Enabling Complex Semantic Queries to Bioinformatics Databases through Intuitive Search Over Data

Sima Ana Claudia

UNIL | Université de Lausanne

Faculté de biologie et de médecine

DOCTORAL THESIS

Enabling Complex Semantic Queries to Bioinformatics Databases through Intuitive Search Over Data

---

Author:
Ana Claudia SIMA

Jury:
Prof. Christophe Dessimoz, thesis director,
Prof. Kurt Stockinger, thesis co-director,
Prof. Marc Robinson-Rechavi, thesis co-director,
Prof. Donald Kossman, expert,
Prof. Ioannis Xenarios, expert,
Prof. Niklaus Schäfer, president

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy (PhD) in the

Department of Ecology and Evolution
Faculty of Biology and Medicine

Lausanne
(2020)

UNIL | Université de Lausanne
Faculté de biologie
et de médecine

Ecole Doctorale
Doctorat ès sciences de la vie

Imprimatur

Vu le rapport présenté par le jury d'examen, composé de

Président·e	Monsieur	Prof.	Niklaus	**Schäfer**
Directeur·trice de thèse	Monsieur	Prof.	Christophe	**Dessimoz**
Co-directeurs·trices	Monsieur	Prof.	Kurt	**Stockinger**
	Monsieur	Prof.	Marc	**Robinson-Rechavi**
Expert·e·s	Monsieur	Prof.	Ioannis	**Xenarios**
	Monsieur	Prof.	Donald	**Kossmann**

le Conseil de Faculté autorise l'impression de la thèse de

Madame Ana Claudia Sima

Master in computer science, EPFL, Suisse

intitulée

Enabling complex semantic queries to bioinformatics databases through intuitive search over data

Lausanne, le 26 octobre 2020

pour le Doyen
de la Faculté de biologie et de médecine

Prof. Niko GELDNER
Directeur de l'Ecole Doctorale

Abstract

Data integration promises to be one of the main catalysts in enabling new insights to be drawn from the wealth of biological data already available publicly. However, the heterogeneity of the existing data sources still poses significant challenges for achieving interoperability among biological databases. Furthermore, merely solving the technical challenges of data integration, for example through the use of common data representation formats, leaves open the larger problem. Namely, the steep learning curve required for understanding the data models of each public source, as well as the technical language through which the sources can be queried and joined. As a consequence, most of the available biological data remain practically unexplored today.

In this thesis, we address these problems jointly, by first introducing an ontology-based data integration solution in order to mitigate the data source heterogeneity problem. We illustrate through the concrete example of Bgee, a gene expression data source, how relational databases can be exposed as virtual Resource Description Framework (RDF) graphs, through relational-to-RDF mappings. This has the important advantage that the original data source can remain unmodified, while still becoming interoperable with external RDF sources.

We complement our methods with applied case studies designed to guide domain experts in formulating expressive federated queries targeting the integrated data across the domains of evolutionary relationships and gene expression. More precisely, we introduce two comparative analyses, first within the same domain (using orthology data from multiple, interoperable, data sources) and second across domains, in order to study the relation between expression change and evolution rate following a duplication event.

Finally, in order to bridge the semantic gap between users and data, we design and implement Bio-SODA, a question answering system over domain knowledge graphs, that does not require training data for translating user questions to SPARQL. Bio-SODA uses a novel ranking approach that combines syntactic and semantic similarity, while also incorporating node centrality metrics to rank candidate matches for a given user question. Our results in testing Bio-SODA across several real-world databases that span multiple domains (both within and outside bioinformatics) show that it can answer complex, multi-fact queries, beyond the current state-of-the-art in the more well-studied open-domain question answering.

Résumé

L'intégration des données promet d'être l'un des principaux catalyseurs permettant d'extraire des nouveaux aperçus de la richesse des données biologiques déjà disponibles publiquement. Cependant, l'hétérogénéité des sources de données existantes pose encore des défis importants pour parvenir à l'interopérabilité des bases de données biologiques. De plus, en surmontant seulement les défis techniques de l'intégration des données, par exemple grâce à l'utilisation de formats standard de représentation de données, on laisse ouvert un problème encore plus grand. À savoir, la courbe d'apprentissage abrupte nécessaire pour comprendre la modélisation des données choisie par chaque source publique, ainsi que le langage technique par lequel les sources peuvent être interrogés et jointes. Par conséquent, la plupart des données biologiques publiquement disponibles restent pratiquement inexplorés aujourd'hui.

Dans cette thèse, nous abordons l'ensemble des deux problèmes, en introduisant d'abord une solution d'intégration de données basée sur ontologies, afin d'atténuer le problème d'hétérogénéité des sources de données. Nous montrons, à travers l'exemple de Bgee, une base de données d'expression de gènes, une approche permettant les bases de données relationnelles d'être publiés sous forme de graphes RDF (Resource Description Framework) virtuels, via des correspondances relationnel-vers-RDF (« relational-to-RDF mappings »). Cela présente l'important avantage que la source de données d'origine peut rester inchangé, tout en devenant interopérable avec les sources RDF externes.

Nous complétons nos méthodes avec des études de cas appliquées, conçues pour guider les experts du domaine dans la formulation de requêtes fédérées expressives, ciblant les données intégrées dans les domaines des relations évolutionnaires et de l'expression des gènes. Plus précisément, nous introduisons deux analyses comparatives, d'abord dans le même domaine (en utilisant des données d'orthologie provenant de plusieurs sources de données interopérables) et ensuite à travers des domaines interconnectés, afin d'étudier la relation entre le changement d'expression et le taux d'évolution suite à une duplication de gène.

Enfin, afin de mitiger le décalage sémantique entre les utilisateurs et les données, nous concevons et implémentons Bio-SODA, un système de réponse aux questions sur des graphes de connaissances domaine-spécifique, qui ne nécessite pas de données de formation pour traduire les questions des utilisateurs vers SPARQL. Bio-SODA utilise une nouvelle approche de classement qui combine la similarité syntactique et sémantique, tout en incorporant des métriques de centralité des nœuds, pour classer les possibles candidats en réponse à une question utilisateur donnée. Nos résultats suite aux tests effectués en utilisant Bio-SODA sur plusieurs bases de données à travers plusieurs domaines (tantôt liés à la bioinformatique qu'extérieurs) montrent que Bio-SODA réussit à répondre à des questions complexes, engendrant multiples entités, au-delà de l'état actuel de la technique en matière de systèmes de réponses aux questions sur les données structures, en particulier graphes de connaissances.

Acknowledgements

First, I would like to thank Prof. Kurt Stockinger, my main supervisor, for his guidance throughout my entire PhD, from teaching me the pragmatism that is often required in order to get things done, to providing a very friendly atmosphere at work and in our weekly discussions.

Next, I want to thank Prof. Christophe Dessimoz for his support and patience in helping me navigate through the challenges of interdisciplinary work and for his dedication to making all the members of his lab, even those only remotely connected, including me, feel as an integral part of the lab.

I am thankful to Prof. Marc Robinson-Rechavi for his timely advice, feedback on the progress of my thesis and active participation throughout the Bio-SODA project. I am grateful to all thesis committee members for their support from early on in my PhD, which gave me the confidence to continue pursuing this work.

Importantly, I want to thank the whole Bio-SODA team, especially Tarcisio and Erich, for their friendly advice and continued contributions to the Bio-SODA project, without whom many of the ideas in this thesis would not have materialized into working, usable and reusable prototypes! I am truly lucky to have been part of a great team that worked well together towards a common goal, in spite of our different locations, backgrounds and sometimes busy schedules. Thanks to the Swiss National Science Foundation, we had the valuable funding that made the Bio-SODA project and this thesis possible. I also want to say a big thank you to my colleagues at the ZHAW, for sharing parts of this roller-coaster journey with me, both through its ups and downs.

Finally, I would like to thank my sources of inspiration: my parents, for teaching me from an early age, through their own personal example, the role of perseverance in achieving goals even throughout harsher times; my husband, for his unconditional love and support, helping me maintain a much-needed balance throughout these years. Last, but not least, I thank all my professors, from past and present, for their pursuit in instilling in students a passion for learning and a drive to always keep an open mind.

Zurich & Lausanne,
2020

List of Publications

The following peer-reviewed publications present original contributions by the author, which are part of this dissertation:

1. **Sima, Ana Claudia**, Kurt Stockinger, Tarcisio Mendes de Farias, and Manuel Gil. *"Semantic integration and enrichment of heterogeneous biological databases."* In Evolutionary Genomics, pp. 655-690. Humana, New York, NY, 2019.

2. **Sima, Ana Claudia**[1], Tarcisio Mendes de Farias[1], Erich Zbinden, Maria Anisimova, Manuel Gil, Heinz Stockinger, Kurt Stockinger, Marc Robinson-Rechavi, and Christophe Dessimoz. "*Enabling semantic queries across federated bioinformatics databases*." Database 2019.

3. **Sima, Ana Claudia**, Christophe Dessimoz, Kurt Stockinger, Monique Zahn-Zabal, and Tarcisio Mendes de Farias. *"A hands-on introduction to querying evolutionary relationships across multiple data sources using SPARQL."* F1000Research 8, no. 1822 (2019): 1822.

In addition, the following publication is currently under preparation for submission:

1. **Sima, Ana Claudia**, Tarcisio Mendes de Farias, Maria Anisimova, Christophe Dessimoz, Marc Robinson-Rechavi, Erich Zbinden and Kurt Stockinger, *"BioSODA - A Question Answering System for Domain Knowledge Graphs"*.

Finally, the following publications are part of the author's PhD research, but present results that are not covered in this thesis:

1. **Sima, Ana Claudia**, Kurt Stockinger, Katrin Affolter, Martin Braschler, Peter Monte, and Lukas Kaiser. "*A hybrid approach for alarm verification using stream processing, machine learning and text analytics*." In EDBT 2018, Vienna, Austria, 26-29 March 2018. ACM, 2018.

2. Lehmann, Claude, Lilach Goren Huber, Thomas Horisberger, Georg Scheiba, **Sima, Ana Claudia** and Kurt Stockinger. "*Big Data architecture for intelligent maintenance: a focus on query processing and machine learning algorithms*." Journal of Big Data 7, no. 1 (2020): 1-26.

3. Sihem Amer-Yahia, Georgia Koutrika et al. (INODE consortium), *INODE: "Intelligent Data Exploration - Leveraging Machine Learning for Data Management Research" [submitted, under review]*.

[1] Equal contributions

Contents

List of Figures

List of Tables

List of Abbreviations

ABox	Assertional box
Bgee	dataBase for Gene Expression Evolution, https://bgee.org/
FK	Foreign key in a relational database
HBB	Hemoglobin unit beta gene
HDF5	Hierarchical Data Format version 5
HOG	Hierarchical orthologous group
IRI	Internationalized Resource Identifier
KG	Knowledge Graph
KGQA	Question Answering over Knowledge Graphs
OBDA	Ontology-based data access
OBDI	Ontology-based data integration
OMA	Orthologous Matrix, a database for the inference of orthologs among complete genomes, https://omabrowser.org
PAM	Point Accepted Mutation
PK	Primary key in a relational database
PK-FK	Primary key-foreign key relationship; enables joining two tables in a relational database
RDB	Relational database
RDF	Resource Description Framework
SODA	Search Over Data Warehouses [1]
SQL	Structured Query Language
SPARQL	SPARQL Protocol and RDF Query Language
TBox	Terminological box
URI	Uniform Resource Identifier

Chapter 1

Introduction

1.1 Motivation

Biological databases are growing at an exponential rate, both in number and in size, currently being among the major producers of Big Data, almost on par with commercial generators, such as YouTube or Twitter [2]. For instance, over one hundred key resources are featured in the yearly Nucleic Acids Research annual database issue [3]. While traditionally biological databases evolved as independent silos, each purposely built by a different research group in order to answer specific research questions, more recently significant efforts have been made toward integrating the available sources. However, the heterogeneity of these data sources, both at the syntactic and the semantic level, still poses significant challenges for achieving interoperability among biological databases. As a consequence, the burden ultimately remains on the end user to find the relevant sources, as well as the connections between them, in order to benefit from the data available publicly. This is today a largely manual and time-consuming process.

Data integration promises to be one of the main catalysts in enabling new insights to be drawn from the wealth of biological data available publicly. For instance, by comparing disease phenotypes in humans with phenotypes produced by particular mutations in model species, it is possible to infer which human genes are involved in the disease [4]. Semantic Web technologies have been key enablers for data integration, opening the path for new insights into the unified data, which were not visible at the level of each independent database. In the first part of this thesis, we propose an Ontology-Based Data Integration solution, that enables exposing existing relational databases as virtual RDF graphs, in order to make these interoperable with external RDF data sources.

However, merely solving the *technical* challenges of data integration, for example by publishing all data in a standard format, such as RDF, does not automatically render the data *usable* to the larger audience. This is because the technical query languages required to access this data (in particular, SPARQL) remain outside the realm of casual users and even of most domain experts. In addition, understanding the *data models* of each source is also a time-consuming process, which must be repeated for each new data source. It has been shown [5] that in fact natural language interfaces are a more suitable data access modality to the Semantic Web for casual users. It is perhaps surprising that the progress in information retrieval over documents, made by commercial search engines, has not been yet paired with equally powerful tools for exploring *structured* data, which still remains an open research problem today.

In this thesis, we describe the challenges of question answering over knowledge graphs (KGQA), focusing on *domain* knowledge graphs, such as scientific datasets. To tackle these challenges, we design and implement Bio-SODA, a generic KGQA

system, that does not rely on prior training data (which in many cases is scarce) for translating user questions to SPARQL. More precisely, considering that many databases have only recently been made available in RDF format, there are usually few question-answer pairs which the system can be trained or fine-tuned on. Bio-SODA overcomes this challenge by using a generic, graph-based approach, combined with a new ranking algorithm that takes into account node centrality metrics for scoring candidate query graphs. We demonstrate across three datasets that Bio-SODA can answer complex, multi-fact queries, beyond the current state-of-the-art in question answering over knowledge graphs. We conclude by outlining research directions for future work, including desirable design goals to increase the adoption of question answering systems in new domains in the future.

1.2 Contributions

This dissertation is a result of interdisciplinary research I have performed in the context of a collaboration between the University of Lausanne (UNIL) and the Zurich University of Applied Sciences (ZHAW). The thesis therefore has a strong *applied research* focus. As a result, its contributions are two-fold:

I. **Methods**

This thesis brings contributions to two broad research topics, namely Data Integration – which is a pre-requisite for semantic search over heterogeneous data sources – and Question Answering over Domain Knowledge Graphs.

In the following, I outline my contributions to the published work on which this dissertation is based (see List of Publications):

- I performed an experimental evaluation of the Ontology-Based Data Integration (OBDI) solution presented in Chapter 3 (Results section 3.4). The evaluation showed that relational-to-RDF mappings result in reasonable performance for practical applications and are therefore a feasible option for interoperating relational databases with existing RDF stores, through federated SPARQL queries.
- I designed and implemented, under the supervision of the Dessimoz group at UNIL, hands-on tutorials to guide domain experts in making use of the integrated data for comparative analyses, through federated SPARQL queries. The tutorials cover use-cases both within the same domain (orthology, in Chapter 4.1), as well as at the intersection between connected domains (gene expression and orthology, Chapter 4.2). Performing such analyses was made possible through the integration of data from the Bgee gene expression database and the OMA orthology database.
- I designed and implemented Bio-SODA (Chapter 5), a generic Question Answering System over Domain Knowledge Graphs, that does not require training data to generate SPARQL queries from a user question in natural language.
- I investigated new ranking algorithms that combine syntactic and semantic similarity, as well as node centrality in the knowledge graph. Many existing question answering systems either rely on simple metrics for ranking, such as the length of the answer query graph, or require extensive training data in order to learn a ranking function. Through the Evaluation in Section 5.5, I show that by taking into account all three factors

for ranking candidate queries (syntactic and semantic similarity, as well as node importance), Bio-SODA consistently outperforms existing state-of-the-art Question Answering systems across three benchmark datasets tested (both within and outside the Life Sciences).

- Finally, I provide an analysis of the limitations of Bio-SODA, outlining desirable design goals for generic question answering systems over domain knowledge graphs, in Chapter 5. This complements existing studies from literature, which have generally focused on open-domain question answering, and can thus serve as both a research agenda for future work and as a guideline for increasing the adoption of question answering systems in new domains in the future.

II. **Software**

For each method introduced in this thesis, I also made available, together with my co-authors, an open-source prototype implementing the method in order to facilitate reproducibility and re-use for future research. This is especially relevant in the context of domain question answering, where not many systems were publicly available for testing at the time of writing.

In more details, Chapter 3 proposes an ontology-based data integration approach, which relies on relational-to-RDF mappings in order to expose relational data as a virtual RDF graph, without requiring changes in the original data source. The mappings created for our example case-study, using the Bgee gene expression database, are made available in a github repository[1]. Furthermore, an intuitive template-based search interface that enables easy access to the integrated data has been developed by co-authors of [6]. All the code for this interface is available in the same github repository[1], while a demo of this is currently available at `http://biosoda.expasy.org/`.

Chapter 4 proposes two applied case studies, for which we make both data and code available. More precisely, Section 4.1 is a hands-on introduction to querying orthology databases using SPARQL. An accompanying Jupyter notebook, which enables easy testing of all protocols proposed in this chapter, as well as Supplementary Materials, are available at `https://github.com/biosoda/tutorial_orthology`. Similarly, data and code for the case study of the correlation between expression and evolutionary rates, presented in Section 4.2, is available in the github repository `https://github.com/anazhaw/tutorial_branch_length`.

Finally, the Bio-SODA system, introduced in Chapter 5, is available open-source at `https://github.com/anazhaw/Bio-SODA` and a demo is currently deployed for testing at `http://biosoda.expasy.org/welcome/`.

The software published has been reused in both academia (the INODE European project[2], where I am part of the INODE consortium at the time of writing and in which Bio-SODA is used for exploring RDF datasets of various domains, such as cancer research, policy research and astrophysics), as well as industry. The results presented in this thesis have partly contributed to new collaborations with industrial partners at both the University of Lausanne and the Zurich University of Applied Sciences.

[1] `https://github.com/biosoda/bioquery`

[2] see `http://www.inode-project.eu/`

1.3 Thesis Outline

This dissertation generally follows the order of publications I have co-written (either as first or co-first author).

Chapter 2, largely based on the book chapter [7], offers an introduction to the topics covered throughout the rest of this thesis. For example, the interested reader can learn about different database models - in particular relational and graph databases - as well as their advantages and limitations in the context of data integration. The chapter also introduces basic Semantic Web Technologies, such as the Resource Description Framework (RDF), the RDF schema (RDFS), ontologies and Ontology-Based Data Integration (OBDI), definitions that provide the background for understanding all subsequent chapters. Finally, the chapter also gives an overview of past and ongoing research efforts in data integration in the Life Sciences, therefore putting our current work into context.

Chapter 3, based on the Database journal publication [6], presents an Ontology-Based Data Integration solution, applied to the case of a relational database of gene expression (Bgee). We illustrate relational-to-RDF mappings, which enable the original relational data source to be exposed as a virtual RDF graph and by these means made interoperable with several external RDF bio-datasets, such as OMA and UniProt. This chapter also introduces BioQuery, a template-based search interface which enables domain experts to re-use and refine questions from a catalog of federated queries across the domains of gene expression, orthology and protein information.

Chapter 4 introduces two applied case studies illustrating the potential of data integration in testing new biological hypotheses. Therefore, this chapter is mainly addressed to domain experts across two fields of bioinformatics: evolutionary relationships (orthology) and gene expression. In more details, the first part of this chapter, Section 4.1, based on the publication [8], introduces a hands-on tutorial for querying orthology data across multiple data sources (OMA, OrthoDB, MBGD and EBI) using federated SPAQL queries. The second part of this chapter, Section 4.2 introduces a more in-depth case study on the correlation between gene expression evolution and branch length following a duplication. Rather than presenting novel results, Chapter 4 focuses on describing the relevant data models and exemplifying federated SPARQL queries, both within the orthology domain, as well as in connection with the gene expression domain. The queries presented here can be used as a starting point in testing new research hypotheses with the interoperable RDF data available across multiple sources. Readers interested strictly in the computational aspects of this thesis might skip this chapter and go directly to Chapter 5.

Chapter 5 introduces Bio-SODA, a question answering system for domain knowledge graphs. Here, we first discuss the challenges of question answering over domain datasets, such as scientific datasets. Next, we present the design and implementation of the Bio-SODA system, as well as results obtained across three real-world datasets, which we compare against state-of-the-art question answering systems. Finally, we conclude with lessons learned from our experience in using Bio-SODA across real-world datasets and testing the system with early adopters. We formulate design goals for generic question answering systems over domain knowledge graphs, meant to highlight research directions that can help to increase the practical utility and adoption of such systems in new domains in the future.

Chapter 6 provides a more general outlook into future work and Chapter 7 concludes this dissertation.

Chapter 2

Semantic Integration and Enrichment of Heterogeneous Biological Databases

Biological databases are growing at an exponential rate, currently being among the major producers of Big Data, almost on par with commercial generators, such as YouTube or Twitter [2]. While traditionally biological databases evolved as independent silos, each purposely built by a different research group in order to answer specific research questions, more recently significant efforts have been made toward integrating these heterogeneous sources into unified data access systems or interoperable systems using the FAIR [9], [10] principles of data sharing. Semantic Web technologies have been key enablers in this process, opening the path for new insights into the unified data, which were not visible at the level of each independent database. In this chapter[1], we first provide an introduction into two of the most used database models for biological data: relational databases and RDF stores. Next, we discuss ontology-based data integration, which serves to unify and enrich heterogeneous data sources. We present an extensive timeline of milestones in data integration based on Semantic Web technologies in the field of Life Sciences. Finally, we discuss some of the remaining challenges in making ontology-based data access systems easily accessible to a larger audience. In particular, we introduce natural language search interfaces, which alleviate the need for database users to be familiar with technical query languages. We illustrate the main theoretical concepts of data integration through concrete examples, using two well-known biological databases: a gene expression database, Bgee [11], and an orthology database, OMA [12].

2.1 Introduction

Biological databases have grown exponentially in recent decades, both in number and in size, owing primarily to modern high-throughput sequencing techniques [2]. Today, the field of genomics is almost on par with the major commercial generators of Big Data, such as YouTube or Twitter, with the total amount of genome data doubling approximately every seven months [2]. While most biological databases have initially evolved as independent silos, each purpose-built by a different research group in order to collect data and respond to a specific research question, more recently significant efforts have been made towards integrating the different data sources, with the aim of enabling more powerful insights from the aggregated data, which would not be visible at the level of individual databases.

[1] Parts of this chapter have been published in [7]

[2] http://www.sciencenews.org/article/gene-sequencing-future-here

Let us consider the following example. An evolutionary biologist might want to answer the question *"What are the human-rat orthologs, expressed in the liver, that are associated with leukemia?"*. Getting an answer for this type of question usually requires information from at least three different sources: an orthology database (e.g. OMA [12], OrthoDB [13] or EggNog [14]), a gene expression database, such as Bgee [11], ArrayExpress [15] or The Human Protein Atlas [16], and a proteomics database containing disease associations (e.g. UniProt [17]). In the lack of a unified access to the three data sources, obtaining this information is a largely manual and time-consuming process. First, the biologist needs to know which databases to search through. Second, depending on the interface provided by these databases, he or she might need to be familiar with a technical query language, such as SQL or SPARQL (note: a list of acronyms is provided at the beginning of this dissertation). At the very least, the biologist is required to know the specific identifiers (IDs) and names used by the research group that created the database, in order to search for relevant entries. An integrated view, however, would allow the user to obtain this information automatically, without knowing any of the details regarding the structure of the underlying data sources nor the type of storage these databases use and eventually not even specific IDs (such as protein or gene names).

Biological databases are generally characterized by a large heterogeneity, not only in the type of information they store, but also in the model of the underlying data store they use: examples would be relational databases, file-based stores, graph-based etc. Some of the databases considered fundamental to research in the life sciences can be found in the ELIXIR Europe's Core Data Resources, available online at `https://www.elixir-europe.org/platforms/data`. In this chapter, we will mainly discuss two types of database models: the relational model (i.e. relational databases) and a graph-based data model, namely RDF (the Resource Description Framework).

Database systems have been around since arguably the same time as computers themselves, serving initially as "digitized" copies of tabular paper forms, for example in the financial sector, or for managing airline reservations. In biology, some of the earliest examples include the 1965 Atlas of Protein Structures [18] and the 1971 Protein Databank [19]. Relational databases, as well as the mathematical formalism underlying them, namely, the relational algebra, were formalized in the 1970s by E.F. Codd, in a foundational paper that now has surpassed 10,000 citations [20]. The Relational Model is designed to structure data into so-called tuples, according to a predefined schema. Tuples are stored as rows in tables (also called "relations"). Each table usually defines an entity, such as an object, a class or a concept, whose instances (the tuples) share the same attributes. Examples of relations are "Gene", "Protein", "Species", etc. The attributes of the relation will represent the columns of the table, for example "gene name". Furthermore, each row has a unique identifier. The column (or combination of columns) that stores the unique identifier is called a Primary Key and can be used not only to uniquely identify rows within a table, but also to connect data between multiple tables, through a Primary Key-Foreign Key relationship. Doing such a connection is called a join. In fact, a join is only one of the operations defined by relational algebra. Other common operations include projection, selection and others. The operands of relational algebra are the database tables, as well as their attributes, while the operations are expressed through the Structured Query Language (SQL). For a more in-depth discussion on relational algebra we refer the reader to the original paper by E.F. Codd [20].

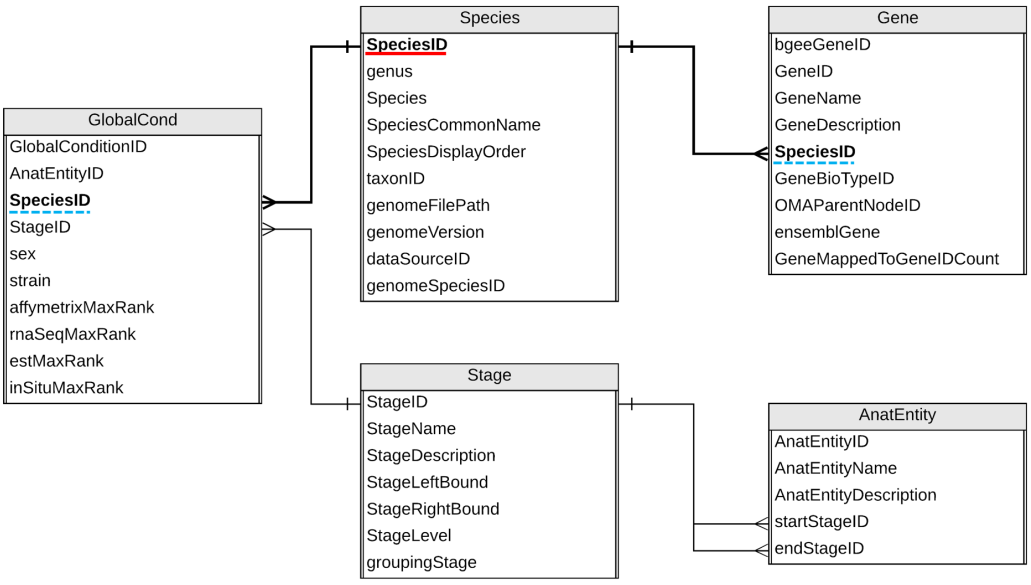

FIGURE 2.1: Sample Relational Database (extracted from the gene expression database Bgee). Tables are connected through Primary Key - Foreign Key relationships. Two example relationships, based on the SpeciesID, are higlighted in bold. The Primary Key attribute is underlined in red, while the two Foreign Keys are underlined with a dash line in blue. The PK-FK relationships enable joining the connected tables, for instance to retrieve information about the *Species* where a given *Gene* can be found.

This chapter is structured as follows. In Section 2.2, we give a brief introduction to relational databases, through the concrete example of the Bgee gene expression database. We introduce the basics of Semantic Web technologies in Section 2.3. Readers who are already familiar with the Semantic Web stack might skip Section 2.3 and jump directly to Section 2.4, which presents an applied use case of Semantic Web technologies in the life sciences: modeling the Bgee and OMA databases. Section 2.5 represents the core of this chapter. Here, we present Ontology Based Data Integration (Section 2.5.1) and illustrate it through the concrete example of a unified ontology for Bgee and OMA (Section 2.5.2), as well as the mechanisms required to further extend the integrated system with other heterogeneous sources such as the UniProt protein knowledge base (Section 2.5.3). We introduce natural language interfaces, which enable easy data access even for non-technical users, in Section 2.5.4. We present an extensive timeline of milestones in data integration based on Semantic Web technologies in the field of Life Sciences in Section 2.6. Finally, we conclude in Section 2.7.

2.2 Modeling a Biological Database with Relational Database Technology

In this section we will demonstrate how to model a biological database with relational database technology.

Figure 2.1 illustrates the data model of a sample extracted from the Bgee database.

The sample contains 5 tables and their relationships, shown as arrows, where the direction of the arrow is oriented from the Foreign Key of one table to the Primary Key of a related one. For example, the Primary Key (PK) of the Species table, underlined in red, is the SpeciesID. Following the relationships highlighted in bold, we see that the SpeciesID also appears in the two tables connected to Species: GlobalCond and Gene. In these tables, the attribute plays the role of a Foreign Key (FK), highlighted in the figure with dash underline in blue. The PK-FK relationships allow combining or aggregating data from related tables. For example, by joining Species and Gene, through the SpeciesID, we can find to which species a gene belongs. Concretely, let's assume we want to find the species where the gene "HBB" can be found. Given that this information is stored in the *SpeciesCommonName* attribute, we can retrieve it through the following SQL query:

```
SELECT SpeciesCommonName from Species JOIN Gene
WHERE Gene.GeneName = "HBB" and Species.SpeciesID = Gene.SpeciesID;
```

This query enables retrieving (via the 'SELECT' keyword) the attribute corresponding to the species name (*SpeciesCommonName*) by joining the Species and Gene tables, based on their Primary Key - Foreign Key relationship, namely, via the SpeciesID, on the condition that the GeneName exactly matches "HBB". For a more detailed introduction to the syntax and usage of SQL, we refer the reader to an online introductory tutorial [3], as well as the more comprehensive textbooks [21], [22].

Taking this a step further, we can imagine the case where a second relational database also stores information about genes, but perhaps with some additional data, such as associations with diseases. Can we still combine information across these distinct databases? Indeed, as long as there is a common point between the tables in the two databases, such as the GeneID or the SpeciesID, it is usually possible to combine them into a single, federated database, and use SQL to query it through federated joins. An example of using federated databases for biomedical data is presented in [23].

2.2.1 Limitations and emerging solutions for data integration

So far, we have seen that relational databases are a mature, highly optimized technology for storing and querying structured data. Also, combined with a powerful and expressive query language, SQL, they allow users to federate (join) data even from different databases.

However, there are certain relationships that are not natural for relational databases. Let us consider the relationship "hasOrtholog". Both the domain and the range of this relationship, as defined in the Orthology Ontology [24], are the same – a Gene. For example, the Hemoglobin (HBB) gene in human has the Hbb-bt orthologous gene in the mouse (expressed via the relation hasOrtholog). In the relational database world, this translates into a so-called "self-join". As the name suggests, this requires joining one table in this case, Gene, with itself, in order to retrieve the answer. These types of "self-join" relations, while frequent in the real world (e.g. a manager of an Employee is also an Employee; a friend of a Person is also a Person, etc.), are inefficient in the context of relational databases. While there are sometimes ways to avoid self-joins, these require even more advanced SQL fluency on the part of the programmer [4].

[3] https://www.w3schools.com/sql/sql_intro.asp

[4] http://sqltouch.blogspot.ch/2013/07/self-join-incurs-more-io-activities-and.html

Moreover, relational databases are typically not well-suited for applications that require frequent schema changes. Hence, NoSQL stores have gained widespread popularity as an alternative to traditional relational database management systems [25], [26]. These systems do not impose a strict schema on the data and are therefore more flexible than relational databases in the cases where the structure of the data is likely to change over time. In particular, graph databases, such as Virtuoso [27], are very well suited for data integration, as they allow easily combining multiple data sources into a single graph. We discuss this in more detail in Section 2.3.

These and other considerations have led to the vision of the Semantic Web, formalized in 2001 by Tim Berners Lee et al. [28]. At a high-level, the Semantic Web allows representing the semantics of data in a structured, easy to interlink, machine-readable way, typically by use of the RDF graph-based data model. The gradual adoption of RDF stores, although widespread in the web context and in the life sciences in particular, did not replace relational databases altogether, which lead to a new challenge: how will these heterogeneous data sources now be integrated?

Initial integration approaches in the field of biological databases have been largely manual: first, many of these sources (either relational or graph-based) have included cross-references to other sources. For example, UniProt contains links to more than 160 other databases. However, this raises a question for the user: which of the provided links should be followed in order to find relevant connections? While a user can be assumed to know the contents of a few related databases, we can hardly expect anyone to be familiar with more than 160 of them! To avoid this problem, other databases have chosen an orthogonal approach: instead of referencing *links* to other sources, simply *copy* the relevant data from those sources into the database. This approach also has a few drawbacks. First, it generates redundant data (which might result in significant storage space consumption) and, most importantly, it might lead to the use of stale, outdated results. Moreover, this approach is contradictory to best practices of data warehousing used widely across various domains in industry. For a discussion on this we refer the reader to [29].

Databases such as UniProt are highly comprehensive, with new results being added to each release, results that may sometimes even contradict previous results. Duplication of this data into another database can quickly lead to missing out the most recent information or to high maintenance efforts required to keep up with the new changes. In the following sections, we discuss an alternative approach: integrating heterogeneous data sources through the use of a unifying data integration layer, namely, an integrative ontology, that aligns, but also enriches the existing data, with the purpose of facilitating knowledge discovery.

Throughout the remainder of this chapter we will combine theoretical aspects of data integration with concrete examples, based on the SODA project [1], which aimed to enable keyword serch over data warehouses, as well as from our ongoing research project, Bio-SODA, where we are currently building an integrated data access system for biological databases (starting with OMA and Bgee), using a natural language search interface. In the context of this project, Semantic Web technologies, such as RDF, are used to enhance interoperability among heterogeneous databases at the semantic level (e.g. RDF graphs with predefined semantics). Moreover, currently, several life science and biomedical databases such as OMA [30], UniProt [17], neXtProt [31], The European Bioinformatics Institute (EMBL-EBI) RDF data [32], the WorldWide Protein Data Bank [33] already provide RDF data access, which also justifies an RDF based approach to enable further integration efforts to include these databases. A recent initiative for (biological) data sharing is based on the FAIR principles [9], aiming to make data **f**indable, **a**ccessible, **i**nteroperable and **r**e-usable.

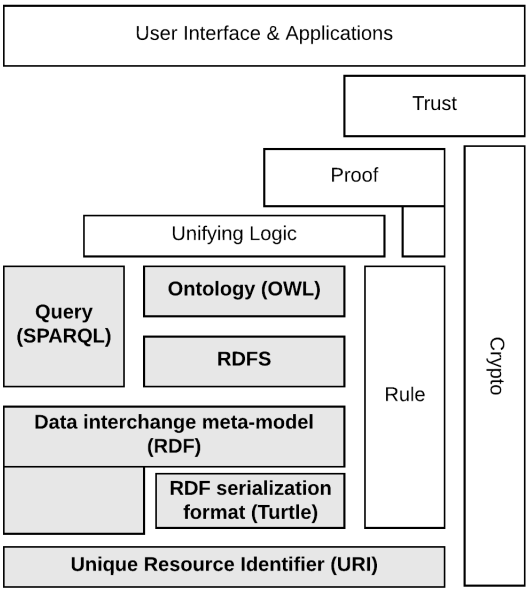

FIGURE 2.2: The Semantic Web stack (modified from [38]).

2.3 Semantic Web Technologies

The Semantic Web, as its name shows, emerged mainly as a means to attach semantics (meaning) to data on the Web [28]. In contrast to relational databases, Semantic Web technologies rely on a graph data model, in order to enable interlinking data from disparate sources available on the Web. Although the vision of the Semantic Web still remains an ideal, many large datasets are currently published based on the Linked Data principles [34] using Semantic Web technologies (e.g. RDF). The Linked Open Data Cloud illustrates a collection of a large number of different resources including DBPedia, UniProt and many others.

In this section, we will describe the semantic web (SW) stack, focusing on the technologies that enhance data integration and enrichment. For a more complete description of the SW stack, we refer the reader to the comprehensive introductions in [35], [36], [37].

The Semantic Web stack is presented in Figure 2.2. We will focus on the following standards or layers of the stack: URI, the syntax layer (e.g. Turtle (TTL), an RDF serialization format), RDF, OWL, RDFS and SPARQL. These layers are highlighted in grey in Figure 2.2.

2.3.1 Unique Resource Identifier (URI)

A Uniform Resource Identifier (URI) is a character sequence that identifies an abstract or physical resource. A URI is classified as a locator, a name, or both. The Uniform Resource Locators (URLs) are a subset of URIs that, in addition to identifying a resource, provide a means of locating the resource by describing its primary access or network "location". For example, https://bgee.org is a URI that identifies a resource (i.e. the Bgee gene expression website) and it implies solely a representation of this resource (i.e. an HTML web page). This resource is accessible through the HTTPS protocol.

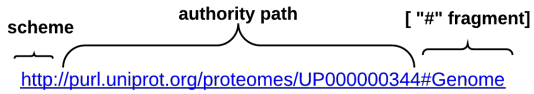

FIGURE 2.3: An example of a UniProt URI with a fragment.

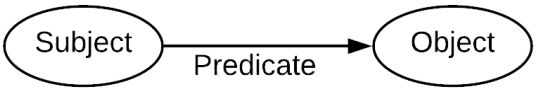

FIGURE 2.4: An RDF graph with two nodes (Subject and Object) and an edge connecting them (Predicate).

The Uniform Resource Name (URN) is also a URI that refers to both the "urn" scheme [5], which are URIs required to remain globally unique and persistent even when the resource does not exist anymore or becomes unavailable, and to any other URI with the properties of a name. For example, the URN `urn:isbn:978-1-61779-581-7` is a URI that refers to a previous edition of this book by using the International Standard Book Number (ISBN). However, no information about the location and how to get this resource (book) is provided.

The URI syntax consists of a hierarchical sequence of components referred to as the scheme, authority, path, query, and fragment[6]. Figure 2.3 describes a UniProt URI that includes these components.

An individual scheme does not have to be classified as being just one of "name" or "locator". Instances of URIs from any given scheme may have the characteristics of names (URN) or locators (URL) or both (URN + URL). Further examples of URIs with variations in their syntax components are:

- `ftp://ftp.bgee.org/current/download/calls/expr_calls/Sus_scrofa_expr_simple_development.tsv.zip`
- `http://www.ensembl.org/Multi/Search/Results?q=BRCA2`
- `mailto:Bgee@sib.swiss`
- `urn:miriam:pubmed:26615188`
- `https://www.ncbi.nlm.nih.gov/pubmed/26615188`

2.3.2 Resource Description Framework (RDF)

The Resource Description Framework (RDF) is a framework for describing information about resources in the World Wide Web, which are identified with URIs. In the previous section, we have seen that data in relational databases is organized into tables, according to some predefined schema. In contrast, in RDF stores data is mainly organized into triples, namely <*subject, predicate, object*>, similarly to how sentences in natural language are structured. An informal example would be: <*Bob, isFriendOf, Alice*>. A primer on triples and the RDF data model, using this simple example, is available online [7]. Figure 2.4 illustrates the RDF triple: the Subject represents the resource being described, the Predicate is a property of that resource and finally the Object is the value of the property (i.e. an attribute of the subject).

[5] `https://tools.ietf.org/html/rfc2141`
[6] see `https://tools.ietf.org/html/rfc3986`
[7] `https://www.w3.org/TR/rdf11-primer/`

Triples can be defined using the RDF framework. The data store for RDF data is also called a "triple store". Moreover, in analogy to the data model (or the schema) of a relational database, the high-level structure of data in a triple store can be described using an ontology. According to Studer et al. [39], an ontology is a formal, explicit specification of a shared conceptualization. "Formal" refers to the fact that the expressions must be machine readable, hence natural language is excluded. In this context, we can mention description logic (DL) [40] based languages, such as OWL 2 DL (see Subsection 2.3.3 for further details) to define ontologies. A DL ontology is the equivalent of a knowledge base (KB). A KB is mainly composed of two components that describe different statements in ontologies: the terminological box (TBox, *i.e.* the schema) and the assertional box (ABox, *i.e.* the data). Therefore, the conceptual statements form the set of TBox axioms, whereas the instance level statements form the set of ABox assertions. To exemplify this, we can mention the following DL axioms: $Man \equiv Human \sqcap Male$ (a TBox axiom that states a Man is a Human and Male) and *john:Man* (an ABox assertion that states john is an instance of Man).

Given that one of the goals of the Semantic Web is to assign unambiguous names to resources (URIs), an ontology should be more than a simple description of data in a particular triple store. Rather, it should more generally serve as a description of a domain, for instance Genomics (see Gene Ontology [41]) or Orthology (see Orth Ontology [24]). Different instantiations of this domain, for example, by different research groups, should reuse and extend this ontology. Therefore, constructing good ontologies requires careful consideration and agreement between domain specialists, with the goal of formally representing knowledge in their field. As a consequence, ontologies are usually defined in the scope of consortiums – such as the Gene Ontology Consortium [42] or the Quest for Orthologs Consortium [43]. A notable collaborative effort is the Open Biological and Biomedical Ontology (OBO) Foundry [44]. It established principles for ontology development and evolution, with the aim of maximizing cross-ontology coordination and interoperability, and provides a repository of Life-Science Ontologies, currently, including about 140 ontologies.

To give an example of RDF data in a concrete life sciences use case, let us consider the following RDF triples, which illustrate a few of the assertions used in the OMA orthology database to describe the human hemoglobin protein ("HBB"), using the first version of the ORTH ontology [24]:

```
oma:PROTEIN_HUMAN04027 rdf:type orth:Protein;
    oma:geneName "HBB";
    biositemap:description "Hemoglobin subunit beta";
    obo:RO_0002162 <http://www.uniprot.org/taxonomy/9606>.
```

This simple example already illustrates most of the basics of RDF. The instance that is being defined – the HBB protein in human has the following URI in the OMA RDF store: `http://omabrowser.org/ontology/oma#PROTEIN_HUMAN04027` .

The URI is composed of the OMA prefix, `http://omabrowser.org/ontology/oma#` (abbreviated here as "oma:") and a fragment identifier, PROTEIN_HUMAN04027. The first triple describes the type of this resource – namely, an *orth:Protein* – based on the Orthology Ontology, prefixed here as "orth:", `http://purl.org/net/orth#`. As mentioned previously, this is a higher-level ontology, which OMA reuses and instantiates. It is important to note that other ontologies are used as well in the remaining assertions: for example, the last triple references the UniProt taxonomy ID 9606. This is based on the National Center for Biotechnology Information (NCBI) organismal taxonomy [45]. If we follow the link in a Web browser, we see that

it identifies the "Homo Sapiens" species, while the property obo:RO_0002162 (i.e. http://purl.obolibrary.org/obo/RO_0002162) simply denotes "in taxon" in OBO [44]. Lastly, the concept also has a human-readable description, "Hemoglobin subunit beta".

2.3.3 RDF Schema (RDFS)

RDF Schema (RDFS) provides a vocabulary for modelling RDF data and is a semantic extension of RDF. It provides mechanisms for describing groups (i.e. classes) of related resources and the relationships between these resources. The RDFS is defined in RDF. The RDFS terms are used to define attributes of other resources such as the domains (*rdfs:domain*) and ranges (*rdfs:range*) of properties. Moreover, the RDFS core vocabulary is defined in a namespace informally called rdfs here and it is conventionally associated with the prefix *rdfs:*. That namespace is identified by the URI http://www.w3.org/2000/01/rdf-schema#.

In this section, we will mostly focus on the RDF and RDFS terms used in this chapter. Further information about RDF/RDFS terms is available in [46].

Classes:

- **rdfs:Resource** all things described by RDF are called resources, which are instances of the class rdfs:Resource (i.e. rdfs:Resource is an instance of rdfs:Class).
- **rdfs:Class** is the class of resources that are RDF classes. Resources that have properties (attributes) in common may be divided into classes. The members of a class are instances.
- **rdf:Property** is a relation between subject and object resources, i.e. a predicate. It is the class of RDF properties.
- **rdfs:Literal** is the class of literal values such as textual strings and integers. rdfs:Literal is a subclass of rdfs:Resource.

