
2) antibio:Amoxicillin rdfs:subClassOf antibio:Antibiotic.
3) antibio:Antibiotic rdfs:subClassOf drug:Drug.

DCO Ontology:

1) :PortionOfMixture owl:disjointWith :PortionOfMonosubstance.

Alignments: Agfa $\langle$====$\rangle$ DCO

(i) drug:Drug rdfs:subClassOf :PortionOfMonosubstance.
(ii) antibio:Fluoroquinolone owl:equivalentClass :PortionOfMixture.
(iii) antibio:Amoxicillin owl:equivalentClass :PortionOfMixture.

Logic Program $\mathcal{P}_{UTI}$:

{?A owl:equivalentClass ?B. ?X a ?A} $\Rightarrow$ {?X a ?B}.
{?A owl:equivalentClass ?B} $\Rightarrow$ {?B owl:equivalentClass ?A}.
{?A owl:equivalentClass ?B. ?B owl:equivalentClass ?C} $\Rightarrow$ {?A owl:equivalentClass ?C}.

{?C rdfs:subClassOf ?D. ?X a ?C} $\Rightarrow$ {?X a ?D}.

{?C rdfs:subClassOf ?D. ?D rdfs:subClassOf ?E} ⇒ {?C rdfs:subClassOf ?E}.
{?X a ?A. ?X a ?B. ?A owl:disjointWith ?B} ⇒ false.

Minimal Contradictory Triple Sets (MCTSs):

E_1 = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, i:coTrimoxazole2 a :PortionOfMixture, i:coTrimoxazole2 a :PortionOfMonosubstance}.

E_2 = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:amoxicillin1 a :PortionOfMixture, ns3:amoxicillin1 a :PortionOfMonosubstance}.

E_3 = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:fluoroquinolone1 a :PortionOfMixture, ns3:fluoroquinolone1 a :PortionOfMonosubstance}.

E_4 = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:ciprofloxacin1 a :PortionOfMixture, ns3:ciprofloxacin1 a :PortionOfMonosubstance}.

Asserted Ancestor Triple Sets (AATSs):

$\mathbb{S}_1$ = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, i:coTrimoxazole2 a :PortionOfMixture, drug:Drug rdfs:subClassOf :PortionOfMonosubstance, antibio:Fluoroquinolone rdfs:subClassOf drug:Drug, antibio:Fluoroquinolone owl:equivalentClass :PortionOfMixture}.

$\mathbb{S}_2$ = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:amoxicillin1 a antibio:Amoxicillin, antibio:Amoxicillin owl:equivalentClass :PortionOfMixture, drug:Drug rdfs:subClassOf :PortionOfMonosubstance, antibio:Antibiotic rdfs:subClassOf drug:Drug, ns3:amoxicillin1 a antibio:Antibiotic}.

$\mathbb{S}_3$ = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:fluoroquinolone1 a antibio:Fluoroquinolone, antibio:Fluoroquinolone owl:equivalentClass :PortionOfMixture, drug:Drug rdfs:subClassOf :PortionOfMonosubstance, antibio:Antibiotic rdfs:subClassOf drug:Drug, ns3:fluoroquinolone1 a antibio:Antibiotic}.

$\mathbb{S}_4$ = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance, ns3:ciprofloxacin1 a antibio:Fluoroquinolone, antibio:Fluoroquinolone owl:equivalentClass :PortionOfMixture, drug:Drug rdfs:subClassOf :PortionOfMonosubstance, antibio:Antibiotic rdfs:subClassOf drug:Drug, ns3:ciprofloxacin1 a antibio:Antibiotic}.

Generated MIRCs ($M_1, \ldots, M_{32} << M_1, \ldots, \ldots, M_{578}$):

Generated MIRC: M_1 = {:PortionOfMixture owl:disjointWith :PortionOfMonosubstance}.

MIRC-Descendants (M_1) = $\emptyset$

numberofTriplesRemove = 1

Generated MIRC: M_2 = {drug:Drug rdfs:subClassOf :PortionOfMonosubstance}.

MIRC-Descendants (M_2) = {ns3:amoxicillin1 a :PortionOfMonosubstance, ns3:fluoroquinolone1 a :PortionOfMonosubstance, ns3:ciprofloxacin1 a :PortionOfMonosubstance, i:coTrimoxazole2 a :PortionOfMonosubstance}

numberofTriplesRemove = 5

Generated MIRC: M_3 = {antibio:Fluoroquinolone owl:equivalentClass :PortionOfMixture, antibio:Amoxicillin owl:equivalentClass :PortionOfMixture}

MIRC-Descendants (M_3) = {ns3:fluoroquinolone1 a :PortionOfMixture, antibio:Fluoroquinolone rdfs:subClassOf :PortionOfMixture, ns3:ciprofloxacin1 a :PortionOfMixture, i:coTrimoxazole2 a :PortionOfMonosubstance, i:coTrimoxazole2 a drug:Drug, i:coTrimoxazole2 a antibio:Fluoroquinolone, :PortionOfMixture rdfs:subClassOf antibio:Fluoroquinolone, :PortionOfMixture owl:equivalentClass antibio:Fluoroquinolone, ns3:amoxicillin1 a :PortionOfMixture, antibio:Amoxicillin rdfs:subClassOf :PortionOfMixture}

numberofTriplesRemove = 12

⋮ ⋮ ⋮ ⋮

Generated MIRC: M_{32} = {i:coTrimoxazole2 a :PortionOfMixture, ns3:ciprofloxacin1 a antibio:Antibiotic, ns3:fluoroquinolone1 a antibio:Antibiotic, ns3:amoxicillin1 a antibio:Antibiotic}

MIRC-Descendants (M_{32}) = {i:coTrimoxazole2 a :PortionOfMonosubstance, i:coTrimoxazole2 a drug:Drug, i:coTrimoxazole2 a antibio:Fluoroquinolone, ns3:ciprofloxacin1 a :PortionOfMonosubstance, ns3:ciprofloxacin1 a drug:Drug, ns3:fluoroquinolone1 a :PortionOfMonosubstance, ns3:fluoroquinolone1 a drug:Drug, ns3:amoxicillin1 a :PortionOfMonosubstance, ns3:amoxicillin1 a drug:Drug}

numberofTriplesRemove = 13

There were 4 unique AATSs computed (shown in the *Asserted Ancestor Triple Sets* section) from the contradiction derivations under $\mathcal{P}_{UTI}$, where each AATS consists of 6 asserted triples from the merged UTI ontology $\mathcal{O}_{UTI-Merge}$. There are 578 possible *inconsistent resolve candidates*, where removing any of these candidates from $\mathcal{O}_{UTI-Merge}$ results in a consistent sub-ontology, but not necessarily a maximal consistent sub-ontology. However using our algorithm (see Algorithm 3), we computed only 32 MIRCs (shown in the *Minimal Inconsistent Resolve Candidates* section in Example 7.1). These are in fact all possible MIRCs, where removing any of these minimal candidates from the merged UTI ontology $\mathcal{O}_{UTI-Merge}$ results in a maximal consistent sub-ontology of $\mathcal{O}_{UTI-Merge}$ w.r.t. $\mathcal{P}_{UTI}$. For each extracted MIRC M_i, we generated a list of its derived triples and also provided the total number of (asserted and their derived) triples that will be removed in the maximal consistent sub-ontology by removing M_i from $\mathcal{O}_{UTI-Merge}$ (as shown in Example 7.1). Based on the cardinality of M_i, cardinality of its derived triples set and the total number of triples removed for each MIRC M_i, the computed MIRCs can be ordered and shown to the users. Such an ordering can be helpful for the users in selecting an MIRC suitable for user's needs. The user-selected MIRC can then be removed from an inconsistent ontology to extract a maximal consistent sub-ontology.

7.6 Results: Morphed Therapeutic Knowledge Generated by $\mathcal{K}$-$\mathcal{MORPH}$

Based on the correspondences found by our matchers TOM and POM (see Section 7.4), $\mathcal{K}$-$\mathcal{MORPH}$ merged the Agfa and DCO/BioTop contextualized sub-ontologies to generate a merged ontology that can provide comprehensive therapeutic knowledge about UTIs. Based on the found correspondences, there were $39,317$ new inferred relations (between Agfa and DCO/BioTop ontologies) found

in the generated merged ontology. Hence the generated merged ontology provided the combined and extended therapeutic knowledge about UTIs by inferring $39,317$ new knowledge links that were not initially present in the source ontologies.

While obtaining the merged ontology, inferred relations were generated from both atomic and complex correspondences. Since explaining and understanding all inferred results generated from the complex correspondences are seemingly difficult, in this section we will focus only on atomic correspondences and present some of their inferred results (i.e. new therapeutic knowledge) generated by $\mathcal{K}$-$\mathcal{MORPH}$.

Table 7.8: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 1

Correspondence	**Inferred Knowledge**
event:Evaluating $\sqsubseteq$ biotop:BiologicalAction	clineva:Diagnosing $\sqsubseteq$ biotop:BiologicalAction clineva:Evaluating $\sqsubseteq$ biotop:BiologicalAction clineva:Prognosing $\sqsubseteq$ biotop:BiologicalAction ns3:diagnosing1 a biotop:BiologicalAction

Case # 1: `event:Evaluating` $\sqsubseteq$ `biotop:BiologicalAction`

In this case (see Table 7.8), given the correspondence that `event:Evaluating` (in Agfa ontologies) is a `biotop:BiologicalAction` (in DCO/BioTop ontologies), therefore all related kinds of clinical actions modelled in Agfa ontologies are now known as a `biotop:BiologicalAction`. As a result, an instance `diagnosing1` (of the concept `Diagnosing`) is also an instance of `biotop:BiologicalAction`.

Table 7.9: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 2

Correspondence	**Inferred Knowledge**
heca:Action $\sqsubseteq$ biotop:BiologicalAction	heca:HealthCaring $\sqsubseteq$ biotop:BiologicalAction heca:Encounter $\sqsubseteq$ biotop:BiologicalAction heca:Action $\sqsubseteq$ biotop:Action heca:HealthCaring $\sqsubseteq$ biotop:Action heca:Encounter $\sqsubseteq$ biotop:Action ns3:hcAction1 a biotop:Action ns3:hcAction1 a biotop:BiologicalAction

Case # 2: heca:Action ⊑ biotop:BiologicalAction

Similarly, based on the correspondence heca:Action ⊑ biotop:BiologicalAction, various kinds of healthcare actions (modelled in Agfa ontologies) now can be realized as a biotop:BiologicalAction (see Table 7.9).

