**SEMANTIC DATA MANAGEMENT**

# Knowledge Graphs Lab

Team members:
**Laura Forero (laura.forero)**
**Chidiebere Ogbuchi (chidiebere.ogbuchi)**
**Bogdana Živković (bogdana.zivkovic)**

Date:
**18th May 2023**

# A. Overview of Implementation

The creation of TBOX was done with gra.fo tool, the resulting owl can be found in the file (KnowledgeGraphSDMLab\data\tbox.owl). The KnowledgeGraphSDMLab folder contains the Java project for creating the ABOX (KnowledgeGraphSDMLab\src\main\java\org\example\ontology\ABOX.java) and the nt file of the resulting graph is located at (KnowledgeGraphSDMLab\data\abox.nt). Additionally, for part B4 the queries are found in the file (Team12E_B4_OGBUCHI_FORERO_ZIVKOVIC.sparql).

Below is an overview of the implementation:

## 1. Technology

### 1.1 GraphDB

GraphDB is a graph database compliant with RDF and SPARQL specifications. It supports open APIs based on the RDF4J (ex-Sesame) project and enables fast publishing of linked data on the web. The Workbench is used for searching, exploring and managing GraphDB semantic repositories.

### 1.2 Apache Jena

Apache Jena is an open source Java framework for building Semantic Web and Linked Data applications. It is quite powerful as it can be used to create ontologies, query and add constraints (via SHACL) in a semantic web world. For the purpose of this lab, we have used Apache Jena API to create TBOX and ABOX (and their links) for our publications' data.

### 1.3 Ontology

The TBOX (Terminology Box) represents the metadata or schema of the knowledge graph, defining concepts (Classes) and their relationships (Properties).

The ABOX (Assertion Box) represents the data instances and their relationships. By combining the TBOX and ABOX, we create an ontology that enables various possibilities for querying and analyzing the data.

## 2. Dataset

We used the output_article data generated from DBLP data (2023) (which was converted from XML to csv). The final data is sliced and stored in datafinal folder due to its size and then preprocessed for the creation of the ABOX data.

## 3. Preprocess

In order to create the correct topology (TBOX and ABOX), we pre-process our data first. We wrote a python script which you can use to get the preprocessed data. Run the following to obtain the instance_data.csv file:

```
$ git clone https://github.com/Lala341/SDM-lab03.git or unzip the Code file
```

```
$ cd SDM-lab03/KnowledgeGraphSDMLab/data/csv
```

Execute preprocess_dblp.ipynb

<!—Notebook can be converted to py script using in case of absence of notebook compiler like jupyter--!>

```
$ jupyter nbconvert --to script preprocess_dblp.ipynb
```

```
python preprocess_dblp.py
```

## 4. Generate TBOX

Obtained from gra.fo saved into

```
$ cd SDM-lab03\KnowledgeGraphSDMLab\data
```

Confirm tbox.owl and the previously generated csv exists.

## 5. Generate ABOX

Execute the Java main class (i.e source file name ‘App’)to generate and save the ABOX

After running the above mentioned commands, you should have these files under cd SDM-lab03\KnowledgeGraphSDMLab\data directory: Tbox.owl, Abox.nt, csv/ instance_data.csv

Now, you can load Tbox.owl and Abox.nt in GraphDB and start querying the data.

# B. ONTOLOGY CREATION

## B1. TBOX Definition

The TBOX code is programmatically generated from https://app.gra.fo/ and serves as a base for the final Tbox file named Tbox.owl. The developed ontology for academic publications provides a structured representation of the entities, relationships, and properties relevant to the scholarly domain. By defining classes, object properties, and data properties, the ontology enables the modeling and organization of information related to concepts of paper, authorship, publication and review, and other associated entities.