Properties:

- **rdfs:range** is an instance of rdf:Property. It is used to state that the values of a property are instances of one or more classes. For example, orth:hasHomolog rdfs:range orth:SequenceUnit (see Figure 2.5a). This statement means that the values of orth:hasHomolog property can only be instances of orth:SequenceUnit class.
- **rdfs:domain** is an instance of rdf:Property. It is used to state that any resource that has a given property is an instance of one or more classes. For example, orth:hasHomolog rdfs:domain orth:SequenceUnit (see Figure 2.5b). This statement means that resources that assert the orth:hasHomolog property must be instances of orth:SequenceUnit class.
- **rdf:type** is an rdf:Property that is used to state that a resource is an instance of a class.
- **rdfs:subClassOf** is an rdf:Property to assert that all instances of one class are instances of another. For example, if C1 rdfs:subClassOf C2 then an instance of C1 is also an instance of C2 but not vice versa.

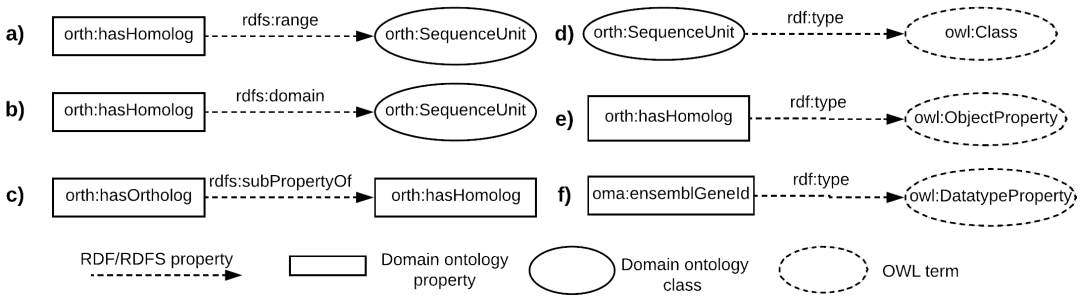

FIGURE 2.5: Examples of RDF/RDFS statements.

- **rdfs:subPropertyOf** is used to state that all resources related by one property (i.e. the subject of rdfs:subPropertyOf) are also related by another (i.e. the object of rdfs:subPropertyOf, the "super-property"). For example, all orthologous relations are also homologous relations. Because of this, in the latest release candidate of the Orthology ontology [24], it is stated that orth:hasOrtholog is a sub-property of orth:hasHomolog. Figure 2.5c illustrates this statement.

2.3.4 Web Ontology Language (OWL)

The first level above RDF/RDFS in the Semantic Web stack (see Figure 2.2) is an ontology language that can formally describe the meaning of resources[8]. If machines are expected to perform useful reasoning tasks on RDF data, the language must go beyond the basic semantics of RDF Schema. Because of this, OWL and OWL 2 (i.e. Web Ontology languages) include more terms for describing properties and classes, such as relations between classes (e.g. disjointness, owl:disjointWith), cardinality (e.g. "exactly 2", owl:cardinality), equality (i.e. owl:equivalentClass), richer typing of properties, characteristics of properties (e.g. symmetry, owl:SymmetricProperty), and enumerated classes (i.e. owl:oneOf). The owl: prefix replaces the following URI namespace: `http://www.w3.org/2002/07/owl#`.

As a full description of OWL and OWL 2 is beyond the scope of this chapter we refer the interested reader to [47]. In the following, we focus solely on some essential modeling features that the OWL languages offer in addition to RDF/RDFS vocabularies.

- **owl:Class** is a subclass of rdfs:Class. Like rdfs:Class, an owl:Class groups instances that share common properties. However, this new OWL term is defined due to the restrictions on DL-based OWL languages (e.g. OWL DL and OWL Lite; OWL 2 DL and its syntactic fragments EL, QL and RL). These restrictions imply that not all RDFS classes are legal OWL DL / OWL 2 DL classes. For example, the orth:SequenceUnit entity in the ORTH ontology is stated as an OWL class (*i.e.* orth:SequenceUnit rdf:type owl:Class Figure 2.5d illustrates this axiom). Therefore, orth:SequenceUnit is also an RDFS class since owl:Class is a subclass of rdfs:Class.

- **owl:ObjectProperty** is a subclass of rdf:Property. The instances of owl:ObjectProperty are object properties that link individuals to individuals (*i.e.* members of an owl:Class). For example, the orth:hasHomolog object property (see Figure 2.5e) relates one orth:SequenceUnit individual to another one. Figure 2.6a illustrates this example.

[8] see `https://www.w3.org/TR/owl-features/`

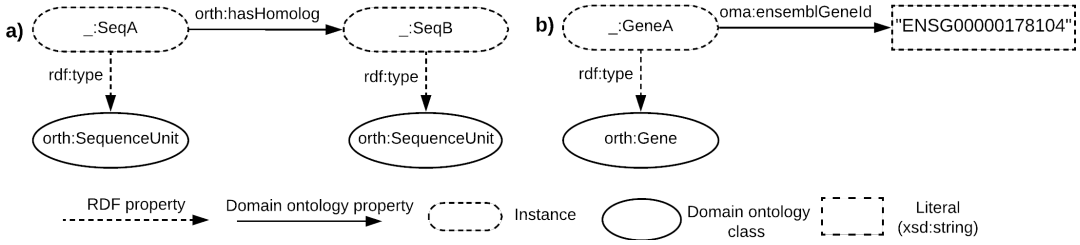

FIGURE 2.6: Examples of instances of orth:SequenceUnit and orth:Gene and object and datatype property assertions.

- **owl:DatatypeProperty** is a subclass of rdf:Property. The instances of owl:DatatypeProperty are datatype properties that link individuals to data values. To illustrate a datatype property, we can mention the oma:ensemblGeneId (see Figure 2.5f and Figure 2.6b). This property asserts a gene identifier to an instance of an orth:Gene.

Further information about OWL languages are available as World Wide Web Consortium (W3C) recommendations in the OWL2 overview online [9] and the OWL reference online [10].

2.3.5 RDF Serialization Formats

RDF is a graph based data model which provides a grammar for its syntax. Using this grammar, RDF syntax can be written in various concrete formats which are called RDF serialization formats. For example, we can mention the following formats: Turtle[11], RDF/XML (an XML syntax for RDF)[12] and JSON-LD (a JSON syntax for RDF) footnotehttps://www.w3.org/TR/json-ld/. In this section, we will solely focus on the Turtle format.

The Turtle language (TTL) allows for writing RDF graph in a compact textual form. To exemplify this serialization format, let us consider the following turtle document that defines the homologous and orthologous relations:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix orth: <http://purl.org/net/orth#> .

# http://purl.org/net/orth#SequenceUnit
orth:SequenceUnit rdf:type owl:Class .

orth:hasHomolog rdf:type owl:ObjectProperty ;
                rdf:type owl:SymmetricProperty ;
                rdfs:domain orth:SequenceUnit ;
                rdfs:range orth:SequenceUnit .

orth:hasOrtholog rdf:type owl:ObjectProperty ;
                 rdfs:subPropertyOf orth:hasHomolog .
```

[9] https://www.w3.org/TR/owl2-overview/
[10] https://www.w3.org/TR/owl-ref/
[11] http://www.w3.org/TR/turtle/
[12] https://www.w3.org/TR/rdf-syntax-grammar/

This example introduces many of features of the Turtle language: @prefix and prefixed names (e.g. *@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>*); predicate lists separated by ";" (e.g. *orth:hasOrtholog rdf:type owl:ObjectProperty; rdfs:subPropertyOf orth:hasHomolog.*); comments prefixed with "#" (e.g. # `http://purl.org/net/orth#SequenceUnit`) and a simple triple where the subject, predicate and object are separated by white spaces and ended with a "." (e.g. *orth:SequenceUnit rdf:type owl:Class.*).

Further details about TTL serialization is available as a W3C recommendation in the Turtle recommendation online [13].

2.3.6 Querying the Semantic Web with SPARQL

Once we have defined the knowledge base (TBox and ABox), how can we use it to retrieve relevant data? Similar to SQL for relational databases, data in RDF stores can be accessed by using a query language. One of the main RDF query languages, especially used in the field of life sciences, is SPARQL [14]. A SPARQL query essentially consists of a graph pattern, namely, conjunctive RDF triples, where the values that should be retrieved (the unknowns either subjects, predicates, or objects), are replaced by variable names, prefixed by "?". Looking again at the previous example, if we want to get the description of the "HBB" protein from OMA, we would simply use a graph pattern, where the value of the 'description' the one we want to retrieve is replaced by a variable as follows:

```
SELECT ?description WHERE {
        ?protein oma:geneName "HBB".
        ?protein biositemap:description ?description.
}
```

The choice of variable name itself is not important (we could have used "?x", "?var", etc. as well, albeit with a loss of readability). Essentially, we are interested in the description of a protein about which we only know a name "HBB".

2.4 Modeling Biological Databases with Semantic Web Technologies

In this section we show a concrete example of how we can use Semantic Web technologies to model the two biology databases Bgee and OMA.

Figure 2.7 illustrates a fragment of a candidate ontology describing the relational database sample from Bgee (see Figure 2.1). The ellipses illustrate classes of the ontology, either specific to the Bgee ontology, such as AnatomicEntity (the equivalent of the anatEntity table in the relational view), or classes from imported ontologies, such as the Taxon class (the prefix "up:" denoting the UniProt ontology, `http://purl.uniprot.org/core/`). The advantage of using external (i.e. imported) classes is that integration with other databases which also instantiate these classes will be much simpler. For example, we will see that the class Gene serves as the "join point" between OMA and Bgee. Arrows define properties of the ontology: either datatype properties (similar to attributes of a table in the relational world), such as the speciesName or the stageName, or object properties, which are similar to Primary-Key/Foreign-Key relationships, given that they link instances of one class to those of another. If we compare Figure 2.7 (the ontology view) against Figure

[13] `http://www.w3.org/TR/turtle/`

[14] see `https://www.w3.org/TR/rdf-sparql-query/`

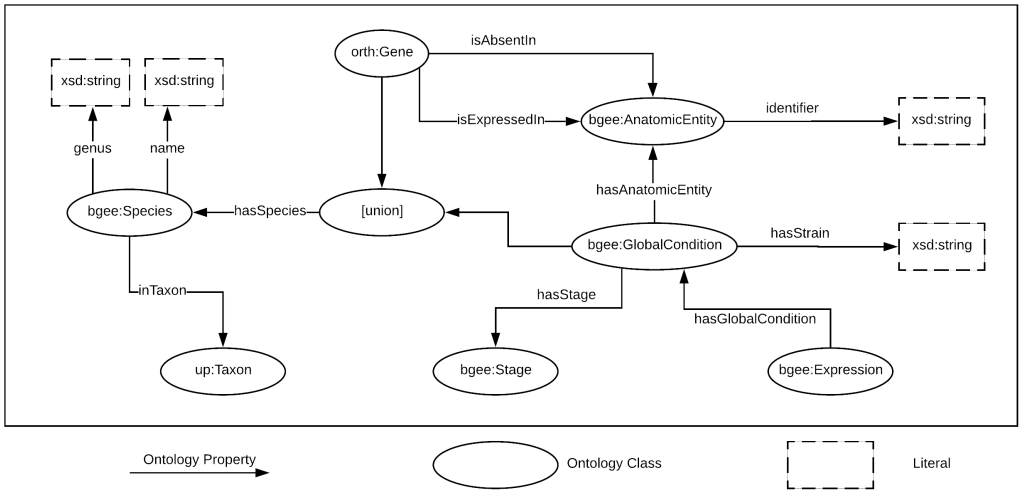

FIGURE 2.7: A portion of the ontology defined over the relational database sample from Bgee. For readability purposes, we omitted the namespace (*"bgee:"*) for the ontology properties.

2.1 (the relational view), we notice that the object properties isExpressedIn and isAbsentIn only appear explicitly in the ontology. This is because the values of these properties will actually be calculated on-the-fly, from multiple attributes in the relational database. Given that Bgee is mainly used to query gene expression, these properties are exposed as new semantic properties in the domain ontology, namely expression or absence of expression of a gene in a particular anatomic entity. This is one of the means through which the semantic layer can not only describe, but also enrich the data available in the underlying layers (in this case, in the relational database). The domain of both the isExpressedIn and isAbsentIn properties is in this case a Gene, while the range is an anatomic entity, such that triples that instantiate this relationship will have the structure: <*Gene, isExpressedIn, AnatomicEntity*>.

Given that the OMA ontology is significantly larger than the one for Bgee, we only show here the class hierarchy in Figure 2.8. The most important concepts in the ontology are shown in the top right corner, namely, the Cluster of Orthologs and the Cluster of Paralogs, which store information about gene orthology (or paralogy) in a hierarchical tree structure (the Gene Tree Node). Similarly to the Bgee ontology, the Gene class in OMA is external. Arrows indicate the "rdfs:subClassOf" relationship for example, both the 'Cluster of Orthologs' and the 'Cluster of Paralogs' classes are subclasses of the 'Cluster of Homologs' class. For a description of the ontology, as well as a discussion regarding its design within the Quest for Orthologs Consortium, we point the reader to [24]. Furthermore, the ontology can be explored or visualized in WebVOWL [48] using the web page of the OMA SPARQL Endpoint [12] available online at `https://sparql.omabrowser.org/sparql`.

Until here we have explored a few relatively simple examples in order to get familiar with the basics of Semantic Web technologies (URIs, RDF triples and SPARQL). However, we can now introduce a more complex query, that will better illustrate the expressivity of the SPARQL query language for accessing RDF stores that is, for integrating and joining data across different databases.

Since all RDF stores structure data using the same standard model for data interchange, the main requirements in order to efficiently join multiple sources are:

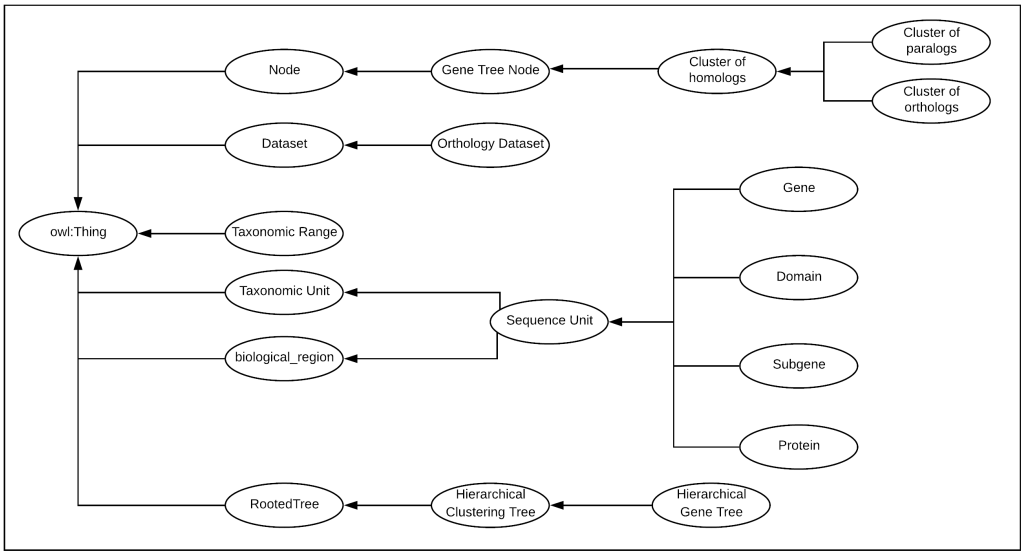

FIGURE 2.8: The class hierarchy of the OMA ontology. Ellipses indicate class labels, while arrows indicate the *"rdfs:subClassOf"* property. Further details are available in [24].

1. that they each expose data through a SPARQL endpoint that supports federation (SPARQL 1.1) and

2. that the sources share URIs or ontologies.

This is the reason why already today we can jointly query for example OMA and UniProt – essentially, integrating the two databases by means of executing a federated SPARQL query.

To illustrate this, let us consider the following example: *what are the human genes available in the OMA database, that have a known association with leukemia*? OMA does not contain any information related to diseases, however, UniProt does. In this case, since OMA already cross-references UniProt with the oma:xrefUniprot property, we can write the federated SPARQL query shown in listing 2.4, which will be running at the OMA SPARQL endpoint:

```
select distinct ?proteinOMA ?proteinUniProt
where {
    service <http://sparql.uniprot.org/sparql> {
        ?proteinUniProt a up:Protein .
        ?proteinUniProt up:organism taxon:9606 .  # Homo Sapiens
        # annotations of this protein entry
        ?proteinUniProt up:annotation ?annotation .
        ?annotation rdfs:comment ?text.
        # only those containing the text "leukemia"
        filter(regex(str(?text), "leukemia") )
    }

    ?proteinOMA a orth:Protein.
    ?proteinOMA oma:xrefUniprot ?proteinUniProt.
}
```

We skip the details regarding the prefixes used in the example and focus on the new elements in the query. The main part to point out is the block "*service <http://sparql.uniprot.org/sparql>*", delimited between the inner brackets. This enables

using the SPARQL endpoint of UniProt remotely, as a service. Through this mechanism, the query will first fetch from UniProt all instances of proteins that are annotated with a text that contains "leukemia" (this is achieved by the filter keyword in the service block). Then, using the cross-reference *oma:xrefUniprot* property, the query will return all the equivalent entries from OMA. From here, the user can explore, either in the OMA browser or by further refining the SPARQL query, other properties of these proteins: for example, their orthologs in a given species available in the database. For a survey of federation techniques for RDF data, we refer the reader to [49].

The mechanisms illustrated so far, while indeed powerful for federating distinct databases, have a major drawback: they require the user to know the schema of the databases (otherwise, how would we know which properties to query in the previous examples?) and, more importantly, they require all users to be familiar with a technical query language, such as SPARQL. While very expressive, formulating such queries can quickly become overwhelming for non-programmer users. In the following, we will look at techniques that aim to overcome these limitations.

2.5 Ontology-Based Integration of Heterogeneous Data Stores

So far we have seen some of the alternatives available for storing biological data relational databases and triple stores. In this section, we look at how these heterogeneous sources can be integrated and accessed in a unified, user-friendly manner, that does not require knowledge of the location or structure of the underlying data, nor of the technical language (SQL or SPARQL) used to retrieve the data. The architecture we present is inspired by work presented in [1], which focused strictly on keyword search in relational databases.

2.5.1 A System's Perspective

We start with a bottom-up description of the layers that make up an integrated data access system, followed by a concrete example using the two bioinformatics databases introduced above: the orthology database OMA, and the gene expression database Bgee.

The main four layers of an integrated data access system, as shown in Figure 2.9, are:

1. Base Data Layer

   This represents the physical storage layer, where all the actual data, for example experimental results, annotations, etc. are kept. Figure 2.1 illustrates only a few of the possible storage types, namely relational databases, hierarchical data stores (e.g. HDF5) and RDF stores. At this low-level layer, the data are usually structured so as to optimize machine parameters, such as storage space, complexity of joins required to answer physical queries etc. Therefore, it is not designed for human readability. Furthermore, tables, column names or even IDs may not match any real terms. For example, the Bgee relational database uses the table name *'anatEntity'* to refer to the term 'Anatomic Entity', while others may be even further away from the original terms.

2. Data Model Layer

   This layer is used to describe, at a higher level of abstraction, the data contained in the physical storage. Here, for example, original names for terms are

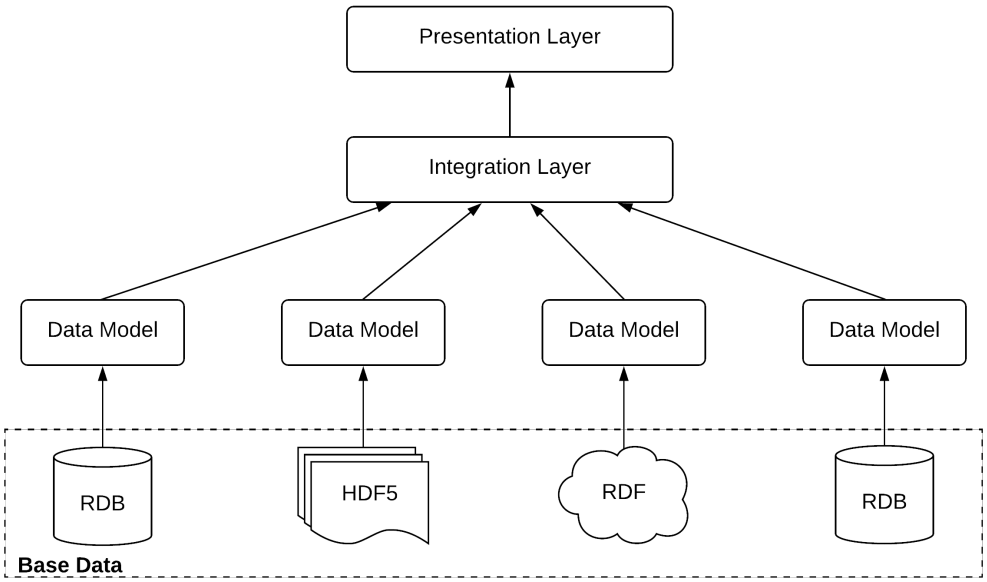

FIGURE 2.9: Integrated Data Access System.

recovered, while also creating a mapping between these higher level terms ('Anatomical Entity') and their corresponding physical layer location (table 'anatEntity' in schema Bgee). The data model layer can be viewed as the first *semantic* layer in the system, as it allows representing the actual terms referred to in the underlying physical storage, while abstracting away the details of the actual structure of the physical storage. The data model layer can be understood as an ontology, however, only applicable to the level of an individual database.

3. Integration Layer

   The integration layer performs a similar task to the data model layer, in that it defines a mapping between high-level concepts ('Anatomical Entity') and all the occurences where these concepts can be found in the physical storage (table 'anatEntity' in schema Bgee, class 'Anatomic Entity' in UniProt, etc.). In doing so, the integration layer also aligns the different data models, by defining which identifiers from one data model correspond to which ones from the others. In the case of biological databases, this is usually done by taking into account *cross-references*, which already exist between most databases, as we have seen in the SPARQL query in Section 2.4.

   While the data model layer can be seen as a local ontology, the integration layer will serve as a global ontology. The integration layer can be queried using, for example, SPARQL. However, in order to get the results from the underlying sources, the SPARQL query needs be translated in the native query languages of the underlying sources (e.g. SQL for relational databases). This is achieved by using the mappings defined in the global ontology. For example, the keyword "expressed in" does not have a direct correspondence in Bgee, but it can be translated into an SQL procedure (in technical terms, it represents an SQL view of the data). Without going into details, at a high-level, the property "gene A expressed in anatomic entity B" will be computed by looking at the number of experiments stored in the database, showing the expression of A

in B. It is conceivable that in another database, which could also form part of the integrated system, this information is available explicitly. In this case the mapping would simply be a 1-to-1 correspondence to the property value stored in the database. The role of the integration layer is to capture all the occurrences where a certain concept (entity or property) can be found, along with a mapping for each of the occurrences, defining how information about this concept can be computed from the base data.

To summarize, the integration layer abstracts away the location and structure of data in the underlying sources, providing users a unified access through a global ontology. One of the drawbacks of this approach is that, in the lack of a presentation layer, such as a user-friendly query interface (e.g. a visual query builder or a keyword-based search interface), the data represented in the global ontology is accessible mainly through a technical query language, such as SPARQL. Therefore, in order to be able to access the data, users are required to become fluent in the respective query language.

It is worth mentioning that most data integration systems available at the time of this writing only offer the three layers presented so far. Examples of such systems, generically denoted as Ontology Based Data Access Systems (OBDA), are Ontop [50], Ultrawrap [51], or D2RQ [52].

4. Presentation Layer

   The three layers presented so far already achieve data integration, but with a significant drawback: the user is required to know a technical query language, such as SPARQL. The role of the Presentation Layer is to expose data from all integrated resources in an easy to access, user-friendly manner. The presentation layer abstracts away the structure of the integration layer and exposes data through a search interface that users (including non-programmers) are familiar with, such as keyword search [1], [53], or even full natural language search [54], [55].

   The challenges in building the presentation layer are many-fold: first, human language is inherently ambiguous. As an example, let us assume a user asks *"Is the HBB gene expressed in the blood?"*. What does the user mean? The hemoglobin gene (HBB) in general? Or just in the human? The system should be proactive in helping the user clarify the semantics or intents of the question, before trying to compute the underlying SPARQL query. Second, the presentation layer should provide not only raw results, but also an explanation for example, what sources were queried, how many items from each source have been processed in order to generate the response etc. This enables the user to validate the generated results or to otherwise continue refining the question. Third, the presentation layer must also rank the results according to some relevance metric, similarly to how search results are scored in web search engines. Given that the number of results retrieved from the underlying sources can easily become overwhelming (for example, searching for "HBB" in Bgee returns over 200 results), it is important that the most relevant ones are shown first.

   From a technical point of view, the presentation layer maintains an index (i.e., the vocabulary) of all keywords stored in the lower-layers, both data and metadata (descriptions, labels, etc.), such that each keyword in a user query can be mapped to existing data in the lower layers. An important observation is that the presentation layer highly relies on the quality of the annotations available

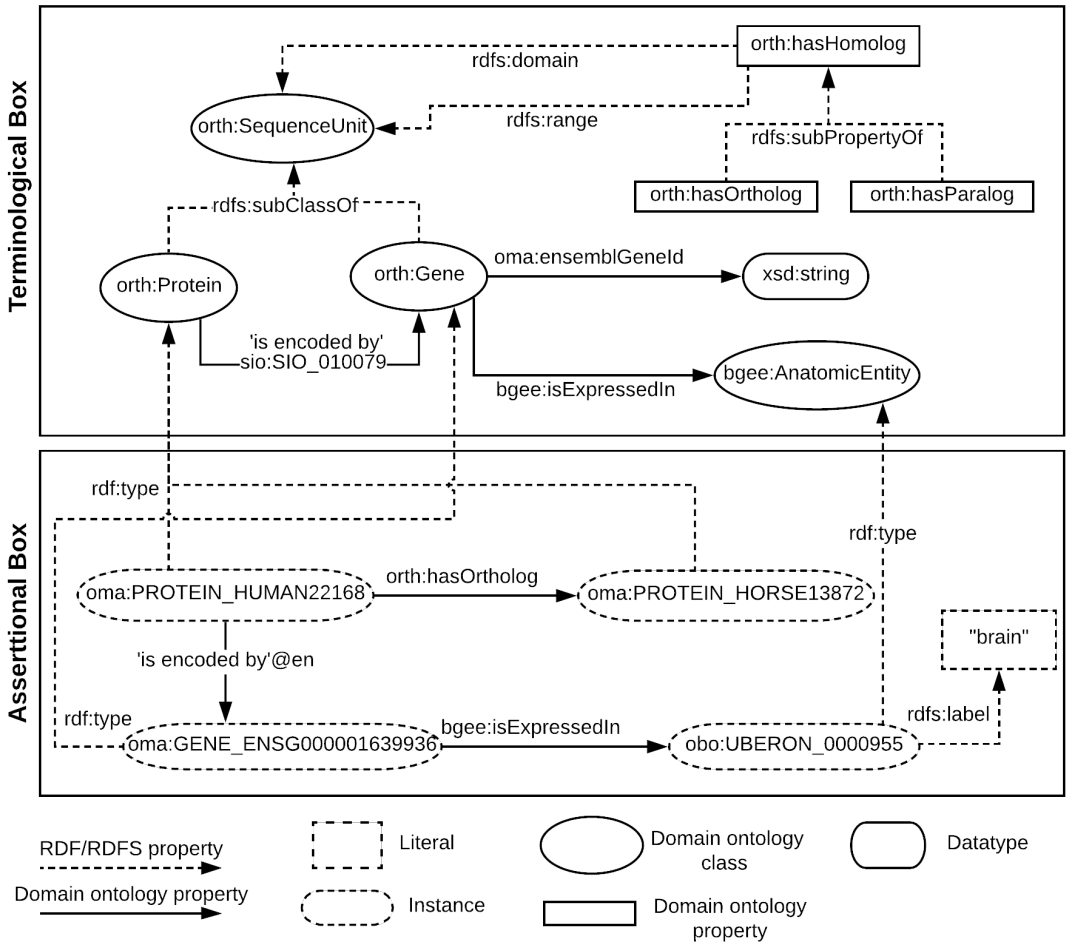

FIGURE 2.10: A sample global ontology for integrating OMA & Bgee and an example assertion.

in the lower layers. In the lack of human readable labels and descriptions in the global ontology, the vocabulary collected by the presentation layer will miss useful terms that the user might search for. One way to detect and fix this problem is to always log user queries and improve the quality of the annotations "on demand", whenever the queries cannot be solved due to missing items in the vocabulary. For a more extended discussion on the topic of labels and their role in the Semantic Web, refer to [56].

Finally, it is worth noting that none of these layers need to be centralized – indeed, even in the case of the integration layer, although its role is to build a common view of all data in the physical storage, it can be distributed across multiple machines, just as long as the presentation layer knows which machine holds which part of the unified view.

2.5.2 A Global Ontology to Unify OMA and Bgee

So far we have seen an abstract view of a system for data integration across heterogeneous databases. It is time to look at how this translates into a real-world example, using the Bgee relational database and the OMA RDF database.

The top part of Figure 2.10, the Terminological Box, illustrates part of the global ontology (layer 3, Integration Layer) for the two databases, with most of the terms

being part of OMA, except for Anatomic Entity, which is specific to Bgee. As mentioned previously, OMA extends the ORTH ontology, which is why the corresponding terms in the ontology are prefixed with "orth:". The Gene concept can actually be found in both Bgee and OMA, therefore the global ontology will define mappings to both sources. As we can see in the ontology, the Gene is the common point that joins together OMA and Bgee. The gene IDs used in both databases are Ensembl IDs [57], stored in the ensemblGeneId string property. For example, the human hemoglobin gene, "HBB", which we previously showed as an example entry in OMA, corresponds to the ENSG00000244734 Ensemble ID, and can also be found in Bgee.

The lower part of Figure 2.10, the Assertional Box, illustrates an example assertion – in this case, that the protein HUMAN22168 in OMA is orthologous to the protein HORSE13872 and that furthermore, this protein is encoded by the gene with the Ensemble ID ENSG000001639936. Moreover, this gene is expressed in the brain (the Uberon ID for this being "UBERON:0000955"). The human readable description is stored in the String literal label as, for example, the name of the anatomic entity, "brain", shown in the bottom-right corner in the figure. Without labels, much of the available data would not be easily searchable by a human user, nor by an information retrieval system.

Note that with this sample ontology we can already answer questions related to orthology and gene expression jointly, such as the first part of our introductory query: "What are the human-rat orthologs, expressed in the liver…?". This question essentially refers to pairs of orthologous Genes (those in human and rat), and their expression in a given Anatomic Entity (the liver). Apart from the Species class, which is not explicitly shown, all of the information is already captured by the ontology in Figure 2.10. A similar mechanism can be used to further extend this to UniProt (for instance, based again on gene IDs as the "join point", or by using existing cross-references, as we have shown in the previous section), therefore enabling users to ask even more complex queries.

2.5.3 How to Link a Database with an Ontology?

One of the main challenges in implementing technologies for the Semantic Web was recognized from early on (see the study published in 2001 by Calvanese et al [58]) to be the problem of integrating heterogeneous sources. In particular, one of the observations made was that integrating legacy data will not be feasible through a simple 1-to-1 mapping of the underlying sources into an integrative ontology (e.g. mapping all attributes of tables in relational databases to properties of classes in an ontology), but rather through more complex transformations, that map views of the data into elements of the global ontology [58].

To illustrate this with a concrete example, let us consider again the unified ontology for OMA and Bgee that we introduced in the previous section. Although Figure 2.10 shows properties such as *"gene isExpressedIn"* or *"gene hasOrtholog"*, this data is actually not explicitly stored in the underlying databases, but rather needs to be computed on-the-fly based on the available data. For example, the *"isExpressedIn"* property can be computed based on the number of experiments which show the expression of a gene in a certain anatomic entity in Bgee. Deciding the exact threshold for when a gene is considered as "expressed" according to the data available is not straightforward and needs to be agreed upon by domain specialists. Therefore, the integration layer will also serve to enrich the data available in the underlying layers,

by defining new concepts based on this data (e.g. the presence or absence of gene expression in an anatomic entity).

At this point it is worth clarifying an important question: why are mappings necessary? Why is it not enough to replicate the data in the different underlying formats into a single, uniform way (for example, translate all RDB data into RDF)? The answer is that not only would such a translation require a lot of engineering effort, but more importantly, it would transform the data from a format that is highly optimized for data access, into a format that is optimized for different purposes (data integration and reasoning). Querying relational databases still is, today, the most efficient means of accessing very large quantities of structured data. Transforming all of it into RDF would in many cases mean downgrading the overall performance of the system. In some cases storing RDF data in the relational format was proven to be more efficient [59].

So how are mappings then created? One of the main mechanisms to achieve this is currently the W3C standard R2RML, available as a W3C recommendation online[15]. R2RML enables mapping relational data to the RDF model, as chosen by the programmer. For a concrete example of how mappings can be defined and what are the advantages of this approach, we refer the reader to [60]. A mapping essentially defines a view of the data, which is a query (in this case, an SQL query) that allows retrieving a relevant portion of the underlying data, in order to answer a higher-level question (e.g. what is "expressed in"?). The materialization of this query (the answer) will be returned in RDF format, on-demand, according to the mapping. This avoids duplicating or translating data in advance from the underlying relational database into RDF until it is really needed, in order to answer a user query.

For a discussion regarding the limitations of R2RML and alternative approaches to define mappings from relational data to RDF, we refer the reader to [61].

2.5.4 Putting Things Together

So far we have seen how individual sources can be represented into a single, unified ontology and we had a high-level view of a data access system that enables users to ask queries and get responses in a unified way, without knowledge of where data is located or how it is structured. In this section we finally look at how all of these components can work together in answering natural language queries on biological databases. Although there are multiple alternatives to natural language interfaces, including visual query interfaces, or keyword-based search interfaces, it has been showed that natural language interfaces are the most appropriate means to query Semantic Web data for non-technical end-users [5]. As a consequence, natural language querying, based on semantic web technologies, is currently one of the active areas of research, examples of recent systems implementing an ontology-based natural language interface including the Athena [54] and TRDiscover [55] systems.

First, recall the user question we formulated in the beginning of this chapter: "What are the human-rat orthologs, expressed in the liver, that are associated with leukemia?" Let us assume the resources at hand to answer this question are the biological databases OMA, Bgee and UniProt. The four main steps required to translate the natural language question into the underlying query languages of OMA, Bgee and UniProt will be:

a) Identify entities in the query

[15] https://www.w3.org/TR/r2rml/

This is the inverted index search, which extracts the main terms the user is interested in, based on the keywords of the input query: orthologs, human, rat, expressed, liver, associated, leukemia.

b) Identify matches of the entities in the integrative ontology

In this step, also known as entity linking, the extracted terms will be searched for in the vocabulary of the presentation layer, resulting in one or multiple URIs, given that a keyword can match multiple concepts. For example, the keyword 'orthologs' can match either the entity 'OrthologCluster' or the property 'hasOrtholog' of a gene in OMA. The index of the presentation layer will also return the location the URI originates from (OMA or Bgee or UniProt).

c) Construct sub-queries for each of the matches

The extracted URIs will be used to construct subqueries on each of the underlying data sources. This step requires translating the original query into the native language of each underlying database, with specific mechanisms for each type of database (relational or triple store). At a high-level, the translation process involves finding the minimal sub-schema (or sub-graph in the case of RDF data) that covers all the keywords matched from the input query. Taking the example previously shown in Figure 2.10, the minimal subgraph that contains "orthologs" and "expressed" will essentially contain only 2 nodes of the entire graph: Gene (which is both the domain and the range of the "hasOrtholog" property in the Orthology Ontology) and AnatomicEntity (which is the range of the "isExpressedIn" property in the Bgee ontology). All the unknowns of the query (for example, which ortholog genes) are replaced by variables. The final sub-queries for OMA and Bgee might therefore (informally) look like this:

```
# OMA:
select ?gene1 ?gene2 where {
       ?protein1 a Protein.
       ?protein1 inTaxon "Homo Sapiens".
       ?protein1 isEncodedBy ?gene1.
       ?protein1 hasOrtholog ?protein2.
       ?protein2 inTaxon "Rattus Norvegicus".
       ?protein2 isEncodedBy ?gene2.
}
```

Note that we have simplified the actual query for readability purposes (using the literals "Homo Sapiens" and "Rattus Norvegicus" instead of their corresponding URIs). This subquery will cover the keywords: ortholog, human, rat. Notice that the query should return genes, not proteins, because the join point between OMA and Bgee is the Gene class.