Table 7.10: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 3

Correspondence	Inferred Knowledge
event:Evaluation ⊑ biotop:Immaterial-NonphysicalEntity	clineva:Diagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:PrimaryDiagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:SecondaryDiagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:PresumptiveDiagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:CertainDiagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:Evaluation ⊑ biotop:ImmaterialNonphysicalEntity clineva:AbsentDiagnosis ⊑ biotop:ImmaterialNonphysicalEntity clineva:Prognosis ⊑ biotop:ImmaterialNonphysicalEntity event:whereinExamined rdfs:domain biotop:ImmaterialNonphysicalEntity event:whereinEvaluated rdfs:domain biotop:ImmaterialNonphysicalEntity ns3:diagnosis1 a biotop:ImmaterialNonphysicalEntity ns3:evaluation1 a biotop:ImmaterialNonphysicalEntity

Case # 3: event:Evaluation ⊑ biotop:ImmaterialNonphysicalEntity

In this case (see Table 7.10), given event:Evaluation ⊑ biotop:Immaterial-NonphysicalEntity, various kind of diagnostic steps (modelled in Agfa ontologies) can now be extended as biotop:ImmaterialNonphysicalEntity. As a result, properties modelled in Agfa ontologies event:whereinExamined and event:whereinEvaluated are now became the properties of biotop:Immaterial-NonphysicalEntity. Also all instances of diagnosis and evaluation are now also treated as instances of biotop:ImmaterialNonphysicalEntity.

Case # 4: event:Evaluation ⊑ biotop:IntellectualProduct

Similarly, based on the correspondence event:Evaluation ⊑ biotop:IntellectualProduct, various kinds of diagnostic steps (modelled in Agfa ontologies) can now be extended as biotop:IntellectualProduct (see Table 7.11). As a result, properties modelled in Agfa ontologies event:whereinExamined and event:whereinEvaluated now become the properties of biotop:Intellectual-

Table 7.11: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 4

Correspondence	Inferred Knowledge
event:Evaluation ⊑ biotop:Intellectual-Product	clineva:Diagnosis ⊑ biotop:IntellectualProduct clineva:PrimaryDiagnosis ⊑ biotop:IntellectualProduct clineva:SecondaryDiagnosis ⊑ biotop:IntellectualProduct clineva:PresumptiveDiagnosis ⊑ biotop:IntellectualProduct clineva:CertainDiagnosis ⊑ biotop:IntellectualProduct clineva:Evaluation ⊑ biotop:IntellectualProduct clineva:AbsentDiagnosis ⊑ biotop:IntellectualProduct clineva:Prognosis ⊑ biotop:IntellectualProduct event:whereinExamined rdfs:domain biotop:IntellectualProduct event:whereinEvaluated rdfs:domain biotop:IntellectualProduct event:hasEvaluationTimeStamp rdfs:domain biotop:IntellectualProduct event:evaluatedIn rdfs:range biotop:IntellectualProduct ns3:diagnosis1 a biotop:IntellectualProduct ns3:evaluation1 a biotop:IntellectualProduct

`Product`. Also all instances of diagnosis and evaluation are now also treated as instances of `biotop:IntellectualProduct`.

Table 7.12: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 5

Correspondence	Inferred Knowledge
dco:Cystitis ⊑ dis:UrinarySystemDisorder	dco:Cystitis ⊑ dis:Disorder. dco:Cystitis ⊑ clinobs:Finding. dco:Cystitis ⊑ event:AdverseProcess. dco:Cystitis ⊑ event:Action. dco:Cystitis ⊑ event:Finding. dco:Cystitis ⊑ event:Event. dco:Cystitis ⊑ event:Process. i:cystitis2 a dis:UrinarySystemDisorder. i:cystitis2 a dis:Disorder. i:cystitis2 a clinobs:Finding. i:cystitis2 a event:AdverseProcess. i:cystitis2 a event:Action. i:cystitis2 a event:Event.

Case # 5: `dco:Cystitis` ⊑ `dis:UrinarySystemDisorder`

In this case (see Table 7.12), based on the correspondence `dco:Cystitis` ⊑ `dis:UrinarySystemDisorder`, a DCO concept `dco:Cystitis` can be further classified and related with other concepts in Agfa ontologies. Hence, now `dco:Cystitis` can be seen as a `dis:UrinarySystemDisorder`, `dis:Disorder`, `clinobs:Finding`, `event:AdverseProcess`, `event:Action`, `event:Finding`, `event:Event`, or

event:Process. As a result, all instances of dco:Cystitis now became instances of the above-mentioned concepts.

Table 7.13: *K-MORPH* Results: Case # 6

Correspondence	Inferred Knowledge
dco:AutoimmuneDisorder ⊑ dis:ImmuneSystemDisorder	dco:MyastheniaGravis ⊑ dis:ImmuneSystemDisorder dco:AutoimmuneDisorder ⊑ dis:Disorder dco:AutoimmuneDisorder ⊑ clinobs:Finding dco:AutoimmuneDisorder ⊑ event:AdverseProcess dco:AutoimmuneDisorder ⊑ event:Action dco:AutoimmuneDisorder ⊑ event:Finding dco:AutoimmuneDisorder ⊑ event:Event dco:AutoimmuneDisorder ⊑ event:Process dco:MyastheniaGravis ⊑ dis:Disorder dco:MyastheniaGravis ⊑ clinobs:Finding dco:MyastheniaGravis ⊑ event:AdverseProcess dco:MyastheniaGravis ⊑ event:Action dco:MyastheniaGravis ⊑ event:Finding dco:MyastheniaGravis ⊑ event:Event dco:MyastheniaGravis ⊑ event:Process i:autoimmuneDisorder2 a dis:ImmuneSystemDisorder i:autoimmuneDisorder2 a dis:Disorder i:autoimmuneDisorder2 a clinobs:Finding i:autoimmuneDisorder2 a event:AdverseProcess i:autoimmuneDisorder2 a event:Action i:autoimmuneDisorder2 a event:Event

Case # 6: dco:AutoimmuneDisorder ⊑ dis:ImmuneSystemDisorder

Similarly, in this case (see Table 7.13), based on the correspondence dco:AutoimmuneDisorder ⊑ dis:ImmuneSystemDisorder, various kinds of immune disorders (modelled in DCO/BioTop ontologies) can be further classified and related with other concepts in Agfa ontologies. Hence, now DCO immune disorders can be seen as a dis:UrinarySystemDisorder, dis:Disorder, clinobs:Finding, event:AdverseProcess, event:Action, event:Finding, event:Event, or event:Process. As a result, all instances of DCO immune disorders now became instances of the above-mentioned concepts.

Case # 7: ther:Therapy ⊑ dco:TherapeuticOrPreventiveProcedure

Based on the correspondence ther:Therapy ⊑ dco:TherapeuticOrPreventiveProcedure, following inferences were generated (see Table 7.14):

Table 7.14: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 7

Correspondence	**Inferred Knowledge**
ther:Therapy ⊑ dco:TherapeuticOr-PreventiveProcedure	ther:CausalTherapy ⊑ dco:TherapeuticOrPreventiveProcedure ther:NonCausalTherapy ⊑ dco:TherapeuticOrPreventiveProcedure ther:BlindTherapy ⊑ dco:TherapeuticOrPreventiveProcedure ther:EmpiricalTherapy ⊑ dco:TherapeuticOrPreventiveProcedure ther:SurgicalProcedure ⊑ dco:TherapeuticOrPreventiveProcedure ther:EvacuationProcedure ⊑ dco:TherapeuticOrPreventiveProcedure ther:DrainageProcedure ⊑ dco:TherapeuticOrPreventiveProcedure ther:Therapy ⊑ dco:HealthCareActivity ther:Therapy ⊑ biotop:Action ther:hasIntent rdfs:domain dco:TherapeuticOrPreventiveProcedure ther:replaces rdfs:domain dco:TherapeuticOrPreventiveProcedure ther:addedTo rdfs:domain dco:TherapeuticOrPreventiveProcedure ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ther:replaces rdfs:range dco:TherapeuticOrPreventiveProcedure ther:addedTo rdfs:range dco:TherapeuticOrPreventiveProcedure ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ther:CausalTherapy ⊑ dco:HealthCareActivity ther:CausalTherapy ⊑ biotop:Action ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ns3:therapy1 a dco:TherapeuticOrPreventiveProcedure ns3:therapy1 a dco:HealthCareActivity ns3:therapy1 a biotop:Action

1. Various therapies modelled in the Agfa ontologies, such `ther:CausalTherapy`, `ther:NonCausalTherapy`, `ther:BlindTherapy`, `ther:EmpiricalTherapy`, `ther:SurgicalProcedure`, `ther:EvacuationProcedure` and `ther:DrainageProcedure`, can now be classified under `dco:TherapeuticOrPreventiveProcedure`.

2. Therapies can now be referred as `dco:HealthCareActivity` and `biotop:Action`.

3. Properties related to therapies, such as `ther:hasIntent`, `ther:replaces`, `ther:addedTo`, now also deal with `dco:TherapeuticOrPreventiveProcedure`.

Table 7.15: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 8

Correspondence	**Inferred Knowledge**
dco:AntiInfectiveSubstance $\sqsubseteq$ drug:Drug	dco:AntimycoticSubstance $\sqsubseteq$ drug:Drug dco:ImmuneSerum $\sqsubseteq$ drug:Drug dco:Immunoglobulin $\sqsubseteq$ drug:Drug dco:Vaccine $\sqsubseteq$ drug:Drug dco:BacterialVaccine $\sqsubseteq$ drug:Drug dco:Caspofungin $\sqsubseteq$ drug:Drug dco:DiphteriaVaccine $\sqsubseteq$ drug:Drug dco:Fluconazole $\sqsubseteq$ drug:Drug dco:Flucytosine $\sqsubseteq$ drug:Drug dco:HelminthiasisVaccine $\sqsubseteq$ drug:Drug dco:Itraconazole $\sqsubseteq$ drug:Drug $\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots$ dco:ImmuneSerum $\sqsubseteq$ ther:Substance dco:ImmuneSerum $\sqsubseteq$ biotop:AmountOfSubstance dco:ImmuneSerum $\sqsubseteq$ dco:PortionOfMonosubstance dco:ImmuneSerum $\sqsubseteq$ biotop:CollectiveMaterialEntity dco:ImmuneSerum $\sqsubseteq$ biotop:MaterialEntity $\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots$

Case # 8: `dco:AntiInfectiveSubstance` $\sqsubseteq$ `drug:Drug`

In this case (see Table 7.15), based on the correspondence `dco:AntiInfectiveSubstance` $\sqsubseteq$ `drug:Drug`, following inferences were generated:

1. Various kinds of anti-infective substances, such as `dco:AntimycoticSubstance`, `dco:ImmuneSerum`, `dco:Immunoglobulin`, `dco:Vaccine`, `dco:BacterialVaccine`, `dco:Caspofungin`, `dco:DiphteriaVaccine`, `dco:Fluconazole`, `dco:Flucytosine`, `dco:HelminthiasisVaccine`, `dco:Itraconazole`, are now classified under `drug:Drug`.

2. All kinds of anti-infective substances are also further classified under other related kinds of substances and material entities, such as `ther:Substance`, `biotop:AmountOfSubstance`, `dco:PortionOfMonosubstance`, `biotop:CollectiveMaterialEntity`, `biotop:MaterialEntity`.