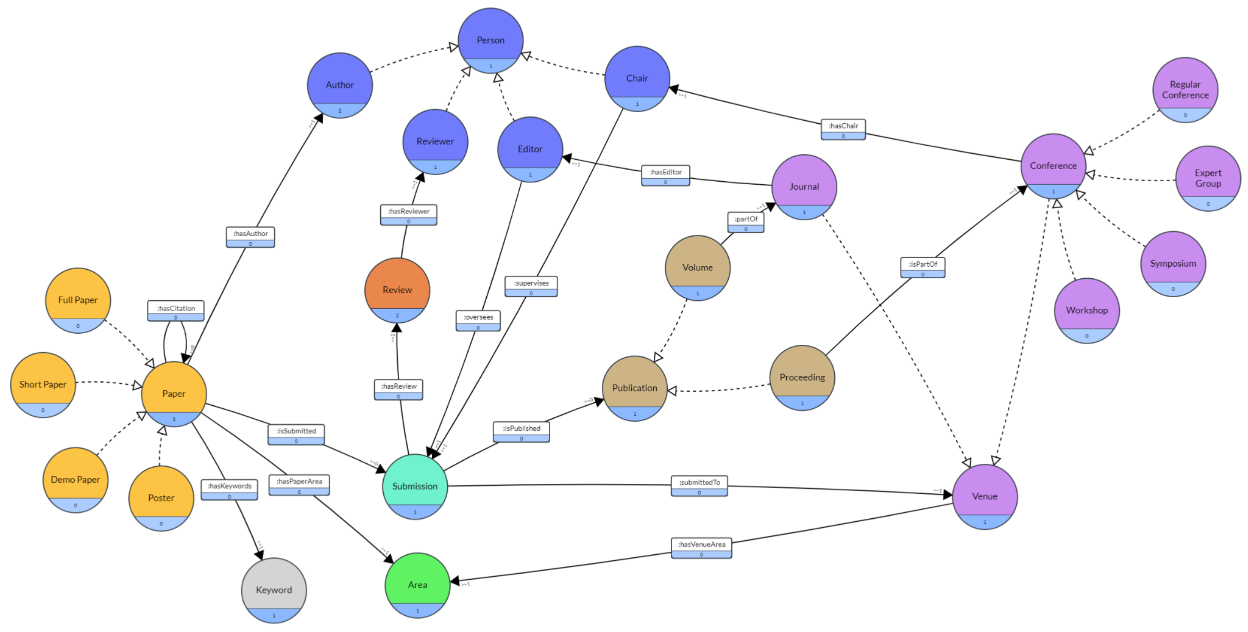

Figure 1: graphical representation of TBOX ontology

Below table1 and 2 gives a summary of the ontologies structure and relationships between the data elements. The ontology defines various domain-specific relationships and attributes to represent the structure and semantics of the scientific data management domain.

| Domain | Property | Range | Cardinality |
|---|---|---|---|
| Paper | :hasAuthor | Author | >=1 |
| Paper | :hasCitation | Paper | >=0 |
| Paper | :hasKeywords | Keywords | >=1 |
| Paper | :hasPaperArea | Area | >=1 |
| Paper | :isSubmitted | Submission | >=0 |
| Chair | :supervises | Submission | >=1 |
| Editor | :oversees | Submission | >=1 |
| Conference | :hasChair | Chair | >=1 |

|  |  |  |  |
|---|---|---|---|
| Journal | :hasEditor | Editor | >=1 |
| Venue | :hasVenueArea | Area | >=1 |
| Review | :hasReviewer | Reviewer | >=1 |
| Proceeding | :isPartof | Conference | ==1 |
| Volume | :partof | Journal | ==1 |
| Submission | :hasReview | Review | ==2 |
| Submission | :isPublished | Publication | >=0 |
| Submission | :Submittedto | Venue | >=1 |