```
# Bgee:
select ?gene where {
       ?gene a Gene.
       ?gene isExpressedIn ?anatomicEntity.
       ?anatomicEntity rdfs:label "liver".
}
```

This subquery will therefore cover the expressed and liver keywords. The final step will be then to get the similar subquery for UniProt (which we omit here for brevity) and to compute the joint result, namely, the intersection between all the sets returned by the subqueries.

d) Join the results from each of the subqueries

This final step is essential in keeping the performance of the system to an acceptable level. Joining (federating) the results of several subqueries into a unified result is not an easy task and requires a careful ordering of the operations from all subqueries. To understand this problem, let us consider again our example and try to see how many results each of the subqueries will return. First, if we take a look at the OMA browser and try to find all orthologs between human and rat, this will amount to more than 21,000 results. However, is the user really interested in all of them? Certainly not, as the input query shows – the user is only interested in a small fraction of the orthologs, namely those that are expressed in the liver and have an association with leukemia (according to the data stored in Bgee and UniProt). How many are these? If we now refer to UniProt and look for the disease leukemia, we will find that there are only 20 entries which illustrate the association with this disease. Clearly, getting only the orthologs of these 20 entries will be much more efficient than retrieving all 21,000 pairs from OMA first, and then removing most of them to only keep relevant ones.

However, note that in this case, we only know this information because we constructed the queries and tried them out by hand first. How should the system estimate the number of results (i.e. the cardinality of each subquery) in advance? This question has been an active area of research for a long time. Some of the methods used to tackle this problem are either to pre-compute statistics regarding the number of results available in different tables of the underlying sources [62], or to use statistics regarding previously asked queries to optimize the new ones, for example via statistical machine learning [63]. In the first case, we would, for instance, store the individual counts of different orthologous pairs, while also keeping statistics about diseases if we expect these types of questions to be asked frequently, whereas in the second case we would simply look at the number of results similar subqueries generated in the past, to optimize which results to fetch first. For a recent study of optimization methods for federated SPARQL queries, see [64].

e) Present the final results to the user

Finally, the joined results are returned to the user, ideally along with an explanation regarding the constructed query and the entities that were matched in order to construct it. In this way, the user has the opportunity to validate the correctness of the answer or otherwise to further refine the question.

For a more in-depth discussion regarding natural language query interfaces in ontology-based data access systems, we refer the reader to Athena [54] and TRDiscover [55].

2.6 Timeline of Ontology-Based Data Integration Milestones in Life Sciences

The field of Life Sciences has been an early adopter of Semantic Web technologies, due to the need of interoperability and integration of biological data spread across different databases. In this section, we provide a brief timeline (summarised in Figure 2.11), including the example ontologies introduced in this chapter.

1995: Davidson et al. [65] suggest basic steps to integrate bioinformatics data (common data model, match semantically related objects, schema integration, transform data into federated database, match semantically equivalent data).

1995	Davidson et al. suggest **basic steps to integrate bioinformatics data**
2000	**TAMBIS**: Transparent Access to Multiple Bioinformatics Information Sources
	Gene Ontology
2001	**BioMoby:** a unified registry of web services for life scientists
2003	**Integrating biological databases** in Nature Reviews Genetics
	UniProt: The Universal Protein Knowledge
2004	First **International Workshop on Data Integration in the Life Sciences**
2005	**HCLS IG**: Semantic Web Health Care and Life Sciences Interest Group
2006	**OBO Foundry**: Open Biological and Biomedical Ontology Foundry
	OLS: Ontology Lookup Service
2007	**National Center for Biomedical Ontology (NCBO) BioPortal**: a web portal to biomedical ontologies
2008	**BioMoby**: interoperable access to over 1400 bioinformatics resources
	BioGateway: a semantic systems biology tool for the life sciences
	Special issue **Database Integration in Life Sciences** in Briefings in Bioinformatics
2009	Review on **Ontologies and Semantic Web Technologies** in Briefings in Bioinformatics
2010	**NCBO** launches a **SPARQL endpoint**
2012	**Semantic Web meets Integrative Biology**
2016	**Orthology Ontology**

FIGURE 2.11: A selective timeline of data integration efforts in the Life Sciences

2000: TAMBIS (Transparent Access to Multiple Bioinformatics Information Sources) [66] proposes a unified ontology covering many aspects of the bioinformatics knowledge space.

2000: The "Gene Ontology a tool for the unification of biology" [41] is the first significant milestone in unifying diverse biological databases, focusing on gene functions. Even before the publication of the Semantic Web paper by Tim Berners Lee (in the following year), the GO highlighted the benefits of controlled vocabularies and standardized naming, both precursors of Semantic Web technologies, which were adopted in the GO in the year 2002 [67]. Today it is, arguably, the most comprehensive resource of computable knowledge regarding gene functions and products.

2001: Launch of the BioMoby project [68] providing a unified registry of Web Services for life scientists using a consensus driven approach. It listed, for instance, all services converting gene names to GO terms, or all databases accepting GO terms. The registry is currently no longer maintained.

2003: A Nature Reviews Genetics article on Integrating biological databases [69] highlights the "database surfing" problem (i.e. the time-consuming process of manually visiting multiple databases to answer complex biological research questions), and argues for standardized naming of biological objects to overcome the problem. Link integration, view integration and data warehousing are proposed for data integration. Arguably, link integration has since become the most adopted solution.

2003: Launch of UniProt [70] by the UniProt Consortium, a collaboration between the Swiss Institute of Bioinformatics (SIB), the European Bioinformatics Institute (EBI), and the Protein Information Resource (PIR). UniProt is the world's most

comprehensive freely accessible resource on protein sequences and functional annotation. Since 2008 the data is published in RDF, and since 2013 a SPARQL endpoint is provided [71].

2004: The first International Workshop on Data Integration in the Life Sciences, held in Leipzig, promotes "a Bioinformatics Semantic Web" and highlights solutions for heterogeneous data integration. The workshop continues to be held every year and its proceedings (e.g. [72]) provide a good overview of advances in the field.

2005: The W3C Consortium launches the Semantic Web Health Care and Life Sciences Interest Group (HCLS IG) to develop the use of Semantic Web technologies to improve Health Care and Life Science research. Today, the HCLS Linked Data Guide[16] provides best practices for publication of biological Linked Data on the Web.

2006: The OBO Foundry [44] establishes principles for ontology development and evolution to support biomedical data integration through a suite of orthogonal interoperable reference ontologies.

2006: Publication of the Ontology Lookup Service (OLS), a repository for biomedical ontologies with the aim to provide a single point of access (with controlled vocabulary queries) to the latest ontology versions. It allows interactive browsing, as well as programmatic access [73].

2007: Launch of the National Center for Biomedical Ontology (NCBO) BioPortal [74], a web portal to biomedical ontologies. OBO ontologies are a central component. The portal started with 50 ontologies; to date it is the most comprehensive repository with currently 852 biomedical ontologies and more than 8 million classes.

2008: Launch of the BioMoby Consortium [75] and the first release of the BioMoby Semantic Web Service, at the time providing interoperable access to over 1400 bioinformatics resources worldwide.

2008: BioGateway [76] provides a single SPARQL entry point to all OBO candidate ontologies, the GO annotation files, the SWISS-PROT protein set, the NCBI taxonomy, and several in-house ontologies.

2008: The Briefings in Bioinformatics Journal launches a special issue dedicated to Database Integration in Life Sciences [77], acknowledging the major challenge of integrating data scattered over millions of publications and thousands of heterogeneous databases.

2008: Bio2RDF [78] applies semantic web technology to various publicly available databases (converting them into RDF format, and linking with normalized URIs and a common ontology). Updates continue to be provided for increased interoperability among bioinformatics databases [79], [80]

2009: Briefings in Bioinformatics publishes a review on Biological Knowledge Management [81], highlighting the transforming role of ontologies and Semantic Web technologies in enabling knowledge representation and extraction from heterogeneous bioinformatics databases.

2010: NCBO launches a SPARQL endpoint, available at `http://sparql.bioontology.org/`.

2012: Publication of a survey highlighting the benefits of integration using Semantic Web technologies in the field of Integrative Biology [82]

2016: Publication of the Orthology Ontology [24].

[16] see `https://www.w3.org/2001/sw/hcls/notes/hcls-rdf-guide/`

2.7 Conclusions and Outlook

Data integration is arguably one of the most important enablers of new scientific discoveries, given that research data is currently growing at an unprecedented rate. This is especially true in the case of biological databases. While data integration poses many challenges, the emergence of standards, integrative ontologies, as well as the availability of cross-references between many of the biological databases make the problem easier to tackle. This chapter has provided a brief introduction to the methods that can be used to integrate heterogeneous databases using Semantic Web Technologies, while also providing a concrete example of achieving this goal for three well-known existing biological databases: OMA, Bgee and UniProt.

Although there would be many more aspects to cover and much of the work for achieving wide-scale data integration still remains to be done, I would like to end this chapter by reinforcing the following conclusion, extracted from a study of Biological Ontologies for Biodiversity Knowledge Discovery [83]:

"We hope that current work will spur interest and feedback from scientists and bioinformaticians who see data integration, interoperability, and reuse as the solution to bringing the past 300 years of biological exploration of the planet into currency for science and society."

Chapter 3

Enabling Federated Queries Across Bioinformatics Databases

Data integration promises to be one of the main catalysts in enabling new insights to be drawn from the wealth of biological data available publicly. However, the heterogeneity of the different data sources, both at the syntactic and the semantic level, still poses significant challenges for achieving interoperability among biological databases.

We introduce an ontology-based federated approach for data integration[1]. We applied this approach to three heterogeneous data stores that span different areas of biological knowledge: 1) Bgee, a gene expression relational database; 2) OMA, a Hierarchical Data Format 5 (HDF5) orthology data store, and 3) UniProtKB, a Resource Description Framework (RDF) store containing protein sequence and functional information. To enable federated queries across these sources, we first defined a new semantic model for gene expression called GenEx. We then show how the relational data in Bgee can be expressed as a virtual RDF graph, instantiating GenEx, through dedicated relational-to-RDF mappings. By applying these mappings, Bgee data are now accessible through a public SPARQL endpoint. Similarly, the materialised RDF data of OMA, expressed in terms of the Orthology ontology, is made available in a public SPARQL endpoint. We identified and formally described intersection points (i.e. virtual links) among the three data sources. These allow performing joint queries across the data stores. Finally, we lay the groundwork to enable non-technical users to benefit from the integrated data, by providing a natural language template-based search interface, BioQuery.

Project URL: `http://biosoda.expasy.org`,
Project code: `https://github.com/biosoda/bioquery`

3.1 Introduction

One key promise of the postgenomic era is to gain new biological insights by integrating different types of data [e.g. 84, 85]. For instance, by comparing disease phenotypes in humans with phenotypes produced by particular mutations in model species, it is possible to infer which human genes are involved in the disease [4].

A wealth of biological data is available in public data repositories; over one hundred key resources are featured in the yearly Nucleic Acids Research annual database issue [3]. However, these databases vary in the way they model their data (e.g. relational, object-oriented, or graph database models), in the syntaxes used to

[1] Parts of this chapter have been published in the Database journal [6]

represent or query the data (e.g. markup or structured query languages), and in their semantics. This heterogeneity poses challenges to integrating data across different databases.

Ontologies have been widely used to achieve data integration and semantic data exchange [86–93]. In this thesis, by ontology, we adopt the broadly accepted definition in data and knowledge engineering of "*a formal, explicit specification of a shared conceptualization*" [94]. The relevance of ontologies in life sciences can be illustrated by the fact that repositories such as BioPortal [95] contain more than seven hundred biomedical ontologies, and the OBO Foundry [96] more than 170 ontologies. Moreover, major life-sciences databases use ontologies to annotate and schematize data, such as UniProt [97] or ChEBI [98]. Ontologies are important to enable knowledge sharing.

Currently, however, even when resources describe their data with ontologies, aligning these ontologies and combining information from different databases remain largely manual tasks, which require intimate knowledge of the way the data is organised in each source. This is despite a plethora of existing literature on data integration approaches, in particular in biological research (surveys of these approaches, as well as the challenges involved, include [99–101]). Projects such as KaBOB [102], Bio2RDF [78] and Linked Life Data [103] link different life science resources using a common ontology and data conventions. However, their centralised architecture makes it difficult to remain up-to-date and to scale up. For example, when querying the number of UniProt protein entries over the outdated and centralised Linked Life Data approach, we can only count around 10% of the 230 million entries that are in the current UniProt release (see Supplementary Material Section A.1 for further explanations). To avoid this issue, federated approaches have recently been proposed [104–107], but to the best of our knowledge, none of them proposes a vocabulary and patterns to extensively, explicitly and formally describe how the data sources can be interlinked further than only considering "same as"-like mappings; in effect, they put the burden on the users to find out precisely how to write a conjunctive federated query. An emerging research direction entails automatically discovering links between datasets using Word Embeddings [108]. We did not pursue this approach, given that it is computationally expensive and that for our study writing the relational-to-RDF mappings proved more straightforward. However, Word Embeddings would be important for the case of integrating more data sources for which the connecting links (join points) are not clearly known.

To address the problem of semantic, syntactic and data model heterogeneity, we propose an ontology-driven linked data integration architecture. We apply this architecture to build a system that federates three bioinformatics databases containing: evolutionary relationships among genes across species (OMA), curated gene expression data (Bgee), and biological knowledge on proteins (UniProt). In Supplementary Material, we summarise the key data provided by Bgee, OMA and UniProt (Table A.3). Each of the three databases uses a different technical approach to store information: a Hierarchical Data Format 5 (HDF5, `http://www.hdfgroup.org/HDF5/`) data store for OMA [109]; a relational database for Bgee [11]; and a Resource Description Framework (RDF) store for UniProt [97]. Our main contribution is to enable researchers to jointly query (i.e. conjunctive queries) the three heterogeneous databases using a common query language, by introducing and leveraging "virtual links" between the three sources. Furthermore, we show how relational data can be made interoperable with RDF data *without* requiring the original relational data to be duplicated into an RDF storage engine. This can be achieved by constructing dedicated relational-to-RDF mappings, allowing the unmodified original data to be queried

via the structured query language SPARQL [110]. In our proposed architecture, we illustrate this through the example of the Bgee relational database.

Moreover, for the purpose of building the federated data access system, we make the following additional contributions: (i) a semantic model for gene expression; (ii) an extension and adaptation of the Vocabulary of Interlinked Datasets (VoID) [111]; (iii) public SPARQL 1.1 [110] query endpoints for OMA and Bgee; and (iv) a user-friendly search interface based on an extensible catalogue of query templates in plain English.

This chapter is structured as follows. In Section 3.2, we describe the individual databases, as well as our approach and the semantic models used in this work. In Section 3.3, we provide the implementation details of the three layers of our proposed architecture (data store, structured query interface, and application). In Section 3.4, we evaluate the performance of the system on a catalogue of 12 representative federated biological queries. We conclude with a discussion and outlook.

3.2 System and Methods

To understand more concretely the problem of integrating data from multiple sources, consider the following motivating example: *"What are the human genes which have a known association to glioblastoma (a type of brain cancer) and which furthermore have an orthologous gene expressed in the rat's brain?"*. To answer this question, we would need to integrate information currently found in different databases:

1. Human proteins associated with glioblastoma can be obtained from **UniProt Knowledge Base**, a database providing a comprehensive, high-quality sequence and functional information on proteins [97]. In the rest of the chapter, we will use the name "UniProt" for readability.

2. The orthologs of these proteins in the rat can be obtained from **OMA** (Orthologous Matrix), a database of orthology inferences [109]. Orthologs are genes in different species that evolved from a common ancestral gene by speciation. The orthologs are normally thought to retain the same function in the course of evolution. Other homology information, such as one-to-one orthology or paralogy, can be derived from the Hierarchical Orthologous Groups (HOGs) data structure [92, 112].

3. The genes expressed in the rat brain can be obtained from **Bgee**, a database of curated gene expression patterns in animals [11]. Bgee version 14.0 includes gene expression data for 29 species such as human, mouse, or hedgehog. Currently, Bgee data are stored in a MySQL relational database [113].

In the following, we first provide a high-level description of our approach, then introduce the semantic models involved.

3.2.1 A federated, ontology-driven data integration approach

In order to achieve semantic interoperability between Bgee, OMA and UniProt, we have chosen a federated approach based on ontologies (Figure 3.1). The advantage of a federated approach is to avoid imposing a common global schema or metamodel on all data sources, and to facilitate the integration of further resources in the future. In doing so, we avoid, for example, the fastidious and time-consuming task of maintaining a centralised, integrated knowledge base. Instead, we provide

a homogeneous data access layer to query the heterogeneous data sources. This homogeneous layer is part of a new generation of federated databases, such as polystores [114], that provide seamless access to distinct data models of storage engines (e.g. MySQL and RDF stores). Unlike that approach, we do not seek to optimise query performance by transferring data on-the-fly between disparate storage engines [114], but rather focus on solving syntactic and semantic heterogeneities among data stores.

To solve the syntactic heterogeneity, we rely on a structured query language—SPARQL—as the homogenous syntax to query all the data [110]. We favoured SPARQL 1.1 over alternatives because it is World Wide Web Consortium (W3C) compliant and the data to be integrated are on the web; because it supports federated queries; and because one of our target data stores, UniProt, is already accessible through a SPARQL 1.1 endpoint, alongside a growing number of other biological databases [115]. Indeed, although our initial prototype integrates data from Bgee, OMA and UniProt, we plan to extend the system to include more data sources in the future.

To reduce semantic heterogeneity among the databases, we rely on ontologies further described in section 3.2.2, which are defined with the Web Ontology Language 2 (OWL 2), and thus based on the RDF model and syntax (`https://www.w3.org/TR/owl2-overview/`). RDF-based modelling decisions were taken to mitigate this heterogeneity when adopting ontological terms and instances to structure and represent the non-RDF data — OMA HDF5 and Bgee relational data. For example, by considering OMA, Bgee and UniProt databases, UniProt covers all of others with regard to the taxonomic lineage information for an organism. Therefore, we rely on UniProt classes and instance IRIs when representing and modelling taxonomy-related data in the OMA and Bgee RDF serialisations. To further exemplify, we can also mention the representation of genes among these three data sources. OMA genes completely overlaps Bgee genes but not all OMA genes has a corresponding one in UniProt, and *vice-versa*. Because of this, we decided to model Bgee genes the same way as in OMA, thereby easing interoperability between gene expression and orthology data. The next sections describe in more details the semantic models and the federated architecture proposed.

3.2.2 Semantic models

Ontology-based data integration requires as a preliminary step that each of the individual resources composing the federated system provide an explicit ontological description of their data. To minimise the need for semantic reconciliation—i.e. the process of identifying and resolving semantic conflicts [116], for example, by matching concepts from heterogeneous data sources [117]—we sought to rely as much as possible on existing ontologies when defining new semantic models.

Prior to our current work, among the three databases considered in this chapter, only UniProt provided an RDF representation of its data, as well as a SPARQL endpoint. The current UniProt RDF release comprises over 55 billion triples, and is based on the OWL 2 Full UniProt core ontology described in Redaschi, Consortium, et al. [118].

For the orthology data in **OMA**, we adopted the Orthology (ORTH) ontology [119], which was recently devised by the Quest for Orthologs Consortium [120] as a common data schema for integrating Orthology databases, such as OMA. We use ORTH to structure the OMA data, which is primarily stored in an HDF5 data store. Furthermore, during the conception of a second version of ORTH, design decisions such as the adoption of taxon-related terms from the UniProt ontology were

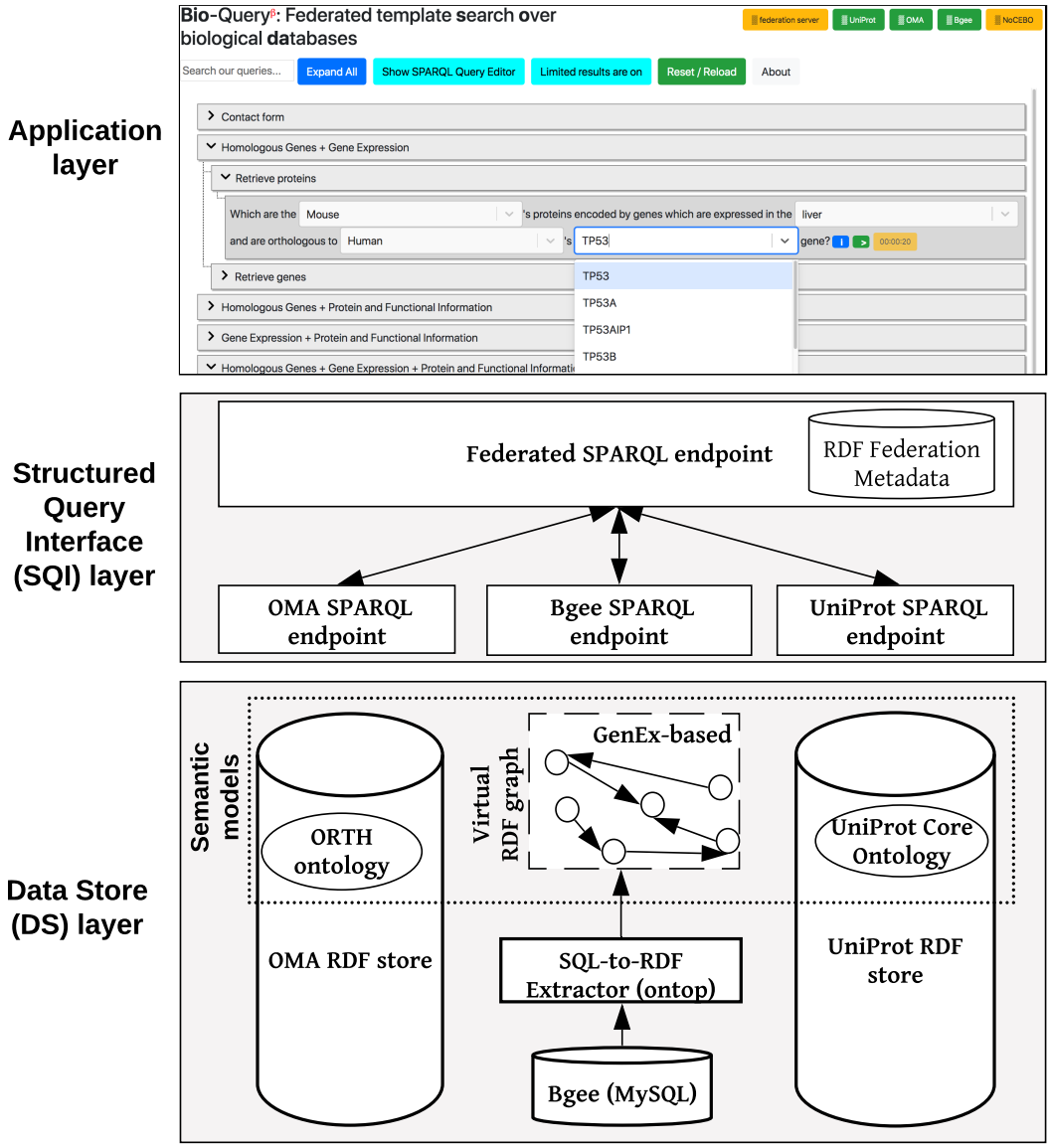

FIGURE 3.1: Overview of the ontology-driven federated data integration architecture applied to Bgee, OMA, and UniProt. The application layer depicts a Web search interface with editable templates to jointly query the data stores. Available online at `http://biosoda.expasy.org`.

made in order to enhance interoperability, enabling us to establish links among the data stores (section 3.3.2). Therefore, the work presented in this chapter also contributed towards a new, improved version of the ORTH ontology, which is described in Farias, Chiba, and Fernández-Breis [92].

In the case of **Bgee**, representing the original data in RDF proved to be a challenge, due to a general lack of a comprehensive ontology to serve as a data schema for describing knowledge in the field of gene expression. This may seem surprising considering the ubiquity of gene expression analyses in molecular biology and the existence of multiple well-established resources for gene expression—not only Bgee, but also Expression Atlas (EA) [121], Genevestigator [122], or the Tissue Expression database [123]. We note here that different gene expression databases often use distinct criteria to assert *expressed in* or *absent in* relations.

To the best of our knowledge, two semantic models currently exist as initial attempts to structure gene expression related data: the Relation Ontology [124] and the Expression Atlas model [125]. The Relation Ontology (RO) defines only a few terms within the domain of gene expression and is not specifically designed for this knowledge domain. Notably, it contains *expressed in* and *expresses* relations. The Expression Atlas defines a semantic model related to gene expression that mainly focuses on modelling the Expression Atlas (EA) data itself and not the domain of gene expression generally. In this EA model, additional data interpretations (i.e. semantics) are not explicitly represented, such as a given gene *is expressed* or *lowly expressed* in some sample relative to others. Although it would be possible to obtain this information through a more complex query on the Expression Atlas SPARQL endpoint, we lack an explicit representation, which would allow us to compare gene expression data from these different databases.

To provide a first step toward a general-purpose gene expression ontology, we drafted a new semantic model called GenEx. GenEx is aligned with the Relation Ontology and Expression Atlas models to facilitate interoperability with existing RDF stores. We also included semantic rules and terms to address (i) the representation of additional information related to gene expression, such as developmental stages, as well as *absent in* and *highly expressed* relations; and (ii) the trade-off between virtualisation and materialisation for the sake of query execution time and data storage. Furthermore, we reuse parts of the data schemas of the ORTH and UniProt core ontologies to provide (iii) the capacity to interoperate with other biological databases from different knowledge domains which are still relevant to the gene expression domain. For example, integrating orthology and gene expression data is relevant since we might want to predict gene expression conservation for orthologous genes. The draft GenEx is available online and documented in `https://biosoda.github.io/genex/`.

We stress that GenEx is currently in draft state. To become a standard, it needs to be endorsed and supported by multiple key stakeholders. We plan to initiate discussions with representatives of Bgee, Expression Atlas, Genevestigator, and Tissue database teams, and intend to solicit involvement from others, for example, the Model Organisms Databases (`http://www.alliancegenome.org`).

3.3 Implementation

Our federated data integration architecture comprises three layers: the data store (DS) layer, the structured query interface (SQI) layer, and the application layer (Figure 3.1). The DS layer contains all data stores to be integrated, including ontologies

and methods to solve semantic and data model heterogeneities, such as relational-to-RDF mappings (Section 3.3.1). The SQI layer provides a homogeneous query language syntax and exploits common instances and literals (i.e. virtual links) to retrieve data from the DS layer (Section 3.3.2). The application layer includes any software tools that access the data stores through the SQI layer, for example a web search interface (Section 3.3.3). Figure 3.1 illustrates this architecture applied to our use case: the Bgee, OMA and UniProt databases.

The three layers are described in the next subsections, and source code is available at `https://github.com/biosoda/bioquery`.

3.3.1 Data store layer

The **UniProt** data were already available in an RDF model and accessible through a SPARQL endpoint at the start of our project. Therefore, we could use UniProt data as is.

The core of our work on the data store layer consisted in exposing data from **Bgee** and **OMA** as RDF, with the goal of solving data model heterogeneity. We focused our efforts on including the domain-specific, most "value-added" aspects of Bgee and OMA to the data store layer—leaving out information already available in UniProt. As a result, the Bgee and OMA data accessible through our system are subsets of their original contents. We provide an overview of the types of information available in the original sources versus in their RDF representation in the Supplementary Material (Table A.3). This reduced the development work and data duplication among the databases, without loss of information considering that our federated approach enables directly retrieving this data from its original source (i.e. UniProt).

The **Bgee** data are stored in a relational database, meaning that integration between RDF stores and relational databases would still require substantial effort. There are two main methods to overcome this issue. First, the existing data could be represented entirely as RDF, which consequently would replace the relational model. A second approach would be to express the existing relational data as a virtual RDF graph, defined over ontological concepts and relations. We have chosen the latter approach, also referred to as *ontology based data access* (OBDA) [126]. Our choice is justified by the fact that changing the Bgee data store into an RDF model would either lead to data duplication or would require significant changes in the current Bgee analysis pipeline [see 11]. This is because Bgee is now adapted to the relational model for storing raw and preprocessed data from multiple data sources such as Ensembl, GEO, ArrayExpress and others (`https://bgee.org/?page=source`).

To implement OBDA over the Bgee relational database, we used the Ontop platform [126] version 3.0-beta-2. We defined several relational-to-RDF model mappings, which dynamically instantiate the gene expression semantic model described in Section 3.2.2. Figure 3.2 shows a simplified example of OBDA mappings that serve to express data from the relational model in the RDF model. Namespace prefixes such as *up:* shown in Figure 3.2 and used in the rest of this chapter are defined in Supplementary Table A.1. Some of the mappings can be simple 1-to-1 correspondences—for example, a gene name (shown in red color on the right) can directly be used as a label of a *orth:Gene* class instance. Other mappings require transforming the original attributes in the relational data for interoperability – for example, replacing ":" with "_" in the case of anatomical entity identifiers from Bgee to be compliant with the existing UBERON ontology IRI (Internationalized resource identifier) terms [127], as shown in green color with the example of *UBERON:0000955* in

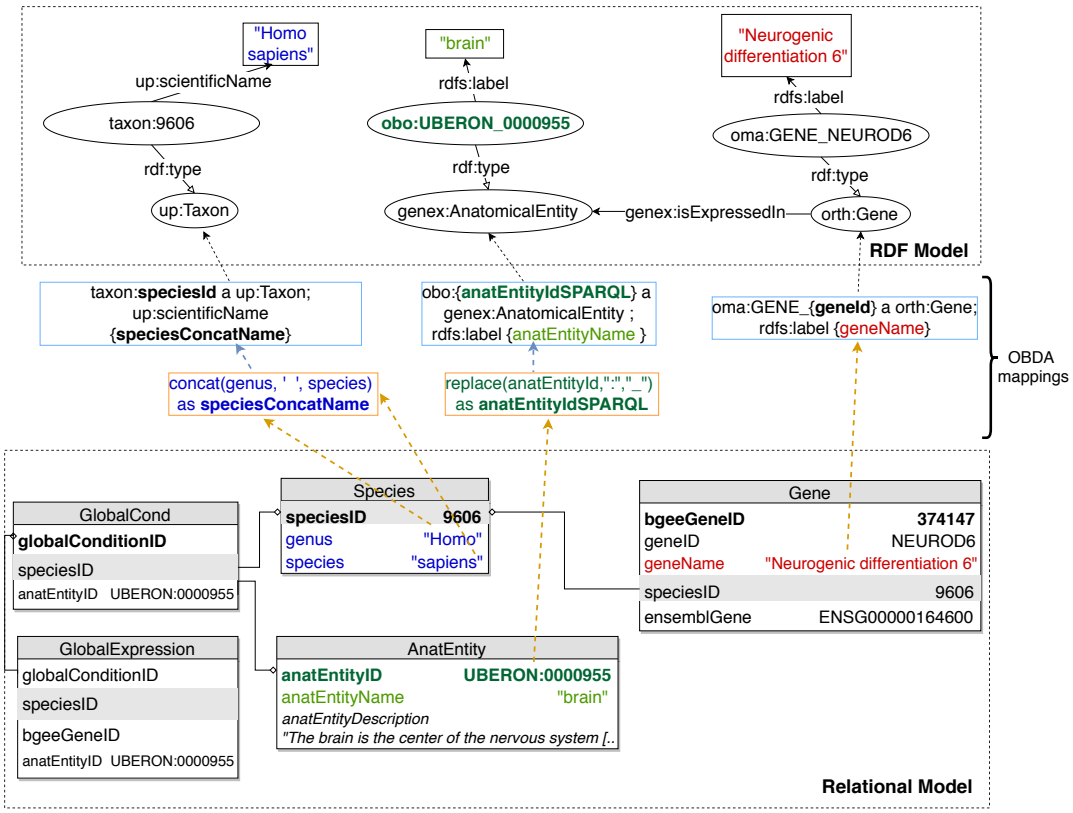

FIGURE 3.2: An illustration of relational-to-RDF mappings on a sample of the Bgee database. These mappings address both *schema-level* heterogeneity (an example is shown in blue), as well as *data-level* heterogeneity (shown in green). A mapping can also be a simple 1-to-1 correspondence between a relational attribute (e.g. *geneName*, shown in red) and its equivalent RDF property (in this case, an *rdfs:label* of an *orth:Gene* instance). Namespace prefixes are defined in Supplementary Table A.1.

Figure 3.2. Another type of transformation can be even combining multiple columns to instantiate a concept, as in the case of expressing *species* data from Bgee in terms of instances of *up:Taxon*. In this case, the OBDA mapping serves to concatenate the *genus* and *species* columns from Bgee in order to form the scientific name in compliance with the UniProt taxonomy. The scientific names of species in UniProt are denominated through the *up:scientificName* property, composed of both genus and species. This is illustrated in the left-most set of mappings (in blue color) in Figure 3.2. For further details of this OBDA mapping, see the Supplementary Material (Section A.2). The full set of OBDA mappings used to expose Bgee relational data as virtual RDF triples are provided in `https://github.com/biosoda/bioquery`.

The code fragment in Listing 3.1 illustrates a mapping expressed with the Ontop relational-to-RDF mapping syntax, where the *source* is a SQL SELECT statement and the *target* consists of the corresponding RDF-based properties and classes. While direct and simple mappings (around 80% of the total) could in principle be automatically generated, complex ones such as the *isExpressedIn* relationship shown in Listing 3.1 can only be manually defined. Further explanations about this are available in Supplementary Material (Section A.2).

Once relational-to-RDF mappings have been defined with Ontop between the Bgee MySQL database and GenEx, the original data can be queried with SPARQL,

```
target  oma:GENE_{geneId} genex:isExpressedIn
        uberon:{anatEntityIdSPARQL} .
source  SELECT g.geneId,
        REPLACE(gc.anatEntityId,":","_") AS anatEntityIdSPARQL
        FROM globalExpression AS ge
        JOIN globalCond AS gc
        ON ge.globalConditionId = gc.globalConditionId
        JOIN gene AS g ON g.bgeeGeneId = ge.bgeeGeneId
```

LISTING 3.1: Ontop mapping to infer the "is expressed in" GenEx relation (i.e. target schema) based on the Bgee relational database (i.e. data source). Prefixes are defined in Supplementary Table A.1.

through the Bgee RDF virtual model. At query time, Ontop will translate SPARQL queries into SQL on-the-fly, using the mappings, and execute these over the Bgee relational database. Ontop has the advantage of supporting federated queries as part of SPARQL 1.1 and of being open source. In order to enable researchers to directly use the RDF representation of Bgee, we made available a public SPARQL 1.1 endpoint at `http://biosoda.expasy.org/rdf4j-server/repositories/bgeelight` as a query service (without any webpage associated to it). Nonetheless, the OBDA solution with Ontop has some limitations – we discuss some of these in Section A.2 in the Supplementary Material.

The **OMA** data are internally stored in an Hierarchical Data Format 5 (HDF5) file. This is not a database management system (DBMS) such as MySQL, but rather a data model and file format along with an API, libraries, and tools. Similarly to Bgee, we have to homogenise the OMA database model and syntax in order to enable integration with other biological RDF data stores (either virtual or materialised). For OMA, we chose to materialise the key parts of OMA data as an RDF graph, by implementing a hybrid approach, that combines materialisation and a possible RDF graph virtualisation for the sake of semantic enrichment and knowledge extraction, as described in detail in `https://qfo.github.io/OrthologyOntology`. The OMA RDF data and ORTH ontology are stored in a Virtuoso 7.2 triple store and a SPARQL endpoint is available at `https://sparql.omabrowser.org/sparql`. Further explanations regarding the OMA RDF data materialisation are available in Supplementary Section A.6.

3.3.2 Structured query interface layer

Once the data stores are accessible through SPARQL endpoints, as depicted in section 3.3.1, we can exploit means to link them at the data level. To do so, we identify common class instances and literals (e.g. strings) in order to establish "virtual links". We define a virtual link as an intersection data point between two data stores. The links are required in order to enable performing federated queries, given that they act as join points between the federated sources. Figure 3.3 illustrates virtual links among UniProt, Bgee and OMA. For example, OMA and Bgee describe complementary information about common genes (instances of the *orth:Gene* class), as well as taxa (instances of the *up:Taxon* class), both of which can serve as virtual links to connect the two sources. A federated SPARQL query written based on the virtual links is described in Supplementary Material Section A.3. To formally and explicitly describe virtual links, we adapted and extended the VoID RDF schema vocabulary [111] to include the concept of virtual links. We call this vocabulary Extended VoID (VoIDext). VoIDext is fully specified and exemplified in `https://biosoda.github.io/voidext/`. The entire metadata of virtual links among UniProt,

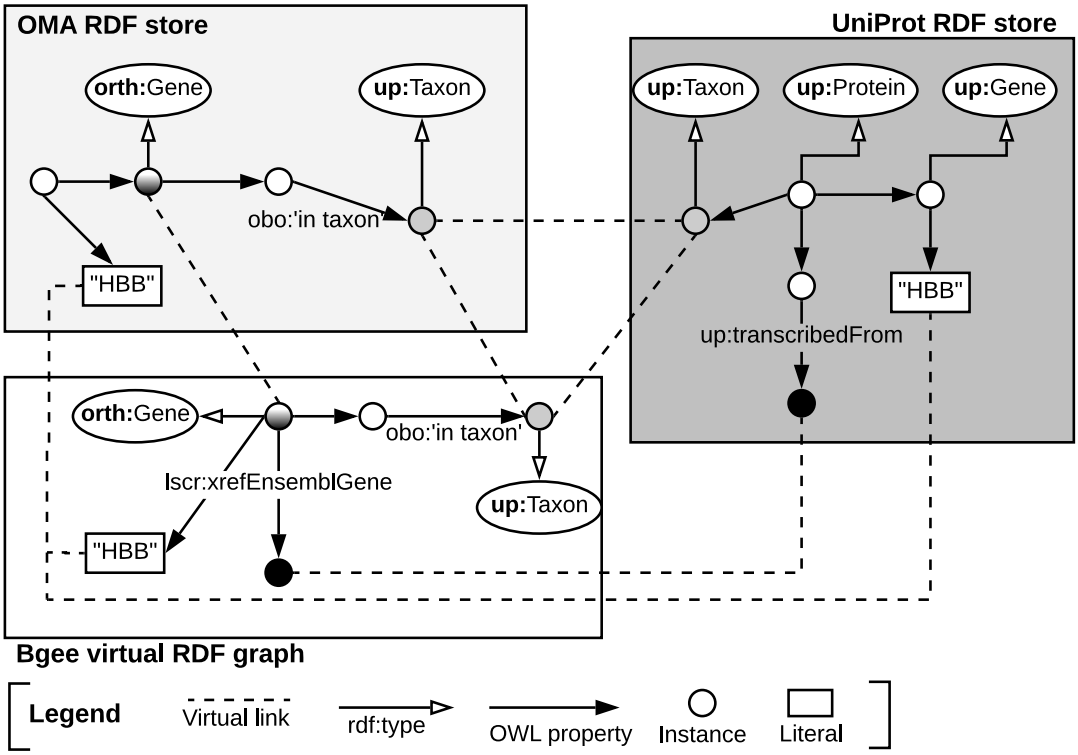

FIGURE 3.3: Example of virtual links among UniProt, OMA and Bgee data stores.

Bgee, and OMA RDF stores for the work depicted in this chapter are available at `http://purl.org/query/bioquery`. In VoIDext specification, we also depict the SPARQL queries to retrieve the virtual links among OMA, Bgee and UniProt that support the writing of joint federated SPARQL queries. These queries can be executed on the federated SPARQL endpoint illustrated in Fig. 3.1 `http://biosoda.expasy.org:8890/sparql`.

To leverage virtual links between Bgee and the other databases, we took advantage of the flexibility provided by Ontop when defining the Bgee OBDA mappings. We aimed at mapping Bgee data into corresponding instance IRIs and literals that already exist in OMA and UniProt RDF graphs. For example, species-related instance IRIs in the Bgee virtual graph are indeed exact matches of *up:Taxon* instance IRIs that are stored in the UniProt database.

Likewise, the code fragment in Listing 3.1 asserts the (re)use of **OMA** gene instances as part of the **Bgee** virtual RDF graph rather than creating new Bgee ones. In this way, we avoid additional *owl:sameAs* assertions to state that the two instances are actually the same. Thus, *orth:Gene* instances are intersection nodes (i.e. virtual links) between the Bgee and OMA graphs. Figure 3.3 (left-hand side) illustrates a shared *orth:Gene* instance between OMA and Bgee graphs. Further information about the virtual links depicted in Figure 3.3 is available in Section A.7 in the Supplementary Material.

Overall, we provide a federated SPARQL query endpoint along with an RDF store that exclusively contains metadata about the virtual links, and the SPARQL endpoints of the Uniprot, OMA and Bgee data stores. These metadata based on the VoIDext schema precisely define and document how the distributed datasets can be interlinked. Therefore, they may significantly facilitate the manual or automatic writing of a SPARQL 1.1 federated query, given that users are no longer required to discover the interlinks between the queried datasets on their own. In [128], we

detail the drawbacks of the current VoID link sets to represent virtual links, and the description of the novel VoIDext specification.

3.3.3 Application layer

The main goal of the application layer in our work is to enable users, even with no prior technical training, easy access to the integrated information from the three biological databases. We developed a user-friendly interface (illustrated in the top part of Figure 3.1), which is accessible at `http://biosoda.expasy.org/`. The interface presents a catalogue of representative query templates drafted together with domain experts. The queries are provided in natural language, with editable fields, and grouped in a tree structure according to the target knowledge domain(s) and information retrieved for each query. A search bar is also provided, which enables filtering the templates by keywords of interest (e.g. "disease").

For users with more advanced technical expertise, we also provide the option to show and modify the equivalent SPARQL queries. In doing so, our approach has the potential to increase the productivity of domain scientists in exploring the three heterogeneous data sets jointly. Additionally, the catalogue of questions is destined to grow according to user needs and feedback.

Moreover, because of the federated architecture of our system, its performance depends on that of the underlying data sources, e.g. UniProt. The availability of the underlying data stores is indicated by green labels in the top right corner of the webpage. For unavailable sources, the corresponding label is shown in yellow, as illustrated in the application layer in Figure 3.1.

By default, our system limits the total number of results returned, which allows for a faster response – the estimated response time is shown as a tag next to each query. However, the user can turn the limit option off, in order to obtain the full set of results. In this case, the response time may be significantly higher, which can largely be attributed to the SPARQL query execution time on the underlying sources. This has already been noted in similar previous systems [129]. In terms of scalability, the UniProt SPARQL endpoint is a good example, having already an active user base of more than one thousand users per month. As we directly rely on the infrastructure of the underlying sources, we can therefore expect our system to exhibit reasonable performance for approximately the same number of users.

3.4 Results

In this section, we first revisit our motivating example for integrated data access to the three databases (UniProt, OMA and Bgee) and then present experimental results based on a catalogue of 12 federated queries. All results are reproducible through our public interface described in section 3.3.3.

Recall our motivating example from the start of Section 3.2: *"What are the human genes which have a known association to glioblastoma (a type of brain cancer) and which furthermore have an orthologous gene expressed in the rat's brain"*. Answering the question requires solving the following three subqueries:

1. Retrieve human proteins with a disease description related to glioblastoma from UniProt.

2. Retrieve orthologs of these proteins in the rat from OMA.

3. Only keep those orthologs for which there exists evidence of expression in the rat's brain from Bgee.

The above steps translate to the federated SPARQL query in Supplementary Listing 3.1. The query produces a set of 15 human-rat orthologous pairs and can be executed in any SPARQL 1.1 endpoint. The full query, as well as a detailed list of results is available in the Supplementary Material in Section A.3.

We further evaluated the performance, in terms of runtime, of 12 federated queries that illustrate real use cases requiring information across the three databases (see Table 3.1). The results are reproducible through our public template-based search interface. A detailed analysis of the queries, including their natural language description, the equivalent federated SPARQL queries, as well as an explanation of the complexity for each query, can be found at `https://github.com/biosoda/bioquery`.

TABLE 3.1: Descriptions of the 12 federated queries across OMA, Bgee, UniProt used for evaluating our system. The queries can be further refined and executed through our template search interface available at `http://biosoda.expasy.org/`.

Query	Description
Q1	Proteins in OMA encoded by the INS gene and their evidence type from UniProt.
Q2	Rabbit proteins encoded by genes orthologous to the HBB-Y gene in Mouse and their associated information from UniProt.
Q3	Rattus Norvegicus proteins paralogous to Tp53 and their UniProt function annotations.
Q4	Mouse genes expressed in the liver which are orthologous to the human INS gene.
Q5	Genes orthologous to a gene expressed in the fruit fly brain.
Q6	Genes in Primates orthologous to a gene expressed in the fruit fly brain.
Q7	Anatomic entities where the ins zebrafish gene is expressed and the ins gene GO annotations.
Q8	Genes expressed in the human pancreas and their annotation in UniProt.
Q9	Genes expressed in the human brain during the infant stage and their UniProt disease annotation.
Q10	The orthologs of a gene that is expressed in the fruit fly brain and the UniProt annotations of these orthologs.
Q11	The orthologs **in Primates** of a gene that is expressed in the fruit fly brain and the UniProt annotations of the primate orthologs.
Q12	Proteins in humans, with a disease annotation, that are orthologous to a gene expressed in the rat brain.

Table 3.2 shows that most of the queries can be executed in a few seconds – up to 6 seconds for 9 out of 12 queries, with less than half a second for 3 out of these. This holds even for queries with higher complexity (number of triple patterns). A triple pattern is similar to a regular RDF triple, except that any part of the triple can be replaced by a variable [110]. Although preliminary, the results in Table 3.2 are encouraging for the use of SPARQL queries in data exploration tasks or in an interactive environment.

The outlier Q10 calls for discussion. By comparing the natural language description of Q10 against Q11 (see corresponding entries in Table 3.1, where the difference between the two queries is highlighted in bold in the description of Q11), we can intuitively deduce that the complexity stems from the high degree of generality of the sub-query that targets orthology information (OMA). In the case of Q10, retrieving an answer will require scanning the entire available orthology data and retrieving a large intermediate result set (orthologs found in any species, a total of 2269 results). By contrast, Q11 restricts the search space to the "primates" taxon only, which in practice results in a much lower query execution time (and a total of only 81 results). An important lesson derived from this example is that queries should always be

TABLE 3.2: Tests performed to evaluate our approach in terms of query execution time and the number of results. We evaluated 12 federated queries of varying complexity (measured in terms of number of triple patterns). Their description is provided in Table 3.1. All queries were executed twenty times, providing an average runtime and its standard deviation, given in seconds. The longest running query, Q10, is highlighted in bold.

Query	Sources	#Results	Mean run-time (s)	Stdev run-time (s)
Q1	OMA, UniProt	27	5.13	0.13
Q2	OMA, UniProt	3	0.36	0.02
Q3	OMA, UniProt	1	0.47	0.05
Q4	OMA, Bgee	2	0.37	0.05
Q5	OMA, Bgee	5322	4.72	0.2
Q6	OMA, Bgee	38	2.38	0.09
Q7	Bgee, UniProt	16	68.18	105.85
Q8	Bgee, UniProt	58	33.17	25.23
Q9	Bgee, UniProt	6	2.37	0.04
Q10	**Bgee, OMA, UniProt**	**2269**	**349.18**	**4.19**
Q11	Bgee, OMA, UniProt	81	6.4	0.14
Q12	Bgee, OMA, UniProt	3	5.24	0.11

as specific as possible, in order to limit both the search space and the size of intermediate results to the minimum necessary to obtain a relevant answer. Although this query illustrates a worst-case scenario, the results are still returned in less than 6 minutes—a latency which is tolerable for investigations in a biological research context.

3.5 Conclusions and outlook

Data integration across heterogeneous biological databases promises to be one of the catalysts for gaining new biological insights in the postgenomic era. Here, we introduced an ontology-driven approach to bioinformatic resource integration. This approach enables complex federated queries across multiple domains of biological knowledge, such as gene expression and orthology, without requiring data duplication. The integration of the three sources promises to open the path for novel comparative studies across species, for example through the analysis of orthologs (OMA) of human disease-causing genes (UniProt) and their expression patterns in model organisms (Bgee). Thanks to modelling decisions made at the semantic (ontology) and data (assertions) levels, we established various virtual links among Bgee, OMA and UniProt data stores. Moreover, making these virtual links available in VoIDext facilitates the task of writing federated SPARQL queries, since users have an explicit representation of the connections (join points) between the three data sources. We furthermore lay the groundwork for bringing the benefits of integrated data to domain specialists through a template-based search engine available online, which does not require users to know SPARQL in order to pose questions on the integrated data.

The catalogue of federated queries across the three data sources can serve as a starting point towards answering new biological questions that span across the domains of evolutionary relationships and gene expression. The results presented in this study can be easily reproduced through our template search interface. We furthermore make available all source code, including the template search interface

code, relational-to-RDF mappings and the catalogue of queries, with the goal of facilitating reuse of these components for further research. All resources are available in our GitHub repository.

Our experiments show that most queries in our catalogue can be answered within seconds. And although the more complex queries take several minutes to complete, we expect this turnaround time to be tolerable for most interested users—particularly considering the alternative of manually querying the resources and combining the results. In the future, we plan to include more resources in the federated system, focusing primarily on publicly available databases. We plan to start with those that already provide SPARQL 1.1 endpoints, for which the main work would entail defining virtual links to our existing integrated resources. In a second step, we can envision integrating more relational databases, for which the main work required would be to define the relational-to-RDF mappings, analogous to those presented for Bgee in the current work. Our first aim is to make more of the publicly accessible databases interoperable for the purpose of advancing research through integrated data access. Nevertheless, we can envision also integrating access control policies in the future, which would enable including sensitive resources, such as patient databases, in the federated system. A good starting point for understanding the types of existing access controls for RDF data, in order to accommodate these in our federation architecture, is the recent survey [130]. Finally, we plan to add a federated query optimiser to our system to further improve the response time. We also note here that the application interface directly queries the underlying databases without performing additional tasks, such as considering all gene name synonyms to get broader results. We plan to support such features as part of future work. To support virtual link evolution, we aim to develop a tool to automatically detect broken virtual links because of either data schema changes or radical modifications of instances' IRIs and property assertions. Meanwhile, we encourage contributions to the current query catalogue, which will serve in the study of a natural language search interface for the integrated biological data as part of future work.

Chapter 4

Applied Case Studies for Data Integration in the Life Sciences

This chapter introduces two applied case studies that illustrate the benefits of semantic data integration in the Life Sciences, particularly for comparative genomics. These tutorials can be used as starting points in testing new research hypotheses, using federated queries across multiple RDF data sources either within the same domain (orthology, Section 4.1) or within connected domains (gene expression and orthology, Section 4.2). The first part of this chapter therefore illustrates the data models of four databases publicly available in RDF which include orthology information: EBI, MBGD, OMA, OrthoDB and shows how these sources can be (jointly) queried. The second part of this chapter is a more in-depth case study, focusing on the relation between expression evolution and branch lengths of paralogous copies following a gene duplication event, using federated queries across the gene expression database Bgee and the orthology database OMA.

4.1 Querying orthology data using SPARQL

The increasing use of Semantic Web technologies in the life sciences, in particular the use of the Resource Description Framework (RDF) and the RDF query language SPARQL, opens the path for novel integrative analyses, combining information from multiple data sources. However, analyzing evolutionary data in RDF is not trivial, due to the steep learning curve required to understand both the data models adopted by different RDF data sources, as well as the equivalent SPARQL constructs required to benefit from this data – in particular, recursive property paths. In this chapter [1], we provide a hands-on introduction to querying evolutionary data across several data sources that publish orthology information in RDF, namely: The Orthologous MAtrix (OMA), the European Bioinformatics Institute (EBI) RDF platform, the Database of Orthologous Groups (OrthoDB) and the Microbial Genome Database (MBGD). We present four protocols in increasing order of complexity. In these protocols, we demonstrate through SPARQL queries how to retrieve pairwise orthologs, homologous groups, and hierarchical orthologous groups. Finally, we show how orthology information in different data sources can be compared, through the use of federated SPARQL queries.

4.1.1 Introduction

Gene classification based on evolutionary history is essential for many aspects of comparative and functional genomics - reviewed in [131]; [132]. On the one hand,

[1] This section is based on the tutorials published in [8]

evolutionary relations are often described as binary relations. Two genes that share a common ancestor are defined as homologs. We can classify homologs into orthologs, which originate from a speciation event; paralogs, which originate from a gene duplication; and xenologs, which originate from horizontal gene transfer [133]. On the other hand, Hierarchical Orthologous Groups (HOGs) are hierarchical clusters of corresponding genes where each level in the hierarchy refers to a common ancestral gene at a taxonomic level of reference [134]. Further details about orthology, paralogy, xenology and various kinds of groupwise orthology relationships are described in [135]. Identifying orthologs and HOGs is valuable in several contexts such as gene function inference, gene evolution dynamics and comparative genomics. To query and interoperate biological databases, Semantic Web Technologies are being increasingly adopted, in particular the use of the Resource Description Framework (RDF) and SPARQL protocol and RDF query language. However, despite the progress they have enabled in several fields, particularly in the life sciences [136], [137], [138], there are still significant challenges that limit their use for the larger scientific community. In particular, analysing evolutionary relationship data in RDF poses the following challenges:

1. complex data models - for example, while storing data in a hierarchical structure (HOGs) results in significant performance benefits for common analyses, such as computing orthologs of a specific gene in a different model organism, the hierarchy also results in requiring advanced knowledge of the SPARQL language (in particular, recursivity) in order to benefit from the RDF representation of HOGs. In this chapter, we present a series of hands-on examples, in increasing order of complexity, to familiarise the reader with the basic concepts needed to query evolutionary relationships in orthology databases.

2. heterogeneous data models - understanding the data model of a single orthology database might not be sufficient in general, since different databases have made different design decisions. We help overcome this challenge by depicting how the following major Orthology Databases structure their data in RDF, as well as how they can be queried using SPARQL: the Orthologous MAtrix (OMA) [12], the European Bioinformatics Institute (EBI) RDF platform [139], the Database of Orthologous Groups (OrthoDB) [140] and the Microbial Genome Database (MBGD) [141]).

3. overhead of integration into existing analysis pipelines.

   The limited rate of adoption of Semantic Web Technologies can be explained by the reluctance of bioinformaticians to change their existing workflows in order to accommodate new data formats based on the RDF framework. For example, retrieving orthology information using public SPARQL endpoints instead of the more traditional file-based data exchange or full database dumps. A SPARQL endpoint is an access point for receiving and processing SPARQL protocol requests. In this chapter, we show through concrete examples that integrating the results of SPARQL queries into existing analyses is a straightforward task - more specifically, we show how to transform the results into regular Pandas dataframes in Python. Furthermore, we provide an accompanying Jupyter notebook where all the examples presented in this chapter can be directly tested and further refined.

This chapter has several goals:

1. Understanding Orthology Data Models. Become familiar with how evolutionary relationships are represented in RDF across several databases. Learn about the data modelling decisions: common points as well as differences between these data sources to support the choice of one or more of them for a given analysis.

2. Understanding how to query orthology data using SPARQL. To this end, we extend the introduction and examples in [142] [143], while also covering multiple, distributed orthology data sources.

3. Integrating external sources. Leverage connections to other external bioinformatics resources that make their data available in public SPARQL endpoints based on cross-references. In particular, learn about the role of UniProt cross-references as a bridge between different data sources in integrative analyses.

In addition, we show how to use SPARQL to make meta-analyses combining multiple orthology databases. For instance, for a given gene, which are the orthologs in a given database which are not present in another one? Finally, we show how to leverage SPARQL aggregations in order to get useful statistics about orthology data available in the sources. Finally, learn how to leverage SPARQL results in downstream analyses by converting them to Pandas dataframes. This is illustrated through a series of hands-on exercises in the accompanying Jupyter notebook (exercises provided in Python).

The protocols presented in this chapter are aimed at bioinformaticians who are already familiar with the basics of SPARQL and wish to learn how orthology data can be integrated in their research analyses programmatically, through the use of (federated) SPARQL queries.

4.1.2 Materials

In the following paragraphs, we briefly describe the orthology databases considered in this chapter.

OrthoDB [140] contains orthologous genes along with evolutionary and functional annotations. It relies on HOGs to enable different orthology information resolutions with regards to more closely related species. The 2018 OrthoDB version covers thousands of eukaryotes, prokaryotes, and viruses. OrthoDB data is available in RDF through the public SPARQL endpoint at `https://sparql.orthodb.org/sparql`. We note here that the timeout for the public SPARQL endpoint is limited to 100 seconds - more precisely, queries with longer estimated execution time will not be allowed to run.

MBGD [141] is a comparative genomics database that contains orthology information about bacteria, archaea and unicellular eukaryotes. The 2018 MBGD version has more than six thousand genomes. The MBGD SPARQL endpoint is available online at `http://mbgd.genome.ad.jp/sparql/`.

OMA [12] provides orthologous gene inferences covering all three domains of life: Archaea, Bacteria, and Eukarya. Although mainly focusing on orthology information, OMA also provides paralogy information (i.e. genes related by duplication). Other homology information is not explicitly available but might be manually or automatically extracted from HOGs [143]. The 2020 OMA version has 2326 species and can be queried through the SPARQL endpoint at `https://sparql.omabrowser.org/lode/sparql`. OMA reports multiple kinds of pairwise and groupwise orthologous relationships, described in [144].

EBI is one of the largest bioinformatics resource providers in Europe [145]. The EBI RDF platform includes pairwise orthologous genes information from Ensembl database [146]. The SPARQL endpoint to access the EBI data is available at https://www.ebi.ac.uk/rdf/services/sparql. For further details, see https://www.ebi.ac.uk/rdf/documentation/ensembl/.

We group the aforementioned databases based on the orthology information type they provide as follows:

I. **Hierarchical Orthologous Groups (HOGs)**. The three data sources that provide evolutionary relationship data in RDF to represent HOGs are OrthoDB, MBGD and OMA. Although the RDF data models of MBGD and OMA both rely on the ORTH ontology [119], they use different ORTH versions. However, SPARQL queries running over either of the two sources can be formulated in a very similar manner. In the case of OrthoDB, data are organised according to their own internal data model, while also providing cross-references to the UniProt RDF store.

II. **Homologous groups** are sets of homologous genes without any hierarchical grouping ("flat"). All members are homologous to all other members, with no distinction of paralogy or orthology. However, the kind of homologous groups considered in this tutorial do not contain "out-paralogs" [147] —i.e. paralogs which result from gene duplications which took place prior to the last common ancestor of all species in the databases. Furthermore, each homologous group can still be associated with a taxonomic level, which indicates to which species clade its members belong. Example of orthology databases from which we can extract these homologous groups are OMA, OrthoDB and MBGD.

III. **Pairwise orthology**. Apart from the aforementioned orthologous groups, evolutionary data can also be provided in the form of pairwise orthologous genes. Among the sources that provide this type of information in RDF, we consider in this chapter EBI, OMA, OrthoDB and MBGD.

We mention that the SPARQL endpoints of the databases may sometimes be temporarily unavailable, for example, for maintenance purposes. In these cases, the queries provided in this chapter may not be able to run or may become unresponsive due to the unavailability of the corresponding SPARQL endpoint. Should these issues persist for a longer period of time, it is advised to contact the respective database support through the email address indicated on the SPARQL endpoint webpage.

4.1.3 Applicable Ontologies

The main existing ontology to represent and structure the orthology information is the Orthology (ORTH) Ontology [119]that is recommended by the Quest for Orthologs Consortium (QfO). The second version of the ORTH ontology is further described at https://github.com/qfo/OrthologyOntology) [92]. Both OMA and MBGD rely on the Orthology Ontology. More precisely, OMA uses the ORTH version 2, while MBGD version 1. In contrast, OrthoDB relies on an internal data model while the fragment of EBI relevant to this chapter mainly reuses the "is orthologous to" pairwise property from the Semanticscience Integrated Ontology (SIO). In the context of the four databases depicted in this chapter, a way to possibly identify relevant URIs (Uniform Resource Identifiers) for properties of interest such as "in taxon", "is orthologous to", but also for taxonomic identifiers etc, is the Ontology Lookup Service (OLS) [73], available online at https://www.ebi.ac.uk/ols/index.

We note below the other main underlying ontologies, controlled vocabularies and taxonomies used by at least one of the four data sources. Further details are depicted in Table 1 in the Extended data available in our github repository.

- The Gene Ontology (GO) to specify molecular functions, for example.
- The Relation Ontology (part of the Open Biological and Biomedical Ontology (OBO) Foundry) for properties such as "in taxon".
- SIO for properties such as "protein encoded by" and "gene encodes protein".
- ENSEMBL for gene identifiers.
- The National Center for Biotechnology Information (NCBI) taxonomy.
- Dublin Core Initiative Metadata (DCMI) terms to state identifiers and descriptions in OMA and MBGD databases.
- UniProt core ontology – for scientific/common names of species, as well as cross-references.

In the Section 4.1.4 we provide a more detailed introduction to how each of the four data sources considered in this chapter structures orthology data. We illustrate through concrete examples the commonalities as well as differences between the data sources.

4.1.4 Data Models

In this section we provide a brief introduction to the data models of the orthology databases considered in this chapter, in order to facilitate the understanding of the SPARQL queries presented in the Protocols Section.

We first present a simple example to illustrate how SPARQL queries can be formulated, starting from a given data model. Figure 4.1 illustrates a simplified graph abstraction of an RDF data model targeting proteins and genes that encode these proteins, and their related species (taxa). Furthermore, the model includes cross-references to the corresponding UniProt entries. These cross-references can be useful in formulating federated SPARQL queries. Federated queries can retrieve information from multiple RDF data sources, using the UniProt entry as an intersection ("join point").

Figure 4.1 can be used as a guide to formulate simple SPARQL queries that aims to answer questions of interest. Such questions can be related to, for example, proteins found in a given species, or those corresponding to a specific UniProt accession number. For readability, we omitted specific namespaces and exact URIs. Instead, we use the human readable labels of properties, such as "in taxon". The URIs that represents these terms (e.g. Protein, Gene, in taxon) depend on the underlying database being queried. Nevertheless, if the database is reusing existing vocabularies to model their data, a reference for mapping these terms to their corresponding identifiers (URIs), is the Ontology Lookup Service (OLS), and the Linked Open Vocabularies (LOV) `https://lov.linkeddata.es`. For example, searching for "in taxon" in OLS will result in the first answer returned being the OBO URI `http://purl.obolibrary.org/obo/RO_0002162`.

An example question for which the corresponding SPARQL query can be formulated, based on the simplified query graph in Figure 4.1, could be: "What are Rattus Norvegicus proteins available in the database?" In order to retrieve this information,

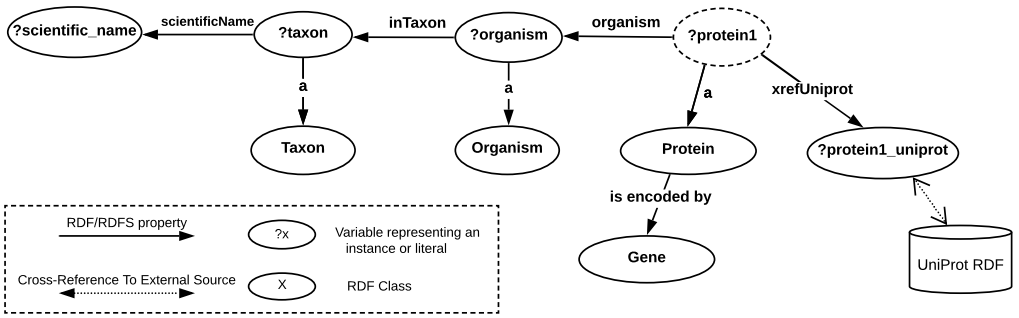

FIGURE 4.1: Simplified query graph that can be used as support for writing SPARQL queries to extract relevant information, such as proteins in a particular species.