Case # 9: `drug:Drug` $\sqsubseteq$ `biotop:AmountOfSubstance`

Based on the correspondence `drug:Drug` $\sqsubseteq$ `biotop:AmountOfSubstance`, following inferences were generated (see Table 7.16):

Table 7.16: *K-MORPH* Results: Case # 9

Correspondence	**Inferred Knowledge**
drug:Drug ⊑ biotop:AmountOfSubstance	antibio:Antibiotic ⊑ biotop:AmountOfSubstance antibio:Fluoroquinolone ⊑ biotop:AmountOfSubstance antibio:Ciprofloxacin ⊑ biotop:AmountOfSubstance antibio:Trimethoprim ⊑ biotop:AmountOfSubstance antibio:Sulfamethoxazole ⊑ biotop:AmountOfSubstance antibio:Benzylpenicillin ⊑ biotop:AmountOfSubstance antibio:Amoxicillin ⊑ biotop:AmountOfSubstance ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ dco:AntiInfectiveSubstance ⊑ biotop:AmountOfSubstance ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ drug:hasGenericName rdfs:domain biotop:AmountOfSubstance drug:hasStrength rdfs:domain biotop:AmountOfSubstance drug:activePartOf rdfs:domain biotop:AmountOfSubstance drug:affects rdfs:domain biotop:AmountOfSubstance drug:affects rdfs:range biotop:AmountOfSubstance drug:hasActivePart rdfs:range biotop:AmountOfSubstance ns3:drug1 a biotop:AmountOfSubstance ns3:antibiotic1 a biotop:AmountOfSubstance ns3:fluoroquinolone1 a biotop:AmountOfSubstance ns3:ciprofloxacin1 a biotop:AmountOfSubstance ns3:trimethoprim1 a biotop:AmountOfSubstance ns3:sulfamethoxazole1 a biotop:AmountOfSubstance ns3:benzylpenicillin1 a biotop:AmountOfSubstance ns3:amoxicillin1 a biotop:AmountOfSubstance

1. Various kind of drugs modelled in Agfa ontologies, such as `antibio:Antibiotic`, `antibio:Fluoroquinolone`, `antibio:Ciprofloxacin`, `antibio:Trimethoprim`, `antibio:Sulfamethoxazole`, `antibio:Benzylpenicillin`, `antibio:Amoxicillin`, etc, are now classified under a high-level concept `biotop:AmountOfSubstance`.

2. Some of the DCO drugs, such as `dco:AntiInfectiveSubstance`, are also classified as a `biotop:AmountOfSubstance`.

3. Properties associated with a drug, such as `drug:hasGenericName`, `drug:hasStrength`, `drug:activePartOf`, `drug:affects` and `drug:hasActivePart`, now also incorporate `biotop:AmountOfSubstance`.

4. Specific instances of various drugs are now also realized as a `biotop:AmountOfSubstance`.

Case # 10: `drug:Drug` ⊑ `biotop:CollectiveMaterialEntity`

Similar to case # 9, based on the correspondence `drug:Drug` ⊑ `biotop:CollectiveMaterialEntity`, following inferences were generated (see Table 7.17):

1. Various kind of drugs modelled in Agfa ontologies, such as `antibio:Antibiotic`, `antibio:Fluoroquinolone`, `antibio:Ciprofloxacin`, `antibio:Trimethoprim`, `antibio:Sulfamethoxazole`, `antibio:Benzylpenicillin`, `antibio:Amoxicillin`, etc, are now classified under a high-level concept `biotop:CollectiveMaterialEntity`.

2. Properties associated with a drug, such as `drug:hasGenericName`, `drug:hasStrength`, `drug:activePartOf`, `drug:affects` and `drug:hasActivePart`, now also incorporate `biotop:CollectiveMaterialEntity`.

3. Various drug instances are realized as `biotop:CollectiveMaterialEntity`.

Table 7.17: $\mathcal{K}$-$\mathcal{MORPH}$ Results: Case # 10

Correspondence	**Inferred Knowledge**
drug:Drug ⊑ biotop:CollectiveMaterialEntity	antibio:Antibiotic ⊑ biotop:CollectiveMaterialEntity antibio:Fluoroquinolone ⊑ biotop:CollectiveMaterialEntity antibio:Sulfamethoxazole ⊑ biotop:CollectiveMaterialEntity ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ dco:BacterialVaccine ⊑ biotop:CollectiveMaterialEntity dco:Caspofungin ⊑ biotop:CollectiveMaterialEntity ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ drug:hasStrength rdfs:domain biotop:CollectiveMaterialEntity drug:activePartOf rdfs:domain biotop:CollectiveMaterialEntity drug:affects rdfs:domain biotop:CollectiveMaterialEntity drug:affects rdfs:range biotop:CollectiveMaterialEntity drug:hasActivePart rdfs:range biotop:CollectiveMaterialEntity ns3:drug1 a biotop:CollectiveMaterialEntity ns3:antibiotic1 a biotop:CollectiveMaterialEntity ns3:fluoroquinolone1 a biotop:CollectiveMaterialEntity ns3:ciprofloxacin1 a biotop:CollectiveMaterialEntity ns3:trimethoprim1 a biotop:CollectiveMaterialEntity ns3:sulfamethoxazole1 a biotop:CollectiveMaterialEntity ns3:benzylpenicillin1 a biotop:CollectiveMaterialEntity ns3:amoxicillin1 a biotop:CollectiveMaterialEntity

7.7 Summary

Clinical decision making demands different reasoning strategies using both strong and weak evidence-based medical knowledge [3]. The importance of integrating multiple medical knowledge sources can be realized in cases when a clinical solution from one medical knowledge source is lacking, or another knowledge source can play a role in extending the initial knowledge to provide alternative solutions. In this experiment, we focused on the same objective and demonstrated the use of our $\mathcal{K}$-$\mathcal{MORPH}$ framework in generating a comprehensive therapeutic knowledge-base for UTIs. By morphing 20 healthcare ontologies in $\mathcal{K}$-$\mathcal{MORPH}$, we achieved a context-driven knowledge sharing and integration among those ontologies by establishing semantic interoperability between them. In this way, therapeutic UTI knowledge—that were separately modelled in the initial ontologies—are now shared and extended to provide knowledge about treatments and drug therapies to UTI patients.

Chapter 8

Conclusion and Future Work

A domain-specific problem is not often solved by an individual knowledge source, but rather requires an interplay between multiple knowledge sources [1]. Since the availability of complete knowledge, especially as one holistic knowledge object, is always challenging; therefore problem-solvers manually integrate knowledge from multiple sources to formulate a comprehensive knowledge object that satisfies the problem's context [2]. In order to obtain context-sensitive and comprehensive knowledge, a context-driven integration of heterogeneous knowledge sources is required to provide a comprehensive and networked view of all knowledge pertaining to a domain-specific problem at hand [12–26].

Knowledge morphing aims to formulate a comprehensive knowledge-base, specific to a given context, through "the intelligent and autonomous fusion/integration of contextually, conceptually and functionally related knowledge objects that may exist in different representation modalities and formalisms, in order to establish a comprehensive, multi-faceted and networked view of all knowledge pertaining to a domain-specific problem" Abidi 2005 [29]. Knowledge morphing extends the traditional notion of knowledge integration by providing the ability to reason over the morphed knowledge to (a) infer context-specific knowledge fragments, and (b) suggest recommendations and actions for solving domain-specific problems.

In this PhD thesis, we have presented our solution approach for Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies ($\mathcal{K}$-$\mathcal{MORPH}$) as shown in Figure 3.1. In $\mathcal{K}$-$\mathcal{MORPH}$, available knowledge sources are required to be modelled as ontologies. These ontology-modelled knowledge sources serve as inputs to $\mathcal{K}$-$\mathcal{MORPH}$. In addition to the source ontologies (i.e. initially modelled knowledge sources), $\mathcal{K}$-$\mathcal{MORPH}$ also requires the *problem-context* from the user as an input. To represent the user-intended context, a problem-context is a data-structure in which the user provides (i) the user-selected concepts (properties and

individuals) from the source ontologies that are pertinent to the context at hand; (ii) context-specific axioms and constraints to be applied on the source ontologies; and (iii) context-specific alignments between the source ontologies. Based on the given source ontologies and the problem-context from the user, $\mathcal{K}$-$\mathcal{MORPH}$ performs a context-driven reconciliation of source ontologies by: (a) extracting contextualized sub-ontologies from the source ontologies. The contextualized sub-ontologies get validated for conceptual/contextual consistency and completeness; (b) aligning and then merging the contextualized sub-ontologies to generate a merged ontology; and (c) detecting and resolving inconsistencies in the merged ontology. In this way, $\mathcal{K}$-$\mathcal{MORPH}$ pursues knowledge morphing by (i) extracting knowledge components from the available ontology-modelled knowledge sources pertinent to the given problem-context by extracting contextualized sub-ontologies; (ii) integrating the extracted knowledge components by merging contextualized sub-ontologies; and (iii) repairing inconsistencies in the morphed knowledge by detecting and resolving inconsistencies in the merged ontology. In this chapter, we will first assess the research contributions of this thesis and then highlight some of the applications and possible future directions for knowledge morphing.

8.1 Assessment of Research Contributions

$\mathcal{K}$-$\mathcal{MORPH}$ provides a framework for context-specific knowledge integration of ontology-modelled knowledge sources by a context-driven reconciliation of ontologies. The $\mathcal{K}$-$\mathcal{MORPH}$ framework is realized by an active interplay of three major research areas in Semantic Web: (i) extracting contextualized sub-ontologies, (ii) aligning and merging ontologies, and (iii) detecting and resolving inconsistencies in ontologies. There have been various attempts these above fields, whereby (a) various extraction methods were proposed to extract sub-ontologies from source ontologies [63–68]; (b) various ontology matching and alignment methods can align and merge ontologies [41, 42]; and (c) available approaches can detect and resolve inconsistencies in the merged ontology [69–72]. However, there are still challenges and limitations in the above-mentioned fields [13, 73]. Hence, in solving the knowledge morphing problem, we have also made scientific contributions in all three research areas (see Chapters 4, 5 and 6). In this section, we will assess

the main research contributions to these areas, made in realizing our $\mathcal{K}$-$\mathcal{MORPH}$ framework, by highlighting their unique features and addressing their limitations in solving the knowledge morphing problem.

8.1.1 Extracting Contextualized Sub-ontologies

In order to perform knowledge morphing using $\mathcal{K}$-$\mathcal{MORPH}$, we have developed a structure-based extraction method for extracting contextualized sub-ontologies (see Chapter 4). In contrast to existing structure-based methods [64–67], in our sub-ontology extraction approach, we restrict the recursive selection of the super-concepts of a selected concept C in order to avoid the unnecessary selection of the siblings and super-concepts of C, which would otherwise result in the expansion of the sub-graph to include concepts that are not relevant. In this way, we avoid a situation where the sub-ontology is overly generalized by the undesired inclusion of higher-level concepts that may even extend all the way to `owl:Thing` (see Chapter 4 for details).