Table 1. List of Object Properties

| Property Class | Data Properties | Range XSD |
|---|---|---|
| Paper | Abstract \| Title \| DOI | str \| str \| int |
| Person | PersonName | str |
| Author | Affiliation \| AuthorOrcid | str \| str |
| Chair | ManagesPanelsContent | boolean |
| Editor | ManagesProductionProcess | boolean |
| Reviewer | YearsReviewerExperience | int |
| Conference | Frequency | int |
| Journal | Journalimpactfactor | float |
| Publication | YearPublished | int |
| Proceeding | Indexnr | int |
| Volume | VolumeNr | int |
| Keyword | Word | str |
| Area | Areaname | str |
| Review | AcceptanceDecision \| ReviewText | boolean \| str |
| Venue | VenueName | str |
| Submission | AcceptanceFinalDecision | boolean |

Table 2. List of Data Properties

All classes and properties have the prefix http://www.sdm.lab/# and follow a similar color code as the table.
The following assumptions were made during the design of the TBOX:

- The Person class is inferred from the subclasses Author, Reviewer, Editor and Chair.
- A submission is handled by an Editor and Chair, so therefore oversee all entities related to the submission including assignment of at least 2 reviewers to each submission.
- Only submission that has AcceptanceFinalDecision as True becomes a publication, while False (rejected papers) do not proceed for publication.
- Conference class is inferred by workshops, Symposium, Expert group & regular conference subclasses.
- Paper class is inferred by full paper, short paper, demo paper or poster. However only conferences can have posters.
- Publication class is inferred by Journal volume and Conference proceedings

- Person class is with subclasses of authors, chairs, editors and reviewers, and an author can also be a reviewer or editor or chair but not for their own paper submission. To ensure the pairwise disjointness and completeness rules are maintained, there is no overlap between instances of different classes within the set.

In addition to the constraints defined by the cardinalities specified on the predicate relationships, some other OWL restrictions are implemented to ensure consistency. Any violation of these ontology constraints will result in irregularities.

- All main classes contain all relevant subclasses as represented by the dotted arrow in figure 1.
- All classes are complete and pairwise disjoint.
- All relationships are also inversely implied. For instance Journals can only publish Volumes, while conference can publish proceedings and conversely volumes can only belong to journals, while proceedings can only belong to conference.

Other possible constraints that are not out of this project scope may exist, but may have not been explicitly stated.

## B2. ABOX Definition

The following describes the methodology used to define the ABOX from non-semantic data.
The Abox.nt code defines an ABOX (Assertional Box) in an ontology from non-semantic data using Java and the Apache Jena library. The ABOX represents the instance-level data that conforms to the ontology's classes, properties, and data properties as implemented in the TBOX.

The methodology involves the following steps:
a) Reading the TBOX (Terminological Box) OWL file: The TBOX contains the ontology's class hierarchy, properties, and data properties. The description of the Tbox is explained in the previous chapter.

b) Creating the OntModel: An OntModel is created using the RDFS_MEM_RDFS_INF reasoning model, which provides inferencing capabilities based on the RDFS semantics. The OntModel combines the TBOX and the ABOX to perform reasoning on the ontology.

c) Defining the classes, properties, and data properties: The code defines the necessary classes, properties, and data properties using the OntModel. These definitions are based on the TBOX's class hierarchy and property definitions.

d) Parsing the non-semantic data: The code reads a created CSV file named *Instance_data.csv* containing the non-semantic data in a specific format. It uses the Apache Commons CSV library to parse the CSV file and retrieve the relevant data for each instance.

e) Creating individuals: For each instance in the CSV file, the code creates individuals of the corresponding class defined in the ontology. It sets the property values using the data properties defined in the ontology.

f) Establishing relationships: The code establishes relationships between individuals using the defined properties. It creates properties and links individuals based on the provided data.

g) Writing the ABOX data: Finally, the code writes the ABOX data to an output file in the N-Triples format, which represents the instance-level data conforming to the ontology.