```
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX orth: <http://purl.org/net/orth#>
SELECT ?protein1 WHERE {
       ?protein1 a orth:Protein.
       ?protein1 orth:organism ?organism.
       ?organism <http://purl.obolibrary.org/obo/RO_0002162> ?taxon.
       ?taxon up:scientificName "Rattus norvegicus".
 }
```

LISTING 4.1: SPARQL query to retrieve Rattus norvegicus proteins from OMA

we can start from the ?protein1 variable, of type Protein, illustrated in the top right corner of the figure, which we need to connect to the taxon scientific name ("Rattus Norvegicus") via "organism". Note that the SPARQL users can define variable names as they want by following the SPARQL syntax where the question mark (?) means that is a variable. Following the arrows from ?protein1 to ?taxon (in Figure 4.1, from right to left) we can formulate the SPARQL query in listing 4.1, which can be executed in the OMA SPARQL endpoint.

Changing the target species of interest only requires changing the corresponding text in between quotes. For example, to find the human proteins available in the data source, change "Rattus norvegicus" to "Homo sapiens" (note: the query is case-sensitive). Any additional information can be retrieved by adding the corresponding triple pattern to the query, while also making sure to add any relevant variables to the select statement. For example, to also retrieve the corresponding UniProt entries for the proteins, we can apply the following modification (see last line of the query statement in listing 4.2):

For a more complete introduction to SPARQL and RDF in the context of biological databases see [7]. Next we will introduce the data models of the orthology databases considered in this chapter.

Figure 4.2 illustrates a few of the members of a HOG, the main data structure in MBGD. In particular, this MBGD cluster has the identifier 28799. Members of an MBGD orthologous cluster can be either genes, domains or other clusters. These nested orthologous clusters are built at specific taxonomic levels in the hierarchy. For example, the cluster highlighted in blue in Figure 4.2 was built at taxonomic level 32, Myxococcus. The hierarchy needs to be traversed in order to reach genes, such as mxa:PL1911 that is highlighted in red in Figure 4.2, or domains (sub-gene

```
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX lscr: <http://purl.org/lscr#>
SELECT ?protein1 ?uniprot_entry WHERE {
       ?protein1 a orth:Protein.
       ?protein1 orth:organism ?organism.
       ?organism <http://purl.obolibrary.org/obo/RO_0002162> ?taxon.
       ?taxon up:scientificName "Rattus norvegicus".
       ?protein1 lscr:xrefUniprot ?uniprot_entry.
 }
```

LISTING 4.2: SPARQL query modified to also retrieve the UniProt entry corresponding to the variable ?protein1 (see last line)

```
?hog_cluster a orth:OrthologsCluster.
?hog_cluster orth:hasHomologous* ?gene_X.
?hog_cluster orth:hasHomologous* ?orthologous_gene_Y.
```

level) which belong to an orthologous cluster at a given taxonomic level. The RDF model is more suitable for representing such hierarchical data structures than the relational data model [7], given that RDF is a graph data model. Moreover, querying orthology RDF data can benefit from SPARQL 1.1 recursive graph patterns such as property paths. The main construct in SPARQL required to retrieve the orthologous genes of a gene of interest X will then be the following recursive pattern:

For example, we can replace *?gene_X* with the URI of the human Hemoglobin Subunit Beta (HBB) gene, namely: `<http://mbgd.genome.ad.jp/rdf/resource/gene/hsa:HSA_4504349>`, which would enable retrieving all orthologs of human HBB through the *?orthologous_gene_Y* variable. The asterisk (*) following the "orth:hasHomologous" property indicates that this property should be matched recursively.

Based on Figure 4.3, SPARQL queries can be formulated by following the directions and labels of arrows in order to formulate triple patterns. For example, to retrieve all genes (i.e. instances of the orth:Gene class) of a given HOG, we can follow the graph structure from root to leaf members by performing the query as shown in the code fragment below. In other words, the *?gene1* variable values illustrated as the left-side member in the cluster *?hog_cluster* (see Figure 4.3).

This SPARQL query will retrieve all the genes in the MBGD Hierarchical Orthologous Group represented with the identifier 28799.

Similarly, the HOG structure in OMA is abstracted in Figure 4.4. Both figures can be used as a guide in formulating SPARQL queries, by following the directions of the arrows in order to formulate triple patterns. Since both the MBGD and the OMA models rely on the ORTH ontology [119], the two graph structures are very similar and therefore SPARQL queries can be formulated with only minor differences for both data sources. Figure 4.5 illustrates the data model of the portion of the EBI RDF

```
PREFIX orth: <http://purl.org/net/orth#>
PREFIX cluster-id:<http://mbgd.genome.ad.jp/rdf/resource/cluster/>
SELECT ?hog_cluster ?gene1 WHERE {
   VALUES ?hog_cluster {cluster-id:2018-01_default_28799}
   ?hog_cluster a orth:OrthologsCluster.
   ?hog_cluster orth:hasHomologous* ?gene1.
   ?gene1 a orth:Gene.
   }
```

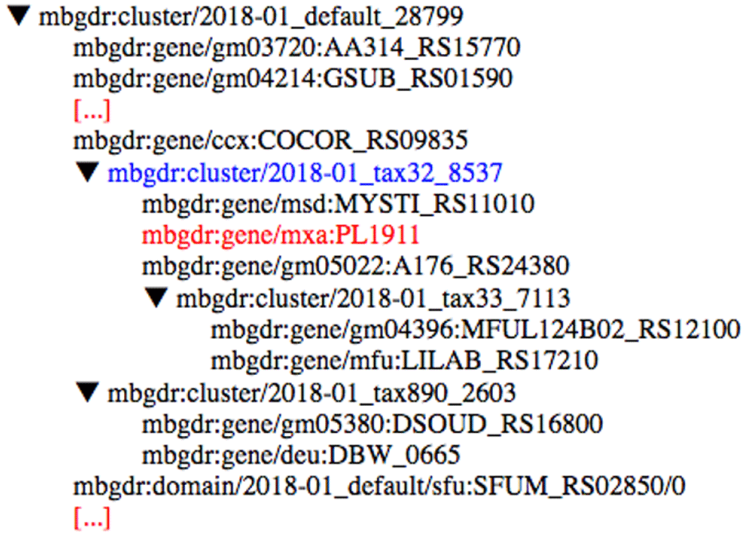

FIGURE 4.2: A fragment of the hierarchical orthologous cluster no. 28799 in MBGD. A cluster can consist of genes, domains (sub-genes) or further nested orthologous clusters. Multiple levels of the hierarchy may need to be traversed recursively in order to reach a given orthologous gene. For example, the gene mxa:PL1911 (highlighted in red) can be reached through the member orthologous cluster 2018-01_tax32_8537 (shown in blue). This can be achieved in SPARQL through a recursive graph pattern, using the hasHomologous property path - a graphical abstraction of the RDF representation is provided in Figure 4.3.

graph describing orthology information. In contrast to OMA, OrthoDB and MBGD, EBI only provides pairwise orthologous genes.

Figure 4.6 illustrates the structure of Orthologous Groups in the OrthoDB RDF. Here, genes are direct members of OrthoGroups built at a given taxonomic level (Clade), e.g. Cyanobacteria. We mention that OrthoDB provides information in RDF, including sequence length, number of exons for gene members, as well as evolutionary rates, functional category and others for orthologous groups (for more details see Extended data available online.

4.1.5 Choosing the relevant target gene identifier in RDF (URIs)

One of the challenges in formulating SPARQL queries is identifying the relevant URIs for the resources of interest. In the case of queries targeting the orthology domain, the resources of interest will usually consist of target genes, for which the orthologous genes need to be retrieved from the RDF data store. A simple way to obtain the relevant URIs in the case of the four data sources considered in this chapter, is to start from the UniProt accession numbers for the target gene of interest. This accession number can be identified through the online search interface of UniProt at `www.uniprot.org`, by searching for the gene name of interest, for example, "HBB" (or "human HBB"). The column "Entry" in the result page contains the corresponding UniProt accession number. From here, the URI can be obtained by concatenating the UniProt namespace prefix: `http://purl.uniprot.org/uniprot/` with this accession number.

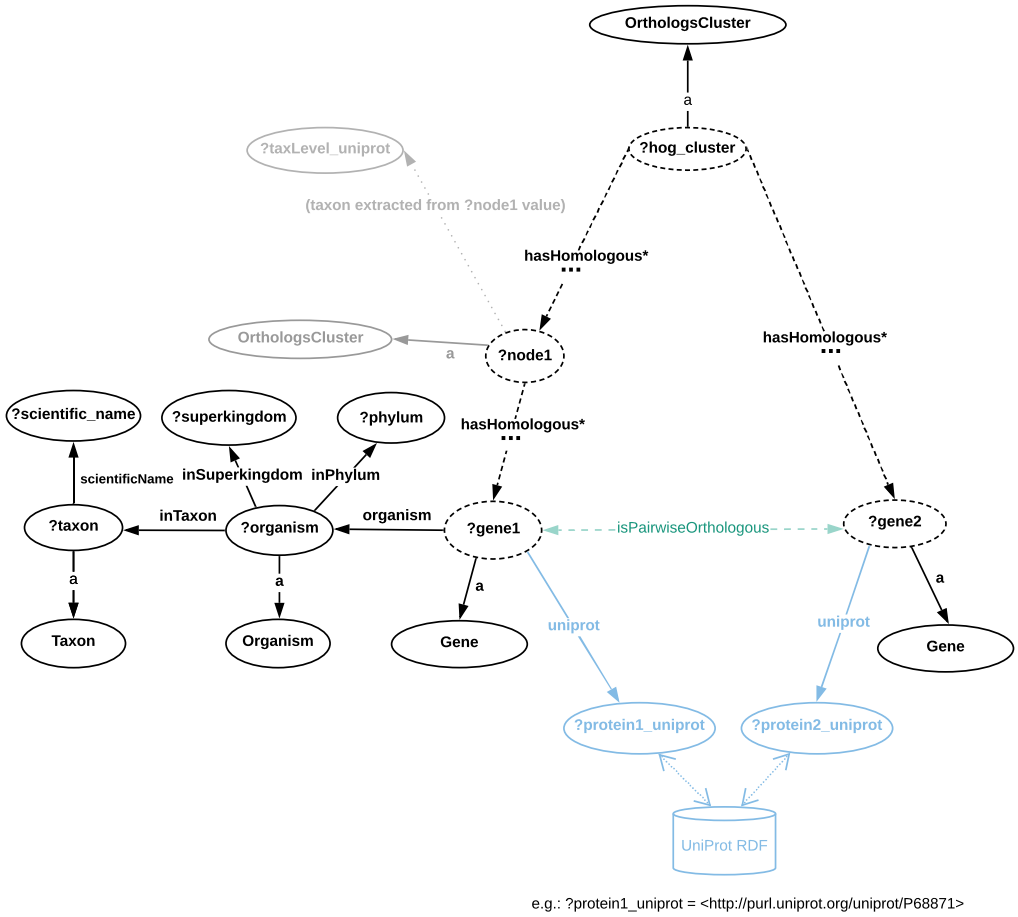

FIGURE 4.3: Directed graph abstraction of a portion of the MBGD RDF graph related to hierarchical orthologous groups. In all data model figures, nodes are either classes or variables, and edges are RDF properties. The terms preceded by a question mark (e.g. *?gene1*) represent variables assigned with either zero or more literals or URIs. Dashed edges illustrate the orth:hasHomologous property that can be stated zero or more times, recursively. URI prefixes were omitted. MBGD is gene-centric and contains taxonomic ranges where HOGs are built are not directly available in RDF - in some cases these can be extracted from the cluster URI (e.g. `http://mbgd.genome.ad.jp/rdf/resource/cluster/2018-01_tax32_8537`) corresponds to taxonomic identifier 32, Myxococcus). By contrast, the taxonomic information per gene entry is richer in MBGD than in OMA, including explicit Superkingdom and Phylum information. Example SPARQL queries based on this graph abstraction are provided in the "Protocols" section, as well as in the accompanying Jupyter notebook. The pairwise orthology information is not directly available (e.g. through an RDF property), but can be extracted from the Orthologs Cluster (to highlight this, the "isPairwiseOrthologous" is shown in green with a dashed arrow).

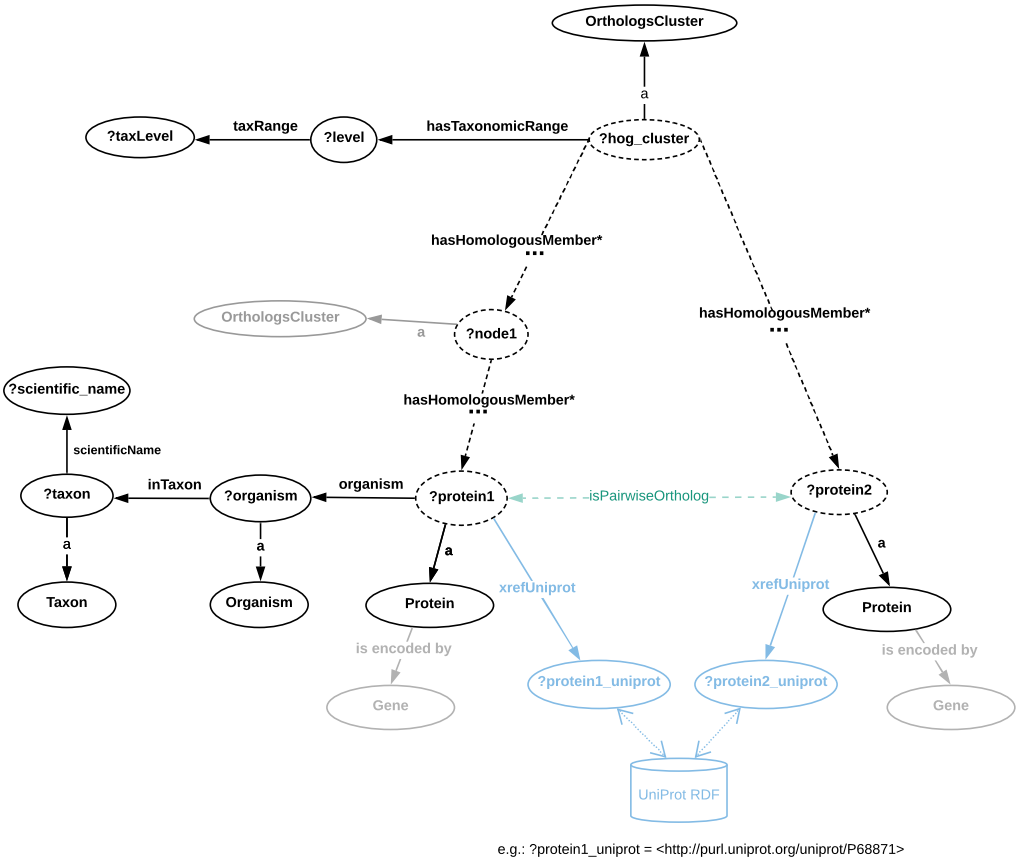

FIGURE 4.4: Directed graph abstraction of a portion of the OMA RDF graph related to hierarchical orthologous groups. In this Figure, dashed edges illustrate the orth:hasHomologousMember property that can be stated zero or more times, recursively. OMA is proteincentric, however the corresponding genes that encode the proteins are also available in RDF through the "is encoded by" property (a cross-reference to Ensembl identifiers is also provided). Furthermore, the taxonomic ranges where HOGs were built are asserted through the "hasTaxonomicRange" property. The pairwise orthology information is not directly available (e.g. through an RDF property), but can be extracted from the Orthologs Cluster (to highlight this, the "isPairwiseOrthologous" is shown in green with a dashed arrow). Note: URI prefixes were omitted.

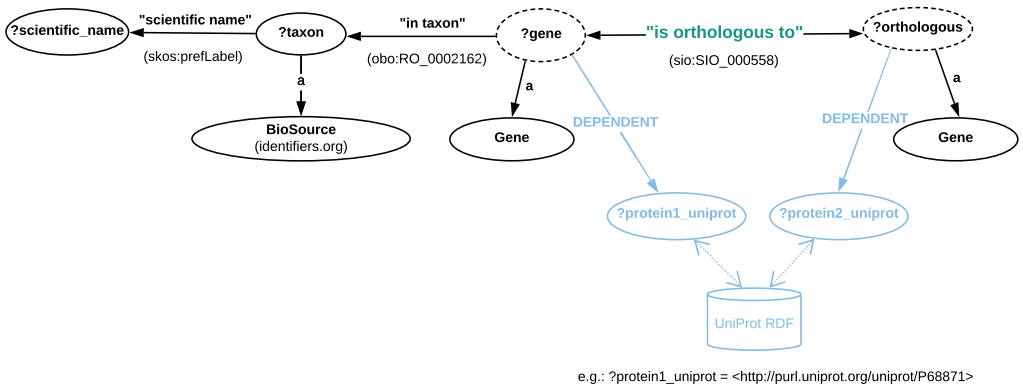

FIGURE 4.5: Directed graph abstraction of a portion of the EBI RDF graph related to pairwise orthologous genes. Moreover, as opposed to the RDF representations in OMA and MBGD, here the pairwise orthology is explicitly asserted through the "is orthologous to" property (more precisely, `http://semanticscience.org/resource/SIO_000558`) as shown in this Figure. However, there is no information available regarding orthologous clusters. Moreover, the Gene class here is in fact the OBO (not ORTH) class, i.e. `http://purl.obolibrary.org/obo/SO_0000704`. Instances of these genes can be specified either through their cross-reference to UniProt (the `http://rdf.ebi.ac.uk/terms/ensembl/DEPENDENT` property) or directly through their ENSEMBL identifier, by fixing the value of *?gene* to the concatenation of `http://rdf.ebi.ac.uk/resource/ensembl/` and the corresponding Ensembl identifier. Finally, the taxonomic identifiers are provided via instances of the BioSource class, `http://www.biopax.org/release/biopax-level3.owl#BioSource`.

For example, in the case of human HBB, the URI will be `http://purl.uniprot.org/uniprot/P68871`. Using this information and the cross-reference properties available in each of the four databases (OMA, OrthoDB, MBGD and OrthoDB), the appropriate SPARQL queries can be formulated, according to the examples shown previously in Figure 4.3 – Figure 4.6. In the "Protocols" section we give concrete examples of queries for each of the four data sources, which can be directly executed in the corresponding SPARQL endpoints.

4.1.6 Protocols - SPARQL queries (OrthoDB, EBI, OMA, MBGD)

In this section, we provide four protocols to (i) retrieve pairwise orthologs through SPARQL queries from EBI, OMA, MBGD, as well as (ii) homologous groups from OMA, MBGD and OrthoDB (iii) restrict the search to a given taxonomic level (iv) perform meta-analyses across multiple data sources providing orthology information, and aggregations using the entire data available in a given data source. All protocols presented below are included in the accompanying Jupyter notebook.

For the sake of simplicity, genes are identified with either their Ensembl identifiers or their cross-reference to the UniProt accession number. In this chapter, we assume the reader already knows the UniProt primary accession number of the searched gene. In general, this number can be found by searching for the corresponding gene name in the UniProt webpage, for example, "HBB" (i.e. "hemoglobin

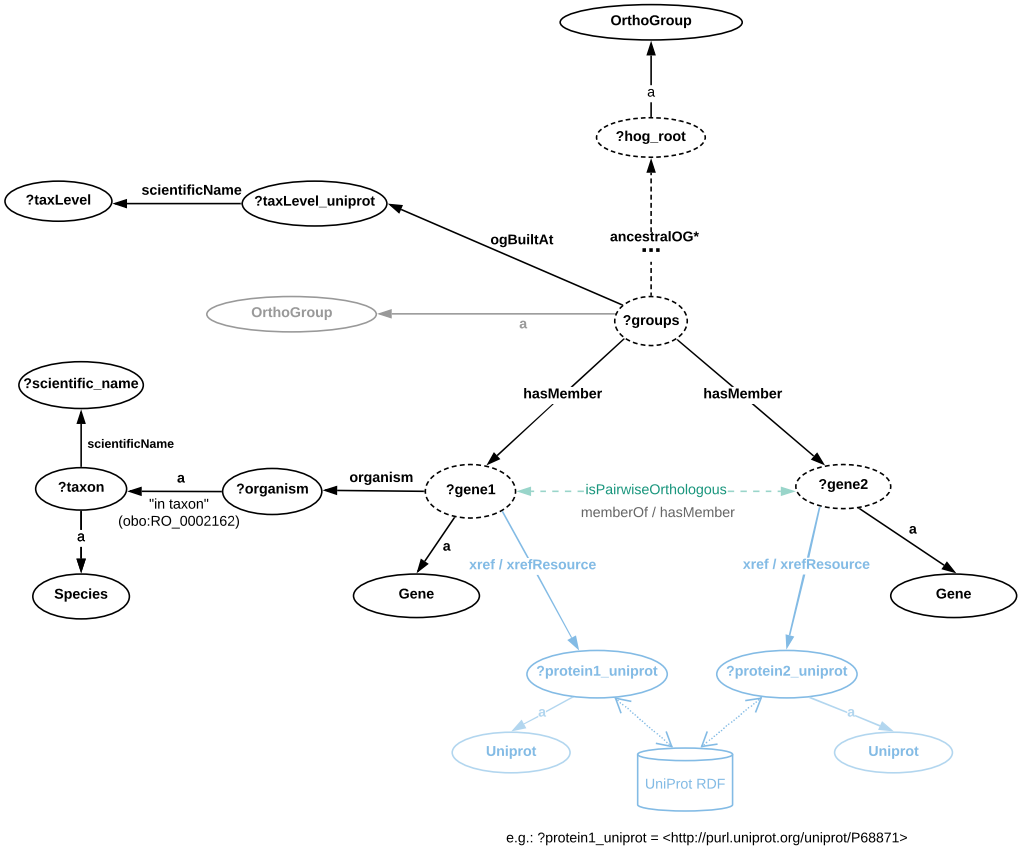

FIGURE 4.6: Directed graph abstraction of a portion of the OrthoDB RDF graph related to orthologous groups. Note that the abstract relation *"?gene1 isPairwiseOrthologous ?gene2"* is derived by considering the concrete property path *"?gene1 :memberOf / :hasMember ?gene2"* that further implies the following joint triples: *"?gene1 :memberOf ?group. ?group :hasMember ?gene2."*. In this Figure, genes are direct members of OrthoGroups built at a given taxonomic level (Clade), e.g. Cyanobacteria, available through the "ogBuiltAt" property. The crossreferences to UniProt (as well as Ensembl and Entrez) are available through a 2-triple pattern (for examples see "Protocols" section).

```
PREFIX : <http://purl.orthodb.org/>
SELECT DISTINCT ?protein1 ?protein2 {
    VALUES(?protein1){(<http://purl.uniprot.org/uniprot/P68871>)}
    ?og a :OrthoGroup.
    ?gene1 :memberOf ?og.
    ?og :hasMember ?gene2.
    ?gene1 :xref ?xref1.
    ?xref1 :xrefResource ?protein1.
    ?gene2 :xref ?xref2.
    ?xref2 :xrefResource ?protein2.
    ?protein2 a :Uniprot.
  }
```

LISTING 4.3: Query Q0

subunit beta"). As a reminder, the UniProt protein identifier in RDF is a URI composed of the UniProt accession number (e.g. P68871) appended to the UniProt namespace prefix: `http://purl.uniprot.org/uniprot/`. For instance, in the case of the human HBB gene, the corresponding URI identifier is `http://purl.uniprot.org/uniprot/P68871`.

Protocol 1: Retrieve pairwise orthologs

In this protocol we illustrate the basic task of retrieving the pairwise orthologs of a given gene, for example the HBB (Hemoglobin subunit beta) human gene. This is illustrated on the four orthology databases that provide pairwise orthology information in RDF: OrthoDB, EBI, OMA and MBGD. The corresponding SPARQL queries to retrieve the pairwise orthologs can be formulated as shown below. We note that the resulting orthologs are also provided using their "clickable" cross-reference link to UniProt. This can directly be used to find out more information about the resulting genes (e.g. name, location, expression) and has the added advantage that results originating from different orthology databases can then be compared against each other. All Protocol 1 queries retrieve a set of 2-tuples *(?a, ?b)*, where *?a* is the gene given as an input to look for its orthologous genes, which are assigned to *?b*. Therefore, the 2-tuple *(?a, ?b)* set represents the binary relation *"?a is orthologous to ?b"*. In all queries except for Q2, these genes are represented with UniProt entries, more specifically, UniProt URIs (see subsection "Choosing the relevant target gene identifiers in RDF" for further details). In Q2, the input gene is represented with an Ensembl gene URI as depicted in b

a) **Retrieving OrthoDB pairwise orthologs**

The code fragment displayed in listing 4.3 shows the SPARQL query to retrieve pairwise orthologs of the human HBB gene from the OrthoDB database. The following link `http://purl.org/orthology/q0` is provided to directly execute the query at the OrthoDB SPARQL endpoint. We denote this query as Q0.

The HBB gene is also represented in the latest code fragment with the UniProt URI `http://purl.uniprot.org/uniprot/P68871`. However, differently to other databases, OrthoDB states cross-references to UniProt by joining three triple patterns: *?g :xref ?x* and *?x :xrefResource ?p* and *?p a :Uniprot*.

b) **Retrieving EBI pairwise orthologs**

The code fragment shown in listing 4.4 and denoted here as Q1, depicts a SPARQL query to retrieve pairwise orthologs of the human HBB gene from

```
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX ensembl: <http://rdf.ebi.ac.uk/resource/ensembl/>
PREFIX ensemblterms: <http://rdf.ebi.ac.uk/terms/ensembl/>
SELECT DISTINCT ?gene_uniprot_uri ?ortholog_uniprot_uri {
   VALUES(?gene_uniprot_uri){(<http://purl.uniprot.org/uniprot/P68871>)
    }
   ?gene sio:SIO_000558 ?ortholog . # "is orthologous to"
   ?gene obo:RO_0002162 ?taxon . # "in taxon"
   ?ortholog obo:RO_0002162 ?ortholog_taxon.
   ?ortholog ensemblterms:DEPENDENT ?ortholog_uniprot_uri.
   ?gene ensemblterms:DEPENDENT ?gene_uniprot_uri.
   FILTER(?taxon != ?ortholog_taxon
            &&
   STRSTARTS(STR(?ortholog_uniprot_uri),
             "http://purl.uniprot.org/uniprot/")
   )
 }
```

LISTING 4.4: Query Q1

Ensembl dataset at the EBI RDF platform (see Figure 4.5 for the respective data schema). You can execute this query directly at the EBI SPARQL endpoint by clicking on the following link: http://purl.org/orthology/q1.

The HBB gene is represented with the UniProt URI http://purl.uniprot.org/uniprot/P68871. To retrieve the orthologs of other genes, we can replace this URI with one that corresponds to another gene, such as human INS (i.e. http://purl.uniprot.org/uniprot/P01308). We can also provide a set of URIs enclosed with parentheses such as follows: *VALUES (?gene_uniprot_uri) (<http://purl.uniprot.org/uniprot/P68871>) (<http://purl.uniprot.org/uniprot/P01308>)* . The sio:SIO_000558 is the « is orthologous to » property, while the obo:RO_0002162 represents the "in taxon" property (see Figure 4.6). We note here that not all EBI gene entries have an assigned cross-reference to UniProt. For example, "ENSG00000139618" identifies an Ensembl gene for which the UniProt cross-reference is missing from the EBI RDF platform. In this case, the previous SPARQL query can be adapted, by assigning in the VALUES statement of the query, the *?gene* variable to the corresponding Ensembl identifier, as depicted in the code fragment shown in listing 4.5, which we denote as Q2. Q2 can be executed at the EBI SPARQL endpoint by clicking on the following link: http://purl.org/orthology/q2.

This code fragment illustrates the SPARQL query to retrieve orthologs for the human BRCA2 gene from the Ensembl dataset. The BRCA2 gene is represented with the UniProt URI ensembl: ENSG00000139618 where ensembl is a prefix that replaces http://rdf.ebi.ac.uk/resource/ensembl/. To retrieve the orthologs of other genes, we can replace ensembl:ENSG00000139618 with a URI that corresponds to another gene such as human INS (i.e. ensembl:ENSG00000254647). We can also provide a set of URIs enclosed with parentheses such as follows: *VALUES(?gene) (ensembl:ENSG00000139618) (ensembl: ENSG00000254647)*.

c) **Retrieving OMA pairwise orthologs**

The code fragment in listing 4.6 shows a SPARQL query to retrieve pairwise orthologs of the human HBB gene which are derived from the HOGs in the OMA database (see Figure 4.4 for the respective data schema). The following

```
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX ensembl: <http://rdf.ebi.ac.uk/resource/ensembl/>
PREFIX ensemblterms: <http://rdf.ebi.ac.uk/terms/ensembl/>
SELECT DISTINCT ?gene ?ortholog_uniprot_uri {
  VALUES(?gene){(ensembl:ENSG00000139618)}
  ?gene sio:SIO_000558 ?ortholog.
  ?gene obo:RO_0002162 ?taxon.
  ?ortholog obo:RO_0002162 ?ortholog_taxon.
  ?ortholog ensemblterms:DEPENDENT ?ortholog_uniprot_uri.
  ?gene ensemblterms:DEPENDENT ?gene_uniprot_uri.
  FILTER(?taxon != ?ortholog_taxon
                &&
   STRSTARTS(STR(?ortholog_uniprot_uri),
             "http://purl.uniprot.org/uniprot/")
  )
 }
```

LISTING 4.5: Query Q2

```
PREFIX oma: <http://omabrowser.org/ontology/oma#>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX lscr: <http://purl.org/lscr#>
SELECT DISTINCT ?protein1 ?protein2 {
VALUES(?protein1){(<http://purl.uniprot.org/uniprot/P68871>)}
 ?cluster a orth:OrthologsCluster.
 ?cluster orth:hasHomologousMember ?node1.
 ?cluster orth:hasHomologousMember ?node2.
 ?node1 orth:hasHomologousMember* ?protein_OMA_1.
 ?node2 orth:hasHomologousMember* ?protein_OMA_2.
 ?protein_OMA_1 lscr:xrefUniprot ?protein1.
 ?protein_OMA_2 lscr:xrefUniprot ?protein2.
FILTER(?node1 != ?node2)}
```

LISTING 4.6: Query Q3

link http://purl.org/orthology/q3 is provided to directly execute the query at the OMA SPARQL endpoint webpage. We denote this query as Q3.

The HBB gene is represented in this code fragment with the UniProt URI http://purl.uniprot.org/uniprot/P68871. More precisely, in OMA the lscr:xrefUniprot represents the Cross-reference to UniProt.

d) **Retrieving MBGD pairwise orthologs**

In a similar manner to the previous code fragment, the code fragment in listing 4.7 depicts a SPARQL query to retrieve pairwise orthologs of the human HBB gene which are derived from the HOGs in the MBGD database (see Figure 4.3 for the respective data schema).

To obtain the results for this query, denoted as Q4, by using the MBGD SPARQL endpoint, we can click on the following link: http://purl.org/orthology/q4.

The HBB gene is represented again with the UniProt URI http://purl.uniprot.org/uniprot/P68871. In the case of MBGD, the mbgd:uniprot represents the cross-reference to UniProt.

```
PREFIX mbgdr: <http://mbgd.genome.ad.jp/rdf/resource/>
PREFIX mbgd: <http://purl.jp/bio/11/mbgd#>
PREFIX orth: <http://purl.org/net/orth#>
SELECT ?protein1 ?protein2
WHERE {
VALUES(?protein1){ (<http://purl.uniprot.org/uniprot/P68871>)}
 ?cluster a orth:OrthologsCluster.
 ?cluster orth:hasHomologous ?node1.
 ?cluster orth:hasHomologous ?node2.
 ?node1 orth:hasHomologous* ?gene1.
 ?node2 orth:hasHomologous* ?gene2.
 ?gene1 mbgd:uniprot ?protein1.
 ?gene2 mbgd:uniprot ?protein2.
FILTER(?node1 != ?node2)}
```

LISTING 4.7: Query Q4

Protocol 2: Retrieve homologous groups

In this protocol we illustrate the task of retrieving the non-hierarchical homologous groups of a target gene, such as the human HBB gene. In addition, we restrict the search to a specific taxonomic level, for example, "only at the primates level". In other words, we depict how to retrieve the homologous groups at a given taxonomic level and including a given gene represented as a UniProt URI. Note that the same query can be executed only providing one of the inputs (i.e. either the taxonomic level or gene). However, it can take longer to return all results or may not even be executed due to runtime constraints at the original databases. The members of a homologous group can be either paralogous or orthologous to one another.

a) **Retrieving OMA Homologous Groups derived from the HOGs**

The code fragment in listing 4.8, denoted as Q5, illustrates the SPARQL query to retrieve homologous groups (i.e. clusters) that contains the human HBB gene in the OMA database. To execute it directly at the OMA SPARQL endpoint webpage at `https://sparql.omabrowser.org/lode/sparql`, you can click on the following link: `http://purl.org/orthology/q5`.

In this code fragment, the HBB gene is represented with its related UniProt entry (i.e. the UniProt URI `http://purl.uniprot.org/uniprot/P68871`). To retrieve the clusters that have other genes, we can replace this URI with one that corresponds to another gene such as human INS (i.e. `http://purl.uniprot.org/uniprot/P01308`). We can also provide a set of URIs enclosed with parentheses such as follows:

VALUES (?protein1_uniprot_URI)

(<http://purl.uniprot.org/uniprot/P68871>) (<http://purl.uniprot.org/uniprot/P01308>).

Similarly, we can change the taxonomic level of reference as follows:

VALUES(?tax_name) ("Hominoidea").

In further details, the Q5 query retrieves a set of *(?cluster, ?protein2_OMA_URI, ?protein2_uniprot_URI, ?tax_name)* tuples. The *?cluster* variable represents the homologous group built at a taxonomic level of reference (i.e. *?tax_name*) that contains the gene represented with a given UniProt entry (e.g. P68871). The *?protein2_OMA_URI* and *?protein2_ uniprot_URI* variables are assigned the homologous genes defined as OMA and UniProt entries, respectively. These

```
PREFIX lscr: <http://purl.org/lscr#>
PREFIX orth: <http://purl.org/net/orth#>
SELECT DISTINCT ?cluster ?protein2_OMA_URI protein2_uniprot_URI ?
   tax_name {
 VALUES(?protein1_uniprot_URI)
    {(<http://purl.uniprot.org/uniprot/P68871>)}
 VALUES(?tax_name){("Primates")}
 ?cluster a orth:OrthologsCluster.
 ?cluster orth:hasHomologousMember* ?protein_OMA_1.
 ?cluster orth:hasHomologousMember* ?protein2_OMA_URI.
 ?protein_OMA_1 a orth:Protein.
 ?protein2_OMA_URI a orth:Protein.
 ?protein_OMA_1 lscr:xrefUniprot ?protein1_uniprot_URI.
 OPTIONAL{
    ?protein2_OMA_URI lscr:xrefUniprot ?protein2_uniprot_URI.
 }

 ?cluster orth:hasTaxonomicRange ?tax.
 ?tax orth:taxRange ?tax_name.