In addition to existing structure-based extraction methods [64–67], there have been attempts towards sub-ontology extraction using logic-based approaches [68, 189]. Logic-based sub-ontology extraction aims to modularize a source ontology based on the semantics of its domain and ontology language. It allows the extraction of relevant and semantically consistent ontology modules from a source ontology [189]. The extracted modules (i.e. sub-ontologies) generated by the logic-based methods provide modular fragments from the source ontology and also their deducible consequences (i.e. deductively obtained axioms and assertions)—under domain-specific and ontology language semantics. Compared to logic-based extraction approaches, structure-based approaches lack the ability to generate semantically consistent and complete ontology modules. Although our proposed extraction method can generate relevant sub-ontologies to support the knowledge morphing problem, we believe that by applying and extending available logic-based approaches in $\mathcal{K}$-$\mathcal{MORPH}$, users can extract more complete and semantically consistent sub-ontologies.

8.1.2 Aligning and Merging Ontologies

K-MORPH performs a context-driven reconciliation of ontologies by merging the contextualized sub-ontologies to generate a merged ontology for facilitating the problem-context at hand. In order to align and merge sub-ontologies, we have developed two ontology matching approaches, *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM), for finding both atomic and complex correspondences between two source ontologies (see Chapter 5) . We evaluated our matchers TOM and POM on data-sets of the Ontology Alignment Evaluation Initiative (OAEI) [167] and compared their results with existing ontology matching systems (see Chapter 5 for details).

Our matching approaches TOM and POM do not perform lexical similarity analysis in their matching process. They find new correspondences between ontology-entities based on the pre-defined correspondences provided in the initially given alignment. However, as demonstrated in Ritze et. al. [176,190], finding lexical and linguistic similarities between ontology-entities upfront can improve the matching process in finding both atomic and complex correspondences [176, 190]. In TOM and POM, we plan to utilize existing lexical matching approaches [41] for finding name-based and thesauri-based similarities between ontology-entities to further enhance our matching process for finding both atomic and complex correspondences.

Correspondences between source ontologies can be complex. As such, representing and validating them becomes a challenging task [176]. In order to apply and evaluate correspondences, they must be represented in a standardize format [169]. Euzenat et. al. proposed the EDOAL: Expressive and Declarative Ontology Alignment Language [172], which extends the original INRIA alignment format [173]. This language allows users to express a correspondence between two complex entities/structures (such as set operators, restrictions applied on entities and relations). In our work, our matchers (TOM and POM) represent correspondences between entities using RDF-Graphs. However, we believe that representing TOM and POM results in a standard format (such as EDOAL) is crucial to enable interoperability with other existing matchers, and also to evaluate complex correspondences.

8.1.3 Detecting and Resolving Inconsistencies in Ontologies

Detecting and resolving inconsistencies is a non-trivial task, as it requires an in-depth understanding of the ontology axioms—in order to select axioms (either manually or automatically), which are either to be removed or repaired for resolving the identified inconsistencies. We have presented our approach for detecting and resolving inconsistencies in ontologies that (i) detects inconsistencies by finding *contradiction derivations* produced under a given logic program; and (ii) generates *minimal inconsistent resolve candidates* MIRCs, where removing any of the MIRCs from the inconsistent ontology results in a maximal consistent sub-ontology w.r.t. the logic program (see Chapter 6 for details).

Our proposed algorithm (see Algorithm 3 in Chapter 6) for computing MIRCs is a form of an apriori/level-wise algorithm [191]. Hence the complexity of our MIRCs algorithm is $O(2^k|\mathbb{M}|)$, where $\mathbb{M}$ is the set of generated MIRCs and $k = |\mathbb{S}|$ is the total number of AATSs. However, the time complexity for MIRCs computation can be improved by formulating this problem as *hypergraph* [192]. A hypergraph $\mathbb{H} = \langle \mathbb{V}, \mathbb{E} \rangle$ is a generalized graph defined on a finite set of vertices $\mathbb{V}$, with every hyper-edge $\mathbb{E}$ of $\mathbb{V}$ being a subset of $\mathbb{V}$. The *transversal hypergraph problem* is the problem of computing, given a hypergraph $\mathbb{H}$, the set of its minimal transversals, i.e. the hypergraph whose hyperedges are all minimal hitting sets of the given one. A minimal transversal $tr(\mathbb{H})$ consists of vertices $\mathbb{V}' \subseteq \mathbb{V}$ such that (a) $\mathbb{V}'$ intersects all hyperedges of $\mathbb{H}$, and (b) no proper subset of $\mathbb{V}'$ does. There have been various attempts towards the *transversal hypergraph problem* to develop a number of optimized algorithms for finding *minimal transversals*. Most efficient algorithm runs in nearly quasi-polynomial time $O(n^{\log n})$, where n is the combined size of the input and the output [193]. In our work, we can consider the asserted ontology triples $\mathbb{I}$ as vertices of a hypergraph $\mathbb{H}_1$, and AATSs $\mathbb{S}$ become hyperedges of $\mathbb{H}_1$. Hence MIRCs can be generated efficiently by computing *minimal transversals* of $\mathbb{H}_1$.

8.2 $\mathcal{K}$-$\mathcal{MORPH}$ Applications: Clinical Decision Making

The demand for context-driven knowledge sharing and integration is realized in various application domains, such as Life Sciences [11–13], E-Business [14–17],

Telecommunications [18], Government Security Services [19,20], Web Portals [21–24] and Marine Sciences [25]. In general, practitioners and domain experts seek an intelligent medium for sharing their behavioural and operational knowledge among other domain experts, in order to evolve a shared understanding between groups, and also to adapt and update their local policies based on the shared knowledge [2,8,26]. On the other hand, knowledge integration becomes crucial when experts are aiming to build a comprehensive knowledge-base for various domain-specific and context-sensitive applications [2,12,13,17].

Our knowledge morphing framework $\mathcal{K}$-$\mathcal{MORPH}$ can be applied in various application domains. In this thesis, we have particularly demonstrated the use of our $\mathcal{K}$-$\mathcal{MORPH}$ framework in the healthcare domain by showing the morphing of various medical ontologies for two different diseases, Prostate Cancer and Urinary Tract Infections, under two different problem-contexts. The discussed test-case can be summarized as follows:

Test-case # 1: *Therapeutic Workflow Knowledge about Prostate Cancer*: For this experiment, we used three location-specific Prostate Cancer (PC) clinical pathways [62], entailing their institution-specific knowledge about the diagnosis, treatments, tests and follow-up care for PC at three different locations: Halifax, Winnipeg and Calgary. $\mathcal{K}$-$\mathcal{MORPH}$ generated a merged PC ontology that provides a comprehensive and context-sensitive knowledge-base for PC management by (i) extracting 3 contextualized sub-ontologies from the three location-specific PC ontologies; (ii) aligning and merging the extracted sub-ontologies; and (iii) detecting and resolving inconsistencies in the merged PC ontology (see Section 3.3 for details).

Test-case # 2: *Therapeutic Knowledge for Treating Urinary Tract Infections:* In this experiment, we considered 20 medical ontologies developed by three groups of medical experts from three different institutions. Hence, these ontologies can be classified into three categories: (i) BioTop Ontology, (ii) DCO Ontology, and (iii) Agfa Ontologies (18 ontologies). These 20 ontologies provide

knowledge about the prognosis, diagnosis, treatment and follow-up for various infectious diseases. $\mathcal{K}$-$\mathcal{MORPH}$ generated a merged ontology to provide a comprehensive and networked knowledge-base for Urinary Tract Infections (UTIs) by (i) extracting 20 contextualized sub-ontologies from DCO, BioTop and AGFA ontologies; (ii) aligning and merging the extracted sub-ontologies; and (iii) detecting and resolving inconsistencies in the merged UTI ontology (see Chapter 7 for details).

8.3 Future Work

Based on the assessment of the research contributions in this PhD thesis (see Section 8.1), we highlight some of the future directions to enhance our $\mathcal{K}$-$\mathcal{MORPH}$ framework for supporting knowledge morphing.

1. Investigating other approaches (e.g. C-OWL, $\mathcal{E}$-connections, etc. [90, 91, 93]) dealing with context-awareness [13] and their support for extracting contextualized sub-ontologies.

2. Based on context-awareness, applying logic-based extraction approaches for identifying entities from the source ontologies that are pertinent to the application context in mind.

3. Dealing with scalability issues: As source ontologies can be very large, performing efficient reasoning for extracting contextualized sub-ontologies [189].

4. Dealing with policy management: Identifying and extracting both local and context-specific constraints—that can be applied on extracted sub-ontologies.

5. Investigating other ontology matching approaches for finding correspondences between the extracted sub-ontologies [176, 194].

6. Representing complex correspondences in a standard format [169] to enable interoperability with other existing matchers, and also to evaluate them.

7. Using defeasible argumentation [96, 97] for resolving inconsistencies. This way, when an inconsistency is raised, 'local' policy (i.e. integrity constraints

and axioms) are considered strict, and only the 'foreign' policy will be defeated by the argumentation process.

8. Optimizing the MIRCs computation by formulating this problem as a *hypergraph* [192], and generating MIRCs efficiently by computing *minimal transversals* of the Hypergraph.

9. Automatically resolving inconsistencies using provenance [195]. This way, when an inconsistency is raised, triples driven from the foreign policies will be removed, while triples driven from local policy will remain unchanged.

10. Dealing with scalability issues: As inconsistencies may appear in very large and networked ontologies, optimized algorithms are required to either resolve inconsistencies, or reason with the inconsistent ontology efficiently [195].

8.4 Final Words

Optimal and complete decision support needs a comprehensive knowledge-base [1]. Developing such a self-contained knowledge-base, as an independent entity, is a non-trivial task [13,29]. There have been various attempts in proposing frameworks and approaches towards knowledge sharing and integration [12–26] (see Chapter 2 for details). However, for achieving a context-driven knowledge integration, Knowledge Morphing aims to formulate a comprehensive knowledge object, specific to a given context, through "the intelligent and autonomous fusion/integration of contextually, conceptually and functionally related knowledge objects that may exist in different representation modalities and formalisms, in order to establish a comprehensive, multi-faceted and networked view of all knowledge pertaining to a domain-specific problem" Abidi 2005 [29].

In this PhD thesis, we have proposed our Semantic Web based framework $\mathcal{K}$-$\mathcal{MORPH}$ to support knowledge morphing among ontology-modelled knowledge sources (see Chapter 3). In order to realize our $\mathcal{K}$-$\mathcal{MORPH}$ approach, we have developed: (i) our sub-ontology extraction method for generating *contextualized sub-ontologies* from the source ontologies pertinent to the problem-context at hand

(see Chapter 4); (ii) two ontology matching approaches, *triple-based ontology matching* (TOM) and *proof-based ontology matching* (POM), for finding both atomic and complex correspondences between two source ontologies (see Chapter 5); and (iii) our approach for resolving inconsistencies in ontologies by generating *minimal inconsistent resolve candidates* (MIRCs), where removing any of the MIRCs from the inconsistent ontology results in a maximal consistent sub-ontology w.r.t. the given logic program (see Chapter 6). Thus, our developed methodologies not only support solving the knowledge morphing problem and its applications in the healthcare domain (see Chapter 7), but also provide scientific contributions in the areas of (a) Sub-ontology Extraction [64–67], (b) Ontology Reconciliation [41,85,86], and (c) Ontology Debugging [116,179–181].