Furthermore, for the creation of the non-semantic data saved in the instance_data.csv, we generated the following column name attributes using DBLP data from Lab1 and faker library in Python. The final attributes are as follows:
['DOI', 'AuthorName', 'Author-orcid', 'VenueType', 'Title', 'YearPublished', 'PaperType', 'Abstract', 'EditorName', 'ChairName', 'ConfName', 'ConfType', 'JournalName', 'ReviewerName1', 'ReviewerName2', 'ReviewText1', 'ReviewText2', 'ReviewDecision1', 'ReviewDecision2', 'FinalDecision', 'Keywords', 'VolumeNr', 'IndexNr', 'Citation_DOI', 'Area', 'Affiliation', 'YearsReviewerExperience1', 'YearsReviewerExperience2', 'ManagesProductionProcess','ManagesPanelsContent', 'Frequency', 'JournalImpactFactor']

The attributes are in tandem with data property classes and conform to the defined TBOX and ABOX. To generate the CSV, first run the XMLToCSV.ipynb to generate the output_article file, then run preprocess_dblp.ipynb python script to generate the instance_data.csv.

ABOX assumptions and constraints:
When defining an ABOX from non-semantic data, several assumptions and constraints are made. Here are some assumptions and constraints that were considered during the process:

Assumptions:

- The TBOX ontology (OWL file) is already available and provides the necessary class hierarchy, properties, and data properties for the ABOX.
- The non-semantic data is provided in a specific format, i.e CSV file, with defined columns and corresponding values.
- The data provided in the CSV file aligns with the structure and semantics of the TBOX ontology.
- Each instance in the non-semantic data corresponds to a specific class defined in the ontology.
- The relationships and associations between instances can be determined based on the available data.

Constraints:

- The ABOX data is created based on the defined classes, properties, and data properties in the TBOX ontology. Any instance that does not align with the ontology's structure cannot be represented in the ABOX.

- The ABOX data follows the restrictions and cardinality constraints defined by the ontology.
- The ABOX data relies on the provided non-semantic data and does not perform any additional reasoning or inference beyond what is explicitly specified in the TBOX ontology.
- The ABOX data assumes that the provided non-semantic data is accurate and complete. Inaccurate or missing data might result in inconsistencies or incomplete representations in the ABOX.

## B3. Ontology Overview and Statistics

Since the ABOX was done programmatically using Jena, for this the TBOX model was loaded. The library allows the creation of instances / individuals based on the classes defined in the TBOX. It was not necessary to define additional conditions, since the TBOX contained the information of the classes, properties, subclasses, domains and ranges. However, in the creation of the ABOX, inference was used to reduce the number of declarations.

1. Inference rdfs:type via rdfs:subClassOf. This was the case of the subclasses, this is the list of cases where inference was used:

| Inferred Classes | Explicit Classes |
|---|---|
| Paper | :FullPaper<br>:ShortPaper<br>:DemoPaper<br>:Poster |
| Person | :Author<br>:Reviewer<br>:Editor<br>:Chair |
| Venue | :Conference<br>:Journal |
| Conference | :RegularConference<br>:ExpertGroup<br>:Symposium<br>:Workshop |
| Publication | :Volume<br>:Proceeding |

Table 3. Inference data

| | |
|---|---|
| explicit | 128068 |
| inferred | 41386 |
| total | 169454 |

Table 4. Numbers of explicit, inferred and total statements in the repository

Additionally, the following statistics were obtained about the resulting graph. The number of classes, properties and instances. We can observe that in the case of disjoint hierarchies, their subclasses add up to the same number of instances as the superclasses. (Volume 536+ Proceeding 747= Publication 1,283).

| Class | Number of instances |
|---|---:|
| owl:Class | 24 |
| owl:ObjectProperty | 16 |
| owl:DatatypeProperty | 20 |

Table 5. Statistics number of classes.