 }
```

LISTING 4.8: Query Q5

genes belong to the same homologous group (i.e. *?cluster*). Because of the fact that the SPARQL results are in a tabular form, to solely retrieve the members of a homologous group that are represented as a UniProt entry, we mainly need to project the *?protein2_uniprot_URI*. To do so, we have to replace the line containing the SELECT keyword in Q5 with the following instruction: *SELECT DISTINCT ?protein2_uniprot_URI*.

b) **MBGD Homologous Groups derived from the HOGs**

The HOGs in MBGD do not provide explicit taxonomic levels at the root level of a HOG. However, the taxon NCBI identifiers of subHOGs (i.e. sublevels) can be extracted in some cases from the cluster URI. Since this requires more advanced knowledge of SPARQL (in particular, for parsing the cluster URIs), we only make it available as part of the Extended data online.

c) **OrthoDB Homologous Groups**

The code fragment in listing 4.9 denoted as Q6 retrieves homologous groups that contains the human HBB gene identified with its corresponding UniProt entry accession number P68871. The query can be executed at the OrthoDB SPARQL endpoint webpage at `https://sparql.orthodb.org`. To inspect the results, we can execute Q6 by accessing the following link: `http://purl.org/orthology/q6`.

This SPARQL query will retrieve flat homologous groups that contains the human HBB gene in OrthoDB. More specifically, these homologous groups are indeed orthologous groups, similarly to groups in which all genes are related to each other by pairwise orthologous relations [144]. Moreover, the HBB gene is represented with its related UniProt entry (i.e. the UniProt URI `http://purl.uniprot.org/uniprot/P68871`).

In more details, the Q6 query retrieves a set of 6-tuples (?group ?species_name ?protein1_uniprot ?gene1 ?taxLevel_ uniprot ?taxLevel). The *?group* variable represents the homologous group built at a taxonomic level of reference

```
PREFIX orthodb: <http://purl.orthodb.org/>
PREFIX up: <http://purl.uniprot.org/core/>
SELECT DISTINCT ?group ?species_name ?protein1_uniprot ?gene1
?taxLevel_uniprot ?taxLevel

WHERE {
  VALUES ?protein2_uniprot {<http://purl.uniprot.org/uniprot/P68871>}
  VALUES ?taxLevel {"Primates"}
  ?gene2 a orthodb:Gene.
  ?gene2 orthodb:memberOf ?group.
  ?gene1 a orthodb:Gene.
  ?gene1 orthodb:memberOf ?group.
  ?gene1 up:organism ?organism.
  ?organism a ?taxon.
  ?taxon up:scientificName ?species_name.
  ?group orthodb:ogBuiltAt ?taxLevel_uniprot.
  ?taxLevel_uniprot up:scientificName ?taxLevel.
  ?gene2 orthodb:xref ?xref2.
  ?xref2 orthodb:xrefResource ?protein2_uniprot.
  ?protein2_uniprot a orthodb:Uniprot.
  ?gene1 orthodb:xref ?xref.
  ?xref a orthodb:Xref.
  OPTIONAL {
      ?xref orthodb:xrefResource ?protein1_uniprot.
      ?protein1_uniprot a orthodb:Uniprot.
  }
} ORDER BY ?group, ?taxLevel


```

LISTING 4.9: Query Q6

(i.e. ?taxLevel) that contains the gene represented with a given UniProt entry (e.g. P68871). The ?gene1 and ?protein1_uniprot variables are assigned the orthologous genes defined as OrthoDB and UniProt entries, respectively. These genes belong to the same homologous group (i.e. *?group*). In addition, ?species_name and ?taxLevel_uniprot variables are assigned, respectively, a species scientific name where ?gene1 is found, and ?taxLevel_uniprot is the corresponding UniProt URI of a ?taxLevel value (i.e. a taxonomic level name, e.g. "Primates"). To solely retrieve the members of an OrthoDB orthologous group that are represented as UniProt entries, we just need to project the ?protein1_ uniprot variable in the SELECT query form.

Protocol 3: Retrieve Hierarchical Orthologous Groups (HOGs)

In this protocol we show how to retrieve the HOGs containing a target gene, such as the human HBB gene, in the three orthology databases OMA, MBGD and OrthoDB. The Ensembl dataset in the EBI RDF platform is not considered because it does not provide HOG information.

a) **Retrieving HOGs from OMA**

The code fragment in listing 4.10 denoted as Q7 retrieves hierarchical orthologous groups (HOGs) that contains a gene identified with the UniProt entry accession number P68871. Q7 can be executed at the OMA SPARQL endpoint webpage at `https://sparql.omabrowser.org/lode/sparql`. The query along with its results are available at `http://purl.org/orthology/q7`.

```
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX taxon: <http://purl.uniprot.org/taxonomy/>
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX lscr: <http://purl.org/lscr#>
SELECT DISTINCT ?root_hog ?species_name ?protein1_uniprot
(?protein1 as
 ?protein1_OMA) ?taxLevel {
 VALUES ?protein2_uniprot {<http://purl.uniprot.org/uniprot/P68871>}
 ?root_hog obo:CDAO_0000148 ?hog_cluster. #has_Root
 ?hog_cluster orth:hasHomologousMember* ?node1.
 ?node1 a orth:OrthologsCluster.
 ?node1 orth:hasTaxonomicRange ?level.
 ?level orth:taxRange ?taxLevel.
 ?node1 orth:hasHomologousMember* ?protein1.
 ?hog_cluster orth:hasHomologousMember* ?protein2.
 ?protein1 a orth:Protein.
 ?protein1 orth:organism ?organism.
 ?organism obo:RO_0002162 ?taxon.
 ?taxon up:scientificName ?species_name.
 OPTIONAL {
    ?protein1 lscr:xrefUniprot ?protein1_uniprot.
 }

 ?protein2 a orth:Protein.
 ?protein2 lscr:xrefUniprot ?protein2_uniprot.

} ORDER BY ?taxLevel
```

LISTING 4.10: Query Q7

This SPARQL query will retrieve the root hierarchical orthologous group that contain the human HBB gene in the OMA database. The HBB gene is represented with its related UniProt entry (i.e. the UniProt URI `http://purl.uniprot.org/uniprot/P68871`). More specifically, the Q7 query retrieves a set of 5-tuples (?root_hog, ?species_name, ?protein1_uniprot, ?protein1_ OMA, ?taxLevel). The ?root_hog variable represents the root HOG (i.e. the deepest common ancestor for all the present species) that contains the gene represented with a given UniProt entry (e.g. P68871). The ?protein1_OMA and ?protein1_uniprot variables are assigned the genes defined as OMA and UniProt entries, respectively. These genes belong to the same root HOG (i.e. ?root_hog). In addition, ?species_name and ?taxLevel variables are assigned, respectively, the species scientific name and the taxonomic level (e.g. "Tetrapoda") where the gene is found. Moreover, the taxonomic levels implicitly represent speciation events and ancestral genes in the context of HOGs.

b) **Retrieving HOGs from MBGD**

The SPARQL query to retrieve HOGs from MBGD is similar to the previous query over OMA and therefore we make it available as Extended data online. As a reminder, although both the OMA and MBGD databases rely on different versions of the ORTH ontology, they structure their HOG data similarly.

c) **Retrieving HOGs from OrthoDB**

The code fragment in listing 4.11 denoted as Q8 retrieves hierarchical orthologous groups that contain the human HBB gene in the OrthoDB database. The HBB gene is represented with its related UniProt entry (i.e. the UniProt

```
PREFIX orthodb: <http://purl.orthodb.org/>
PREFIX up: <http://purl.uniprot.org/core/>
SELECT DISTINCT ?hog_root ?species_name ?protein1_uniprot ?gene1
?taxLevel_uniprot ?taxLevel
WHERE {
  VALUES ?protein2_uniprot   {<http://purl.uniprot.org/uniprot/P68871>}
  ?gene2 a orthodb:Gene.
  ?gene2 orthodb:memberOf ?groups.
  ?gene2 orthodb:memberOf ?hog_root.

  FILTER NOT EXISTS {?hog_root orthodb:ancestralOG ?ancestor.}

  ?groups orthodb:ancestralOG* ?hog_root.
  ?gene1 a orthodb:Gene.
  ?gene1 orthodb:memberOf ?groups.
  ?gene1 up:organism ?organism.
  ?organism a ?taxon.
  ?taxon up:scientificName ?species_name.
  ?groups orthodb:ogBuiltAt ?taxLevel_uniprot.
  ?taxLevel_uniprot up:scientificName ?taxLevel.
  ?gene2 orthodb:xref ?xref2.
  ?xref2 orthodb:xrefResource ?protein2_uniprot.
  ?protein2_uniprot a orthodb:Uniprot.
  ?gene1 orthodb:xref ?xref.
  ?xref orthodb:xrefResource ?protein1_uniprot.
  ?protein1_uniprot a orthodb:Uniprot.

} ORDER BY ?hog_root, ?taxLevel
```

LISTING 4.11: Query Q8

URI http://purl.uniprot. org/uniprot/P68871). The query can be executed at the OrthoDB SPARQL endpoint webpage at https://sparql.orthodb.org. To execute Q8 and inspect its results, we provide the following link: http://purl.org/orthology/q8. Note that unlike OMA rootHOGs where a gene can only be in one root HOG, in OrthoDB a gene can belong to multiple "root" HOGs. This is because OrthoDB might not explicitly relate the orthologous group built at the highest taxonomic level (i.e. the actual root) to lower level orthologous groups in the hierarchy. For example, for the HBB gene the Q8 query retrieves three distinct "root HOGs" with the following highest taxonomic levels: Eukaryota, Metazoa, and Vertebrata.

In further details, the Q8 query retrieves a set of 6-tuples (*?hog_root ?species_name ?protein1_uniprot ?gene1 ?taxLevel_uniprot ?taxLevel*). The ?hog_root variable represents the "root HOG" that contains the gene represented with a given UniProt entry (e.g. P68871). The ?protein1_uniprot and ?gene1 variables are assigned the genes defined as UniProt and OrthoDB entries, respectively. These genes belong to the same root HOG (i.e. ?hog_root). In addition, ?species_name and ?taxLevel variables are assigned, respectively, the species scientific name and the taxonomic level (e.g. "Eukaryota") where the gene is found. The ?taxLevel_uniprot is the corresponding UniProt URI of a ?taxLevel value.

Protocol 4: Meta-analysis - comparing OMA and MBGD data

In this protocol, we show how to compare orthology information across multiple databases with SPARQL 1.1. Although the example in the following code fragment

```
PREFIX oma: <http://omabrowser.org/ontology/oma#>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX lscr: <http://purl.org/lscr#>
PREFIX mbgd: <http://purl.jp/bio/11/mbgd#>
SELECT ?protein2 ?species WHERE {
 SERVICE<http://sparql.nibb.ac.jp/sparql> {
   # retrieve via federated query from MBGD SPARQL endpoint
   SELECT ?protein2 ?species ?uniprot_entry where {
    VALUES ?uniprot_entry  {<http://purl.uniprot.org/uniprot/K9Z723>}
    ?cluster_mbgd a orth:OrthologsCluster.
    ?cluster_mbgd orth:hasHomologous ?node1_mbgd.
    ?cluster_mbgd orth:hasHomologous ?node2_mbgd.
    ?node1_mbgd orth:hasHomologous* ?gene1.
    ?node2_mbgd orth:hasHomologous* ?gene2.
    ?gene1 mbgd:uniprot ?uniprot_entry.
    ?gene2 mbgd:uniprot ?protein2.
    ?gene2 mbgd:organism ?taxon.
    OPTIONAL {
        ?taxon mbgd:species ?species.
    }
    FILTER(?node1_mbgd != ?node2_mbgd)
   }
 }

 # keep only those that do not exist in OMA
 FILTER NOT EXISTS {
    ?cluster a orth:OrthologsCluster.
    ?cluster orth:hasHomologousMember ?node1.
    ?cluster orth:hasHomologousMember ?node2.
    ?node1 orth:hasHomologousMember* ?protein_OMA_1.
    ?node2 orth:hasHomologousMember* ?protein_OMA_2.
    ?protein_OMA_1 lscr:xrefUniprot ?uniprot_entry.
    ?protein_OMA_2 lscr:xrefUniprot ?protein2.
    FILTER(?node1 != ?node2)
  }
}
```

LISTING 4.12: Query Q9

is restricted to OMA and MBGD, similar queries over different combinations of the orthology databases mentioned in this chapter can be derived based on the Code Fragments in Protocols 1, 2 and 3.

For a given UniProt entry such as the accession number K9Z723, retrieve orthologs that are only in MBGD, but not in OMA. Alternatively, to retrieve only those that appear in both sources, simply remove the "NOT" keyword in the FILTER clause below. To execute the query in listing 4.12, denoted as Q9, at the OMA SPARQL endpoint, https://sparql.omabrowser.org/lode/sparql, we provide the following link: http://purl.org/orthology/q9.

This federated SPARQL query will retrieve pairwise orthologous genes of the Cyanobacterium-aponinum psb27 gene that are found in the MBGD database but are not present in OMA. The psb27 gene is represented with its related UniProt entry, thus the UniProt URI http://purl.uniprot.org/uniprot/K9Z723. Aggregations in SPARQL: Combining data from multiple resources By exploiting orthology databases that represent information with the same framework for data interchange (i.e. RDF) allow us to further query the data with the same query language (i.e. SPARQL). As a result, we can aggregate and combine data from multiple databases

more efficiently. This is because we avoid the needs of syntactic conversions and changes in the original data models and structures by using SPARQL and RDF. These conversions and changes are often required by traditionals methods that combines different file-based data exchange formats or full database dumps. In the Extended data online, we provide additional examples showing how to retrieve the top 10 entries with most orthologs in OMA and MBGD for a given species, e.g. 'Drosophila melanogaster'. These examples illustrate a few more advanced SPARQL features, such as aggregation and ordering by a criterion in order to select the top N results.

4.1.7 Conclusions

We provide four protocols that show how to query evolutionary relationships (pairwise orthologs, as well as HOGs) across four major databases available through SPARQL 1.1 endpoints: EBI, OMA, MBGD and OrthoDB. These protocols can serve as a good practical source for bioinformatics researchers and practioners for understanding the RDF data models of these data sources, as well as the basics of retrieving orthology information through SPARQL queries. Finally, we have shown how aggregations in SPARQL can be used to quickly generate an overview of the data available in each considered database, and how this data can be compared across the data sources.

To sum up, we hope these protocols provide a useful introduction into analysing evolutionary relationships among genes with SPARQL, as well as enriching these analyses by integrating information from external data sources, through federated queries. We have also integrated the queries in this chapter in the BioQuery search interface [6] available at `http://biosoda.expasy.org/`. Researchers to directly execute or further refine these queries in a more user-friendly environment.

We encourage readers to experiment with the example queries presented in this chapter, which are provided in the accompanying Jupyter notebook, to be directly re-used or integrated into existing research analysis pipelines.

4.2 On the correlation between gene expression evolution and branch length following duplication

4.2.1 Introduction

Phylogenetic analyses of gene expression have great potential for providing new biological insights, for example by enabling to identify genes that have evolutionary shifts in expression correlated with evolutionary changes in specific biological processes of interest [148]. Furthermore, *comparative* gene expression studies can enable a better understanding of the evolution of gene expression patterns, for example by providing insights into expression profile changes following gene duplication [149].

Most of the existing studies so far have focused on phylogenetic analyses within species. However, the authors of [148] highlight the need for new phylogenetic comparative analyses of gene expression *across* species. The authors define three significant challenges that need to be addressed for such studies to fully realize their potential:

1. Gene expression data that is comparable across species must be available. At a high-level, expression data represents the presence or absence of gene transcripts in a given tissue or organ. The challenge for comparative studies, however, is to define equivalent tissues or equivalent organs across different

species. For example, in order to compare expression levels between the human and other simpler, well-studied organisms, such as the zebrafish, a correspondence must be defined between the organs in the human body and the equivalent organs in the zebrafish. A simple such correspondence would be the equivalence between the lung in the human and the swim bladder in zebrafish. However, defining these correspondences in general is a difficult task.

2. New comparative methods must be developed, suitable for large multidimensional datasets, given that expression data is generally sparse, involving a large number of genes but only across a small number of species.

3. Comparative gene expression studies must also consider gene duplication and loss. More precisely, the article [148] points out that previous studies of gene expression across species have focused on the subset of genes that have only strict orthologs [150] [151], which simplifies analyses, but discards a large fraction of the data and therefore precludes the investigation of many phenomena of interest, such as the evolution of gene expression following gene duplication.

Here, we illustrate the potential of our data integration approach in jointly addressing all three challenges highlighted above, through a case study involving gene expression information from Bgee [11], as well as orthology information from OMA [12]. The availability of gene expression information, comparable across 29 species at the time of writing in Bgee, integrated with orthology information - more specifically Hierarchical Orthologous or Paralogous Groups [134] in OMA, offers many opportunities for new comparative expression analyses using the aggregated knowledge across the two fields of biology.

4.2.2 Background

Before describing in more detail our proposed case study, we provide here a few useful definitions for the type of information we will use in this study. First, we recall that **Hierarchical Orthologous Groups (HOGs)** are defined as sets of genes that descended from a single common ancestor within a taxonomic range of interest [134], either through speciation or through duplication. After a single-gene duplication event, the resulting sets of genes represent paralogous copies (or paralog groups) of the ancestral gene. In the following, we will refer to these as **sub-HOGs**. In our case study, we chose to focus on the paralogous sub-HOGs resulting from a duplication event after Vertebrata speciation, as illustrated in Figure 4.7. Each component gene of sub-HOGs, shown as colored dots in Figure 4.7, can originate from a different species. Therefore, HOGs aggregate genomic information across many species, making them a very useful tool in the study of evolution.

We furthermore define the **branch length** of a sub-HOG as a measure of the evolutionary distance between the sub-HOG and the ancestral gene. More precisely, a smaller branch length (such as the branch length of sub-HOG1 compared to sub-HOG2 in Figure 4.7) indicates a smaller number of substitutions in sequence with respect to the ancestral gene. This could be an indication that the member genes of this sub-HOG may have conserved the ancestral gene function.

Next, we will denote the **expression profile** of a sub-HOG to be the aggregate gene expression information over all member genes of the sub-HOG. The expression pattern of genes is considered indicative of their function. A concrete example is shown in Figure 4.8. Here, the expression profile is illustrated as a matrix, where

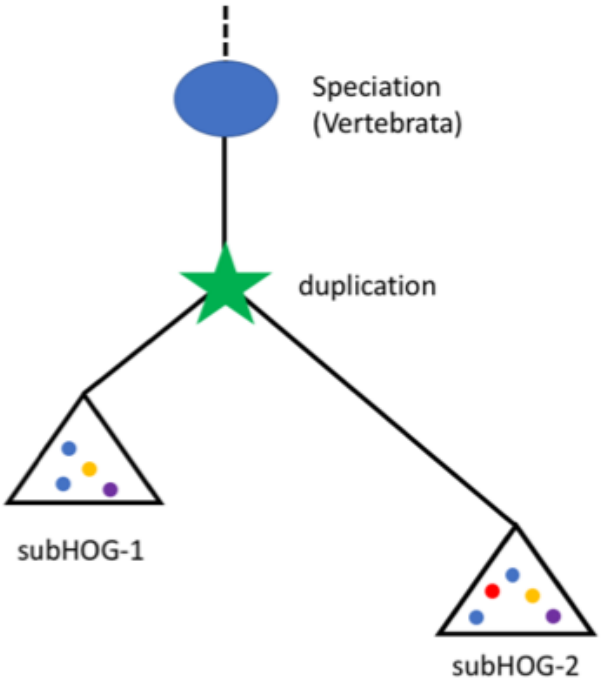

FIGURE 4.7: Schematic representation of a phylogenetic tree showing two paralogous groups (sub-HOGs) following a duplication event. In our case study, we only consider sub-HOGs which were formed after the Vertebrate speciation. Each sub-HOG consists of orthologs and paralogs (represented by color dots).

columns denote component genes of the sub-HOG, while rows denote tissues or organs, such as the lung or the liver, where these genes are expressed. An expressed gene is marked in red in the expression profile matrix in Figure 4.8. The aggregate expression profile per sub-HOG will be computed as follows: for each tissue or organ where >=50% of all component genes in the sub-HOG are expressed, we will consider an overall "expressed" value in the aggregate profile. For example, in sub-HOG1 in Figure 4.2, genes are considered overall expressed in the lung, the heart and the testes. By expression breadth we will denote the total number of tissues or organs where genes are expressed on average within a sub-HOG. For example, sub-HOG1 in Figure 4.8 has an overall expression breadth of 3 (given there are 3 organs where genes are expressed), while sub-HOG2 has an expression breadth of 2 (only 2 organs where genes are on average expressed). Finally, by expression variance we will denote the difference in expression profiles (tissues where each component gene is expressed) within a single sub-HOG.

4.2.3 Research Hypothesis

In this case study, we are primarily interested in understanding the evolution of gene expression patterns (and hence of gene function) following a duplication event. In principle, we would aim to investigate whether after a single-gene duplication one of the genes evolves rapidly, perhaps receiving a new function (a process called neofunctionalization), while the other conserves the function of the ancestral gene. However, we recall here that the only observable (measurable) gene expression profiles are those corresponding to the paralogous copies and generally *not* the ancestral gene function, which is unknown. Therefore, we cannot easily measure the amount

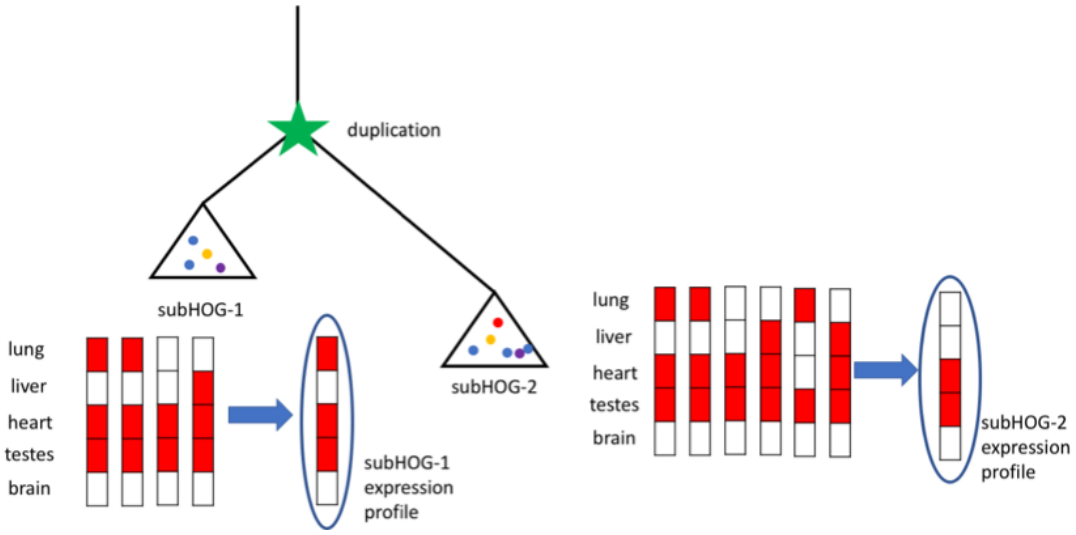

FIGURE 4.8: Expression profiles of two paralogous sub-HOGs

of change *between* the ancestral gene and any of its copies following a duplication, given that doing so would require reconstructing the ancestral gene state, which is beyond the scope of the current study (one solution is discussed in Section 4.2.6).

Instead, we investigate here the following hypothesis: does the *evolution rate* (as measured by the branch length) correlate with the amount of change in the expression profile *between pairs of paralogous copies (subHOGs)*, as measured by the pairwise difference between their expression profiles? In other terms, can we confirm, using the available data across OMA and Bgee, that more substitutions in sequence correlate with more changes in the expression profile?

For this purpose, we combine gene duplication data - in the form of paralogous HOGs, which represent gene families following a gene duplication event, from OMA - with expression data, in the form of comparable expression levels across tissues and across species, from Bgee. We focus on the paralogous HOGs resulting from a duplication following the Vertebrata speciation. In order to estimate the evolutionary distance between the vertebrate ancestral gene and each of its subsequence paralogous copies (sub-HOGs), we will integrate a new type of information, namely the branch length of the corresponding sub-HOGs. A smaller branch length indicates stronger sequence similarity between the duplicated gene and the ancestral gene, while a longer branch indicates many substitutions in sequence between the duplicate and the ancestor. We therefore expect to see a correlation between the difference in expression profiles among paralogous copies and their difference in branch lengths. Moreover, we will investigate whether a longer branch also correlates with a higher expression breadth (on average more tissues where genes are expressed within a sub-HOG).

Finally, we will also investigate the correlation between expression breadth (at the level of each sub-HOG) and evolutionary rate. Previous studies (see [152]) have shown a strong negative correlation between expression breadth and evolutionary rate. Given that we do not know the ancestral state, we will only consider the the following special case: pairs of paralogous subHOGs where one expression profile (*e.g.* in Figure 4.8, subHOG-1) is a *superset* of the other (e.g. subHOG-2). This case may be indicative of a *specialization* in the less broadly expressed copy (subHOG-2 in Figure 4.8), while the superset (*i.e.* subHOG-1, the more broadly expressed copy) might be closer in function to the ancestral gene.

```
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX genex: <http://purl.org/genex#>
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX dct: <http://purl.org/dc/terms/>
select distinct ?id ?anatName ?taxonName where {
  # OMA SPARQL endpoint
  service <https://sparql.omabrowser.org/sparql/> {
     select * where {
        <http://omabrowser.org/ontology/oma#
    GROUP_498805_Euteleostomi_3106> orth:hasHomologousMember* ?protein.
        ?protein a orth:Protein.
        ?protein a orth:Protein.
        # here is where the ENSEMBL identifier is stored in OMA
        ?protein dct:identifier ?id.
        }
     }
    # federate with Bgee SPARQL endpoint to retrieve expression pattern
    service <https://bgee.org/sparql/> {
      select * where {
        ?gene dct:identifier ?id.
        ?gene genex:isExpressedIn ?anatEntity.
        ?anatEntity rdfs:label ?anatName.
        ?gene orth:organism/obo:RO_0002162/up:scientificName ?taxonName
       }
     }
}
```

LISTING 4.13: Sample federated query for retrieving expression pattern of OMA HOG members from the Bgee database

4.2.4 Materials and Methods

Data collection

The **branch length** information was not available through the official OMA SPARQL endpoint at the time of writing and was instead provided by the Dessimoz lab as an external resource. The branch length is measured in PAM (Point Accepted Mutation) units [153], *i.e.* the average number of substitutions per site divided by 100 (in the duplicated gene sequence compared to the ancestral gene).

A local RDF copy of the relevant fraction of OMA (paralogous copies following duplication events immediately after the Vertebrata speciation), which included the branch length information, was therefore used. Unique duplication events with at least two comparable (same taxonomic level) paralogous copies (subHOGs) were used for this analysis. For each member gene of a subHOG, the expression profile was extracted from Bgee through a federated query, based on the ENSEMBL identifier of the member gene (present both in OMA and Bgee). The RDF data from Bgee was used, which provides only a binary expressed / not expressed value (we did not take into account expression *levels* in this study).

An example federated query for retrieving the **expression profile** of member genes of a given HOG (in this example the *GROUP_498805_Euteleostomi_3106*), is shown in Listing 4.13. The query can be run in any SPARQL endpoint and will retrieve a table of genes and the anatomical entities and species where these genes are expressed. The results are illustrated in Table 4.1. In order to retrieve all relevant subHOG IDs, the query in listing 4.14 can be used.

id	anatName	taxonName
"ENSACAG00000014973"	"female gonad"	"Anolis carolinensis"
"ENSACAG00000014973"	"skeletal muscle tissue"	"Anolis carolinensis"
"ENSACAG00000014973"	"kidney"	"Anolis carolinensis"
"ENSXETG00000020872"	"testis"	"Xenopus tropicalis"
"ENSXETG00000020872"	"embryo"	"Xenopus tropicalis"
"ENSXETG00000020872"	"heart"	"Xenopus tropicalis"
"ENSXETG00000020872"	"brain"	"Xenopus tropicalis"
"ENSXETG00000020872"	"female gonad"	"Xenopus tropicalis"
"ENSXETG00000020872"	"skeletal muscle tissue"	"Xenopus tropicalis"
"ENSXETG00000020872"	"gastrula"	"Xenopus tropicalis"
"ENSXETG00000020872"	"egg cell"	"Xenopus tropicalis"
"ENSXETG00000020872"	"mesonephros"	"Xenopus tropicalis"
"ENSXETG00000020872"	"blastula"	"Xenopus tropicalis"

TABLE 4.1: Sample expression pattern for a subHOG obtained by federating data from OMA and Bgee

Finally, we considered that a paralogous copy (subHOG) of a duplicated gene is overall expressed in a given tissue in the **expression profile** of the subHOG if at least 50% of the member genes of the subHOG are expressed in that tissue. We computed the pairwise **expression difference** between paralogous copies as the Hamming distance (the total number of tissues only present in one of the subHOGs in the pair of paralogous copies plus the the total number of tissues only present in the other) between the two expression profiles. For example, in Figure 4.8, the Hamming distance between subHOG1 and subHOG2 is exactly 1 (subHOG1 is overall expressed in the lung, whereas subHOG2 is not).