Publications Related to the Thesis:

Conference Publications:

- Sajjad Hussain, Jos De Roo, Ali Daniyal and SSR. Abidi: Detecting and Resolving Inconsistencies in Networked Ontologies. *In IEEE Conference on Computer Software and Applications*, July 18-22 2011, Munich, Germany, IEEE Press (Accepted).

- Sajjad Hussain and SSR. Abidi: Extracting and Merging Contextualized Ontology Modules. *In: 4th International Workshop on Modular Ontologies - FOIS 2010*, May 11 2010, Toronto, Canada, IOS Press.

- Sajjad Hussain and SSR. Abidi: Integrating Heterogeneous Healthcare Knowledge for Clinical Decision Support: A semantic web based approach for healthcare knowledge morphing. *In: 12th Conf. on Artificial Intelligence in Medicine (AIME09)*, July 18-22 2009, Verona, Springer.

- Sajjad Hussain and SSR. Abidi: A Rule-based Method for Extracting RDF(S) and OWL Sub-ontologies. *In: 2nd Canadian Semantic Web Working Symposium*, May 24 2009, Kelowna.

- Sajjad Hussain and SSR. Abidi: $\mathcal{K}$-$\mathcal{MORPH}$: A Semantic Web based Knowledge Representation and Context-driven Morphing Framework. *In: Workshop on Context and Ontologies, at ECAI'2008*, July 21-25 2008, Patras, Greece.

- SSR. Abidi and Sajjad Hussain: Medical Knowledge Morphing via a Semantic Web Framework. *In: 20th IEEE Symposium on Computer-Based Medical Systems*, June 20-22 2007, IEEE Press.

Doctoral Consortium Publications

- Sajjad Hussain: $\mathcal{K}$-$\mathcal{MORPH}$: Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies. *In: PhD Symposium, 7th Extended Semantic Web Conference (ESWC 2010)*, May 31 2010, Heraklion, Greece.

- Sajjad Hussain: $\mathcal{K}$-$\mathcal{MORPH}$: Knowledge Morphing via Reconciliation of Contextualized Sub-ontologies. *In: Doctoral Consortium, 12th International Conference on the Principles of Knowledge Representation and Reasoning*, May 9-13 2010, Toronto, Canada.

- Sajjad Hussain: $\mathcal{K}$-$\mathcal{MORPH}$: A Semantic Web based Knowledge Representation and Context-driven Morphing Framework. *In: Graduate Student Symposium, Canadian Conference on Artificial Intelligence*, May 25-27 2009, Kelowna, British Columbia, Springer.

Bibliography

[1] Keith J. Holyoak and Robert G. Morrison, editors. Cambridge Handbook of Thinking and Reasoning. Cambridge University Press, 2004.

[2] Alexander Schatten, Stefan Biffl, and A Min Tjoa. Closing the gap: From nescience to knowledge management. In EUROMICRO Conference, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[3] VL Patel, JF Arocha, and J Zhang. Cambridge Handbook of Thinking and Reasoning, chapter Thinking and reasoning in medicine. Cambridge University Press, 2004.

[4] Sajjad Hussain, S Abidi, and S S R Abidi. Semantic web framework for knowledge-centric clinical decision support systems. In 11th Conference on Artificial Intelligence in Medicine (AIME 07), Amsterdam, The Netherlands, July 07-11 2007. Springer Verlag Press.

[5] S S R Abidi, C Yu-N, and J Curran-Smith. A knowledge creation infostructure to acquire and crystallize the tacit knowledge of healthcare experts. In IEEE Transactions on Information Technology in Biomedicine, volume 9, pages 193–204, June 2005.

[6] Yuh-Jen Chen. Development of a method for ontology-based empirical knowledge representation and reasoning. Decision Support Systems, 50(1):1–20, 2010.

[7] M. P. A. Thompson and G. Walsham. Placing knowledge management in context. Journal of Management Studies, 41(5):725–747, 2004.

[8] Geoff Walsham. Knowledge management systems: Representation and communication in context. Journal on Communication, Information Technology and Work, 1(1):6–18, 2005.

[9] Ikujiro Nonaka. A Dynamic Theory of Organizational Knowledge Creation. Organization Science, 5(1):14–37, 1994.

[10] Ikujiro Nonaka and Hirotaka Takeuchi. The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press, 1995.

[11] Martin Bishop, editor. Briefings in Bioinformatics, volume 9 of Special Issue:Database Integration in Life Sciences. Oxford University Press, November 2008.

[12] Jon F. Kerner. Knowledge translation versus knowledge integration: A "funder's" prespective. Jounral of Continuing in the Health Professions, 6(1):72–80, 2006.

[13] Antoine Zimmermann, Ratnesh Sahay, Ronan Fox, and Axel Polleres. Heterogeneity and context in semantic-web-enabled HCLS systems. In Robert Meersman, Tharam S. Dillon, and Pilar Herrero, editors, 8th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE 2009), LNCS, pages 1165–1182, Vilamoura, Algarve, Portugal, November 2009. Springer.

[14] Yogesh Malhotra. Knowledge management for e-business performance: Advancing information strategy to "internet time". Information Strategy: The Executive's Journal, 16(4):5–16, 2000.

[15] Richard Herschel and Ira Yermish. Knowledge management in business intelligence. In William R. King, editor, Knowledge Management and Organizational Learning, volume 4 of Annals of Information Systems, pages 131–143. Springer US, 2009.

[16] W. F. Cody, J. T. Kreulen, V. Krishna, and W. S. Spangler. The integration of business intelligence and knowledge management. IBM System Journal, 41(4):697–713, 2002.

[17] Aykut Firat, Stuart Madnick, and Benjamin N. Grosof. Knowledge integration to overcome ontological heterogeneity: Challenges from financial information systems. In International Conference on Information Systems, Barcelona, Spain, Dec. 16-18 2002.

[18] MS Elashaheb. A knowledge management framework for the telecommunication industry: the KMFTI model. PhD thesis, University of Salford, UK, May 2005.

[19] Uffe Kock Wiil, Nasrullah Memon, and Jolanta Gniadek. Knowledge management processes, tools and techniques for counterterrorism. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 29–36. INSTICC Press, 2009.

[20] Zamira Dzhusupova, Adegboyega K. Ojo, and Tomasz Janowski. Organizing and managing knowledge for e-government - issues, practices and challenges. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 206–211. INSTICC Press, 2009.

[21] Alexander Stocker and Klaus Tochtermann. Exploring the value of enterprise wikis - a multiple-case study. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 5–12. INSTICC Press, 2009.

[22] João de Sousa Saraiva and Alberto Rodrigues da Silva. Webc-docs - a cms-based document management system. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 21–28. INSTICC Press, 2009.

[23] Kamla Ali Al-Busaidi. A knowledge management perspective on corporate portal - processes and benefits. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 69–74. INSTICC Press, 2009.

[24] Dimitrios A. Koutsomitropoulos, Georgia D. Solomou, Andreas D. Alexopoulos, and Theodore S. Papatheodorou. Knowledge management and acquisition in digital repositories - a semantic web perspective. In Kecheng Liu, editor, Proceedings of the International Conference on Knowledge Management and Information Sharing, pages 117–122. INSTICC Press, 2009.

[25] S S R Abidi, A. Abusharek, A. Daniyal, M. Kuan, F. Mehdi, S. Abidi, F. Abbas, P. Yeo, F. Jamal, and R. Fathzadeh. A service oriented e-research platform for ocean knowledge management. In 6th IEEE World Congress on Services (Services 2010), Miami, USA, July 5-10 2010.

[26] Dave Robertson, Fausto Giunchiglia, Frank van Harmelen, Maurizio Marchese, Marta Sabou, Marco Schorlemmer, Nigel Shadbolt, Ronnie Siebes, Carles Sierra, Chris Walton, Srinandan Dasmahapatra, Dave Dupplaw, Paul Lewis, Mikalai Yatskevich, Spyros Kotoulas, Adrian Perreau de Pinninck, and Antonis Loizou. Open knowledge semantic webs through peer-to-peer interaction. Technical Report DIT-06-034, University of Trento, 2006.

[27] R. Telesko and D. Karagiannis. Knowledge selection with neural networks. In International Joint Conference on Neural Networks (IJCNN'99), pages 2486–2489, 1999.

[28] Dimitris Karagiannis, Franz Kurfess, and Heinz-Wilhelm Schmidt. Knowledge selection in large knowledge bases. In Michael Papazoglou and John Zeleznikow, editors, The Next Generation of Information Systems: From Data to Knowledge, volume 611 of Lecture Notes in Computer Science, pages 291–310. Springer, 1992.

[29] S S R Abidi. Medical knowledge morphing: Towards the integration of medical knowledge resources. Computer-Based Medical Systems, June 23-24 2005.

[30] S S R Abidi and Shapoor Shayegani. Modeling the form and function of clinical practice guidelines: An ontological model to computerize clinical practice guidelines. In Knowledge Management for Healthcare Processes Workshop at ECAI 2008, Patras, 2008.

[31] Sajjad Hussain and S S R Abidi. Integrating healthcare knowledge artifacts for clinical decision support: Towards semantic web based healthcare knowledge morphing. In C. Combi, Y. Shahar, and A. Abu-Hanna, editors, 12th Conference on Artificial Intelligence in Medicine (AIME 09), volume 5651 of LNAI, Verona, Italy, July 18-22 2009. Springer Verlag Press.

[32] Evimaria Terzi, Athena Vakali, and Mohand-Saïd Hacid. Knowledge representation, ontologies, and the semantic web. In Y. Zhang X. Zhou and M.E. Orlowska, editors, Web Technologies and Applications, volume 2642 of LNCS, pages 382–387. Springer, 2003.

[33] T Berners-Lee, J Hendler, and O Lassila. The semantic web. Scientific American, 284(5):34–43, 2001.

[34] Timon ten Berge and Rene van Hezewijk. Procedural and declarative knowledge: An evolutionary perspective. Theory & Psychology, 9(5):605–624, October 1999.

[35] Paolo Ciccarese, Marco Ocana, Sudeshna Das, and Tim Clark. Ao: An open annotation ontology for science on the web. Journal of BioMedical Semantics, May 2010. Accepted.

[36] Peter Haase and Ljiljana Stojanovic. Consistent evolution of owl ontologies. In Asunción Gómez-Pérez and Jérôme Euzenat, editors, Proceedings of the Second European Semantic Web Conference, Heraklion, Greece, 2005, volume 3532 of Lecture Notes in Computer Science, pages 182–197. Springer, May 2005.

[37] Ljiljana Stojanovic. Methods and Tools for Ontology Evolution. PhD thesis, Universität Karlsruhe (TH), Universität Karlsruhe (TH), Institut AIFB, D-76128 Karlsruhe, 2004.

[38] Natalya Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. Knowledge and Information Systems, 6(4):428–440, 2004.

[39] Eduard Hovy. Combining and standardizing large-scale, practical ontologies for machine translation and other uses. In Proc. 1st International Conference on Language Resources and Evaluation (LREC), pages 535–542, Granada (ES), 1998.