| Class | Number of instances |
|---|---:|
| :Paper | 2,474 |
| :DemoPaper | 666 |
| :ShortPaper | 743 |
| :Poster | 313 |
| :FullPaper | 752 |
| :Venue | 400 |
| :Conference | 200 |
| :RegularConference | 50 |
| :Workshop | 50 |
| :Symposium | 50 |
| :ExpertGroup | 50 |
| :Journal | 200 |
| :Person | 5,163 |
| :Author | 5,163 |
| :Reviewer | 1,675 |
| :Chair | 910 |
| :Editor | 906 |
| :Review | 4,898 |
| :Keyword | 37 |
| :Area | 7 |
| :Submission | 2,449 |
| :Publication | 1,283 |
| :Proceeding | 747 |
| :Volume | 536 |

| Object property | Times used |
|---|---:|
| :hasauthor | 7,356 |
| :haschair | 1,192 |
| :hascitation | 2,474 |
| :haseditor | 1,278 |
| :haskeywords | 11,432 |
| :haspaperarea | 2,474 |
| :hasreview | 4,898 |
| :hasreviewer | 4,948 |
| :hasvenuearea | 1,669 |
| :ispartof | 774 |
| :ispublished | 1,579 |
| :issubmitted | 2,474 |
| :oversees | 1,281 |
| :partof | 803 |
| :submittedto | 2,474 |
| :supervises | 1,193 |

| Data property | Times Used |
|---|---:|
| :abstract | 2,474 |
| :acceptancedecision | 4,905 |
| :acceptancefinaldecision | 2,454 |
| :affiliation | 7,347 |
| :areaname | 7 |
| :authororcid | 7,023 |
| :doi | 2,474 |
| :frequency | 707 |
| :indexnr | 747 |
| :journalimpactfactor | 1,281 |
| :managespanelscontent | 1,026 |
| :managesproductionprocess | 1,075 |
| :personname | 5,363 |
| :reviewtext | 4,948 |
| :title | 2,474 |
| :venuename | 1,110 |
| :volumenr | 536 |
| :word | 37 |
| :yearpublished | 1,553 |
| :yearsreviewerexperience | 4,301 |
| rdfs:comment | 1 |
| rdfs:label | 61 |

Table 6. Statistics number of instances per class and per Object and Data Property.

## B4. Querying

The queries are in the file named Team12E_B4_OGBUCHI_FORERO_ZIVKOVIC.sqarql, the prefixes used are:

```
PREFIX sdm: <http://www.sdm.lab#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
```

1. **Find all Authors.**

```
SELECT ?author where {

    ?author rdf:type sdm:Author

}
```

2. **Find all properties whose domain is Author.**

```
SELECT ?property where {
        ?property rdfs:domain sdm:Author

}
```

3. **Find all properties whose domain is either Conference or Journal.**

```
SELECT ?property where {

   {?property rdfs:domain sdm:Conference} UNION {?property rdfs:domain sdm:Journal}

}
```

4. **Find all the papers written by a given author that were published in database conferences.**

```
SELECT ?author (GROUP_CONCAT(?paperName; SEPARATOR=", ") AS ?paperNames) ?area
where {

?paper rdf:type sdm:Paper;

   sdm:issubmitted ?submition;

   sdm:hasauthor ?author ;

   sdm:title ?paperName .

?submition rdf:type sdm:Submission;

   sdm:submittedto ?venue.

?venue rdf:type sdm:Conference ;

   sdm:hasvenuearea ?area .

?area rdf:type sdm:Area;

   sdm:areaname "database"

}

GROUP BY ?author ?area
```

This query obtains the name of all the papers (concatenated in a string) that have been published in a conference in the database area, for each of the authors. In other words, the papers that have been submitted (issubmitted) in a conference-type venue (submittedto) are obtained, and that are conferences of the area databases (hasvenuearea).

## References

DBLP data (2023). Accessed from (https://dblp.uni-trier.de/xml/) at 8th May 2023.

Dblp-to-csv. Accessed from github (https://github.com/ThomHurks/dblp-to-csv) at 8th May 2023.

Apache Jena documentation. Accessed from https://jena.apache.org/documentation/ at 8th May 2023.

Gra.fo. Accessed from https://gra.fo/ at 4th May 2023.

GraphDb. Accessed fromhttps://www.ontotext.com/products/graphdb/download at 4th May 2023.

SDM lecture Notes UPC.