```
1
2  PREFIX orth: <http://purl.org/net/orth#>
3
4  SELECT distinct (?cluster as ?duplication) (?orth_cluster_1 as ?
       SubHOG_after_duplication) WHERE
5  {
6    #The tree that contains paralogs. The leafs are proteins.
7    #This graph pattern defines the relationship protein1 is paralogous
       to protein2
8    ?cluster a orth:ParalogsCluster.
9    ?cluster orth:hasHomologousMember ?orth_cluster_1.
10   ?cluster orth:hasHomologousMember ?orth_cluster_2.
11   ?orth_cluster_1 orth:hasHomologousMember* ?protein1.
12   ?orth_cluster_2 orth:hasHomologousMember* ?protein2.
13   ?protein2 a orth:Protein.
14   ?protein1 a orth:Protein.
15
16   ### duplication immediately after the Vertebrata speciation
17   ?parent_hog orth:hasHomologousMember ?cluster.
18   ?parent_hog orth:hasTaxonomicRange/orth:taxRange "Vertebrata".
19
20   filter(?orth_cluster_1 != ?orth_cluster_2 )
21 }
```

LISTING 4.14: Sample federated query for retrieving subHOGs following a duplication event after the Vertebrata speciation

The data and code for reproducing our results are available at https://github.com/anazhaw/tutorial_branch_length.

4.2.5 Results

In this section we present preliminary results obtained by performing the analyses using the available data from OMA and Bgee, as well as the external information of the branch lengths. We follow-up in Section 4.2.6 with a discussion, as well as an analysis of the limitations of this case study in Section 4.2.7, which could in part explain the results obtained and provide direction for future work.

I. **Relation between expression breadth and branch length**

In a first step, we are interested in analysing the correlation between the expression breadth versus the branch length of each paralogous copy following a duplication. The study [152] has shown that in general there is a strong negative correlation between expression breadth (EB) and branch length.

Our results from the analysis of approximately 15,000 paralogous copies (both direct lineage genes following duplication - which form a majority of the data - as well as subHOGs, only around 500 data points) do confirm a negative correlation, albeit a very weak one: Spearman correlation coefficient of -0.247. Figure 4.9 plots the two variables, namely expression breadth versus branch length of paralogous copies. *Note:* here we report results for the entire data range. Outlier removal did not significantly change the results - we report details in the accompanying Jupyter notebook.

II. **Relation between expression change and number of mutations in sequence**

Going one step further, we want to investigate if there is a correlation between the amount of change in expression observed between pairs of paralogous copies (following a duplication) and the total number of substitutions in sequence, as measured by the *sum of the branch lengths* of the two copies. For this purpose, we computed the pairwise Hamming distance between the expression profiles of paralogous copies and compared this against the sum of branch lengths per same pair. Figure 4.10 illustrates the data, namely Hamming distance between pairs of expression profiles versus the pairwise sum of branch lengths. The total number of pairs compared was approximately 231,000. The results are similar to the previous case, namely a relatively weak negative correlation: Spearman correlation coefficient of -0.228.

III. **Relation between expression change and branch length difference**
(case where one expression profile is a subset of the second)

Finally, in order to estimate if there is a correlation between the difference in evolutionary rate between pairs of paralogous copies and their difference in expression profile, we analysed only data corresponding to the case where one expression profile is the subset of the other in the pair. We assume this might indicate that one of the copies *specialized*, while the other may have *retained* the ancestral gene function. In this way, we consider the copy with the broader expression profile to be a proxy for the ancestral gene and measure the difference in expression profile compared to this reference versus the delta in evolutionary rate. Perhaps interestingly, the number of pairs compared for which this property holds is slightly above 90,000 (around 40% of the total number of pairwise comparisons).

Figure 4.11 illustrates the data. No significant correlation was found (Spearman coefficient of -0.034). However, we additionally performed here a t-test to pairwise-compare the branch lengths of the two paralogous copies, namely,

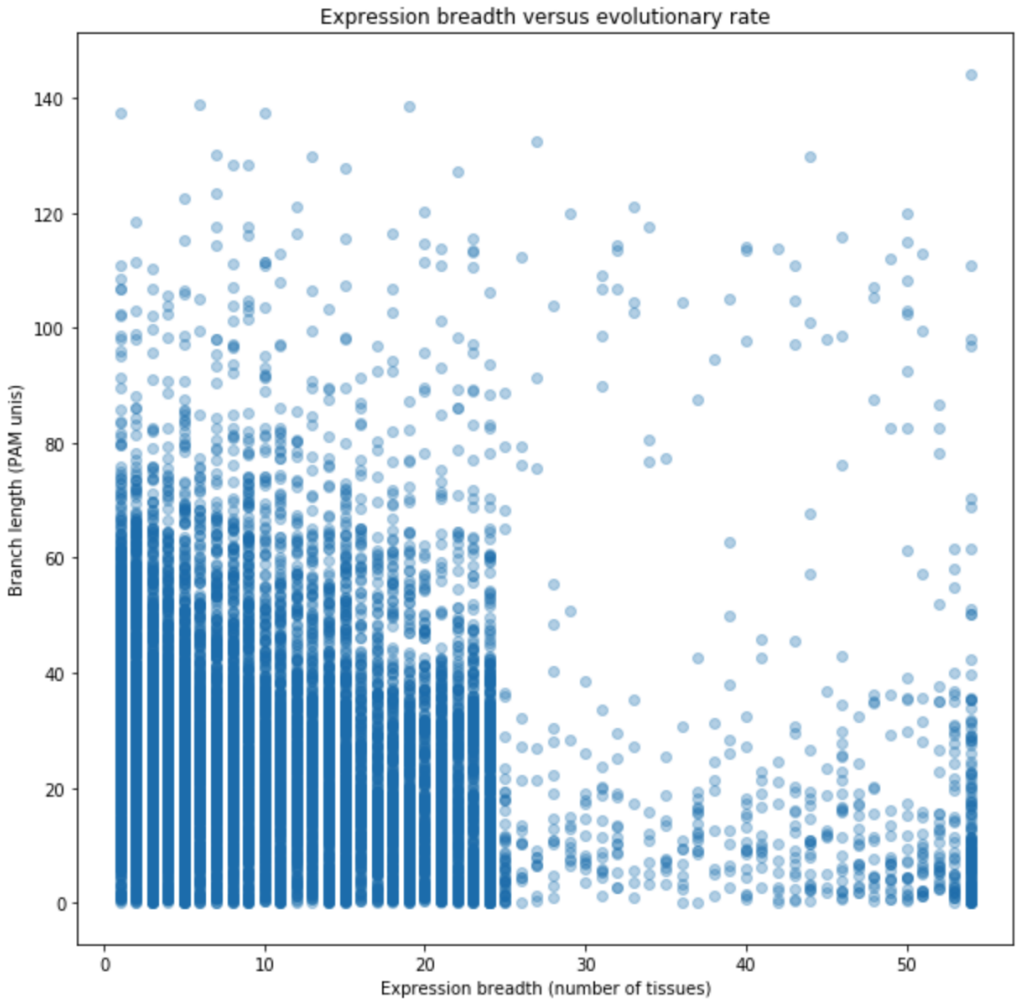

FIGURE 4.9: Expression breadth versus evolutionary rate of paralogous copies following a duplication event

the set of branch lengths corresponding to the more *broadly* expressed copy compared with the set of branch lengths corresponding to subset. Perhaps surprisingly, here we obtained a significant *positive* difference, with a *t-value* of $+21.85$; *p-value* < 0.005).

We discuss the results, while also analysing some of the limitations of our current study, in the following sections.

4.2.6 Discussion

We break down the discussion according to the three cases tested:

1. Expression Breadth versus Branch Length

   Our experiments confirmed a negative correlation between expression breadth and branch length of paralogous copies following a duplication, which can be indicative of *specialization* along the longer branches. However, previous studies [152] indicated a stronger negative correlation than our results have shown. One possible explanation for this is that we did not account for differences between the total number of tissues for which data is available in Bgee across

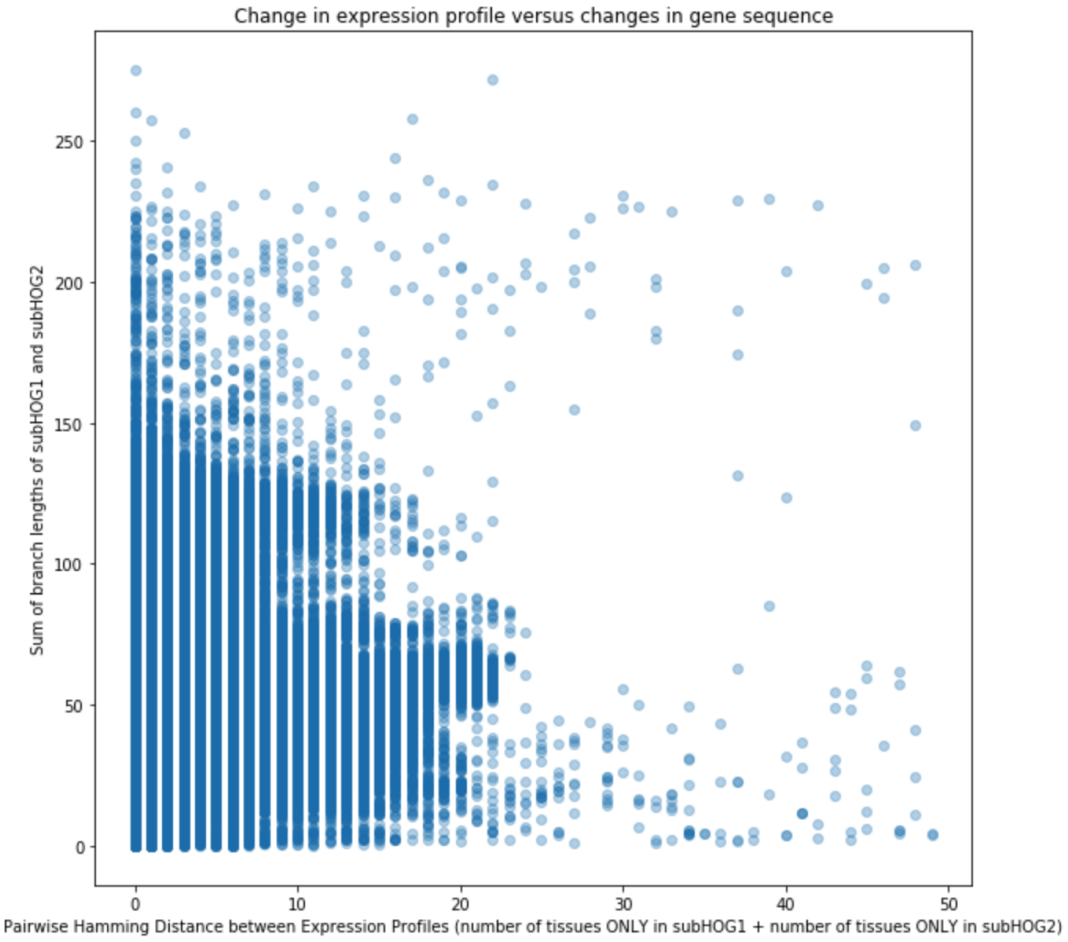

FIGURE 4.10: Pairwise Hamming distance between expression profiles versus total change in gene sequence (sum of branch lengths)

different species. Indeed, some species can have very rich expression profiles, given the availability of expression data across thousands of their tissues (this is the case for the mouse), whereas others inherently have only a small number of tissues with available data. Therefore, the expression breadth will on average be significantly larger for species such as the mouse, with no connection to the evolutionary rate following the duplication event.

One further interesting aspect to note in Figure 4.9 is that there appears to be a cut-off around the expression breadth value of 25, from which the trend seems to *reverse*. However, cases with expression breadths larger than 25 are in fact rare: only 700 data points out of the total 15,000 (< 5%). Excluding these from the analysis did not change the initial results.

2. Expression Change versus Number of Mutations in Sequence

   In the lack of the ancestral gene expression profile, we analysed here if there is a correlation between the total amount of change in expression in pairs of paralogous copies (assuming that one of the two copies is closer to the ancestral gene, while the other perhaps specialized), and the total amount of change in sequence, as measured by the sum of the branch lengths. Interestingly, the negative correlation here is similar to the first case. An important direction would

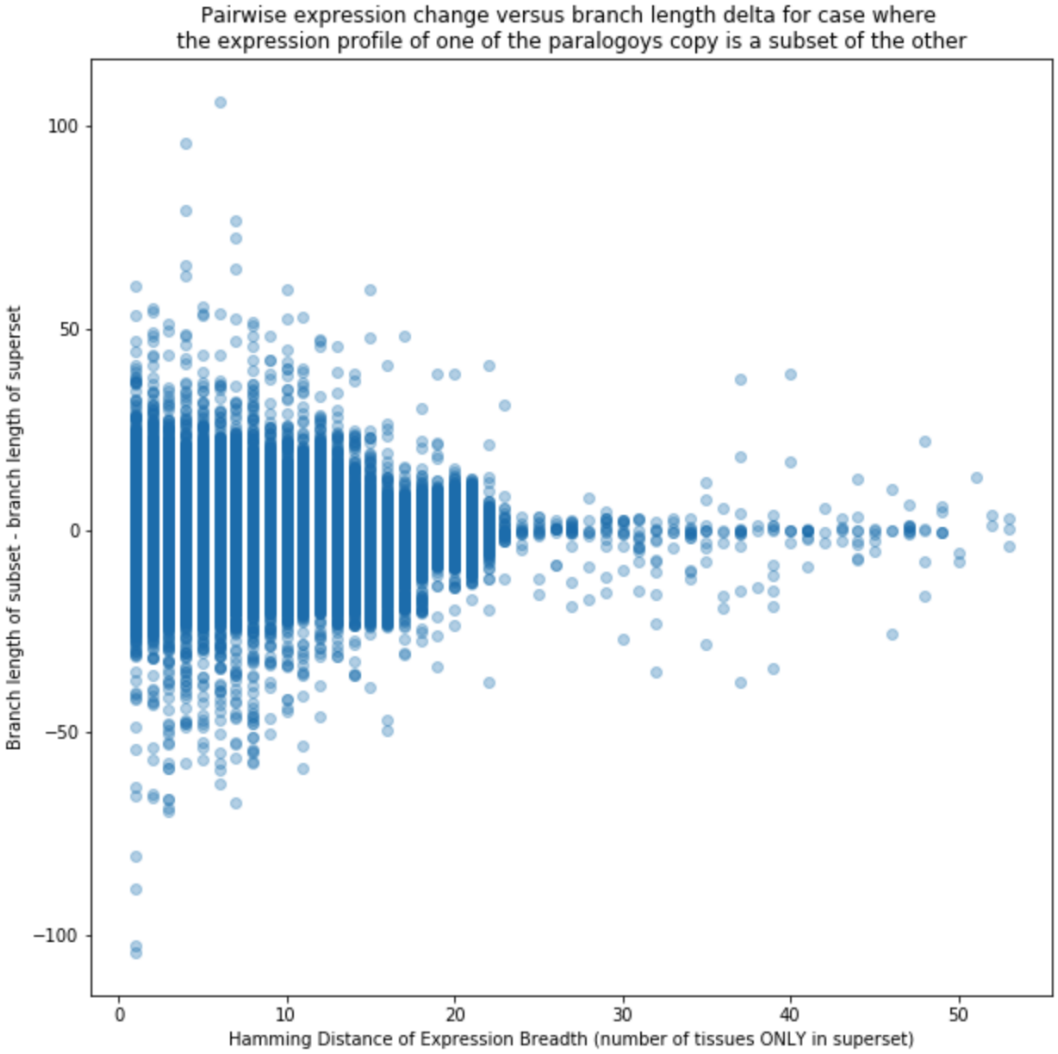

FIGURE 4.11: Pairwise Hamming distance between expression profiles (for case where one paralogous copy has an expression profile that is a subset of the other in the pair) versus delta between branch length of the subset (presumed to be closer to the ancestral gene) and the superset

be re-do the analysis after reconstructing the ancestral gene state (possible solutions for this are provided further in this section) in order to test what is the impact of computing the expression change relative to the ancestral gene, as opposed to relative to the other paralogous copy in the pair. We explain why this is an important distinction in the limitations discussion in Section 4.2.7.

3. Expression Change versus Branch Length Difference
(case where one expression profile is a subset of the second)

   The *positive* difference between the branch lengths of *broader* expression copies versus branch lengths of subsets is an unexpected result and seems to indicate that the more broadly expressed paralogous copies are on average *further* from the ancestral gene than the copies whose expression profile is a subset. From previous studies, we would expect the subset to be on a longer branch, indicative of a specialization event.

However, the difference in means between the two groups is extremely small (the mean branch length for the superset in the pair is 27.68, while for the subset it is only slightly smaller, at 26.91). We also recall here that this case covers 90,000 out of the total 230,000 pairwise comparisons, therefore it might not be representative in general. In addition, the robustness of the result can only be tested by repeating the analysis with knowledge of the ancestral gene function, as there might actually be changes in expression profile on *both* branches, as we explain in the following section.

4.2.7 Limitations

The main limitation of our study is the lack of the ancestral gene expression profile. To understand why this is an important problem, consider the scenario shown in Figure 4.12. Without knowing the ancestral state, we could consider that subHOG2 *specialized*, while subHOG1 perhaps *conserved* the ancestral gene function. However, this hypothesis might be wrong. Taking into account now the ancestral gene state, shown in the example figure on the top-right, both subHOG1 and subHOG2 are at a distance of 2 compared to the ancestral gene (subHOG2 lost the expression in liver and heart, whereas subHOG1 is now also expressed in the lung, but lost the expression in liver), therefore *none* of the two are representative of the ancestral state. One possible solution for this problem would be to consider the Amphioxus as a reference outgroup to the Vertebrata, as has been done in previous studies [154], [155], [156]. In particular, the authors of [156] used Amphioxus expression data to show that a large fraction of paralogous copies following whole-genome duplication in Vertebrata indeed have undergone *specialization* rather than *subfunctionalization*. However, at the time of writing, Bgee did not yet include Amphioxus expression information. Therefore, we could not directly perform this analysis, which remains an interesting direction for future work.

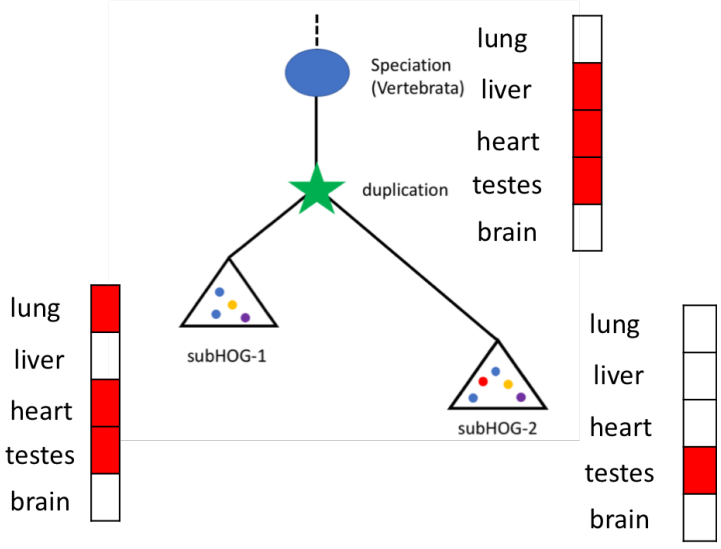

FIGURE 4.12: Example scenario where ancestral gene state is known. The expression profile of subHOG1 is a superset of the expression profile of subHOG2. Without knowing the ancestral state, we might conclude this is a case of specialization. However, in reality both subHOG1 and subHOG2 are equally distant in terms of expression profile to the ancestral gene (in this figure, 2 tissues are different).

Other problems might be related to data quality and size. First, we note that the branch length information itself might be partially inaccurate, since it has not yet been validated through more studies. For instance, a significant number of paralogous copies have a branch length of zero - these have been excluded from our analysis. One possible solution for this problem is to re-do this analysis with a secondary information source for the branch lengths, for example Selectome [157], a new release of which will be available soon. Second, we might lose a significant fraction of the duplication subHOGs by only looking at those with expression data available in Bgee. Third, the expression data itself might not be well-balanced, meaning that for some model species (*e.g. Mus musculus*) the number of well-studied genes, as well as the overall number of tissues for which expression data is available, is significantly higher than for others, which might affect the final results of this study, given that we only consider as expressed subHOGs where at least 50% of the total number of genes are expressed. Another limitation of this study is that we consider the absence of information essentially as "not expressed", which might bias the results. Future work could mitigate this problem by refining the expression profile computation to consider homologous tissues across species and removing tissues for which data is not available across *all* members of a subHOG. Currently, for subHOGs with a large number of member genes, but which might not have a lot of expression information available, this might artificially reduce ("shrink") the final expression profile breadth. However, we recall here that this only affects a small number of data points in our analysis, given that a large majority of pairwise comparison (99.7%) are of *single-gene* paralogous copies. Finally, one possibility is that the scope of the analysis is too coarse-grained: it may be that considering a binary "expressed" / "not expressed" value in constructing the expression profiles, instead of including expression *levels*, might simply result in too much information loss.

4.2.8 Conclusions and Outlook

This section introduced an applied case study for the correlation between evolutionary rate and expression profile. The main goal of this tutorial was to exemplify how the integrated data can be used for testing research hypotheses across two connected bioinformatics domains: orthology and gene expression.

Many possible improvements are left for future work. As a first step, the expression profile information could be combined with the branch lengths to try to infer information about the ancestral gene function. For this purpose, the expression profile of the ancestral gene could be computed as a weighted average of the expression profiles of paralogous copies following duplication, where the weight factor for each copy could be inversely proportional to the branch length of the paralog. This would incorporate - in the ancestral gene function inference - the intuition that a smaller branch implies higher functional similarity to the ancestral gene. Once outgroup expression data for the Vertebrata (from Amphioxus) becomes available in Bgee, it will also be possible to compare the inferred expression profile with the known expression information. Further improvements could be to include expression levels in the analysis or to narrow the study to only a set of particular tissues of interest (*e.g.* 5 representative tissues, such as brain, liver, lung *etc*) or to a more focused set of genes (for example, excluding "housekeeping" genes, which are broadly expressed). However, performing these refinements requires more in-depth domain expertise, remaining therefore outside the scope of this thesis.

All in all, the integrated information of gene expression across species from Bgee, as well as orthology information and more specifically paralogous HOGs from OMA,

combined with novel data access tools that reduce the manual effort required to benefit from this data, promise to open the path for many new insights in comparative genomics. In particular, the study of gene function evolution across species following duplications, is becoming significantly easier today due to the interoperability of the existing relevant data sources, such as Bgee an OMA. Moreover, reusing the federated queries and analyses introduced in this case study, new semantic concepts could be added in the ontologies of Bgee and OMA, such as "expression profile" and "branch length", which would also enable direct access to this information through richer data exploration tools, such as the Bio-SODA semantic search engine (see Chapter 5), in the future.

Chapter 5

Bio-SODA - A Question Answering System for Domain Knowledge Graphs

The problem of question answering over structured data has become a growing research field, both within the relational database and the Semantic Web community, with significant efforts involved in question answering over knowledge graphs (KGQA). However, many of these approaches are specifically targeted at *open-domain* question answering, and often cannot be applied directly in complex *closed-domain* settings of scientific datasets.

In this chapter, we focus on the specific challenges of question answering over *closed-domain knowledge graphs* and derive design goals for KGQA systems in this context. Moreover, we introduce our prototype implementation, Bio-SODA, a question answering system that does not require training data in the form of question-answer pairs for generating SPARQL queries over closed-domain KGs. Bio-SODA uses a generic graph-based approach for translating questions to a ranked list of candidate queries. Furthermore, we use a novel ranking algorithm that includes node centrality as a measure of relevance for candidate matches in relation to a user question. Our experiments with real-world datasets across several domains, including the last official closed-domain Question Answering over Linked Data (QALD) challenge – the QALD4 biomedical task – show that Bio-SODA outperforms generic KGQA systems available for testing in a closed-domain setting by increasing the F1-score by at least 20% across all datasets tested. Finally, our experiments with a new benchmark of complex questions developed by domain experts across two fields of bioinformatics, show that Bio-SODA can serve as a powerful tool for scientific data exploration.

5.1 Introduction

The problem of question answering over structured data has gained significant traction, both in the Semantic Web community – with a focus on answering natural language questions over RDF data stores [158–160] – and in the relational database community, where the goal is to answer questions by finding their semantically equivalent translations to SQL [54, 161, 162]. Significant research efforts have been invested in particular in *open-domain* question answering over knowledge graphs. These efforts often use the DBpedia and/or Wikidata knowledge bases that are composed of structured content from various Wikimedia projects such as Wikipedia. A growing ecosystem of tools is therefore becoming available for solving subtasks of the KGQA problem, such as entity linking [163–166] or query generation [167]. However, most

of these tools are specifically targeted at open-domain question answering, mostly over DBpedia [168], which cast doubts on their applicability to other contexts, such as for scientific datasets.

On the one hand, encouraged by the recent success of machine learning methods, several new benchmarks for training and evaluating KGQA systems have been published [169, 170]. On the other hand, not only are these datasets generally costly to construct for a *new* domain, since they involve manual curation, but also most of the existing ones are synthetic (*i.e.*, not based on real query logs). Moreover, most datasets are limited to DBpedia or Wikidata, which may not be representative of other, single-domain knowledge graphs - which we refer to as *closed-domain*.

For example, one of the major question answering datasets over DBpedia, LC-Quad [169], as well as its updated version, LC-Quad 2.0 [170], include only simple multi-fact questions that connect at most two facts. In other words, these queries cover at most two or three triple patterns, with a query graph of at most two hops. However, real-world questions may be much more complex. In particular, a study of SPARQL query logs [171] across multiple knowledge graphs, including DBpedia, has shown that a significant fraction of real-world queries have 10 triple patterns or more. It therefore remains unclear whether existing training sets can serve as representative for real-world question answering systems over knowledge graphs in general.

All in all, an important unknown still remains as to how many of the lessons learned in the open-domain question answering world can be easily applied to closed-domain datasets, such as scientific datasets. In these domains, an equivalent ecosystem of tools is not readily available. As a consequence, data access and retrieval remain challenging for domain experts who are not familiar with structured query languages. Lastly, the problem is further aggravated by the *scarcity of open-source* question answering systems that can also be tested outside of the DBpedia ecosystem.

As a step towards bridging the current gap in closed-domain question answering, we introduce Bio-SODA, a system designed to answer questions across multiple domain knowledge graphs where no prior training data are available. Bio-SODA relies on a generic graph-based approach in order to translate natural language questions into SPARQL queries. Furthermore, Bio-SODA is designed to compensate for incompleteness in the data—either due to missing schema information or, to some extent, due to missing labels. Although these situations should not occur when following ontology engineering best practices for representing data in RDF, our experience in working with real-world datasets shows that these problems are frequent in practice.

We make our prototype implementation available open-source[1]. The prototype enables both keyword search, as well as full question answering in English. We chose bioinformatics as our primary target domain, motivated by the rapid growth of publicly available RDF data in this domain. Specifically, around 8% of the Linked Open Data Cloud originates from the Life Sciences [104]. For the purpose of evaluating our system, we use several real-world datasets stemming from different domains. For example, we use the latest *closed-domain* question answering challenge released as part of the official Question Answering on Large Databases (QALD) series, namely the QALD4 biomedical task [172]. Subsequent QALD challenges have since focused on open-domain question answering.

[1]Code at `https://github.com/anazhaw/Bio-SODA`

Furthermore, we introduce a new benchmark of 30 representative question-answer pairs drafted in collaboration with domain experts from two different subfields of bioinformatics: orthology (two genes are called orthologs if they were the same in the last common ancestor) and gene expression (a measure of gene activity). The questions represent information needs related to two in-use and publicly available bio-databases, Orthologous MAtrix (OMA), a database of evolutionary relationships among genes found in different species [12], and Bgee, a gene expression database [11]. It has been shown that the number of triple patterns in the expected SPARQL answer queries is inversely proportional to the performance of question answering systems in generating these queries [173]. With an average complexity of 7 triple patterns per query, our catalog highlights that real-world questions in these domains surpass the current benchmarks in complexity and therefore require new approaches. Finally, as an application of Bio-SODA to an entirely different, non-bioinformatics domain, we use the CORDIS dataset describing European Union (EU) funded research projects[2].

This chapter makes the following contributions:

- We introduce Bio-SODA—a novel question answering system over domain knowledge graphs that *does not require prior training data* (question-answer pairs) for translating natural language questions into SPARQL.
- We define a ranking algorithm that combines *syntactic and semantic similarity*, as well as *node centrality* in the knowledge graph. Many existing question answering systems either rely on simple metrics for ranking, such as the length of the answer query graph [54], or require extensive training data in order to learn a ranking function [174]. To the best of our knowledge, our approach is the first to take into account all three factors (syntactic and semantic similarity, as well as node centrality) for ranking candidate queries.
- Our experiments on various real-world datasets show that Bio-SODA outperforms state-of-the-art KGQA systems by 20% on the F1-score using the official QALD4 biomedical benchmark and by an even higher factor on the more complex bioinformatics dataset. These differences generally stem from the higher complexity of the questions composing these datasets compared to existing benchmarks in open-domain question answering: higher number of triple patterns per query, but also extensive use of literals (numbers, strings), filters, negations, and so on. We make the benchmark targeting Bgee and OMA publicly available[3] for future research in this direction.
- Finally, we outline the challenges of *closed-domain* question answering and identify requirements for generic question answering systems over domain knowledge graphs.

The rest of this chapter is structured as follows: we introduce some of the specific challenges of closed-domain question answering in Section 5.2. We illustrate the question answering pipeline, through a concrete example from the biomedical domain, in Section 5.3. We present the system architecture of Bio-SODA in Section 5.4. Next, we describe the datasets used for the evaluation, their specific challenges and the results obtained in Section 5.5. Section 5.7 places our contribution in the context of the related work. We conclude by outlining possible directions for future work in Section 5.8.

[2]https://cordis.europa.eu/projects

[3]See Benchmarks folder in the Bio-SODA github repository

5.2 Problem Statement

In this section we introduce some of the specific challenges of closed-domain question answering, which shape the architecture of the Bio-SODA system (described in Section 5.4).

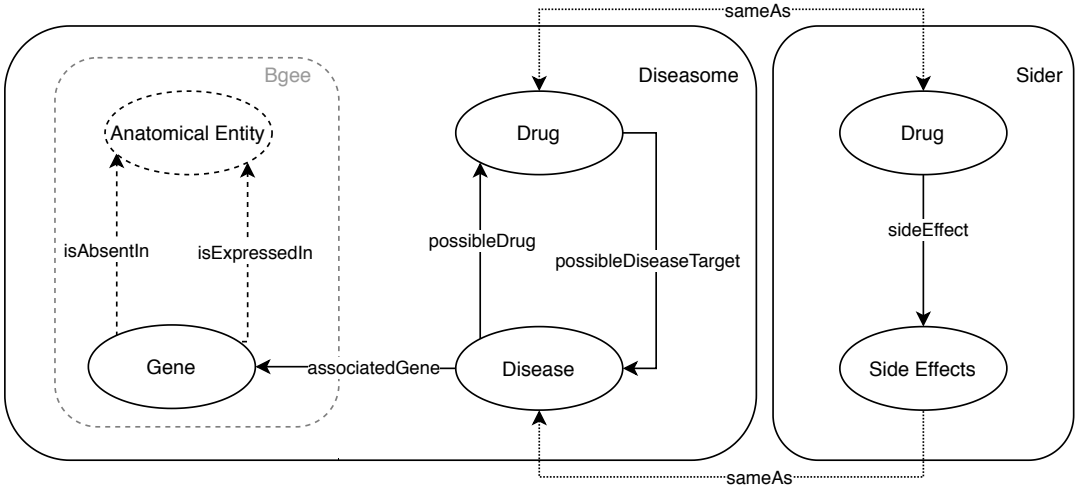

FIGURE 5.1: Simplified data model, inspired from the QALD4 Diseasome, Drugbank and Sider datasets, as well as the Bgee data model (which includes the *Anatomical Entity* class). The data model is a multigraph, including disjoint properties – such as *isAbsentIn* and *isExpressedIn*, as well as inverse properties, such as *possibleDiseaseTarget* and *possibleDrug*. The information represented by one of these properties can be derived from the another one. This is an example of redundancy at the data level that could be avoided. Furthermore, the *sameAs* property between instances of the two distinct *Drug* classes represents the join point between the two datasets. To make matters more complicated, a *Side Effect* and a *Disease* can be described by the same terms, with instances of the two classes being related via the *sameAs* property. As a result, even simple questions such as *"which drugs might lead to strokes?"* are hard to automatically translate correctly in the absence of external knowledge (*i.e.* "lead to" = "side effect" as opposed to "possible disease target").

- **Lack of training data.**

  For many of the closed-domain knowledge graphs there is no sufficiently long and diverse log of queries in order to derive a representative training set for a machine learning-based solution. So far, existing training corpora have proven costly to construct [169], with the added drawback that any semi-automatically generated dataset risks compiling a set of question-answer pairs that are non-representative for the information needs of real users of the KGQA system, in this case, domain experts.

- **Specialized terminology and vocabulary.**

  Closed-domain knowledge graphs, such as scientific datasets, often make use of specialized vocabularies.

  On the one hand, this means that data-driven methods for entity and property linking (*e.g.* N-gram-based matching, using an inverted index over the

data) are likely to perform better than “off-the-shelf" solutions. These solutions mostly revert to open-domain knowledge graphs for linking terms, such as Falcon [163] or Spotlight [165] for DBpedia. However, specialized terms cannot always be linked to DBpedia.

On the other hand, the target user is a domain expert and is therefore expected to be more familiar with the domain terminology. As a result, the lexical gap [175] and question ambiguity may be smaller problems in this context than in open-domain question answering.

- **Rule-based approaches perform well, but are costly to build and maintain.**

  So far, state-of-the-art solutions for closed-domain question answering have been mostly rule-based systems, relying on manually handcrafted rules. For example, GFMed [176] and Pomelo [177], the top 2 ranked systems in the QALD4 biomedical challenge, have achieved very good results in the challenge, but at the cost of very little generality. In essence, these systems suffer from significant overfitting: to be applicable to a new domain, their rule sets would need extensive or even complete rewriting. Moreover, even for a new dataset within the same domain, for which the schema differs, new rules need to be added in order to accommodate the differences.

  In some cases it is beneficial to incorporate a minimal set of rules in KGQA systems, particularly for deriving complex concepts. However, this should be a last resort and not the main translation mechanism, given that a large rule set is hard to maintain and scale.

- **Schema-less, incomplete data with imprecise labels.**

  Real-world datasets, particularly RDF knowledge graphs, often exhibit poor structure, for example, properties with missing or generic domains and ranges. In such cases, a question answering system should infer the schema based on instance-level data.

  An added challenge is the quality of labels. Given the manual nature of the annotation process of most RDF data, *i.e.* adding labels, descriptions and comments to data, it can be error-prone, leading to imprecise or missing information. Note that many properties in the official QALD4 biomedical dataset do *not* have any labels associated. For example, `<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/foodInteraction>` does not have any label associated in the official challenge dataset, although 4 (out of 50) questions specifically target this property. In these cases, a question answering system should try to infer definitions based on other available information, for example, Uniform Resource Identifier (URI) fragments.

  Label imprecision can also affect the performance of question answering systems, given that it hampers correct entity linking. Consider the following example: let us assume an RDF dataset with two different classes: class A, with a label “Gene Tree" and class B, with a much more verbose label, such as “This class describes genes along with their identifiers according to the information in PubMed January 2020". A search for the term “gene tree" would indeed return a single candidate match, namely class A. However, if the user would instead search for “gene", both class A and class B are potential matches. Although according to pure string similarity measures class A is a better match, it is clear that in fact class B should be returned. This shows that pure string

similarity metrics are not always sufficient for entity linking. For an extended study on the quality of labels in the web of data, see [56].

Finally, a related problem in scientific datasets and particularly prevalent in biological data, is the complexity of ontologies. These can define thousands of terms, out of which however only few are actually *used*. For example, the Gene Ontology [41], widely instantiated by bioinformatics databases, contains many terms not used at all, for example the term GO:0062165[4]. This suggests that KGQA systems using these data should take into account additional metrics for entity linking, such as the importance of a term in the knowledge graph, in order to reduce the number of potential matches for a given user term. This can be achieved, for example, by taking into account the PageRank score of the matched node in the knowledge graph.

- **Integration with external datasets.**

  Since a closed-domain knowledge graph might be an information silo, it is often beneficial to integrate it with external sources in order to obtain new knowledge. In fact, an increasing number of domain datasets are being made available in the form of knowledge graphs for this exact purpose [78, 80]. Figure 5.1 illustrates the integration of Diseasome and Sider (both part of the QALD4 biomedical task) through the *sameAs* equivalence relationship between the instances of two *Drug* classes. This enables joint queries across the datasets. A further integration could be done between Diseasome and Bgee, based on instances of the *Gene* class.

  A question answering system for closed-domain knowledge graphs should therefore also be easily extensible to new datasets, as well as updates in the existing data sources. Ideally, these updates should not require the re-indexing of all previously available data for each new release. As a result, the system should support incremental indexing.

- **High level of redundancy.**

  Although not specific to closed-domain knowledge graphs, redundancy can also be an issue in this context, even more so when multiple datasets are integrated (as is the case for the QALD4 benchmark considered in this article). Redundancy can occur both at the schema level, as well as at the instance level. For example, in Figure 5.1, the *Drug* concept is defined by two different classes, whose instances are however connected via *sameAs* predicates. The *Drug* concept is therefore the *join point* between the two integrated datasets, *Diseasome* and *Sider*. Moreover, the *Drug* and *Disease* concepts are connected through two different properties, *possibleDrug* and *possibleDiseaseTarget*, that essentially represent the same information. This results from the integration of *Diseasome* and *DrugBank* and in principle implies that one of the two properties can be inferred from another one (i.e. $(x, y) \in R \Leftrightarrow (y, x) \in R^{-1}$ where R^{-1} is the inverse relation of an R relation). Furthermore, redundancy can also occur at the instance level: in the EU projects dataset, the same organization name can be assigned to an instance of the *Organization* class, as well as to an instance of a *Participant* in a project.

[4] See https://www.ebi.ac.uk/QuickGO/term/GO:0062165

5.3 Question Answering Pipeline

In this section we introduce the question answering pipeline in Bio-SODA. For this purpose, we start from the following motivating example. Assume that a domain expert is interested in answering the question:

"*What are the drugs for diseases associated with the BRCA genes?*"

Note that, based on the biomedical literature, mutations in the two *BRCA* genes, BRCA1 and BRCA2 (stemming from *BReast CAncer*) are known to be associated with multiple types of cancer. Moreover, assume that the domain expert aims to answer the question using information from the QALD4 biomedical dataset, which integrates three different sources: *Drugbank*, a database containing information about drugs; *Sider*, a database describing the side-effects of drugs; and *Diseasome*, a database about diseases, as well as possible drugs that can be prescribed for each disease.

The question answering pipeline of Bio-SODA is illustrated in Figure 5.2. The main steps involved in the answering process are the following: first, the system retrieves, from an Inverted Index, matches for all tokens in the user question. In this example, all tokens are of length one, *i.e.* composed of a single word. The index will enable retrieving not only the URI of the candidate match, but also its PageRank score (an example is shown in parentheses for the first two tokens in the Figure), as well as the class and property names of the match (omitted in the figure due to space constraints). For example, the lookup for "*BRCA*" retrieves instances of the class *Diseasome:Genes*, where the *rdfs:label* property matches the user token ("*BRCA1*", "*BRCA2*"). A few simplified examples of Inverted Index entries are provided in Table 5.1.

In a second step, candidates are grouped together according to class and property (a `FILTER` for the token *BRCA* is created on the *Diseasome:Genes* class), as well as ranked according to string similarity and PageRank score. From here, the algorithm detailed in the following sections (see Algorithm 1) is applied to construct candidate query graphs. Due to space constraints, Figure 5.2 only shows the query graph obtained for the top ranked candidate matches. However, Bio-SODA will generate multiple alternative interpretations, for example, also including the interpretation considering *Sider:Drugs* instead of the *DrugBank:Drugs*. This can be tested in the demo page of Bio-SODA for QALD4. From the query graph, the corresponding SPARQL query is generated by also including labels for diseases and drugs. Finally, the result set shown in the bottom of Figure 5.2 is returned.

In the following section, we introduce the system architecture of Bio-SODA and describe the design of the question answering pipeline in more detail.

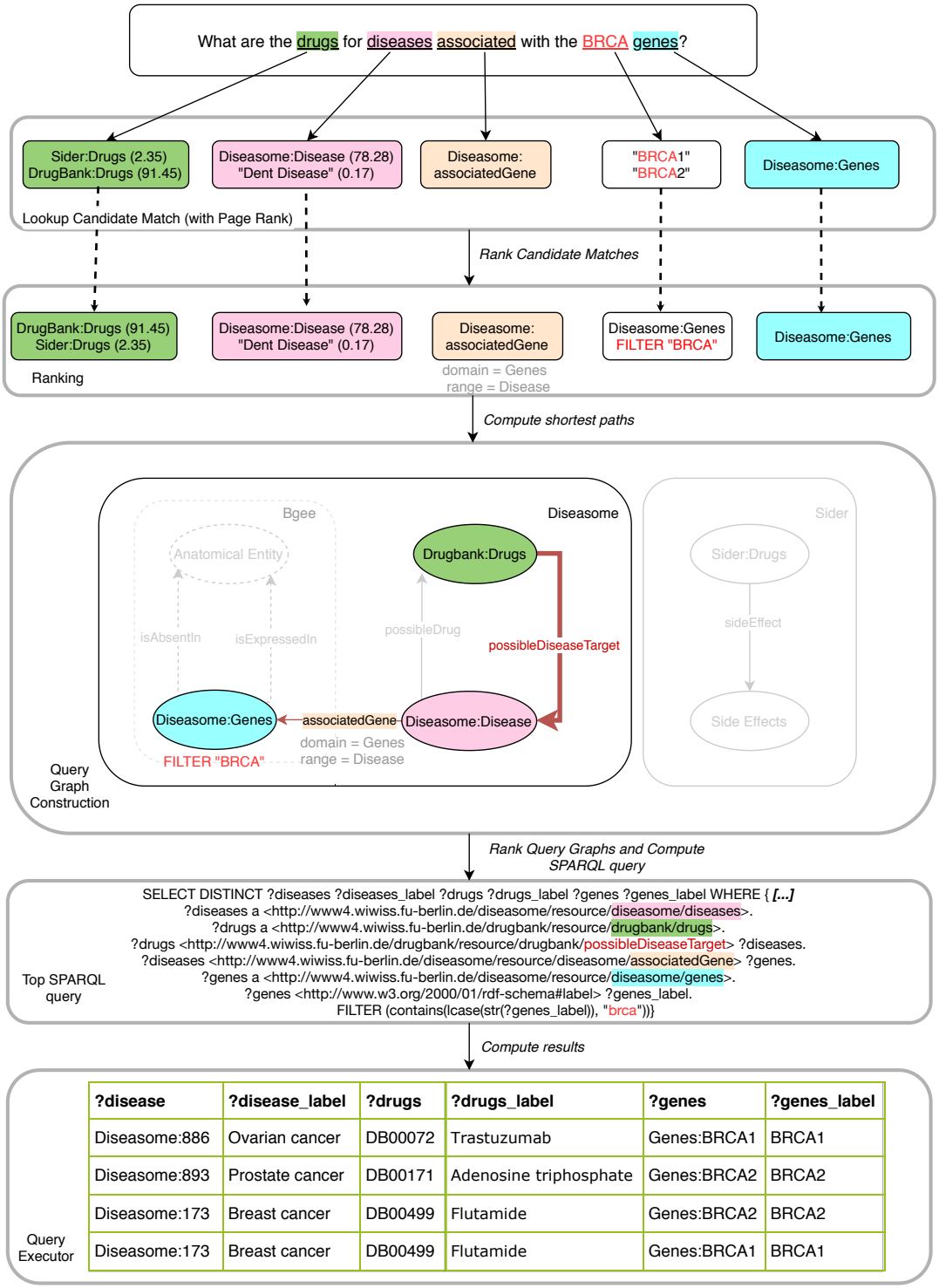

FIGURE 5.2: Simplified answer pipeline for the query *"What are the drugs for diseases associated with the BRCA genes?"*. For the sake of simplicity, PageRank scores are solely displayed when more than one match is found.

5.4 System Architecture

In this section, we describe the Bio-SODA system architecture, shown in Figure 5.3. The main building blocks of the system are the following:

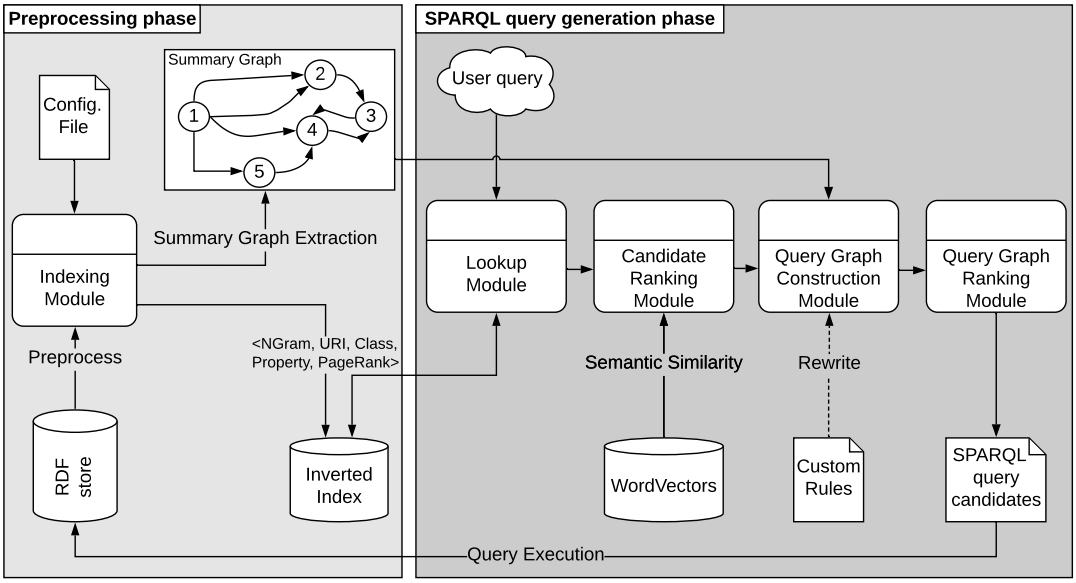

FIGURE 5.3: Bio-SODA System Architecture

1. **Preprocessing phase**:

   - *Indexing Module*—composed of:

     (a) *Inverted Index*: stores the vocabulary of the system. More precisely, all the properties that should be searchable from the RDF data store are indexed, according to a configuration file that specifies the list of properties of interest (by default, all string literals will be indexed). A further configuration option is whether URI fragments should also be parsed and indexed. In this case, these fragments are split by a predefined punctuation list, and through a camel case regex (*e.g.*, "possibleDiseaseTarget" will be indexed as the corresponding keywords "possible disease target").
     The inverted index is stored in a relational database for fast searches and it is used to match tokens (sequences of keywords in a user query) against the RDF data. More precisely, the index stores: keywords (N-grams of literals indexed), the indexed instance URI, the class of this instance, the property from which the keywords were indexed (*e.g.* label), as well as the PageRank score of the instance. PageRank scores are computed using the approach presented in [178]. We further explain the role of each indexed information item in Subsection 5.3. An example is shown in Table 5.1.

     (b) *Summary Graph Extractor*: used in order to infer the schema of the knowledge graph(s) based on instance-level data from the RDF store. The Summary Graph is essentially a schema of the integrated RDF data (note that multiple RDF sources can be combined, as long as they are semantically aligned - *i.e.* they have at least one common concept, such as *Gene*) and serves as the basis for constructing candidate query graphs from selected entry points (*i.e.*, matches for tokens in a user question). Computing a Summary Graph allows the system to compensate for incomplete schema information, for example, in cases where domains and ranges for properties are either missing or ill-defined. A second benefit of the Summary Graph is that

Lookup Key	URI	Class	Property	PageRank
stroke	side_effects:C0038454	sider:side_effects	sider:side-EffectName	0.34
drug	drugbank:drugs	owl:Class	rdfs:label	91
drug	sider:drugs	owl:Class	rdfs:label	2.3
possible disease target	diseasome:possible-DiseaseTarget	rdf:Property	uri_match	80

TABLE 5.1: Inverted Index Sample. The lookup key is used for fast searches based on keywords from a user question. The remaining information is used in attaching candidate matches to the Summary Graph (see description in Section 5.4) in order to construct the corresponding query graphs. A lookup key can consist of multiple keywords. The same lookup key can appear multiple times.

it enables joining together multiple schemas from different knowledge graphs in cases where several datasets should be integrated and queried through the system, such as for the QALD4 biomedical benchmark, composed of *DrugBank*, *Diseasome* and *Sider*.
Extracting the summary graph is achieved via SPARQL queries that compute, for example, domains and ranges of all properties, based on the classes of the instances which they connect. As a simplified example, a triple asserting "Migraine ->possibleDrug ->Ibuprofen" will result in *Disease ->possibleDrug ->Drug* being added to the Summary Graph.

We note here that indexing is a pre-processing step that is only required once, when the system is initialized. Afterwards, updates to the RDF store can be incorporated periodically through incremental updates (appends) to the inverted index, while the Summary Graph will only need to be recomputed in case of schema changes.

2. **SPARQL query generation phase**:

Given a natural language question, the goal of the Bio-SODA system is to translate it into a set of ranked candidate SPARQL queries, such that the top ranked query is the closest to the user's query intent. In the following, we detail the role of each component involved in this translation process:

- *Lookup Module*:
  The lookup module has the role of retrieving the best candidate matches for tokens identified in a user query. A token is defined by the longest sequence of keywords that matches an entry in the inverted index. For example, in the question "*What are the possible disease targets of Ibuprofen?*" the two tokens extracted will be "*possible disease target*" (corresponding to an RDF property name) and "*Ibuprofen*" (corresponding to one or more Drug instances).
- *Candidate Ranking Module*:
  The lookup module can return a large number of candidate matches per token. Due to this, the ranking module helps group together equivalent matches and rank them in order of relevance to the initial query.

More precisely, instances of the same class and with an identical property that assigns the matched values (for example, instances of the class *Drug* where the *rdfs:label* matched - a concrete case would be the *BRCA1* and *BRCA2* genes as a match for the *BRCA* keyword, see question illustrated in Figure 5.2) are grouped together into a single candidate, which in turn will be used to create a FILTER in the SPARQL query.

Furthermore, both *string similarity* and *node importance* are taken into account when ranking. Including the PageRank score as a measure of importance in the knowledge graph reduces the influence of the quality of labels assigned (labels which can be imprecise, see discussion in Section 5.2). The intuition behind this is that domain knowledge graphs usually cluster around a few important concepts, which will be reflected in the PageRank scores of the corresponding nodes. For example, UniProt[5] [179], a protein knowledge base containing more than 60 billion triples, includes only 177 classes at the time of writing. Out of these, only few classes, such as *Protein* and *Annotation*, have a central role, and will usually be the target of domain expert questions on this source. Likewise, in the case of the Cordis EU projects dataset, two different classes of Projects are available, *EC-Project* and *ERC-Project*. However, there is significantly more information in the dataset for the first class. In the lack of query logs or handcrafted rules for mapping query tokens to the correct candidates, the PageRank score can serve as a good proxy for ranking candidates according to node centrality, similarly to the initial approach used by web search engines [180].

As an added benefit, scoring with PageRank also ensures that metadata matches are prioritized. For example, *Drug* as a class name will rank higher than an instance match.

Finally, to ensure that candidate matches not only have good string similarity, but are also *semantically similar*, word embeddings are also used in the candidate ranking. The similarity comparison ensures that spurious matches, such as *gene* compared to *oogenesis*, are discarded based on a pre-defined similarity threshold in the system configuration. Any word embeddings can in principle be used with Bio-SODA. For the two main bioinformatics use cases considered in this chapter, we use Word Vectors extracted from PubMed, as described in [181]. The candidate ranking module presents to the user top N matches per query token, where N is configurable in the system.

- *Query Graph Construction Module*:
  The goal of this module is to use the matches from the previous step to generate a list of candidate query graphs. We extend the approach presented in [182] to translate matches to query graph patterns. More precisely, we apply the iterative algorithm shown in Algorithm 1: for each set of candidate matches (one match per query token), we augment the Summary Graph by attaching the candidate matches to their corresponding class. Next, we find the minimal subgraph that covers all matches. For this purpose, we solve the approximate Steiner tree problem by computing the minimal spanning tree that covers one match per token. Note that there might be multiple such subgraphs, given that two classes can be connected via multiple properties. However, unless the user can be

[5] https://sparql.uniprot.org/

Algorithm 1: Iterative graph-based approach for constructing query graphs from candidate matches

Data:
$M^{n \times t}$: the matrix of ranked candidate matches, where
n = the number of candidate matches per token,
t = the number of tokens in the user question.
M_i = a set of candidates covering one match per token (i.e. the i^{th} row vector of the $M^{n \times t}$ matrix).
G: Summary Graph of the RDF data
Result: S: the ranked set of candidate query graphs