[40] Ian Niles and Adam Pease. Towards a standard upper ontology. In Proc. 2nd International Conference on Formal Ontology in Information Systems (FOIS), pages 2–9, Ogunquit (ME US), 2001.

[41] Jérôme Euzenat and Pavel Shvaiko. Ontology matching. Springer-Verlag, Heidelberg (DE), 2007.

[42] Marc Ehrig. Ontology alignment: bridging the semantic gap. Semantic web and beyond: computing for human experience. Springer, New-York (NY US), 2007.

[43] R. V. Guha, R. McCool, and R. Fikes. Contexts for the semantic web. In 3rd International Semantic Web Conference ISWC 2004, volume 3298 of LNCS, pages 32–46. Springer, 2004.

[44] Tony Veale and Yanfen Hao. Corpus-driven contextualized categorization. In Contexts and Ontologies workshop at ECAI 2006, Trento, Italy, August 2006.

[45] F Giunchiglia. Contextual reasoning. Technical report, 1992. Technical report.

[46] Aviv Segev and Avigdor Gal. Putting things in context: A topological approach to mapping contexts to ontologies. Journal on Data Semantics IX, LNCS 4601, pages 113–140, 2007.

[47] Deborah L. McGuinness and Frank van Harmelen. OWL: Web Ontology Language Overview. W3C Recommendation, February 2003. http://www.w3.org/TR/owl-features/.

[48] Frank Manola and Eric Miller. RDF Primer. W3C Recommendation, February 10 2004. http://www.w3.org/TR/rdf-primer.

[49] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. OWL 2 Web Ontology Language Profiles:, October 27 2009. http://www.w3.org/TR/owl2-profiles/Reasoning_in_OWL_2_RL_and_RDF_Graphs_using_Rules.

[50] Mike Dean and Guus Schreiber (eds.). OWL web ontology language reference. Recommendation, W3C, February 2004.

[51] Mike Smith, Christopher Welty, and Deborah McGuinness (eds.). OWL web ontology language guide. Recommendation, W3C, February 10 2004.

[52] Ramanathan Guha. Contexts: a formalization and some applications. PhD thesis, Stanford, CA, USA, 1992.

[53] Joelle Coutaz, James Crowley, Simon Dobson, and David Garlan. Context is key. Communications of the ACM, 48(3):49–53, 2005.

[54] J Barwise and J Perry. Situations and Attitudes. MIT Press, 1983.

[55] J McCarthy. Notes on formalizing context. In Proc. of the IJCAI, 1993.

[56] Richmond Thomason. Representing and reasoning with context. In Jacques Calmet and Jan Plaza, editors, Artificial Intelligence and Symbolic Computation, volume 1476 of Lecture Notes in Computer Science, pages 29–41. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0055900.

[57] G Chen and D Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[58] Luciano Serafini and Paolo Bouquet. Comparing formal theories of context in ai. Artificial Intelligence, 155(1-2):41–67, 2004.

[59] Massimo Benerecetti, Paolo Bouquet, and Chiara Ghidini. On the dimensions of context dependence: partiality, approximation, and perspective. In Proc. 3rd International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT), volume 2116 of Lecture notes in computer science, pages 59–72, Dundee (UK), 2001.

[60] A. K. Dey. Understanding and using context. Personal Ubiquitous Computing, 5(1):4–7, 2001.

[61] Andreas Zimmermann, Andreas Lorenz, and Reinhard Oppermann. An operational definition of context. In B. Kokinov et al., editor, CONTEXT 2007, volume 4635 of LNAI, pages 558–571. Springer-Verlag, 2007.

[62] S. Abidi, R Abidi, Sajjad Hussain, and L Butler. Ontology-based modeling and merging of institution-specific prostate cancer clinical pathways. In Knowledge Management for Healthcare Processes Workshop at ECAI 2008, Patras, Greece, July 21-25 2008.

[63] Sajjad Hussain and Syed Sibte Raza Abidi. Extracting and merging contextualized ontology modules. In Jie Bao Bernardo Cuenca Grau Oliver Kutz, Joana Hois, editor, Modular Ontologies - Proceedings of the Fourth International Workshop (WoMO 2010), volume 210 of Frontiers in Artificial Intelligence and Applications, pages 25–40. IOS Press, May 2010.

[64] R Rajugan, Elizabeth Chang, and Tharam S. Dillon. Sub-ontologies and ontology views; a theoretical perspective. Int. J. Metadata Semant. Ontologies, 2(2):94–111, 2007.

[65] Julian Seidenberg. Web ontology segmentation: Extraction, transformation, evaluation. In H. Stuckenschmidt et al., editor, Modular Ontologies, volume 5445 of LNCS, pages 211–243. Springer, 2009.

[66] Natalya F. Noy and Mark A. Musen. Traversing ontologies to extract views. In H. Stuckenschmidt et al., editor, Modular Ontologies, volume 5445 of LNCS, pages 245–260. Springer, 2009.

[67] Zhuang Miao, Jinpen Wang, Bo Zhou, Yafei Zhang, and Jianjiang Lu. Method for extracting RDF(S) sub-ontology. pages 348–353, 2008.

[68] Heiner Stuckenschmidt and Anne Schlicht. Structure-based partitioning of large ontologies. In H. Stuckenschmidt et al., editor, Modular Ontologies, volume 5445 of LNCS, pages 187–210. Springer, 2009.

[69] Guilin Qi and Anthony Hunter. Measuring incoherence in description logic-based ontologies. In 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC/ASWC'07), volume 4825 of LNCS, pages 381–394. Springer, 2007.

[70] Peter Haase, Frank van Harmelen, Zhisheng Huang, Heiner Stuckenschmidt, and York Sure. A framework for handling inconsistency in changing ontologies. In 4th International Semantic Web Conference (ISWC'05), pages 353–367. Springer, 2005.

[71] Thomas Scharrenbach, Rolf Grütter, Bettina Waldvogel, and Abraham Bernstein. Structure preserving TBox repair using defaults. In Proceedings of the 23rd International Workshop on Description Logics (DL 2010), CEUR-ws. CEUR Workshop Proceedings, 2010.

[72] Jörg Pührer, Stijn Heymans, and Thomas Eiter. Dealing with inconsistency when combining ontologies and rules using DL-programs. In Extended Semantic Web Conference (ESWC2010), volume 6088 of LNCS, pages 183–197, Heraklion, Greece, June 2010.

[73] Pavel Shvaiko and Jérôme Euzenat. Ten challenges for ontology matching. In On the Move to Meaningful Internet Systems: OTM 2008, volume 5332/2008 of Lecture Notes in Computer Science, pages 1164–1182, Monterrey, Mexico, November 9-14 2008. Springer.

[74] Sergio Alejandro Gomez, Carlos Ivan Chesnevar, and Guillermo Ricardo Simari. Reasoning with inconsistent ontologies through argumentation. Applied Artificial Intelligence, 24(1-2):102–148, 2010.

[75] Z Huang, F van Harmelen, and A ten Teije. Reasoning with inconsistent ontologies. In 19th International Joint Conference on Artificial Intelligence(IJCAI 2005), pages 254–259, San Francisco, 2005. Morgan Kaufmann.

[76] Jérôme Euzenat. Towards a principled approach to semantic interoperability. In Proc. IJCAI Workshop on Ontologies and Information Sharing, pages 19–25, Seattle (WA US), 2001.

[77] Jérôme Euzenat and Heiner Stuckenschmidt. The 'family of languages' approach to semantic interoperability. In Borys Omelayenko and Michel Klein,

editors, Knowledge transformation for the semantic web, pages 49–63. IOS press, Amsterdam (NL), 2003.

[78] Yuri Breitbart. Multidatabase interoperability. ACM SIGMOD Record, 19(3):53–60, 1990.

[79] Won Kim and Jungyun Seo. Classifying schematic and data heterogeneity in multidatabase systems. IEEE Computer, 24(12):12–18, 1991.

[80] Richard Hull. Managing semantic heterogeneity in databases: a theoretical prospective. In Proc. 16th Symposium on Principles of Database Systems (PODS), pages 51–61, Tucson (AZ US), 1997.

[81] Vipul Kashyap and Amit Sheth. Semantic heterogeneity in global information systems: The role of metadata, context and ontologies. In Michael Papazoglou and Gunter Schlageter, editors, Cooperative information systems, pages 139–178. Academic Press, New York (NY US), 1998.

[82] Holger Wache, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. Ontology-based integration of information – a survey of existing approaches. In Proc. IJCAI Workshop on Ontologies and Information Sharing, pages 108–117, Seattle (WA US), 2001.

[83] Michel Klein. Combining and relating ontologies: an analysis of problems and solutions. In Proc. IJCAI Workshop on Ontologies and Information Sharing, Seattle (WA US), 2001.

[84] Jérôme Euzenat. An infrastructure for formally ensuring interoperability in a heterogeneous semantic web. In Isabel Cruz, Stefan Decker, Jérôme Euzenat, and Deborah McGuinness, editors, The emerging semantic web, pages 245–260. IOS press, Amsterdam (NL), 2002.

[85] Adil Hameed, Alun Preece, and Derek Sleeman. Ontology reconciliation. In Steffen Staab and Rudi Studer, editors, Handbook on ontologies, chapter 12, pages 231–250. Springer Verlag, Berlin (DE), 2004.

[86] Avigdor Gal, Ateret Anaby-Tavor, Alberto Trombetta, and Danilo Montesi. A framework for modeling and evaluating automatic semantic reconciliation. The VLDB Journal, 14(1):50–67, 2005.

[87] S S R Abidi and Sajjad Hussain. Medical knowledge morphing via a semantic web framework. In 20th IEEE Symposium on Computer-Based Medical Systems. IEEE Press, June 20-22 2007.

[88] Sajjad Hussain and S S R Abidi. K-MORPH: A semantic web based knowledge representation and context-driven morphing framework. In Workshop on Context and Ontologies, at ECAI'2008, Patras, Greece, July 21-25 2008.

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

[90] Alexander Borgida and Luciano Serafini. Distributed description logics: Assimilating information from peer sources. Journal on Data Semantics, I:153–184, 2003.

[91] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. C-OWL – contextualizing ontologies. In Proc. 2nd International Semantic Web Conference (ISWC), volume 2870 of Lecture notes in computer science, pages 164–179, Sanibel Island (FL US), 2003.

[92] J. Bao, D. Caragea, and V. G. Honavar. On the semantics of linking and importing in modular ontologies. In 5th International Semantic Web Conference ISWC 2006, volume 4273, pages 72–86. Springer, 2006.

[93] O O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. $\mathcal{E}$-connections of abstract description systems. Artificial Intelligence, 156(1):1–73, 2004.

[94] A. Zimmermann. Integrated distributed description logics. In D. Calvanese, E. Franconi, V. Haarslev, D. Lembo, B. Motik, S. Tessaris, and A. Y. Turhan, editors, 20th International Workshop on Description Logics, DL'07, pages 507–514. Bolzano University Press, 2007.

[95] J. Y. Bèziau, W. Carnielli, and D. M. Gabbay. Handbook of Paraconsistency. College Publications, 2007.

[96] C. I. Ches nevar, A. G. Maguitman, and R. P. Loui. Logical models of argument. ACM Computing Survey, 32(4):337–383, 2000.

[97] A. J. Garcìa and G. R. Simari. Defeasible logic programming: An argumentative approach. Theory and Practice of Logic Programming, 4(1-2):95–138, 2004.

[98] A. Analyti, G. Antoniou, and C. V. Damàsio. A principled framework for modular web rule bases and its semantics. In 11th International Conference on Principles of Knowledge Representation and Reasoning (KR 2008), pages 390–400. AAAI Press, 2008.

[99] Aykut Firat, Stuart Madnick, and Benjamin Grosof. Contextual alignment of ontologies in the eCOIN semantic interoperability framework. Inf. Tech. and Management, 8(1):47–63, 2007.

[100] Dave Robertson, Fausto Giunchiglia, Frank van Harmelen, Maurizio Marchese, Marta Sabou, Marco Schorlemmer, Nigel Shadbolt, Ronnie Siebes, Carles Sierra, Chris Walton, Srinandan Dasmahapatra, Dave Dupplaw, Paul

Lewis, Mikalai Yatskevich, Spyros Kotoulas, Adrian Perreau de Pinninck, and Antonis Loizou. Open knowledge: Semantic webs through peer-to peer interaction. Technical Report DIT-06-034, University of Trento, Povo, Italy, May 2006.

[101] Fausto Giunchiglia, Fiona McNeill, and Fiona McNeill. Web service composition via semantic matching of interaction specifications. In WWW2007, Canada, 2007. ACM Press.

[102] Yuxin Mao, William K. Cheung, Zhaohui Wu, and Jiming Liu. Dynamic sub-ontology evolution for collaborative problem-solving. In AAAI fall symposium, pages 1–8, 2005.

[103] Dazhou Kang, Baowen Xu, Jianjiang Lu, Peng Wang, and Yanhui Li. Extracting sub-ontology from multiple ontologies. In OTM 2004 Workshop on Ontologies, Semantics, and E-learning (WOSE), volume 3292 of LNCS, pages 731–740, 2004.

[104] Mathieu d' Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Modular Ontologies, volume 5445 of LNCS, chapter Criteria and Evaluation for Ontology Modularization Techniques, pages 67–89. Springer, 2009.

[105] Martin Dzbor, John Domingue, and Enrico Motta. Magpie – towards a semantic web browser. In Proc. 2nd International Semantic Web Conference (ISWC), volume 2870 of Lecture notes in computer science, pages 690–705, Sanibel Island (FL US), 2003.

[106] Jeremy J. Carroll, Christian Bizer, Patrick J. Hayes, and Patrick Stickler. Named graphs. Journal of Web Semantics, 3(4):247–267, 2005.

[107] B Schillit, N Adams, and R Want. Context-aware computing applications. In 1st International Workshop on Mobile Computing Systems and Applications, pages 85–90, 1994.

[108] Carlo Batini, Maurizio Lenzerini, and Shamkant Navathe. A comparative analysis of methodologies for database schema integration. ACM Computing Surveys, 18(4):323–364, 1986.

[109] Amit Sheth and James Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Computing Surveys, 22(3):183–236, 1990.

[110] Cheng-Hian Goh. Representing and reasoning about semantic conflicts in heterogeneous information sources. PhD thesis, MIT, Cambridge (MA US), 1997.

[111] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. Contextualizing ontologies. Journal of Web Semantics, 1(1):325–343, 2004.

[112] Pepijn Visser, Dean Jones, Trevor Bench-Capon, and Michael Shave. Assessing heterogeneity by classifying ontology mismatches. In Proc. 1st International Conference on Formal Ontology in Information Systems (FOIS), pages 148–162, Trento (IT), 1998.

[113] Paolo Bouquet, Marc Ehrig, Jérôme Euzenat, Enrico Franconi, Pascal Hitzler, Markus Krötzsch, Luciano Serafini, Giorgos Stamou, York Sure, and Sergio Tessaris. Specification of a common framework for characterizing alignment. Deliverable D2.2.1, Knowledge web NoE, 2004.

[114] Livia Predoiu, Cristina Feier, Francois Scharffe, Jos de Bruijn, Francisco Martń-Recuerda, Dimitar Manov, and Marc Ehrig. State-of-the-art survey on ontology merging and aligning v2. Technical Report D4.2.2, Digital Enterprise Research Institute, University of Innsbruck, December 31 2005.

[115] Natalya Noy and Mark Musen. Ontology versioning in an ontology management framework. IEEE Intelligent Systems, 19(4):6–13, 2004.

[116] Peter Haase and Guilin Qi. An analysis of approaches to resolving inconsistencies in DL-based ontologies. In International Workshop on Ontology Dynamics, June 2007.

[117] Tim Berners-Lee, Dan Connolly, Lalana Kagal, Yosi Scharf, and Jim Hendler. N3Logic: A logical framework for the world wide web. Theory and Practice of Logic Programming, 8(3):249–269, May 2008.

[118] Jos De Roo. Euler proof mechanism–Eye: last accessed on January 9th, 2011. `http://eulersharp.sourceforge.net/2003/03swap/eye-2009.txt`, 2005.

[119] Christine Parent and Stefano Spaccapietra. Database integration: the key to data interoperability. In Mike Papazoglou, Stefano Spaccapietra, and Zahir Tari, editors, Object-oriented data modeling, chapter 9, pages 221–253. The MIT Press, Cambridge (MA US), 2000.

[120] Erhard Rahm and Philip Bernstein. A survey of approaches to automatic schema matching. The VLDB Journal, 10(4):334–350, 2001.

[121] Hong-Hai Do, Sergei Melnik, and Erhard Rahm. Comparison of schema matching evaluations. In Proc. Workshop on Web, Web-Services, and Database Systems, volume 2593 of Lecture notes in computer science, pages 221–237, Erfurt (DE), 2002.

[122] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. The Knowledge Engineering Review, 18(1):1–31, 2003.

[123] Natalya Noy. Semantic integration: A survey of ontology-based approaches. ACM SIGMOD Record, 33(4):65–70, 2004.

[124] Natalya Noy. Tools for mapping and merging ontologies. In Steffen Staab and Rudi Studer, editors, Handbook on ontologies, chapter 18, pages 365–384. Springer Verlag, Berlin (DE), 2004.

[125] An-Hai Doan and Alon Halevy. Semantic integration research in the database community: A brief survey. AI Magazine, 26(1):83–94, 2005. Special issue on Semantic integration.

[126] Pavel Shvaiko and Jérôme Euzenat. A survey of schema-based matching approaches. Journal on Data Semantics, IV:146–171, 2005.

[127] Tova Milo and Sagit Zohar. Using schema matching to simplify heterogeneous data translation. In Proc. 24th International Conference on Very Large Data Bases (VLDB), pages 122–133, New York (NY US), 1998.

[128] Luigi Palopoli, Giorgio Terracina, and Domenico Ursino. DIKE: a system supporting the semi-automatic construction of cooperative information systems from heterogeneous databases. Software–Practice and Experience, 33(9):847–884, 2003.

[129] Prasenjit Mitra, Gio Wiederhold, and Martin Kersten. A graph-oriented model for articulation of ontology interdependencies. In Proc. 8th Conference on Extending Database Technology (EDBT), volume 1777 of Lecture notes in computer science, pages 86–100, Praha (CZ), 2000.

[130] Silvana Castano, Valeria De Antonellis, and Sabrina De Capitani di Vimercati. Global viewing of heterogeneous data sources. IEEE Transactions on Knowledge and Data Engineering, 13(2):277–297, 2000.

[131] Silvana Castano, Alfio Ferrara, and Stefano Montanelli. Matching ontologies in open networked systems: Techniques and applications. Journal on Data Semantics, V:25–63, 2006.

[132] Natalya Noy and Mark Musen. Anchor-PROMPT: Using non-local context for semantic matching. In Proc. IJCAI Workshop on Ontologies and Information Sharing, pages 63–70, Seattle (WA US), 2001.

[133] Giovanni Modica, Avigdor Gal, and Hasan Jamil. The use of machine-generated ontologies in dynamic information seeking. In Proc. 9th International Conference on Cooperative Information Systems (CoopIS), volume 2172 of Lecture notes in computer science, pages 433–448, Trento (IT), 2001.

[134] Jayant Madhavan, Philip Bernstein, and Erhard Rahm. Generic schema matching with Cupid. In Proc. 27th International Conference on Very Large Data Bases (VLDB), pages 48–58, Roma (IT), 2001.

[135] Hong-Hai Do and Erhard Rahm. COMA – a system for flexible combination of schema matching approaches. In Proc. 28th International Conference on Very Large Data Bases (VLDB), pages 610–621, Hong Kong (CN), 2002.

[136] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: a versatile graph matching algorithm. In Proc. 18th International Conference on Data Engineering (ICDE), pages 117–128, San Jose (CA US), 2002.

[137] Yuan An, Alexander Borgida, and John Mylopoulos. Discovering the semantics of relational tables through mappings. Journal on Data Semantics, VII:1–32, 2006.

[138] Dejing Dou, Drew McDermott, and Peishen Qi. Ontology translation on the semantic web. Journal on Data Semantics, II:35–57, 2005.

[139] Paolo Bouquet, Bernardo Magnini, Luciano Serafini, and Stefano Zanobini. A SAT-based algorithm for context matching. In Proc. 4th International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT), volume 2680 of Lecture notes in computer science, pages 66–79, Stanford (CA US), 2003.

[140] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: a practical OWL-DL reasoner. Journal of Web Semantics, 5, 2007.

[141] Dmitry Tsarkov and Ian Horrocks. FaCT++ description logic reasoner: system description. In Proc. 3rd International Joint Conference on Automated Reasoning (IJCAR), volume 4130 of LNCS, pages 292–297, Seattle (WA US), 2006. Springer.

[142] Fausto Giunchiglia, Mikalai Yatskevich, and Pavel Shvaiko. Semantic matching: Algorithms and implementation. Journal on Data Semantics, IX, 2007.

[143] Konstantinos Kotis, George Vouros, and Konstantinos Stergiou. Towards automatic merging of domain ontologies: The HCONE-merge approach. Journal of Web Semantics, 4(1):60–79, 2006.

[144] 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 Proc. 18th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE), volume 3533 of Lecture notes in computer science, pages 722–731, Bari (IT), 2005.

[145] Than-Le Bach, Rose Dieng-Kuntz, and Fabien Gandon. On ontology matching problems (for building a corporate semantic web in a multi-communities organization). In Proc. 6th International Conference on Enterprise Information Systems (ICEIS), pages 236–243, Porto (PT), 2004.

[146] Rong Pan, Zhongli Ding, Yang Yu, and Yun Peng. A Bayesian network approach to ontology mapping. In Proc. 3rd International Semantic Web Conference (ISWC), volume 3298 of Lecture notes in computer science, pages 563–577, Hiroshima (JP), 2005.