```
1  foreach M_i ∈ M do
2      QG_i = φ (empty graph)
3      foreach candidate match T_j ∈ M_i do
4          if T_j = a RDF property then
5              Get domain D and range R of T_j from G;
6              Add D and R as vertices to QG_i;
7              Add edge T_j between D and R in QG_i;
8              if multiple domains / ranges for T_j then Create a new copy of
                 QG_i per alternative;
9          else
10             Compute in summary graph G:
11               shortest paths between class of T_j and classes of other matches T_z
                 in M_i;
12             Add shortest paths to QG_i
13             if multiple alternatives exist then
14             Create a new copy of QG_i per alternative;
15         end
16     end
17     Add spanning tree extracted from QG_i to result set S (Steiner tree
         approximation)
18 end
19 S_sorted = sort S by sum of match score of composing vertices. On a tie, sort
     by the weight (i.e. the number of edges) of spanning tree.
20 return S_sorted
```

involved in disambiguating, it is important to generate all the variants, given that two equal-length subgraphs might actually have opposite semantics. Recall the example shown in Figure 5.1, where the properties *e.g*, *isAbsentIn versus isExpressedIn* both connect the same two classes, but represent disjoint result sets.

Finally, in some cases handcrafted rules for inferring new concepts or relations are required, due to the complexity of the corresponding query graphs. In such cases translating user questions into SPARQL cannot be done via simple entity linking methods. Therefore, if needed, our approach also supports adding rules to derive implicit information from the original knowledge graph as part of the question answering pipeline. These rules are implemented as sub-queries similar to the `SELECT` SPARQL query form. In this case, the rule head is the SPARQL query projection, and the rule body is the WHERE clause content.

Relation	Query Graph
?X ortholog ?Z	prefix orth: <http://purl.org/net/orth#> ?X a orth:Protein. ?Z a orth:Protein. ?cluster a orth:OrthologsCluster. ?cluster orth:hasHomologousMember ?node1. ?cluster orth:hasHomologousMember ?node2. ?node2 orth:hasHomologousMember* ?X. ?node1 orth:hasHomologousMember* ?Z. filter(?node1 != ?node2)

TABLE 5.2: Example of a custom rule for orthology data

An example for the evolutionary relationship "ortholog", describing genes that descend from the same common ancestor gene, is shown in Table 5.2. The problem of identifying the orthologs of a gene can be considered similar to finding persons descending from a common ancestor at a certain level in an ancestry tree. The *ortholog* relation is not explicitly available as an RDF property in the OMA RDF data, but rather asserted through a complex set of property paths[6]. An extended discussion on this is available in [143]. By providing custom rules, our approach enables semantically enriching the data sources and handling complex concepts which translate to long query subgraphs.

- *Query Graph Ranking Module*:
  The query graph ranking module plays an important role in presenting the user with a meaningful, ordered list of results. In contrast to existing work, we do not return the *overall minimal subgraph* as the top result, but rather the *graph that maximizes the sum of the match scores* of the candidates covered. To understand why this is the case, consider the following question: *"what are the drugs for asthma?"*. This question translates to a 2-hop query graph, joining *Drug* and *Disease* via the *possibleDiseaseTarget* path (see Figure 5.1). However, one likely scenario is that the description of a *Drug* instance includes the keyword *asthma*. In this case, the minimal query graph would be 1-hop only, retrieving only *Drug* instances that explicitly contain the keyword in the description, probably a small subset of all instances which have the corresponding *Disease* as a possible target. In this case, the minimal result would have good precision, but very low recall.
- *Query Executor Module*:
  Finally, the query executor translates the ranked query graphs into SPARQL queries, assigning meaningful variable names, also adding human-readable fields to the result set, as shown in Table 5.6. Importantly, we do not only return the best result, but rather a ranked list of possible interpretations (top N, where N is configurable in the system). This gives the user the opportunity to inspect the results in order to use further solely the interpretation (*i.e.* SPARQL query) that matched the question intent.

[6] https://www.w3.org/TR/sparql11-query/#propertypaths

5.5 Evaluation

5.5.1 Datasets

In this section we introduce the 3 datasets considered for evaluating Bio-SODA. The characteristics of these datasets are shown in Table 5.3. Importantly, all three are real-world, in-use datasets—here we highlight particular challenges that need to be overcome for each dataset in the context of designing a generic question answering system:

1. The QALD4 biomedical dataset is composed of Sider, DrugBank and Diseasome. The particularity of this dataset is that it contains a lot of redundancy, such as multiple *Drug* classes, identical terms describing both *Disease* and *Side Effects* instances, which are connected via *owl:sameAs* properties, and so on.

2. The bioinformatics dataset is composed of the Bgee (gene expression) and OMA (orthology) RDF stores. Given the highly specialized domain information contained in these sources, a particularity of this dataset is that questions can include complex concepts which translate to long SPARQL query graphs. An added challenge deriving from this is that the same concepts can be connected through multiple equal-length paths with semantically different or even opposite meanings.

3. The CORDIS dataset of EU-funded projects. Although this dataset has a simpler schema, the challenge here is that questions can have a higher degree of ambiguity. In some cases, multiple interpretations are valid – for example, many terms are reused often and in a variety of contexts, such as "*Big Data*". This can be either part of a project title, a topic, an organization name, and so on. Therefore, identifying the query intent in some cases (*e.g. Show Big Data projects*) cannot be done without user disambiguation.

Dataset	Sources	#Classes	#Triples	Size on Disk
QALD4-biomedical	Drugbank, Diseasome, Sider	12	0.69 M	200 MB
Bioinformatics	Bgee, OMA	37	430 M	30 GB
CORDIS	EU projects dataset	26	6.5 M	1 GB

TABLE 5.3: Descriptions of the 3 public datasets used in our evaluation.

5.5.2 Queries

We have reused the official 50 queries of the QALD4 biomedical challenge[7]. We do not distinguish between training and test queries. Indeed, we report performance metrics for all systems we tested across the entire set of 50 queries. Given that the test set was also available to participants in the official challenge, we believe this to be a fair evaluation. We do not change the questions in the official challenge, not even in cases where we could identify mistakes in the question. Furthermore, as opposed to previous work using this benchmark [55], we do not materialize triples based on *owl:sameAs* statements and only use the exact dataset, as provided in the official benchmark.

[7] https://github.com/ag-sc/QALD/blob/master/4/data

For the bioinformatics dataset, we created in collaboration with domain experts, a benchmark of 30 queries, in increasing order of complexity, across two datasets, namely Bgee and OMA. The queries represent real information needs of domains experts within the field of gene expression and orthology, using the publicly available RDF data of Bgee[8] and OMA[9]. The average number of triple patterns per query here is 7 (not taking into account joint queries between the two sources, which have even higher complexity), with some questions jointly targeting 4 entities or more (*Gene*, *Species*, *Anatomical Entity*, *Developmental Stage*). In contrast, in existing benchmarks, such as LC-Quad [169], queries with only 2 entities are already considered complex.

In order to test Bio-SODA using an entirely different domain, using the CORDIS dataset of EU funded projects, we created a test set of 30 queries in increasing order of complexity. Given the relatively simple structure of this data model, the average number of triple patterns per query is close to that of existing KGQA benchmarks [169], with an average 2.3 triple patterns per query. However, the complexity stems from the usage of filters, literals in the query, as well as the higher degree of ambiguity.

Queries across the three datasets include aggregations, negations, and make extensive use of filters.

All questions, as well corresponding SPARQL queries, are available in the Evaluation folder of our GitHub repository[10].

5.5.3 Results

In this section we compare the performance of Bio-SODA against state-of-the-art systems which are open-source. Although there is a rich literature introducing question answering systems for knowledge graphs in recent years (to cite a few examples, see [183–185] etc.), only a few of them are publicly available for testing. Some provide at most a web service interface, where the target knowledge base usually cannot be changed[11].

Given these constraints, we focused on open-source systems that are publicly available for testing, in particular systems that show state-of-the-art performance according to a recent survey on natural language interfaces to databases [186]. Specifically, we tested Sparklis [129], a generic query builder system for knowledge graphs[12]. Moreover, we compared against GFMed [176] which was top ranked in the QALD4 biomedical challenge and specifically designed for this dataset. We use GFMed's publicly available grammar[13] to evaluate how the system performs outside of the official QALD4 biomedical dataset. In addition, we compared our approach against SQG [167], a system for query generation over knowledge graphs[14].

We use the standard evaluation metrics of precision (P), recall (R) and F1-score, macro-averaged over all questions in the dataset. For Bio-SODA in particular, although the system generates a ranked list of possible interpretations, we report numbers based on the top answer only (Precision@1). The results are presented in Table 5.4.

[8] https://bgee.org/sparql

[9] https://sparql.omabrowser.org/sparql

[10] Evaluation in https://github.com/anazhaw/Bio-SODA/

[11] e.g. http://wdaqua-frontend.univ-st-etienne.fr/

[12] A live demo can be tested with any SPARQL endpoint at http://www.irisa.fr/LIS/ferre/sparklis/

[13] See http://cs-gw.utcluj.ro/~anca/GFMed/index.html

[14] Available at https://github.com/AskNowQA/SQG/

System/Stats	Precision	Recall	F1
QALD4			
GFMed	1	0.99	0.99
SQG	0.42	0.42	0.42
Sparklis (5.5 steps/query)	0.88	0.88	0.88
Bio-SODA	0.61	0.60	0.60
Bioinformatics			
GFMed	0	0	0
SQG	0.16	0.16	0.16
Sparklis	-	-	-
Bio-SODA	0.6	0.6	0.6
CORDIS			
GFMed	0	0	0
SQG	0.33	0.33	0.33
Sparklis (6.2 steps/query)	1	1	1
Bio-SODA	0.66	0.66	0.66

TABLE 5.4: Evaluation results. By considering a perfect user of the Sparklis tool, the minimum number of manual steps for composing a query (averaged over all queries) is shown between parentheses.

GFMed, the highest scoring in the QALD4 challenge, cannot (nor was it intended to) be used outside this dataset without rewriting the set of grammar rules that are strictly designed for question answering over specific releases of *Diseasome*, *Drugbank* and *Sider*.

SQG on the other hand, originally evaluated on the LC-Quad [169] benchmark, does not support complex multi-hop questions, nor filters or queries involving literals. An example such query is "*Show me projects which started in **2020**?*", where *2020* is a numerical literal, as opposed to a linkable entity. While in the case of LC-Quad these limitations do not impact performance, all the three datasets considered in our evaluation include such features, which explains the poorer performance of SQG: an F-score of 0.42 in the case of QALD4, only 0.33 in the CORDIS dataset and finally 0.16 in the case of the bioinformatics dataset. We note that these results are a theoretical best, since for SQG we assume perfect entity and property linking, leading to the highest performance it can achieve.

Finally, Sparklis is not a question answering system per-se, but rather a query builder, which helps users form the correct question by composing building blocks starting from examples of class names, properties, values etc. Therefore, in order to answer questions, we needed to rephrase them from the available building blocks *manually*. On the positive side, we found Sparklis to be a powerful system, because it enables building a rich variety of query types out-of-the-box. To achieve this, only the SPARQL endpoint URL of the target RDF data store is required. Using the query building methodology of Sparklis, 44 out of 50 questions in the QALD4 biomedical benchmark can be answered. Furthermore, all questions in the CORDIS dataset can also be answered. Although this result might seem surprising, recall that the major challenge of this dataset is disambiguation. The manual query building process in Sparklis addresses exactly this problem, provided that the user knows very well how the data is structured and semantically represented. Therefore, on the negative side, we found that the query building methodology requires precise understanding of the data model, especially if multiple classes have the same label, as is the case in QALD4. For example, answering the question *Which drugs might lead to strokes?*

requires knowing that the *Drugs* class to be used is the one in *Sider*, as opposed to the one in *Diseasome*. Furthermore, formulating questions in Sparklis is a manual and therefore time-consuming process. Even when making the strong assumption that the user has perfect knowledge of the data model, as well as of the features of Sparklis (for example, how to correctly formulate aggregations, which can be challenging), the minimal number of manual steps required to formulate questions is on average 5.5 interactions per question for QALD4 and 6.2 for CORDIS, with a maximum of 10 for the more complex questions. In most cases, the question resulting from composing the building blocks will be significantly different from a true natural language question. We did not pursue this approach on the bioinformatics dataset, also because complex concepts in this dataset (ortholog, paralog) cannot be expressed through the query building mechanism. More precisely, Sparklis does not support complex property paths such as the ones previously introduced in Table 5.2.

Bio-SODA is a middle-ground between the generic, but manual approach of Sparklis, and the grammar-based approach of GFMed, which is not easily transferable to a new domain. More precisely, Bio-SODA achieves relatively good performance (around 0.6 F1 score) across the three datasets without requiring manual intervention. The only exception are two custom rules for the bioinformatics dataset, as described in Table 5.2. These help answer 4 out of 30 queries. Although GFMed has the best results for QALD4, it cannot be used outside this dataset without a complete rewriting of the grammar rules. Sparklis also achieves better results on the two datasets tested, but with the big disadvantage that it is a manual approach, where the user must understand the data model in order to compose questions correctly.

Our findings are further detailed in the Evaluation folder in our GitHub repository. We provide an analysis of the results for Bio-SODA, also highlighting limitations of the current prototype, in the following sections.

5.5.4 Impact of Ranking Algorithm

We conducted an ablation study to quantify the importance of ranking by PageRank score of candidate matches. For this purpose, we disable our ranking algorithm and instead use simple string similarity-based ranking for candidate matches, returning the *overall* minimal subgraph as the top answer.

The results, displayed in Table 5.5, show that for the CORDIS dataset in particular, ranking makes a crucial difference. The reason for this is that, for most of the keywords that describe metadata (such as class names, like *Project Topic* or *Subject Area*), there exists in the dataset a project whose acronym matches exactly. For example, there exist projects with acronyms such as *Topic*, *Area*, *Host*, *Code*, which are (according to string similarity only) classified as best matches for tokens in the original question. Constructing the *overall* minimal subgraph leads to wrong results in almost all cases, except for only 3 out 30 questions, where there is no ambiguity. Note that adding *no other change* aside from considering PageRank scores in ranking enables answering 17 more queries out of 30 for this dataset.

5.5.5 Error Analysis and Remaining Problems

In the QALD4 biomedical benchmark, Bio-SODA correctly answered 30 out of 50 questions with an additional 2 partially correct. We note that 1 question in QALD4 cannot be answered by Sparklis nor Bio-SODA due to missing label information. More precisely, the instance `<http://www4.wiwiss.fu-berlin.de/diseasome/resource/`

Dataset	(a) Correct with Bio-SODA Ranking	(b) Correct with String Similarity Ranking
QALD4	30/50	23/50
Bioinformatics	18/30	12/30
CORDIS	**20/30**	**3/30**

TABLE 5.5: Ablation study: (a) ranking with node centrality measure versus (b) traditional approach with solely string-similarity and overall minimal subgraph as top result.

genes/EDNRB> is the target of the question "Which genes are associated with Endothelin receptor type B?", however, the label *Endothelin receptor type B* is not assigned in the official dataset of the benchmark, nor can it be inferred from the URI fragment, for example. Upon closer inspection, the question is actually ill-formulated. Since *EDNRB* itself is a gene, the correct question should be "Which *diseases* are associated with EDNRB?". In total, we have found at least 4 out of 50 entries in the dataset to contain errors, either in the question formulation, or in the ground truth answer. These have already been discussed in previous studies [55].

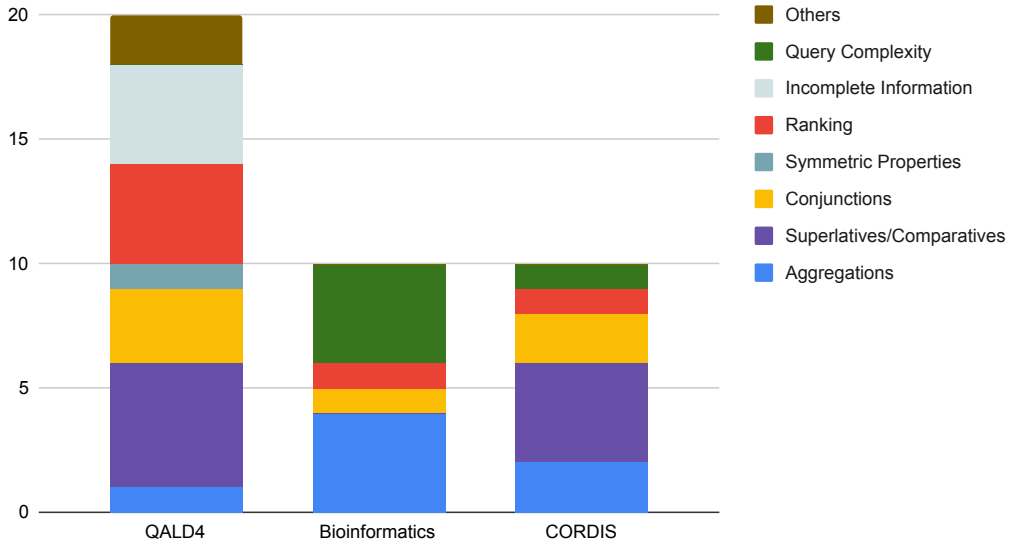

FIGURE 5.4: Bio-SODA failure analysis. Out of the total 50 questions in the QALD4 biomedical benchmark, Bio-SODA cannot correctly answer 20. A further 12 out of 30 cannot be answered in the bioinformatics dataset, mainly due to query complexity (some queries having more than 10 triple patterns). Finally, on the CORDIS dataset 10 out of 30 queries cannot be answered, a large fraction of which include features currently unsupported in Bio-SODA: aggregations, superlatives and comparatives, conjunctions etc.

An additional number of questions cannot be answered by Bio-SODA across the three datasets due to other reasons. We summarise them in Figure 5.4, explained in the following:

- *Aggregations*. Our system currently does not support questions that require aggregations, such as *Count*, *Sum* etc. An example such question would be *Count*

the projects in the life sciences domain. A possible solution to this would be to include pre-defined patterns or training a question classifier for this purpose.

- *Superlatives/Comparatives*. Another unsupported feature in the current prototype is the use of quantifiers (superlatives or comparatives). An example would be *Which drug has the highest number of side-effects?* The main reason for lack of support in this feature is that determining the property which should be quantified requires either using the dependency parse tree (in which case the questions should be grammatically correct) or using heuristics, which may be error-prone. The requirement to support keyword searches in Bio-SODA made it challenging to take into account the dependency parse tree for questions, given that the performance of dependency parsers decreases with missing words (such as keyword searches) [187]. One possible solution for this would be to only use Part of Speech (PoS) tagging to detect superlatives, considering the dependency parse tree when these are present. The direction of the superlative (*e.g. most* as opposed to *least*) would need to be detected via a classifier.

- *Conjunctions*. Conjunctive questions which involve multiple instances of the same class are not supported in the current prototype. An example such case is *List drugs that lead to strokes and arthrosis*. This limitation derives from our methodology in computing the minimal subgraph covering candidate matches, which would require special handling for cases when multiple candidates of the same class are present in a question.

- *Properties with same domain and range*. Stemming from the same limitation mentioned above, these properties are currently not supported. In QALD4, the only instance of this is the *diseaseSubtypeOf* property, which has as both domain and range the *Disease* class. In the bioinformatics dataset we handle symmetric properties describing *ortholog* and *paralog* genes through the custom rules exemplified in Table 5.2.

- *Ranking*. One of the major sources of failure in our prototype remains ranking. In the QALD4 dataset, ranking problems affect 4 out of 50 queries. An example is: *What are the diseases caused by Valdecoxib?*. Here, the system cannot correctly choose *Drug - sideEffect - Side_Effect* over the alternative *Disease - possibleDrug - Drug*. The reason for this is that the *Disease* class matches exactly the term in the question, while the *Drug* class in *Diseasome* has a higher PageRank score than the one in *Sider*. Furthermore, the combination of these two candidate matches also leads to the smallest subgraph. A possible solution to the problem would be incorporating word embeddings or synonyms in the indexing phase, or rephrasing the question. Despite this issue, the user is still able to identify the correct interpretation among the lower ranked answers, based on the explanations provided in the output.

  A more complex corner-case is part of the bioinformatics dataset, *What are the genes with lung in the description?* The term *lung* is commonly used to refer to an *Anatomical Entity*. This also reflects in the node importance of this match in the dataset. Therefore, the system cannot correctly determine that, in the context of this question, it should instead be considered part of the *description* property of a *Gene*. The correct candidate match scores very low, resulting in the correct answer also being ranked too low. A similar example from QALD4 is *Which drugs have bipolar disorder as indication?*, where *bipolar disorder* is matched

against a *Disease* instead of a drug indication. In these cases user disambiguation, at the level of candidate matches, would solve the problem.

- *Incomplete information*. This problem affects mainly the results in the QALD4 dataset, more precisely 4 out of 50 queries. We have already covered the example of the question targeting the *EDNRB* gene, which lacks the correct label in the official dataset. We currently do not enrich the inverted index with synonyms or external information, which means questions must be formulated in terms of the available vocabulary of the dataset. An additional three questions cannot be answered because they refer to URIs that do not have any class defined in the data, therefore the system cannot attach the candidate matches anywhere in the Summary Graph. An example is the *drugType* property, which can take two values, either `http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugtype/experimental` or `http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugtype/approved`. We believe a better modelling of the data should have provided, for example, either these as a *xsd:anyURI* datatype, given they are not used for any other purposes, or defined some class for both.

- *Query complexity*. The bioinformatics dataset covers queries with high complexity, which are difficult to solve especially since they include symmetric properties, with multiple instances of the same class, each filtered according to different conditions. An example of such a question is *Retrieve Oryctolagus cuniculus' proteins encoded by genes that are orthologous to Mus musculus' HBB-Y gene*. Here, the task is to retrieve *Gene* instances in a particular *Taxon* (species), namely the rabbit (*Oryctolagus cuniculus*), which are *orthologs* (symmetric property) of a second instance of *Gene*, labeled *HBB-Y*, in a different species, namely the mouse (*Mus musculus*). The resulting query has over 15 triple patterns, with 3 filters (the 2 species names plus the gene name). Bio-SODA currently cannot answer questions which involve multiple instances of the same class with additional constraints on each instance. This is particularly challenging to solve without considering the dependency parse tree. Therefore, we currently only support generic questions, such as *What are the orthologs of Mus musculus' HBB-Y gene?*.

- *Others*. Two questions in the QALD4 dataset have particular challenges, the first being a stemming error. In the question *Give me drugs in the gaseous state*, the term *gaseous* cannot be correctly stemmed to *gas*. The second type of error is due to unsupported *ASK* queries, *e.g. Are there drugs that target the Protein kinase C beta type?*. Here, Bio-SODA retrieves examples of such drugs, instead of the boolean *True*. However we do not consider this a fundamental limitation and a question type classifier could be added in future work.

We report a more detailed analysis of all systems considered in this chapter in the Evaluation folder in our GitHub repository. For easy accessibility to the system, as well as reproducibility of the results, we also provide a demo page for each of the three datasets, available online [15].

[15] Bio-SODA Demo: `http://biosoda.expasy.org/welcome/`

5.6 Lessons Learned: Design Goals for Practical Question Answering Systems over Domain Knowledge Graphs

Stemming from the challenges of closed-domain question answering, which we introduced in Section 5.2, as well as our experience in using Bio-SODA across several real-world datasets, we derived the following design goals for question answering systems over domain knowledge graphs:

- *Generality*: The system should be easily adaptable to new datasets. In particular, the system should be able to answer questions in a new domain without manual intervention and without relying on extensive training data, which is hard to obtain in many domains. In this line, a desirable property is also the ability to cope with "real-world" datasets, dealing with incompleteness in the data, in the form of:
  - missing schema information (should be inferred from instance-level data);
  - missing labels (should be incorporated from URIs whenever meaningful);
  - other issues; an example is the owl:sameAs property, which is an equivalence relationship, however, in many cases also has a defined directionality. More precisely, an RDF store can assert "A owl:sameAs B" without also asserting "B owl:sameAs A". When "owl:sameAs" inferences are not derived by Web Ontology Language (OWL) reasoners [188], this can have unexpected consequences, given that writing the wrong SPARQL query (*i.e.* wrong directionality) will lead to empty results. This is a known issue in the the QALD4 biomedical dataset [189]. One possible solution is to always consider the UNION of both directions: "A owl:sameAs B" UNION "B owl:sameAs A" in constructing SPARQL queries.
- *Extensibility*: The system should easily work with multiple datasets (provided they are already semantically aligned—*i.e.*, data integration is a prior requirement). Many studies introduce possible approaches for data integration - for the ontology-based data integration approach used as part of this work see Chapter 3.
- *Configurability*: The database owner should be able to specify properties (*e.g.* labels, descriptions) should be searchable using the system. Our experience with real-world datasets showed that in general it is not desirable for *all* properties to be indexed. As an example, in many cases, fields in the queried data sources can be either redundant or too verbose. In bioinformatics, these are abstracts of papers that are assigned as values to an RDF property, whose length can therefore be up to 300 words. Similarly, in the CORDIS dataset, these are the abstracts of the EU projects. These cases should be handled through a dedicated approach, for example, based on classical information retrieval methods (see an example in [190]).
- *Explainability*: The system should clearly guide the user through how a question was processed and interpreted. This starts from explaining which concepts were matched in relation to the original question, continuing with how these candidate matches are composed together in a query graph in order to provide the final SPARQL query. Finally, the query results should be understandable as well. Therefore, the projected variable names should be meaningful. Furthermore, labels and descriptions should be part of the result set, as

opposed to only URIs, which in most cases do not contain human-readable information. A concrete example of two possible interpretations for the question "Which drugs lead to strokes?" are shown in Table 5.6. The first interpretation given on the top is correct, but hard to validate since no labels are shown. The second interpretation is easier to understand given that the SPARQL query includes meaningful variable names and human-readable information (e.g. labels) in the result set.

1) A simplified SPARQL query could be:
Query: SELECT ?x where { ?x sider:sideEffect side_effects:C0038454.}
Results: ?x=sider_drug:163742; ?x=sider_drug:4440;

2) A more meaningful and interpretable result set would be given by:

Query:
```
SELECT * where {
?drug sider:sideEffect ?side_effect.
?drug rdfs:label ?drug_label.
?side_effect sider:sideEffectName ?name.
FILTER(contains(?name), "stroke"). }
```

Results:

?drug	**?drug_label**	?side_effect	**?name**
sider_drug:163742	"Arixtra"	side_effects:C0038454	"stroke"
sider_drug:4440	"Amerge"	side_effects:C0038454	"stroke"

TABLE 5.6: Example showing two equivalent interpretations for the question *"Which drugs lead to strokes?"*. The namespace URIs are omitted for readability and replaced with prefixes (e.g. "sider:") The first interpretation returns correct results, however, they are hard to validate by a non-expert user, given that the response only contains URIs (essentially, opaque numerical identifiers). In contrast, in the second variant, the system provides a more understandable SPARQL query, with meaningful variable names, but more importantly a more comprehensive result set. Note that the human readable name of an instance returned in the result is not necessarily always found in the *label* property (e.g. *sider:sideEffectName*).

5.7 Related Work

The problem of question answering over structured data has been well-studied in recent years, with a growing number of published systems, particularly in open-domain question answering. Recent surveys on natural language interfaces to databases include [186, 191].

In parallel, the biomedical field has seen a growth of dedicated systems for question answering. Examples include GFMed [176] and Pomelo [177] – the two highest ranked systems in the QALD4 biomedical challenge – as well as more recent systems [189]. However, these can generally be considered as expert systems, with lower generalizability to other domains, given that they extensively rely on manually handcrafted rules, which often depend on domain expertise.

Our work aims to bridge the gap between the two parallel efforts by solving the *common case* in a domain-independent manner. For this, Bio-SODA relies on a generic graph-based approach in order to generate a ranked list of candidate SPARQL

queries from a given question. We enable the addition of custom rules only for *corner cases* when needed.

Many recent KGQA systems [158, 160] have been evaluated using the LC-Quad benchmark of 5000 questions over DBpedia [169]. Although this benchmark is an important step forward, particularly for enabling machine learning approaches, it does not cover complex multi-hop questions, which makes it unclear how the results would generalize to this case. As an example, at the time of writing, the current implementation of SQG, which is publicly available [167], would not work for complex question answering on a new knowledge graph without significant changes to the code base, as it is targeted at question answering over DBpedia and more specifically in the format required by the LC-Quad benchmark. We recall here that LC-Quad includes queries with a complexity of at most two hops, which is also reflected in the implementation of SQG at the time of this writing. ExSQG, an extension of SQG, was recently proposed [192] to support richer query types. Although ExSQG supports ordinal questions (*e.g.*, superlatives), it does not support string-based filters, which Bio-SODA makes extensive use of.

More recent KGQA systems, such as [160, 193], support multiple knowledge graphs, but are limited to queries with a complexity of at most three triple patterns. Similarly, existing end-to-end QA systems, based on machine learning approaches, such as [194], can only handle simple questions. These approaches have the added drawback that they only generate a single answer, as opposed to multiple candidates. Furthermore, end-to-end approaches suffer from the problem of lack of explainability, which makes it challenging for users to validate the correctness of the result. Explainability has therefore in this context become an active area of research, with solutions proposed including translating back structured queries into natural language sentences [195–197] or summarizing the entities in the results [178].

Disambiguation is one of the major tasks of question answering systems. One possible solution for this is to limit the interface to a controlled natural language and involve the user in constructing questions from the available building blocks. Sparklis [129] is a query building system that enables answering controlled natural language questions over knowledge graphs out-of-the-box. However, this process is manual and therefore time-consuming, which makes it less convenient than a true natural language interface.

One of the systems closest to ours is the KG-agnostic WDAqua-core1 [158]. The system supports multiple knowledge bases in several languages. However, the system is only available as a demo. Although the authors mention that node relevance can in principle be taken into account in ranking, it is not clear whether the approach was used in the evaluation or whether the ranking function was learned based on training data. Furthermore, the system identifies relations from a user question based on a dictionary and does not use word embeddings for filtering out spurious candidate matches. One of the strong points of the system is its portability, which is demonstrated across 5 datasets. We have also tested Bio-SODA across 3 multi-domain datasets, achieving satisfactory results across all three benchmarks.

5.8 Conclusions and Outlook

In this chapter we have introduced Bio-SODA, a question answering system for domain knowledge graphs, which we evaluated across three real-world datasets pertaining to different domains: biomedical, orthology and gene expression, and finally

EU-funded projects. Our results have shown that Bio-SODA outperforms state-of-the-art systems that are publicly available for testing by a 20% F1-score improvement and more. The main advantage of Bio-SODA over existing open-source systems is that it can handle *complex, multi-triple pattern queries* without requiring user guidance and training data. Bio-SODA uses a novel ranking approach that takes into account both string and semantic similarity, as well as node centrality of candidate matches. Our experiments demonstrate that this improves the quality of results, particularly for datasets which can suffer from redundancy and imprecise labels.

However, making question answering systems for domain knowledge graphs work in the real-world will require a multi-way dialog between disciplines, involving the database community, the natural language processing community and domain experts.

As a first step in future work, we plan to add user feedback to the question answering process, by involving the user in a disambiguation dialog for selecting the best candidate matches, as well as for ranking the best answer among resulting candidate queries. As a long term direction for future research, we envision compiling a benchmark of cross-domain question-answer pairs, similar to the Spider benchmark in the relational database world [198], which would enable research into refining pre-trained KGQA models for new domains.

Chapter 6

Outlook

The research presented in this thesis opens many opportunities for future work. First, Machine Learning methods have the potential to simplify semantic data integration, by automating some of the underlying processes. One example is finding intersection points (Virtual Links, discussed in Chapter 3) between connected datasets. This can be achieved through Entity Matching. Recent approaches have proposed using Transformer Architectures for this step [199], which could be promising for automating Virtual Links discovery between existing datasets.

Other steps of the data integration pipeline are however less likely to be amenable to full automation in the near future. For example, ontology engineering or relational-to-RDF mappings creation (see Chapter 3), require in-depth domain knowledge, as well as agreement between domain experts. Therefore, these are likely to remain manual efforts. Still, new approaches are emerging in reducing these efforts, as they are both challenging and time-consuming tasks. For example, a recent study proposed a "pay-as-you-go" methodology for creating mappings, in order to speed-up the adoption of Ontology Based Data Access systems in practice [200]. A complementary problem, namely knowledge base creation from text, has already been tackled through machine learning methods (e.g. [201]), whose prevalence is likely to increase in this field. All in all, automatic knowledge base creation from heterogeneous sources remains crucial in enabling semantic integration of both non-structured (text) and structured data.

In Life Science research, semantic data integration has the potential to create a shift of paradigm, from the current *status quo* of tedious manual work – involving users searching by identifier lists and compiling results from multiple sources into spreadsheets by hand - to novel data exploration engines, capable of fulfilling such information needs automatically.

However, while data integration will continue to be an enabler for many new applications, it will also give rise to many new challenges. As large volumes of heterogeneous data are becoming logically unified into federated architecture, improved federated query optimizers will be required in order for response times to remain practical. Steps in this direction have already been taken by the creation of benchmarks [202], as well as practical implementations of federated query optimizers such as [203], [64]. Testing these systems with new queries and datasets, such as the catalog of bioinformatics queries over three datasets (Bgee, OMA and UniProt), evaluated in this thesis in Chapter 3, remains an interesting direction for future work.

Additional challenges arise from providing unified access points to federated data: provenance tracking, quality and relevance assessment of existing data sets (in the context of a given research hypothesis), as well as the explainability of results, all of which will play an important role in designing applications with practical impact. One of the pitfalls of making data interoperable and reusable will be its reuse in

scenarios that it was *not* originally intended for. In fact, example such scenarios are already known and discussed in the literature [204], [205], [206].

In this sense, benefiting from the *integrated data* across domains will also require *integrated domain knowledge* and careful interpretation of the novel results that span multiple domains. Therefore, interdisciplinary work, conducted in truly interdisciplinary teams, is likely to play an increasingly important role in going forward, particularly in Life Science research.

Finally, research in the field of Question Answering over Structured Data still has a long road ahead, as recent studies point out [207]. Although Machine Learning methods have brought contributions in this direction too, they have so far been applicable mostly to low-complexity queries, especially in the case of RDF datasets [208], where the focus remains on open-domain questions (for example, over DBpedia), rather than domain datasets. It is therefore not clear how they generalize, particularly in cases where there are only few training examples that a system can learn from or be fine-tuned on. One of the most promising new contributions has been given by the Spider challenge [198] of question answering over relational databases. Systems participating in this challenge, such as IRNet [209] or RatSQL [210] - although designed for question answering over relational databases - have the potential to be applicable for RDF datasets as well, due to their use of an intermediate representation. Additionally, they have the important advantage of being open-source, which was unfortunately not the case for most of the published question answering systems so far.

However, none of these systems will work without some fine-tuning to new domains, particularly in cases of ambiguity and redundancy in the data. On the other hand, the number of new training examples required for this fine-tuning has not yet been systematically analyzed (nor is it clear if such an estimation is feasible in general). It is therefore important to evaluate existing systems on unseen domains and investigate the remaining challenges that most be solved in order to make them usable in practice. Another important direction is to replace dictionary-based solutions (inverted indexes), still used to a large extent in most question answering systems, with new methods based on semantic similarity. This would significantly improve the handling of out-of-vocabulary terms. However, although Word Embeddings, such as GloVe [211], have opened the path for many new applications in natural language processing, currently finding the most semantically similar terms *from within a database* with respect to a given user term (in reasonable time) remains a complex problem. One of the only solutions proposed so far has been [212], but more research is needed - in particular, word embeddings will not solve the problem of multi-word concepts. Knowledge Graphs embeddings [213] on the other hand could be helpful in solving this for the case of KGQA systems. Their application to question answering has only recently started to be analyzed, focusing only on simple questions [214].

Ultimately, it is worth noting that no question answering system will be perfect on its own, given that two users might actually mean different things while asking *the same question*. Therefore, incorporating user feedback will also be one of the key components in improving the accuracy of question answering systems over time.

Chapter 7

Conclusions

This thesis has addressed two key challenges: first, the data integration challenge, stemming from the heterogeneity of publicly available biological databases. Here, we proposed an ontology-based data integration approach, which we illustrated on the concrete use case of an existing, in-use, gene expression database - Bgee. As part of the Bio-SODA project, Bgee was made interoperable with external datasets, such as the OMA orthology database, as well as the UniProt RDF store. As an illustration of the benefits of data integration, I have next introduced two applied case studies, which are designed to help guide domain experts into formulating expressive, federated queries within across the domains of orthology and gene expression. For example, I showed how the integrated data between Bgee and OMA can be used in order to make a comparative study of expression evolution following a duplication.

The second challenge addressed in this thesis is the data access problem, in order to bridge the semantic gap between end users and structured data, particularly knowledge graphs. This can be achieved through an intuitive search interface, that removes from users the burden of understanding the data models of the queried sourced, as well as the technical language required to query them. Here, I introduced Bio-SODA, a question answering system for domain knowledge graphs. Experimental results showed that Bio-SODA achieves good performance across three benchmarks datasets, outperforming state-of-the-art question answering systems that are publicly available for testing. Furthermore, Bio-SODA does not require prior training data, in the form of question-answer pairs (a scarce resource, particularly for domain datasets), in order to translate user questions into their equivalent technical queries in SPARQL.

The techniques presented in this thesis complement each other. Ontology-based data integration is the foundation for enabling new applications - either within the same domain (using multiple, interoperable data source to compare and contrast results) or, perhaps more importantly, across domains, where data integration can be a key enabler to new scientific discovery. The design of Bio-SODA, which I introduced in Chapter 5, is based on the assumption that the underlying data sources are already semantically aligned and interoperable. This makes the methods introduced in Chapter 3 a necessary pre-requisite. However, the practical applications that make use of the integrated data, such as the ones I introduced in Chapter 4, also *can* and *should* help guide the development of question answering systems. Ultimately, the goal of such systems is not merely to answer complex, even if at times artificial, benchmark questions, but rather to assist the main beneficiaries of the data - in particular, domain experts - in answering *meaningful, novel and insightful* questions. In this sense, measuring the utility of question answering systems in accelerating the pace of scientific discovery, in comparison with traditional data access modalities, remains an interesting open topic for future research.

Appendix A

Supplementary Materials for Chapter 3

Supplementary Materials — Enabling Semantic Queries Across Federated Bioinformatics Databases

AC Sima, T Mendes de Farias, et al.

A.1 Example that compares the number of UniProt entries between the Linked Life Data and UniProt RDF stores

The following example illustrates one of the main drawbacks of centralized data integration approaches, namely that in the lack of a good strategy for keeping data in-sync, results can quickly become stale. As an illustration, we can compare the total number of protein entries in the Linked Life Data (LLD) SPARQL endpoint ([103] mentioned in the Introduction section in Chapter 3), with the count according to the latest version of UniProt.

Executing the SPARQL query in the version of UniProt at the time of writing (release 2019_03) will result in a total of around 238 million proteins, whereas in Linked Life Data there are only around 20 million, indicating that the entries in LLD are severely outdated, missing out more than 90% of the total entries in the latest UniProt version. We provide below the required links in order to reproduce these observations:

Query to retrieve the total number of proteins:

```
PREFIX up:<http://purl.uniprot.org/core/>
SELECT (count(?protein) as ?count_uniprot_entries )
WHERE
{ ?protein a up:Protein .}
```

Executing the query at Linked Life Data SPARQL endpoint:
http://www.linkedlifedata.com/sparql

To get the list of results in your browser directly, use the following link:
http://www.linkedlifedata.com/sparql?query=PREFIX+up%3A%3Chttp%3A%2F%2Fpurl.uniprot.org%2Fcore%2F%3E+%0D%0ASELECT+%28count%28%3Fprotein%29+as+%3Fcount_uniprot_entries+%29%0D%0AWHERE%0D%0A%7B%0D%0A%09%3Fprotein+a+up%3AProtein+.%09%0D%0A%7D&_implicit=false&implicit=true&_form=%2Fsparql

Executing the query at UniProt SPARQL endpoint:
https://sparql.uniprot.org/sparql/

To get the list of results in your browser directly, use the following link:
https://sparql.uniprot.org/sparql/?format=html&query=PREFIX+up%3A%3Chttp%3A%2F%2Fpurl.uniprot.org%2Fcore%2F%3E+%0D%0ASELECT+%28count%28%3Fprotein%29+as+%3Fcount_uniprot_entries+%29%0D%0AWHERE%0D%0A%7B%0D%0A%09%3Fprotein+a+up%3AProtein+.%09%0D%0A%7D

A.2 Example Bgee-Ontop relational-to-RDF mapping

The code fragment in Listing A.1 defines how rows (i.e. a species) and columns (i.e. species attributes) of the *species* table in the relational database can be mapped as RDF triples, by instantiating a corresponding GenEx class, namely the *up:Taxon* class (imported from the UniProt core ontology), as well as its individuals. The mapping also addresses *schema-level heterogeneity* by concatenating two source column values (i.e. *genus* and *species*) into the target *up:scientificName* RDF property.

```
target   taxon:{speciesId} a up:Taxon ;
             up:scientificName {speciesSciName} ;
              up:rank  up:Species ;
              up:commonName {speciesCommonName}.
              orth:Organism/{speciesId} a orth:Organism ;
              obo:RO_0002162 taxon:{speciesId} .
source   SELECT speciesId, speciesCommonName,
              CONCAT(genus, ' ', species) AS speciesSciName
         FROM species
```

LISTING A.1: Ontop direct mapping to infer Species related data into the GenEx semantic model (i.e. target schema) based on the Bgee relational database (i.e. data source). Prefixes are defined in Table A.1.

TABLE A.1: In this thesis, we assume the namespace prefix bindings in this table.

Prefix	Namespace Internationalized Resource Identifier IRI
rdfs:	http://www.w3.org/2000/01/rdf-schema#
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#
orth:	http://purl.org/net/orth#
up:	http://purl.uniprot.org/core/
taxon:	http://purl.uniprot.org/taxonomy/
genex:	http://purl.org/genex#
obo:, uberon:	http://purl.obolibrary.org/obo/
oma:	http://omabrowser.org/ontology/oma#
skos:	http://www.w3.org/2004/02/skos/core#
sio:	http://semanticscience.org/resource/
lscr:	http://purl.org/lscr#
void:	http://rdfs.org/ns/void#
voidext:	http://purl.org/query/voidext#

A.3 Example SPARQL federated query across Bgee, OMA, UniProt

What are the human genes which have a known association to glioblastoma (a type of brain cancer) and which furthermore have an orthologous gene expressed in the rat brain?

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX obo: <http://purl.obolibrary.org/obo/>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX taxon: <http://purl.uniprot.org/taxonomy/>
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX lscr: <http://purl.org/lscr#>
PREFIX genex:<http://purl.org/genex#>

SELECT DISTINCT ?protein ?orthologous_protein_rat ?id WHERE {
 SELECT * {
  SERVICE <http://sparql.uniprot.org/sparql> {
    SELECT ?protein WHERE {
      ?protein a up:Protein ;
               up:organism taxon:9606 ;
               up:annotation ?annotation .
      ?annotation rdfs:comment ?annotation_text .
      ?annotation a up:Disease_Annotation .
    FILTER CONTAINS (?annotation_text, "glioblastoma") } }
  SERVICE <https://sparql.omabrowser.org/sparql> {
    SELECT ?orthologous_protein_rat ?protein ?id WHERE {
      ?protein_OMA a orth:Protein .
      ?orthologous_protein_rat a orth:Protein .
      ?cluster a orth:OrthologsCluster .
      ?cluster orth:hasHomologousMember ?node1 .
      ?cluster orth:hasHomologousMember ?node2 .
      ?node2 orth:hasHomologousMember* ?protein_OMA .
      ?node1 orth:hasHomologousMember* ?orthologous_protein_rat .
      ?orthologous_protein_rat orth:organism/obo:RO_0002162 taxon
    :10116. #rattus norvegicus
      ?orthologous_protein_rat sio:SIO_010079/lscr:xrefEnsemblGene ?id.
      ?protein_OMA lscr:xrefUniprot ?protein .
      FILTER(?node1 != ?node2) } }
  SERVICE <http://biosoda.expasy.org:8080/rdf4j-server/repositories/
    bgeelight> {
    ?gene genex:isExpressedIn ?a .
    ?a rdfs:label "brain" .
    ?gene orth:organism ?s .
    ?s obo:RO_0002162 taxon:10116.
    ?gene lscr:xrefEnsemblGene ?id . } } }
```

LISTING A.2: A federated SPARQL 1.1 query to retrieve proteins associated with glioblastoma and the orthologs expressed in the rat brain.

Table A.2 displays the results of executing the SPARQL query above, where:

- The first column, "protein", shows UniProt human proteins with a known association with glioblastoma for which there exists an orthologous protein expressed in the rat's brain. Clicking on any of the links in this column will redirect to the corresponding UniProt entry online.
- The second column, "orthologous_protein_rat", shows the orthologous rat protein (for which there exists known expression in the brain according to data from Bgee)

- The third column, "id", shows the Ensembl ID of the gene encoded by the rat protein (from column 2). Note that the ensemble ID (e.g. ENSRNOG00000008839) can be used in the Bgee search interface at `https://bgee.org/` for validating the results.

The complete list of federated queries is available in our template-based search interface at `http://biosoda.expasy.org`, while an explanation for the complexity of the queries is available in our github repository at `https://github.com/biosoda/bioquery/`.

A.4 Example relational-to-RDF OBDA mappings and discussion

Figure A.1 – provided here as a copy of Figure 3.2 in chapter 3, for readability purposes – illustrates graphically an example of exposing relational data from Bgee (shown in the bottom half of the figure, the relational model) in a virtual RDF graph (a fraction of the ontology and 2 instances are shown in the upper half of the figure, the RDF model).

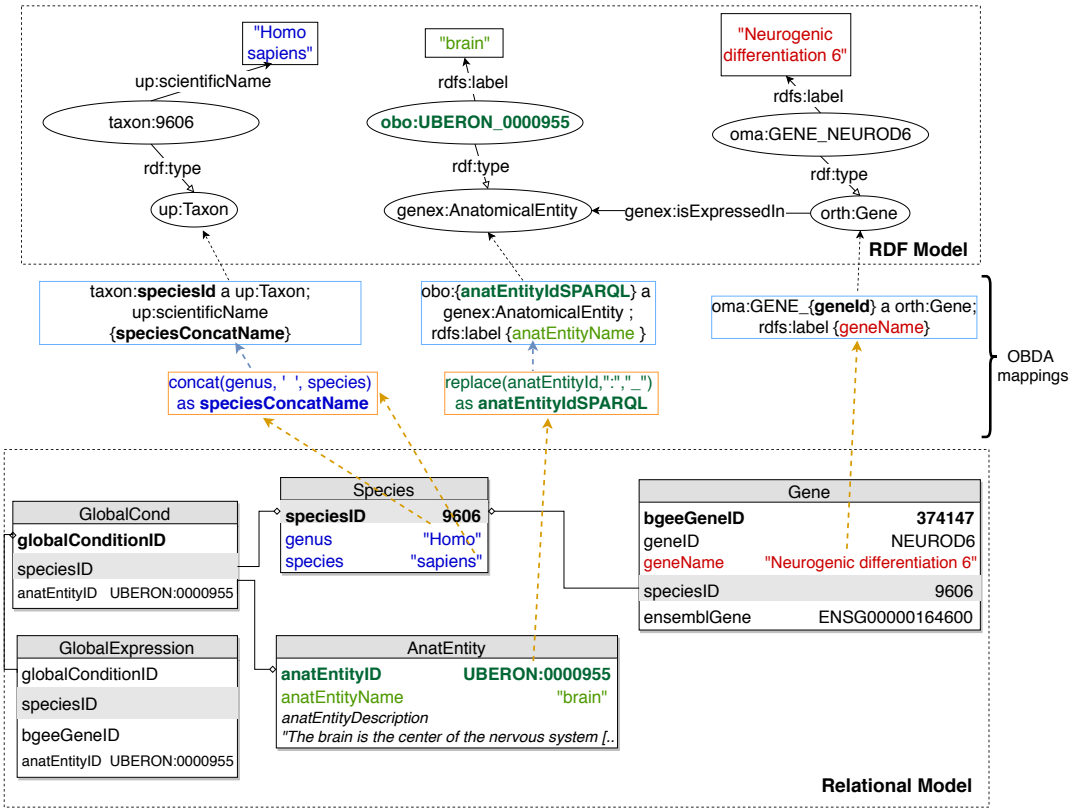

FIGURE A.1: An illustration of relational-to-RDF mappings on a sample of the Bgee database (a copy of Figure 3.2 in Chapter 3, provided here for readability). These mappings address both *schema-level* heterogeneity (an example is shown in blue), as well as *data-level* heterogeneity (shown in green). A mapping can also be a simple 1-to-1 correspondence between a relational attribute (e.g. *geneName*, shown in red) and its equivalent RDF property (in this case, an *rdfs:label* of an *orth:Gene* instance). Namespace prefixes are defined in Supplementary Table A.1.

The orange rectangles in the OBDA mappings layer (in the center of the figure) represent the "source" fragments (a simplified SQL statement) of a relational-to-RDF mapping, while the blue rectangles illustrate the "target" (resulting RDF triples). For readability purposes, we only included 3 sample mappings in this figure. However, the full set of OBDA mappings employed with the Bgee relational data is available in our github repository at `https://github.com/biosoda/bioquery/tree/master/Bgee_OBDA_mappings`.

We note here that more complex mappings OBDA mappings can require joining multiple tables in order to express an ontological concept or property which does not have a direct correspondent (i.e., is *not* explicitly present) in the original database.

TABLE A.2: Results of federated SPARQL query joining Bgee, OMA and UniProt.

protein	orthologous_protein_rat	id
http://purl.uniprot.org/uniprot/P37231	https://omabrowser.org/oma/info/RATNO15188	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000008839
http://purl.uniprot.org/uniprot/P08922	https://omabrowser.org/oma/info/RATNO12308	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000000406
http://purl.uniprot.org/uniprot/P68431	https://omabrowser.org/oma/info/RATNO09038	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000053155
http://purl.uniprot.org/uniprot/P68431	https://omabrowser.org/oma/info/RATNO09042	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000056281
http://purl.uniprot.org/uniprot/P68431	http://omabrowser.org/oma/info/RATNO11352	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000060366
http://purl.uniprot.org/uniprot/Q14956	https://omabrowser.org/oma/info/RATNO14717	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000008816
http://purl.uniprot.org/uniprot/P84243	https://omabrowser.org/oma/info/RATNO18582	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000032401
http://purl.uniprot.org/uniprot/P84243	https://omabrowser.org/oma/info/RATNO06508	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000003220
http://purl.uniprot.org/uniprot/O75140	https://omabrowser.org/oma/info/RATNO07117	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000018144
http://purl.uniprot.org/uniprot/Q12980	https://omabrowser.org/oma/info/RATNO03263	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000020541
http://purl.uniprot.org/uniprot/Q8WTW4	https://omabrowser.org/oma/info/RATNO20234	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000021660
http://purl.uniprot.org/uniprot/Q9BZH6	https://omabrowser.org/oma/info/RATNO02052	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000020430
http://purl.uniprot.org/uniprot/Q9HD26	https://omabrowser.org/oma/info/RATNO12311	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000000408
http://purl.uniprot.org/uniprot/Q9Y243	https://omabrowser.org/oma/info/RATNO06482	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000021497
http://purl.uniprot.org/uniprot/Q9UM73	https://omabrowser.org/oma/info/RATNO17047	http://rdf.ebi.ac.uk/resource/ensembl/ENSRNOG00000008683

For this reason, mappings such as the one in 3.1 from Chapter 3 (shown here in Listing A.3 below) can be interpreted as a *semantic enrichment* of the Bgee original data schema, because they provide an explicit representation for knowledge that is not directly present in the base data. In addition, all RDF triples are indeed virtual, not materialised, hence the data are not duplicated.

```
target oma:GENE_{geneId} genex:isExpressedIn uberon:{anatEntityIdSPARQL
    } .
source  SELECT g.geneId,
       REPLACE(gc.anatEntityId,":","_") AS anatEntityIdSPARQL
       FROM globalExpression AS ge
       JOIN globalCond AS gc
       ON ge.globalConditionId = gc.globalConditionId
       JOIN gene AS g ON g.bgeeGeneId = ge.bgeeGeneId
```

LISTING A.3: Ontop mapping to infer the "is expressed in" GenEx relation (i.e. target schema) based on the Bgee relational database (i.e. data source). Prefixes are defined in Supplementary Table A.1.

In more detail, the code fragment in Listing A.3 asserts the *genex:isExpressedIn* property by relating the projected columns *geneId* and *anatEntityIdSPARQL* from the join between the *globalExpression*, *globalCond* and *gene* tables in the Bgee database. This mapping also addresses *data-level heterogeneity* by applying the SQL REPLACE() function in order to transform the *anatEntityId* attribute (represented in Bgee with the separator ":") into the corresponding standard UBERON IRI (where the "_" separator is used). This transformation is also graphically illustrated in Figure A.1 (in green).

Nonetheless, the OBDA solution with Ontop has some limitations. In particular, Ontop may struggle to apply complex and numerous mappings. In principle, the advantage of Ontop is that, by translating SPARQL queries into SQL, the system can take full advantage of the SQL query optimizer provided by the underlying relational database management system (RDBMS). However, when the relational-to-RDF mappings are very complex, the translation of a given SPARQL query can result in an extremely complex SQL equivalent that can possibly imply an overall poor performance of the system (Ontop + RDBMS) [126]. Moreover, currently not all SPARQL queries can be translated into SQL. For example, aggregation queries (e.g. SUM, COUNT, MAX) are not yet supported.

A.5 Information available in Bgee, OMA and UniProt

In the Table A.3 we provide an overview of the type of information available in Bgee, OMA and UniProt, both in the original representation (relational database for Bgee and HDF5 for OMA), as well as in RDF. More precisely, an "x" represents information available; an added "+" symbolizes information available in RDF; and "o" represents a link to other databases (e.g. OMA homologous groups).

A.6 Discussion regarding choice of RDF materialization for OMA

From the original OMA data, we chose to exclusively materialise in RDF the Hierarchical Orthologous Groups (HOGs) and the cross-references to other resources, which are the main "value-added" parts of OMA. The RDF serialisation relies on the ORTH ontology documented in `https://qfo.github.io/OrthologyOntology/`.

TABLE A.3: The information available on Bgee, OMA and UniProt data stores by also including resulted information (i.e. non-stored) after some data processing. Legend: "x" represents information available; "+" information available in RDF; and "o" represents a link to other databases (e.g. OMA homologous groups).

Information	**Bgee**	**OMA**	**UniProt**
Gene Ontology annotations	x	x	x+
Cross-references	x+	x+	x+
Family and domain	x	x+	
Local synteny	x		
Pairwise homologous genes/proteins	x+		
Homologous groups of genes/proteins	o	x+	o
Hierarchical Orthologous Group (HOG)	x+	o	
Gene expression	x+	o	
Absence of gene expression	x		
Anatomical entity annotations (UBERON)	x+		
Developmental stage annotations	x+		
Species taxonomy (NCBI identifiers)	x+	x+	x+ (fully)
Gene and/or protein names	x+	x+	x+
Subcellular location (including GO annotation)	x	x+	
Sequences	x	x+	
Post-translational modifications and/or processing events	x+		
Protein structures (quaternary, tertiary and secondary)	x+		
Similar proteins based on their membership in UniProt Reference Clusters (UniRef).	x+		

Other information, such as pairwise orthologs (i.e. orth:hasOrtholog property assertions), may be inferred from the HOGs, therefore it does not require to be materialised [92].

The choice of a partial materialisation in the case of OMA is further justified by the following reasons:

(i) To our knowledge, there is currently no OBDA solution to query HDF5 data stores as virtual RDF graphs with SPARQL support. One possible direction would be to translate SPARQL into HDFql (i.e. a recent SQL-like query language for HDF5 data). In analogy to Ontop for Bgee, using such an approach would allow us to expose OMA data as an RDF virtual graph and convert the data to RDF on-the-fly. However, building an OBDA solution for HDF5 from scratch requires substantial development efforts, well beyond the scope of this project.

(ii) Parts of the data in the OMA HDF5 store are already available in UniProt (e.g. the protein sequences or the gene ontology annotations—for further examples see Supplementary Material). Hence, by solely materialising into RDF the OMA HDF5 data which are not already available through the UniProt SPARQL endpoint, we reduce the amount of data duplication, as well as the maintenance efforts required to keep data in sync with changes in UniProt.

A.7 Discussion regarding Virtual Links

At a technical level, note that virtual links among OMA, Bgee and UniProt can be classified into two different types: (i) IRI based (e.g. an instance), and (ii) data literal based (e.g. a label, such as "Homo sapiens" or "HBB"). For example, cross-reference IRIs serve as virtual links of type (i) among the data stores. Concretely, let us consider cross-references to Ensembl genes [215]. On the one hand, the Ensembl gene IRI is assigned to the *up:transcribedFrom* OWL object property in the UniProt RDF store. This IRI is illustrated as a filled black circle in Figure 3.3. On the other hand, we reuse the same IRI as value of the *lscr:xrefEnsemblGene* OWL annotation property in the Bgee and OMA RDF graphs. To illustrate type (ii), we can mention the *rdfs:label* and *skos:prefLabel* properties that assert the same gene names (e.g. "HBB") to an instance of *orth:Gene* in the Bgee graph and *up:Gene* in the UniProt RDF graph, respectively. In Figure 3.3 in Chapter 3, these literals are represented as rectangles.

Bibliography

[1] Lukas Blunschi et al. "Soda: Generating sql for business users". In: *Proceedings of the VLDB Endowment* 5.10 (2012), pp. 932–943.

[2] Zachary D Stephens et al. "Big data: astronomical or genomical?" In: *PLoS biology* 13.7 (2015), e1002195.

[3] Daniel J Rigden and Xosé M Fernández. "The 2018 Nucleic Acids Research database issue and the online molecular biology database collection". en. In: *Nucleic Acids Res.* 46.D1 (Jan. 2018), pp. D1–D7.

[4] Christopher J Mungall et al. "The Monarch Initiative: an integrative data and analytic platform connecting phenotypes to genotypes across species". en. In: *Nucleic Acids Res.* 45.D1 (Jan. 2017), pp. D712–D722.

[5] Esther Kaufmann and Abraham Bernstein. "How useful are natural language interfaces to the semantic web for casual end-users?" In: *The Semantic Web*. Springer, 2007, pp. 281–294.

[6] Ana Claudia Sima et al. "Enabling semantic queries across federated bioinformatics databases". In: *Database* 2019 (2019).

[7] Ana Claudia Sima et al. "Semantic integration and enrichment of heterogeneous biological databases". In: *Evolutionary Genomics*. Springer, 2019, pp. 655–690.

[8] Ana Claudia Sima et al. "A hands-on introduction to querying evolutionary relationships across multiple data sources using SPARQL". In: *F1000Research* 8.1822 (2019), p. 1822.

[9] Mark D Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific data* 3.1 (2016), pp. 1–9.

[10] Anna-Lena Lamprecht et al. "Towards FAIR principles for research software". In: *Data Science* Preprint (2019), pp. 1–23.

[11] Frederic B Bastian et al. "The Bgee suite: integrated curated expression atlas and comparative transcriptomics in animals". In: *Nucleic acids research, under press* (2020).

[12] Adrian M Altenhoff et al. "The OMA orthology database in 2018: retrieving evolutionary relationships among all domains of life through richer web and programmatic interfaces". In: *Nucleic acids research* 46.D1 (2018), pp. D477–D485.

[13] Robert M Waterhouse et al. "OrthoDB: a hierarchical catalog of animal, fungal and bacterial orthologs". In: *Nucleic acids research* 41.D1 (2013), pp. D358–D365.

[14] Sean Powell et al. "eggNOG v3. 0: orthologous groups covering 1133 organisms at 41 different taxonomic ranges". In: *Nucleic acids research* 40.D1 (2012), pp. D284–D289.

[15] Alvis Brazma et al. “ArrayExpress—a public repository for microarray gene expression data at the EBI”. In: *Nucleic acids research* 31.1 (2003), pp. 68–71.

[16] Mathias Uhlen et al. “Towards a knowledge-based human protein atlas”. In: *Nature biotechnology* 28.12 (2010), pp. 1248–1250.

[17] UniProt Consortium. “UniProt: a hub for protein information”. In: *Nucleic acids research* 43.D1 (2015), pp. D204–D212.

[18] MA Chang et al. “Atlas of protein sequence and structure”. In: (1965).

[19] Protein Data Bank. “Protein data bank”. In: *Nature New Biol* 233 (1971), p. 223.

[20] EF Codd. “A relational submodel for large shared data banks”. In: *Commun. Ass. Comput. Mach., June 1970* (1970).

[21] Alan Beaulieu. *Learning SQL: master SQL fundamentals*. " O’Reilly Media, Inc.", 2009.

[22] Chris Fehily. *SQL Database Programming*. Questing Vole Press, 2014.

[23] D Teodoro et al. “Integration of biomedical data using federated databases”. In: *Swiss Medical Informatics* 25.67 (2009), pp. 57–60.

[24] Jesualdo Tomás Fernández-Breis et al. “The Orthology Ontology: development and applications”. In: *Journal of biomedical semantics* 7.1 (2016), pp. 1–11.

[25] Pramod J Sadalage and Martin Fowler. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. Pearson Education, 2013.

[26] Kurt Stockinger et al. “ZNS-Efficient query processing with ZurichNoSQL”. In: *Data & Knowledge Engineering* 112 (2017), pp. 38–54.

[27] Orri Erling and Ivan Mikhailov. “RDF Support in the Virtuoso DBMS”. In: *Networked Knowledge-Networked Media*. Springer, 2009, pp. 7–24.

[28] Tim Berners-Lee, James Hendler, and Ora Lassila. “The semantic web”. In: *Scientific american* 284.5 (2001), pp. 34–43.

[29] Ralph Kimball and Margy Ross. *The Kimball group reader: relentlessly practical tools for data warehousing and business intelligence*. John Wiley & Sons, 2010.

[30] Adrian M Altenhoff et al. “The OMA orthology database in 2015: function predictions, better plant support, synteny view and other improvements”. In: *Nucleic acids research* 43.D1 (2015), pp. D240–D249.

[31] Lydie Lane et al. “neXtProt: a knowledge platform for human proteins”. In: *Nucleic acids research* 40.D1 (2012), pp. D76–D83.

[32] Weizhong Li et al. “The EMBL-EBI bioinformatics web and programmatic tools framework”. In: *Nucleic acids research* 43.W1 (2015), W580–W584.

[33] Helen Berman, Kim Henrick, and Haruki Nakamura. “Announcing the worldwide protein data bank”. In: *Nature Structural & Molecular Biology* 10.12 (2003), pp. 980–980.

[34] T Heath, M Hepp, and C Bizer. *The Story So Far*. 2009.

[35] John Domingue, Dieter Fensel, and James A Hendler. *Handbook of semantic web technologies*. Springer Science & Business Media, 2011.

[36] Pascal Hitzler, Markus Krotzsch, and Sebastian Rudolph. *Foundations of semantic web technologies*. CRC press, 2009.

[37] Peter F Patel-Schneider. "A revised architecture for semantic web reasoning". In: *International Workshop on Principles and Practice of Semantic Web Reasoning*. Springer. 2005, pp. 32–36.

[38] Tim Berners-Lee. *Semantic Web Stack, Presentation available online at https://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/*. 2000.

[39] Rudi Studer, V Richard Benjamins, and Dieter Fensel. "Knowledge engineering: principles and methods". In: *Data & knowledge engineering* 25.1-2 (1998), pp. 161–197.

[40] Franz Baader et al. *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[41] Michael Ashburner et al. "Gene ontology: tool for the unification of biology". In: *Nature genetics* 25.1 (2000), pp. 25–29.

[42] Gene Ontology Consortium. "Gene ontology consortium: going forward". In: *Nucleic acids research* 43.D1 (2015), pp. D1049–D1056.

[43] Christophe Dessimoz et al. "Toward community standards in the quest for orthologs". In: *Bioinformatics* 28.6 (2012), pp. 900–904.

[44] Barry Smith et al. "The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration". In: *Nature biotechnology* 25.11 (2007), pp. 1251–1255.

[45] Scott Federhen. "The NCBI taxonomy database". In: *Nucleic acids research* 40.D1 (2012), pp. D136–D143.

[46] Brian McBride. "The resource description framework (RDF) and its vocabulary description language RDFS". In: *Handbook on ontologies*. Springer, 2004, pp. 51–65.

[47] Boris Motik et al. "OWL 2 web ontology language: Structural specification and functional-style syntax". In: *W3C recommendation* 27.65 (2009), p. 159.

[48] Steffen Lohmann et al. "WebVOWL: Web-based visualization of ontologies". In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2014, pp. 154–158.

[49] Nur Aini Rakhmawati et al. "A comparison of federation over SPARQL endpoints frameworks". In: *International Conference on Knowledge Engineering and the Semantic Web*. Springer. 2013, pp. 132–146.

[50] Diego Calvanese et al. "Ontop: Answering SPARQL queries over relational databases". In: *Semantic Web* 8.3 (2017), pp. 471–487.

[51] Juan F Sequeda and Daniel P Miranker. "Ultrawrap: SPARQL execution on relational data". In: *Journal of Web Semantics* 22 (2013), pp. 19–39.

[52] Christian Bizer and Andy Seaborne. "D2RQ-treating non-RDF databases as virtual RDF graphs". In: *Proceedings of the 3rd international semantic web conference (ISWC2004)*. Vol. 2004. Proceedings of ISWC2004. 2004.

[53] Elisabeth Gasteiger et al. "ExPASy: the proteomics server for in-depth protein knowledge and analysis". In: *Nucleic acids research* 31.13 (2003), pp. 3784–3788.

[54] Diptikalyan Saha et al. "ATHENA: an ontology-driven system for natural language querying over relational data stores". In: *Proceedings of the VLDB Endowment* 9.12 (2016), pp. 1209–1220.

[55] Dezhao Song et al. "TR discover: A natural language interface for querying and analyzing interlinked datasets". In: *International Semantic Web Conference*. Springer. 2015, pp. 21–37.

[56] Basil Ell, Denny Vrandečić, and Elena Simperl. "Labels in the web of data". In: *International Semantic Web Conference*. Springer. 2011, pp. 162–176.

[57] Rhoda J Kinsella et al. "Ensembl BioMarts: a hub for data retrieval across taxonomic space". In: *Database* 2011 (2011).

[58] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. "Ontology of Integration and Integration of Ontologies." In: *Description Logics* 49.10-19 (2001), p. 30.

[59] Mihaela A Bornea et al. "Building an efficient RDF store over a relational database". In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. 2013, pp. 121–132.

[60] Juan Federico Sequeda. *Integrating relational databases with the Semantic Web*. Vol. 22. IOS Press, 2016.

[61] Franck Michel, Johan Montagnat, and Catherine Faron Zucker. "A survey of RDB to RDF translation approaches and tools". In: (2014).

[62] Viktor Leis et al. "How good are query optimizers, really?" In: *Proceedings of the VLDB Endowment* 9.3 (2015), pp. 204–215.

[63] Wentao Wu et al. "Predicting query execution time: Are optimizer cost models really unusable?" In: *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE. 2013, pp. 1081–1092.

[64] Gabriela Montoya, Hala Skaf-Molli, and Katja Hose. "The Odyssey approach for optimizing federated SPARQL queries". In: *International Semantic Web Conference*. Springer. 2017, pp. 471–489.

[65] Susan B. Davidson, Chris Overton, and Peter Buneman. "Challenges in integrating biological data sources". In: *Journal of Computational Biology* 2.4 (1995), pp. 557–572.

[66] Robert Stevens et al. "TAMBIS: transparent access to multiple bioinformatics information sources". In: *Bioinformatics* 16.2 (2000), pp. 184–186.

[67] Aimilia Magkanaraki et al. "Benchmarking rdf schemas for the semantic web". In: *International Semantic Web Conference*. Springer. 2002, pp. 132–146.

[68] Mark D Wilkinson and Matthew Links. "BioMOBY: an open source biological web services proposal". In: *Briefings in bioinformatics* 3.4 (2002), pp. 331–341.

[69] Lincoln D Stein. "Integrating biological databases". In: *Nature Reviews Genetics* 4.5 (2003), pp. 337–345.

[70] Rolf Apweiler, Amos Bairoch, and Cathy H Wu. "Protein sequence databases". In: *Current opinion in chemical biology* 8.1 (2004), pp. 76–80.

[71] UniProt Consortium. "Update on activities at the Universal Protein Resource (UniProt) in 2013". In: *Nucleic acids research* 41.D1 (2012), pp. D43–D47.

[72] Marcos Da Silveira, Cédric Pruski, and Reinhard Schneider. *Data integration in the life sciences*. Springer, 2017.

[73] Richard G Côté et al. "The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries". In: *BMC bioinformatics* 7.1 (2006), p. 97.

[74] Manuel Salvadores et al. "BioPortal as a dataset of linked biomedical ontologies and terminologies in RDF". In: *Semantic web* 4.3 (2013), pp. 277–284.

[75] Mark Wilkinson. "Interoperability with Moby 1.0-it's better than sharing your toothbrush!" In: *Nature Precedings* (2008), pp. 1–1.

[76] Erick Antezana et al. "BioGateway: a semantic systems biology tool for the life sciences". In: *BMC bioinformatics* 10.S10 (2009), S11.

[77] Stephan Philippi, ed. *Special Issue : Database Integration in Life Sciences*. Vol. 9. Springer, 2008.

[78] François Belleau et al. "Bio2RDF: towards a mashup to build bioinformatics knowledge systems". In: *Journal of biomedical informatics* 41.5 (2008), pp. 706–716.

[79] Alison Callahan et al. "Bio2RDF release 2: improved coverage, interoperability and provenance of life science linked data". In: *Extended Semantic Web Conference*. Springer. 2013, pp. 200–212.

[80] Michel Dumontier et al. "Bio2RDF release 3: a larger connected network of linked data for the life sciences". In: *Proceedings of the 2014 International Conference on Posters & Demonstrations Track*. Vol. 1272. 2014, pp. 401–404.

[81] Erick Antezana, Martin Kuiper, and Vladimir Mironov. "Biological knowledge management: the emerging role of the Semantic Web technologies". In: *Briefings in bioinformatics* 10.4 (2009), pp. 392–407.

[82] Huajun Chen, Tong Yu, and Jake Y Chen. "Semantic web meets integrative biology: a survey". In: *Briefings in bioinformatics* 14.1 (2013), pp. 109–125.

[83] Ramona L Walls et al. "Semantics in support of biodiversity knowledge discovery: an introduction to the biological collections ontology and related ontologies". In: *PloS one* 9.3 (2014), e89606.

[84] Marylyn D Ritchie et al. "Methods of integrating data to uncover genotype-phenotype interactions". en. In: *Nat. Rev. Genet.* 16.2 (Feb. 2015), pp. 85–97.

[85] Konrad J Karczewski and Michael P Snyder. "Integrative omics for health and disease". en. In: *Nat. Rev. Genet.* 19.5 (May 2018), pp. 299–310.

[86] Hansi Zhang et al. "An ontology-guided semantic data integration framework to support integrative data analysis of cancer survival". en. In: *BMC Med. Inform. Decis. Mak.* 18.Suppl 2 (July 2018), p. 41.

[87] Michael Baitaluk and Julia Ponomarenko. "Semantic integration of data on transcriptional regulation". en. In: *Bioinformatics* 26.13 (July 2010), pp. 1651–1661.

[88] Chengbin Wang, Xiaogang Ma, and Jianguo Chen. "Ontology-driven data integration and visualization for exploring regional geologic time and paleontological information". In: *Comput. Geosci.* 115 (June 2018), pp. 12–19.

[89] Tarcisio M Farias, Ana Roxin, and Christophe Nicolle. "FOWLA, A Federated Architecture for Ontologies". In: *Rule Technologies: Foundations, Tools, and Applications*. Springer International Publishing, 2015, pp. 97–111.

[90] Sebastian Mate et al. "Ontology-based data integration between clinical and research systems". en. In: *PLoS One* 10.1 (Jan. 2015), e0116656.

[91] Craig A Knoblock and Pedro Szekely. "Exploiting Semantics for Big Data Integration". In: *AI Magazine* 36.1 (2015).

[92] Tarcisio M de Farias, Hirokazu Chiba, and Jesualdo T Fernández-Breis. "Leveraging logical rules for efficacious representation of large orthology datasets". In: Apr. 2017.

[93] Muhammad Shoaib, Adnan Ahmad Ansari, and Sung-Min Ahn. "cMapper: gene-centric connectivity mapper for EBI-RDF platform". en. In: *Bioinformatics* 33.2 (Jan. 2017), pp. 266–271.

[94] Rudi Studer, V Richard Benjamins, and Dieter Fensel. "Knowledge engineering: Principles and methods". In: *Data Knowl. Eng.* 25.1 (Mar. 1998), pp. 161–197.

[95] Patricia L Whetzel et al. "BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications". en. In: *Nucleic Acids Res.* 39.Web Server issue (July 2011), W541–5.

[96] Barry Smith et al. "The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration". en. In: *Nat. Biotechnol.* 25.11 (Nov. 2007), pp. 1251–1255.

[97] UniProt Consortium. "UniProt: the universal protein knowledgebase". en. In: *Nucleic Acids Res.* 46.5 (Mar. 2018), p. 2699.

[98] Janna Hastings et al. "The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013". en. In: *Nucleic Acids Res.* 41.Database issue (Jan. 2013), pp. D456–63.

[99] Carole Goble and Robert Stevens. "State of the nation in data integration for bioinformatics". In: *Journal of biomedical informatics* 41.5 (2008), pp. 687–693.

[100] Zhang Zhang et al. "Data integration in bioinformatics: current efforts and challenges". In: *Bioinformatics-Trends and Methodologies*. IntechOpen, 2011.

[101] Vasileios Lapatas et al. "Data integration in biological research: an overview". In: *Journal of Biological Research-Thessaloniki* 22.1 (2015), p. 9.

[102] Kevin M Livingston et al. "KaBOB: ontology-based semantic integration of biomedical databases". en. In: *BMC Bioinformatics* 16 (Apr. 2015), p. 126.

[103] Vassil Momtchev et al. "Expanding the pathway and interaction knowledge in linked life data". In: *Proc. of International Semantic Web Challenge* (2009).

[104] Ali Hasnain et al. "Biofed: federated query processing over life sciences linked open data". In: *Journal of biomedical semantics* 8.1 (2017), p. 13.

[105] Marija Djokic-Petrovic et al. "PIBAS FedSPARQL: a web-based platform for integration and exploration of bioinformatics datasets". In: *Journal of biomedical semantics* 8.1 (2017), p. 42.

[106] Marko Živanovic. "SpecINT: A framework for data integration over cheminformatics and bioinformatics RDF repositories". In: *semantic-web-journal.net* (2019).

[107] Sarala M Wimalaratne et al. "SPARQL-enabled identifier conversion with Identifiers.org". en. In: *Bioinformatics* 31.11 (June 2015), pp. 1875–1877.

[108] Raul Castro Fernandez et al. "Seeping semantics: Linking datasets using word embeddings for data discovery". In: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE. 2018, pp. 989–1000.

[109] Adrian M Altenhoff et al. "The OMA orthology database in 2018: retrieving evolutionary relationships among all domains of life through richer web and programmatic interfaces". en. In: *Nucleic Acids Res.* 46.D1 (Jan. 2018), pp. D477–D485.

[110] Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. "SPARQL 1.1 query language". In: *W3C recommendation* 21.10 (2013).

[111] Keith Alexander et al. "Describing linked datasets with the VoID vocabulary". In: (Mar. 2011).

[112] Adrian M Altenhoff et al. "Inferring hierarchical orthologous groups from orthologous gene pairs". In: *PLoS One* 8.1 (Jan. 2013), e53786.

[113] Andrea Komljenovic et al. "BgeeDB, an R package for retrieval of curated expression datasets and for gene list expression localization enrichment tests". In: *F1000Res.* 5 (Aug. 2018).

[114] V Gadepally et al. "The BigDAWG polystore system and architecture". In: *2016 IEEE High Performance Extreme Computing Conference (HPEC)*. 2016, pp. 1–6.

[115] Muhammad Saleem, Ali Hasnain, and Axel-Cyrille Ngonga Ngomo. "LargeRDFBench: A billion triples benchmark for SPARQL endpoint federation". In: *Journal of Web Semantics* 48 (Jan. 2018), pp. 85–125.

[116] Michael D Siegel and Stuart E Madnick. "A metadata approach to resolving semantic conflicts". In: (1991).

[117] Avigdor Gal et al. "Automatic ontology matching using application semantics". In: *AI magazine* 26.1 (2005), p. 21.

[118] Nicole Redaschi, Uniprot Consortium, et al. "Uniprot in RDF: Tackling data integration and distributed annotation with the semantic web". In: (2009).

[119] Jesualdo Tomás Fernández-Breis et al. "The Orthology Ontology: development and applications". en. In: *J. Biomed. Semantics* 7.1 (June 2016), p. 34.

[120] Kristoffer Forslund et al. "Gearing up to handle the mosaic nature of life in the quest for orthologs". en. In: *Bioinformatics* (Aug. 2017).

[121] Robert Petryszak et al. "Expression Atlas update—a database of gene and transcript expression from microarray- and sequencing-based functional genomics experiments". In: *Nucleic Acids Res.* 42.D1 (Jan. 2014), pp. D926–D932.

[122] Tomas Hruz et al. "Genevestigator v3: a reference expression database for the meta-analysis of transcriptomes". In: *Adv. Bioinformatics* 2008 (July 2008), p. 420747.

[123] Oana Palasca et al. "TISSUES 2.0: an integrative web resource on mammalian tissue expression". en. In: *Database* 2018 (Jan. 2018).

[124] Barry Smith et al. "Relations in biomedical ontologies". In: *Genome Biology* 6.5 (Apr. 2005), R46. ISSN: 1474-760X. DOI: 10.1186/gb-2005-6-5-r46. URL: https://doi.org/10.1186/gb-2005-6-5-r46.

[125] Simon Jupp et al. "The EBI RDF platform: linked open data for the life sciences". en. In: *Bioinformatics* 30.9 (May 2014), pp. 1338–1339.

[126] D Calvanese, B Cogrel, S Komla-Ebri, et al. "Ontop: Answering SPARQL queries over relational databases". In: *Semant. Pragmat.* (2017).

[127] Christopher J. Mungall et al. "Uberon, an integrative multi-species anatomy ontology". In: *Genome Biology* 13.1 (Jan. 2012), R5. ISSN: 1474-760X. DOI: 10.1186/gb-2012-13-1-r5. URL: https://doi.org/10.1186/gb-2012-13-1-r5.

[128] Tarcisio Mendes de Farias, Kurt Stockinger, and Christophe Dessimoz. "VoIDext: Vocabulary and patterns for enhancing interoperable datasets with virtual links". In: *arXiv preprint arXiv:1906.01950v2* (2019).

[129] Sébastien Ferré. "Sparklis: an expressive query builder for SPARQL endpoints with guidance in natural language". In: *Semantic Web* 8.3 (2017), pp. 405–418.

[130] Sabrina Kirrane, Alessandra Mileo, and Stefan Decker. "Access control and the resource description framework: A survey". In: *Semantic Web* 8.2 (2017), pp. 311–352.

[131] Toni Gabaldón and Eugene V Koonin. "Functional and evolutionary implications of gene orthology". In: *Nature Reviews Genetics* 14.5 (2013), pp. 360–366.

[132] Natasha Glover et al. "Advances and Applications in the Quest for Orthologs". In: *Molecular biology and evolution* 36.10 (2019), pp. 2157–2164.

[133] Walter M Fitch. "Distinguishing homologous from analogous proteins". In: *Systematic zoology* 19.2 (1970), pp. 99–113.

[134] Adrian M Altenhoff et al. "Inferring hierarchical orthologous groups from orthologous gene pairs". In: *PloS one* 8.1 (2013), e53786.

[135] Rosa Fernández, Toni Gabaldon, and Christophe Dessimoz. "Orthology: Definitions, prediction, and impact on species phylogeny inference". In: *Phylogenetics in the Genomic Era* (2020), pp. 2–4.

[136] Paula Duek et al. "Exploring the uncharacterized human proteome using neXtProt". In: *Journal of proteome research* 17.12 (2018), pp. 4211–4226.

[137] Anandhi Iyappan et al. "NeuroRDF: semantic integration of highly curated data to prioritize biomarker candidates in Alzheimer's disease". In: *Journal of biomedical semantics* 7.1 (2016), pp. 1–15.

[138] Antony J Williams et al. "Open PHACTS: semantic interoperability for drug discovery". In: *Drug discovery today* 17.21-22 (2012), pp. 1188–1198.

[139] Simon Jupp et al. "The EBI RDF platform: linked open data for the life sciences". In: *Bioinformatics* 30.9 (2014), pp. 1338–1339.

[140] Evgeny M Zdobnov et al. "OrthoDB v9. 1: cataloging evolutionary and functional annotations for animal, fungal, plant, archaeal, bacterial and viral orthologs". In: *Nucleic acids research* 45.D1 (2017), pp. D744–D749.

[141] Ikuo Uchiyama et al. "MBGD update 2018: microbial genome database based on hierarchical orthology relations covering closely related and distantly related comparisons". In: *Nucleic acids research* 47.D1 (2019), pp. D382–D389.

[142] Hirokazu Chiba, Hiroyo Nishide, and Ikuo Uchiyama. "Construction of an ortholog database using the semantic web technology for integrative analysis of genomic data". In: *PloS one* 10.4 (2015), e0122802.

[143] Tarcisio M de Farias, Hirokazu Chiba, and Jesualdo T Fernández-Breis. *Leveraging logical rules for efficacious representation of large orthology datasets*. 2017.

[144] Monique Zahn-Zabal, Christophe Dessimoz, and Natasha M Glover. "Identifying orthologs with OMA: A primer". In: *F1000Research* 9 (2020).

[145] Catherine Brooksbank et al. "The european bioinformatics institute's data resources 2014". In: *Nucleic acids research* 42.D1 (2014), pp. D18–D25.

[146] Daniel R Zerbino et al. "Ensembl 2018". In: *Nucleic acids research* 46.D1 (2018), pp. D754–D761.

[147] Erik LL Sonnhammer and Eugene V Koonin. "Orthology, paralogy and proposed classification for paralog subtypes". In: *TRENDS in Genetics* 18.12 (2002), pp. 619–620.

[148] Casey W Dunn, Xi Luo, and Zhijin Wu. "Phylogenetic analysis of gene expression". In: *Integrative and comparative biology* 53.5 (2013), pp. 847–856.

[149] Nadezda Kryuchkova-Mostacci and Marc Robinson-Rechavi. "Tissue-specificity of gene expression diverges slowly between orthologs, and rapidly between paralogs". In: *PLoS computational biology* 12.12 (2016), e1005274.

[150] David Brawand et al. "The evolution of gene expression levels in mammalian organs". In: *Nature* 478.7369 (2011), pp. 343–348.

[151] Scott A Rifkin, Junhyong Kim, and Kevin P White. "Evolution of gene expression in the Drosophila melanogaster subgroup". In: *Nature genetics* 33.2 (2003), pp. 138–144.

[152] Seung Gu Park and Sun Shim Choi. "Expression breadth and expression abundance behave differently in correlations with evolutionary rates". In: *BMC evolutionary biology* 10.1 (2010), p. 241.

[153] Margaret O Dayhoff. *Atlas of protein sequence and structure*. National Biomedical Research Foundation., 1972.

[154] Linda Z Holland et al. "The amphioxus genome illuminates vertebrate origins and cephalochordate biology". In: *Genome research* 18.7 (2008), pp. 1100–1111.

[155] Gary W Litman, Jonathan P Rast, and Sebastian D Fugmann. "The origins of vertebrate adaptive immunity". In: *Nature Reviews Immunology* 10.8 (2010), pp. 543–553.

[156] Ferdinand Marlétaz et al. "Amphioxus functional genomics and the origins of vertebrate gene regulation". In: *Nature* 564.7734 (2018), pp. 64–70.

[157] Sebastien Moretti et al. "Selectome update: quality control and computational improvements to a database of positive selection". In: *Nucleic acids research* 42.D1 (2014), pp. D917–D921.

[158] Dennis Diefenbach et al. "Towards a question answering system over the semantic web". In: *Semantic Web* Preprint (2018), pp. 1–19.

[159] Weiguo Zheng et al. "Question answering over knowledge graphs: question understanding via template decomposition". In: *Proceedings of the VLDB Endowment* 11.11 (2018), pp. 1373–1386.

[160] Svitlana Vakulenko et al. "Message Passing for Complex Question Answering over Knowledge Graphs". In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2019, pp. 1431–1440.

[161] Fei Li and HV Jagadish. "Constructing an interactive natural language interface for relational databases". In: *Proceedings of the VLDB Endowment* 8.1 (2014), pp. 73–84.

[162] Fei Li and HV Jagadish. "Understanding natural language queries over relational databases". In: *ACM SIGMOD Record* 45.1 (2016), pp. 6–13.

[163] Ahmad Sakor, Kuldeep Singh, and Maria-Esther Vidal. "FALCON: An Entity and Relation Linking Framework over DBpedia". In: (2019).

[164] Paolo Ferragina and Ugo Scaiella. "Tagme: on-the-fly annotation of short text fragments (by wikipedia entities)". In: *Proceedings of the 19th ACM international conference on Information and knowledge management*. 2010, pp. 1625–1628.

[165] Pablo N Mendes et al. "DBpedia spotlight: shedding light on the web of documents". In: *Proceedings of the 7th international conference on semantic systems*. 2011, pp. 1–8.

[166] Alex Olieman et al. "Entity linking by focusing DBpedia candidate entities". In: *Proceedings of the first international workshop on Entity recognition & disambiguation*. 2014, pp. 13–24.

[167] Hamid Zafar, Giulio Napolitano, and Jens Lehmann. "Formal query generation for question answering over knowledge bases". In: *European Semantic Web Conference*. Springer. 2018, pp. 714–728.

[168] Kuldeep Singh et al. "No one is perfect: Analysing the performance of question answering components over the dbpedia knowledge graph". In: *arXiv preprint arXiv:1809.10044* (2018).

[169] Priyansh Trivedi et al. "Lc-quad: A corpus for complex question answering over knowledge graphs". In: *International Semantic Web Conference*. Springer. 2017, pp. 210–218.

[170] Mohnish Dubey et al. "Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia". In: *International Semantic Web Conference*. Springer. 2019, pp. 69–78.

[171] Angela Bonifati, Wim Martens, and Thomas Timm. "An analytical study of large SPARQL query logs". In: *The VLDB Journal* (2019), pp. 1–25.

[172] Christina Unger et al. "Question answering over linked data (QALD-4)". In: 2014.

[173] Muhammad Saleem et al. "Question Answering Over Linked Data: What is Difficult to Answer? What Affects the F scores?" In: *BLINK/NLIWoD3@ ISWC*. 2017.

[174] Gaurav Maheshwari et al. "Learning to rank query graphs for complex question answering over knowledge graphs". In: *International Semantic Web Conference*. Springer. 2019, pp. 487–504.

[175] Sherzod Hakimov et al. "Applying semantic parsing to question answering over linked data: Addressing the lexical gap". In: *International Conference on Applications of Natural Language to Information Systems*. Springer. 2015, pp. 103–109.

[176] Anca Marginean. "Question answering over biomedical linked data with grammatical framework". In: *Semantic Web* 8.4 (2017), pp. 565–580.

[177] Thierry Hamon et al. "Description of the POMELO System for the Task 2 of QALD-2014." In: *CLEF (Working Notes)* 1212 (2014), p. 28.

[178] Dennis Diefenbach and Andreas Thalhammer. "Pagerank and generic entity summarization for rdf knowledge bases". In: *European Semantic Web Conference*. Springer. 2018, pp. 145–160.

[179] Nicole Redaschi, UniProt Consortium, et al. "Uniprot in RDF: Tackling data integration and distributed annotation with the semantic web". In: *Nature precedings* (2009), pp. 1–1.

[180] Lawrence Page et al. *The pagerank citation ranking: Bringing order to the web.* Tech. rep. Stanford InfoLab, 1999.

[181] SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. "Distributional semantics resources for biomedical text processing". In: *Proceedings of LBM* (2013), pp. 39–44.

[182] Katerina Gkirtzou et al. "Keywords-to-sparql translation for rdf data search and exploration". In: *International Conference on Theory and Practice of Digital Libraries*. Springer. 2015, pp. 111–123.

[183] Dennis Diefenbach, Kamal Singh, and Pierre Maret. "Wdaqua-core1: a question answering service for rdf knowledge bases". In: *Companion Proceedings of the The Web Conference 2018*. 2018, pp. 1087–1091.

[184] Saeedeh Shekarpour et al. "Sina: Semantic interpretation of user queries for question answering on interlinked data". In: *Journal of Web Semantics* 30 (2015), pp. 39–51.

[185] Lei Zou et al. "Natural language question answering over RDF: a graph data driven approach". In: *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 2014, pp. 313–324.

[186] Katrin Affolter, Kurt Stockinger, and Abraham Bernstein. "A comparative survey of recent natural language interfaces for databases". In: *The VLDB Journal* 28.5 (2019), pp. 793–819.

[187] Homa B Hashemi and Rebecca Hwa. "An evaluation of parser robustness for ungrammatical sentences". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 1765–1774.

[188] Tarcisio Mendes [de Farias], Ana Roxin, and Christophe Nicolle. "SWRL rule-selection methodology for ontology interoperability". In: *Data & Knowledge Engineering* 105 (2016). Knowledge Engineering for Enterprise, Integration, Interoperability and Networking: Theory and Applications, pp. 53–72. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2015.09.001. URL: http://www.sciencedirect.com/science/article/pii/S0169023X15000713.

[189] Thierry Hamon, Natalia Grabar, and Fleur Mougin. "Querying biomedical linked data with natural language questions". In: *Semantic Web* 8.4 (2017), pp. 581–599.

[190] Stefanie Nadig, Martin Braschler, and Kurt Stockinger. "Database Search vs. Information Retrieval: A Novel Method for Studying Natural Language Querying of Semi-Structured Data". In: *International Conference on Language Resources and Evaluation (LREC)*. 2020.

[191] Nilesh Chakraborty et al. "Introduction to Neural Network based Approaches for Question Answering over Knowledge Graphs". In: *arXiv preprint arXiv:1907.09361* (2019).

[192] Abdelrahman Abdelkawi et al. "Complex Query Augmentation for Question Answering over Knowledge Graphs". In: *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer. 2019, pp. 571–587.

[193] Dennis Diefenbach et al. "QAnswer KG: Designing a portable Question Answering System over RDF data". In: (2020).

[194] Denis Lukovnikov et al. "Neural network-based question answering over knowledge graphs on word and character level". In: *Proceedings of the 26th international conference on World Wide Web*. 2017, pp. 1211–1220.

[195] Daniel Deutch, Nave Frost, and Amir Gilad. "Explaining Natural Language query results". In: *The VLDB Journal* 29.1 (2020), pp. 485–508.

[196] Axel-Cyrille Ngonga Ngomo et al. "Sorry, i don't speak SPARQL: translating SPARQL queries into natural language". In: *Proceedings of the 22nd international conference on World Wide Web*. 2013, pp. 977–988.

[197] Andreas Kokkalis et al. "Logos: a system for translating queries into narratives". In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 2012, pp. 673–676.

[198] Tao Yu et al. "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task". In: *arXiv preprint arXiv:1809.08887* (2018).

[199] Ursin Brunner and Kurt Stockinger. "Entity matching with transformer architectures-a step forward in data integration". In: *International Conference on Extending Database Technology, Copenhagen, 30 March-2 April 2020*. 2020.

[200] Juan F Sequeda et al. "A Pay-as-you-go Methodology to Design and Build Enterprise Knowledge Graphs from Relational Databases". In: *International Semantic Web Conference*. Springer. 2019, pp. 526–545.

[201] Jaeho Shin et al. "Incremental knowledge base construction using deepdive". In: *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*. Vol. 8. 11. NIH Public Access. 2015, p. 1310.

[202] Muhammad Saleem, Ali Hasnain, and Axel-Cyrille Ngonga Ngomo. "Largerdfbench: a billion triples benchmark for sparql endpoint federation". In: *Journal of Web Semantics* 48 (2018), pp. 85–125.

[203] Muhammad Saleem et al. "Costfed: Cost-based query optimization for sparql endpoint federation". In: *Procedia Computer Science* 137 (2018), pp. 163–174.

[204] Tina Begum and Marc Robinson-Rechavi. "Special care is needed in applying phylogenetic comparative methods to gene trees with speciation and duplication nodes". In: *bioRxiv* (2020), p. 719336.

[205] Adrian M Altenhoff et al. "Resolving the ortholog conjecture: orthologs tend to be weakly, but significantly, more similar in function than paralogs". In: *PLoS Comput Biol* 8.5 (2012), e1002514.

[206] Xiaoshu Chen and Jianzhi Zhang. "The ortholog conjecture is untestable by the current gene ontology but is supported by RNA sequencing data". In: *PLoS Comput Biol* 8.11 (2012), e1002784.

[207] Hyeonji Kim et al. "Natural language to SQL: Where are we today?" In: *Proceedings of the VLDB Endowment* 13.10 (2020), pp. 1737–1750.

[208] Denis Lukovnikov, Asja Fischer, and Jens Lehmann. "Pretrained transformers for simple question answering over knowledge graphs". In: *International Semantic Web Conference*. Springer. 2019, pp. 470–486.

[209] Jiaqi Guo et al. "Towards complex text-to-sql in cross-domain database with intermediate representation". In: *arXiv preprint arXiv:1905.08205* (2019).

[210] Bailin Wang et al. "Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers". In: *arXiv preprint arXiv:1911.04942* (2019).

[211] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[212] Michael Günther. "Freddy: Fast word embeddings in database systems". In: *Proceedings of the 2018 International Conference on Management of Data*. 2018, pp. 1817–1819.

[213] Quan Wang et al. "Knowledge graph embedding: A survey of approaches and applications". In: *IEEE Transactions on Knowledge and Data Engineering* 29.12 (2017), pp. 2724–2743.

[214] Xiao Huang et al. "Knowledge graph embedding based question answering". In: *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 2019, pp. 105–113.

[215] Daniel R Zerbino et al. "Ensembl 2018". en. In: *Nucleic Acids Res.* 46.D1 (Jan. 2018), pp. D754–D761.