[147] Prasenjit Mitra, Natalya Noy, and Anuj Jaiswal. Ontology mapping discovery with uncertainty. In Proc. 4th International Semantic Web Conference (ISWC), volume 3729 of Lecture notes in computer science, pages 537–547, Galway (IE), 2005.

[148] Kevin Chang, Bin He, and Zhen Zhang. Toward large scale integration: Building a metaquerier over databases on the web. In Proc. 2nd Biennial Conference on Innovative Data Systems Research (CIDR), pages 44–55, Asilomar (CA US), 2005.

[149] Jérôme Euzenat. Brief overview of T-tree: the Tropes taxonomy building tool. In Proc. 4th ASIS SIG/CR Workshop on Classification Research, pages 69–87, Columbus (OH US), 1994.

[150] Martin Lacher and Georg Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In Proc. 14th International Florida Artificial Intelligence Research Society Conference (FLAIRS), pages 305–309, Key West (FL US), 2001.

[151] Gerd Stumme and Alexander Mädche. FCA-Merge: Bottom-up merging of ontologies. In Proc. 17th International Joint Conference on Artificial Intelligence (IJCAI), pages 225–234, Seattle (WA US), 2001.

[152] An-Hai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. Ontology matching: a machine learning approach. In Steffen Staab and Rudi Studer, editors, Handbook on ontologies, chapter 18, pages 385–404. Springer Verlag, Berlin (DE), 2004.

[153] Robin Dhamankar, Yoonkyong Lee, An-Hai Doan, Alon Halevy, and Pedro Domingos. iMAP: Discovering complex semantic matches between database schemas. In Proc. 23rd International Conference on Management of Data (SIGMOD), pages 383–394, Paris (FR), 2004.

[154] Jacob Berlin and Amihai Motro. Database schema matching using machine learning with feature selection. In Proc. 14th International Conference on Advanced Information Systems Engineering (CAiSE), volume 2348 of Lecture notes in computer science, pages 452–466, Toronto (CA), 2002.

[155] Jiying Wang, Ji-Rong Wen, Frederick Lochovsky, and Wei-Ying Ma. Instance-based schema matching for web databases by domain-specific query probing. In Proc. 30th International Conference on Very Large Data Bases (VLDB), pages 408–419, Toronto (CA), 2004.

[156] Henrik Nottelmann and Umberto Straccia. sPLMap: A probabilistic approach to schema matching. In Proc. 27th European Conference on Information Retrieval Research (ECIR), pages 81–95, Santiago de Compostela (ES), 2005.

[157] Wen-Syan Li and Chris Clifton. SEMINT: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. Data and Knowledge Engineering, 33(1):49–84, 2000.

[158] Laura Haas, Mauricio Hernández, Howard Ho, Lucian Popa, and Mary Roth. Clio grows up: from research prototype to industrial tool. In Proc. 24th International Conference on Management of Data (SIGMOD), pages 805–810, Baltimore (MD US), 2005.

[159] Marc Ehrig and York Sure. Ontology mapping – an integrated approach. In Proc. 1st European Semantic Web Symposium (ESWS), volume 3053 of Lecture notes in computer science, pages 76–91, Hersounisous (GR), May 2004.

[160] Marc Ehrig and Steffen Staab. QOM – quick ontology mapping. In Proc. 3rd International Semantic Web Conference (ISWC), volume 3298 of Lecture notes in computer science, pages 683–697, Hiroshima (JP), 2004.

[161] Umberto Straccia and Raphaël Troncy. oMAP: Combining classifiers for aligning automatically OWL ontologies. In Proc. 6th International Conference on Web Information Systems Engineering (WISE), pages 133–147, New York (NY US), 2005.

[162] David Embley, Li Xu, and Yihong Ding. Automatic direct and indirect schema mapping: Experiences and lessons learned. ACM SIGMOD Record, 33(4):14–19, 2004.

[163] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in OWL-lite. In Proc. 16th European Conference on Artificial Intelligence (ECAI), pages 333–337, Valencia (ES), 2004.

[164] Yannis Kalfoglou and Marco Schorlemmer. IF-Map: an ontology mapping method based on information flow theory. Journal on Data Semantics, I:98–127, 2003.

[165] Wei Hu, Ningsheng Jian, Yuzhong Qu, and Qanbing Wang. GMO: A graph matching for ontologies. In Proc. K-CAP Workshop on Integrating Ontologies, pages 43–50, Banff (CA), 2005.

[166] Jie Tang, Juanzi Li, Bangyong Liang, Xiaotong Huang, Yi Li, and Kehong Wang. Using Bayesian decision for ontology mapping. Journal of Web Semantics, 4(1):243–262, 2006.

[167] Ontology Alignment Evaluation Initiative. http://oaei.ontologymatching.org/.

[168] Anhai Doan and Alon Y. Halevy. Semantic integration research in the database community: A brief survey. AI Magazine, 26:83–94, 2005.

[169] F. Scharffe. Correspondence Patterns Representation. PhD thesis, University of Innsbruck, 2009.

[170] O. Sváb-Zamazal and Vojtěch Svátek. Towards ontology matching via pattern-based detection of semantic structures in owl ontologies. In In Proceedings of the Znalosti Czecho-Slovak Knowledge Technology conference, 2009.

[171] O. Šváb Zamazal, V. Svátek, J. David, and F. Scharffe. Towards metamorphic semantic models. In Poster Session at European Semantic Web Conference (ESWC2009), LNCS, Heraklion, Greece, 2009. Springer.

[172] Edoal: Expressive and declarative ontology alignment language. http://alignapi.gforge.inria.fr/edoal.html.

[173] J. Euzenat, F. Scharffe, and A. Zimmermann. Expressive alignment language and implementation. deliverable 2.2.10, Knowledge web, 2007.

[174] H. Stuckenschmidt, L. Predoiu, and C. Meilicke. Learning Complex Ontology Alignments A Challenge for ILP Research. In Proceedings of the 18th International Conference on Inductive Logic Programming, 2008.

[175] Han Qin, Dejing Dou, and Paea LePendu. Discovering executable semantic mappings between ontologies. In OTM'07: Proceedings of the 2007 OTM Confederated international conference on On the move to meaningful internet systems, pages 832–849, Berlin, Heidelberg, 2007. Springer-Verlag.

[176] Dominique Ritze, Christian Meilicke, Ondřej Šváb Zamazal, and Heiner Stuckenschmidt. A pattern-based ontology matching approach for detecting complex correspondences. In Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Natasha Noy, and Arnon Rosenthal, editors, International Workshop on Ontology Matching (OM-2009), volume 551. CEUR-WS, October 2009.

[177] V Haarslev and R Möller. RACER system description. In R P Goré, A Leitsch, and T Nipkow, editors, IJCAR 2001, volume 2083 of LNCS (LNAI), pages 701–706. Springer, 2001.

[178] M Ahmad, S S R ABIDI, and B Jafarpour. Ontology based modeling and execution of nursing care plans and practice guidelines. In 13th World Congress on Medical Informatics (MEDINFO 2010), Cape Town, September 12-15 2010. IOS Press.

[179] David Bell, Guilin Qi, , and Weiru Liu. Approaches to inconsistency handling in description-logic based ontologies. In 3rd International IFIP Workshop On Semantic Web & Web Semantics (SWWS'07), volume 4806 of LNCS, pages 1303–1311, Springer, 2007.

[180] Q Ji, P Haase, and G Qi. RaDON-Repair and diagnosis in ontology networks. In 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25 2008.

[181] Xi Deng, Volker Haarslev, and Nematollaah Shiri. Measuring inconsistency in ontologies. In ESWC 2007, volume 4519 of LNCS, pages 326–340, 2007.

[182] Marc Bezem and Thierry Coquand. Automating coherent logic. In 12th International conference on Logic for programming, artificial intelligence, and reasoning, LPAR 2005, volume 3835 of LNCS, pages 246–260, Jamaica, December 2-6 2005. Springer.

[183] Li Ding, Lalana Kagal, Deborah L. McGuinness, and Paulo Pinheiro da Silva. Translating cwm proof into pml. Working Draft, 2006. http://iw.stanford.edu/2006/tami/TR-cwm-pml-translation.htm.

[184] K. Hurley and S S R Abidi. Ontology engineering to model clinical pathways: Towards the computerization and execution of clinical pathways. In 20th IEEE Symposium on Computer-Based Medical Systems, Slovenia, June 20-22 2008. IEEE Press.

[185] Elena Beisswanger, Stefan Schulz, Holger Stenzhorn, and Udo Hahn. Biotop: An upper domain ontology for the life sciences: A description of its current structure, contents and interfaces to OBO ontologies. Applied Ontology, 3(4):205–212, 2008.

[186] Pierre Grenon, Barry Smith, and Louis Goldberg. Biodynamic ontology: Applying BFO in the biomedical domain. In Stud. Health Technol. Inform, pages 20–38. IOS Press, 2004.

[187] Daniel Schober, Martin Boeker, Jessica Bullenkamp, Csaba Huszka, Kristof Depraetere, Douglas Teodoro, Nadia Nadah, Remy Choquet, Christel Daniel, and Stefan Schulz. The debugit core ontology: semantic integration of antibiotics resistance patterns. In C. Safran et al., editor, MEDINFO 2010, pages 1060–1064. IOS Press, 2010.

[188] DebugIT–Detecting and Eliminating Bacteria UsinG Information Technology. http://www.debugit.eu/, Large-scale integrating project (FP7, 2008 - 2011).

[189] Chiara del Vescovo, Bijan Parsia, Uli Sattler, and Thomas Schneider. The modular structure of an ontology: an empirical study. In Jie Bao Bernardo Cuenca Grau Oliver Kutz, Joana Hois, editor, Modular Ontologies - Proceedings of the Fourth International Workshop (WoMO 2010), volume 210 of Frontiers in Artificial Intelligence and Applications, pages 11–24. IOS Press, May 2010.

[190] Dominique Ritze, Johanna Völker, Christian Meilicke, and Ondřej Šváb Zamazal. Linguistic analysis for complex ontology matching. In Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Ming Mao, and Isabel Cruz, editors, ISWC workshop on Ontology Matching (OM-2010), volume 689. CEUR-WS, November 2010.

[191] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In VLDB94, pages 487–499, 1994.

[192] D. J. Kavvadias and E. C. Stavropoulos. An efficient algorithm for the transversal hypergraph generation. Graph Algorithms and Applications, 9(2):239–264, 2005.

[193] Michael L. Fredman and Leonid Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. Journal of Algorithms, 21(3):618–628, November 1996.

[194] Franois Scharffe and Dieter Fensel. Correspondence patterns for ontology alignment. In Aldo Gangemi and Jérôme Euzenat, editors, Knowledge Engineering: Practice and Patterns, volume 5268 of Lecture Notes in Computer Science, pages 83–92. Springer Berlin / Heidelberg, 2008. 10.1007/978-3-540-87696-0_10.

[195] A. Hogan, A. Harth, A. Passant, S. Decker, and A. Polleres. Weaving the Pedantic Web. In Linked Data on the Web Workshop (LDOW2010) at WWW'2010, 2010.