Semantic Data Management Project

Simon Coessens
Student BDMA
simon.coessens@estudiantat.upc.edu
Mohamed Louai Bouzaher
Student BDMA
mohamed.louai.bouzaher@estudiantat.upc.edu
Professor : Oscar Romero

June 10, 2024

1 Introduction

In our Joint BDMA project, we are extensively utilizing graph-based solutions to enhance our data management and analysis capabilities. This report is dedicated to exploring the application of Knowledge Graph solutions, examining how they can benefit our project by enabling sophisticated data integration and relationship mapping. Simultaneously, our colleagues within the same team are conducting a parallel investigation into Property Graphs. Their findings will complement ours, providing a comprehensive understanding of how each graph type can be optimally employed to meet our project's objectives.

Having previously explored this topic during our Semantic Data Management course, we will use the knowledge acquired there to advance our current project.

2 Purpose Statement (M1)

For our project on research paper management and reading recommendations, we will use a graph-based solution to efficiently manage and analyze the complex relationships between papers, authors, citations, and research topics. This approach will enable us to create a highly interconnected and dynamic system that facilitates advanced querying, visualization, and recommendation capabilities.

2.1 Key Features

- **Author Exploration:** Users can click on a paper, then on the author, and find all papers by this author. This enables easy navigation and exploration of an author's body of work.
- **Citation Navigation:** Users can view and navigate through citation networks, exploring which papers cite a given paper and which papers are cited by it.
- **Keyword Search:** Users can search for papers based on specific keywords or topics and see how these are connected within the research network.

- **Institutional Affiliation:** Users can explore the affiliations of authors, discovering papers produced by specific institutions or research centers.
- **Collaborator Networks:** Users can view co-authorship networks to identify frequent collaborators and explore the collaborative landscape of researchers.
- **Trend Analysis:** Users can analyze emerging trends and hot topics in research by viewing aggregated citation and publication data over time.
- **Interactive Visualization:** The application will include interactive graph visualizations to help users intuitively explore the research landscape and relationships between entities.

.

3 Graph Family (M2)

As described before we will focus on Knowledge Graphs in this report. In this way we can find out what the advantages are in using Knowledge Graphs for our project.

Some direct advantages of this approach can be listed as follows:

- Because of the standardized format of knowledge graphs, integrating data from diverse sources becomes more streamlined. In this project, we will specifically focus on assimilating data from Semantic Scholar and ArXiv.
- Knowledge graphs excel in depicting complex relationships and facilitating the inclusion of taxonomies within the data. This capability allows for a nuanced representation of hierarchical and associative data relationships, enriching the dataset's structural complexity and analytical value.

A note here is that our current data doesn't fully utilize complex knowledge graph capabilities. However, including these features in the future could be useful.

4 Graph Design (M3)

For our knowledge graph we have the following graph scheme:

We created the TBOX using Gra.fo, shown in Figure 1 and accessible via this link.
Some decisions and assumptions made here:

- We decided to create a **Person** superclass which has **Author** and **Reviewer** under it.
- We assumed that all reviewers are persons who review and author and authors are the people who only author but do not review.
- We decided to create another superclass, **Organization** and added **Journal** and **Conference** as subclasses, then we assumed that both of them could be related to the same generated fields.

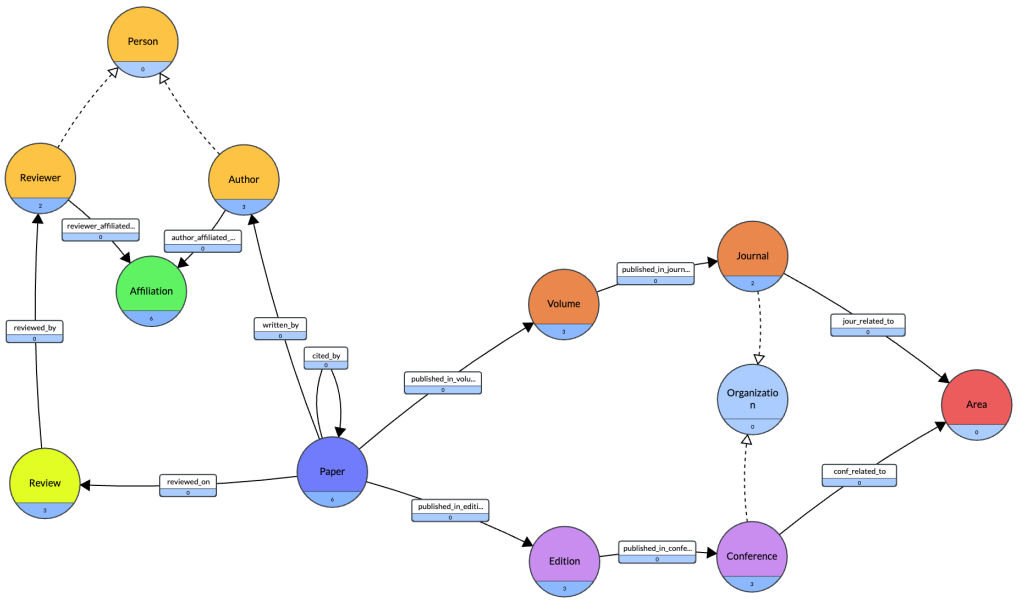

Figure 1: TBOX graphical representation

5 Data Flow (M4)

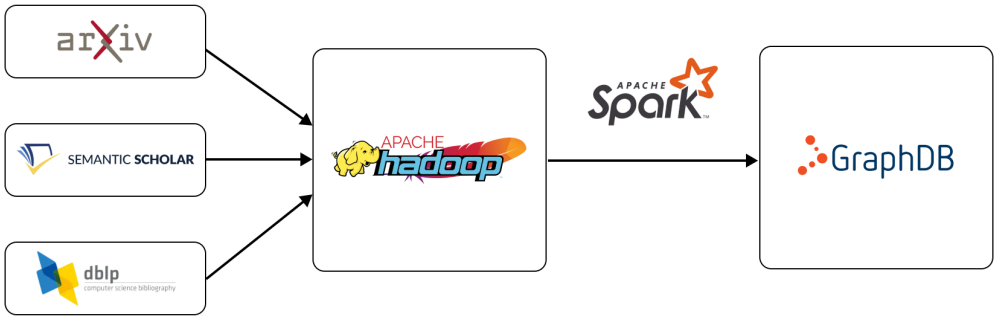

Figure 2: Data Flow

5.1 Data Collection and Ingestion

Semantic Scholar, DBLP, and Arxiv data are collected. This data might include research papers, metadata, author information, citations, abstracts, and other bibliographic details. Data from these sources is ingested into the Hadoop ecosystem for large-scale storage and preliminary processing.

5.2 Data Storage in Hadoop

Apache Hadoop is used to store raw data in a distributed manner. The Hadoop Distributed File System (HDFS) allows for scalable and fault-tolerant storage of the large datasets collected previously.

5.3 Data Processing with Spark

Apache Spark is employed to process and transform the data stored in Hadoop. Spark's powerful data processing capabilities allow for the extraction, transformation, and loading (ETL) processes

necessary to clean, normalize, and structure the data. Specific operations might include parsing bibliographic records, extracting meaningful metadata, and transforming the data into a format suitable for graph database ingestion.

5.4 Populating the GraphDB

The processed data is then used to populate the GraphDB. This involves creating nodes and edges that represent entities such as authors, papers, institutions, and their relationships (e.g., authorship, citations, affiliations).

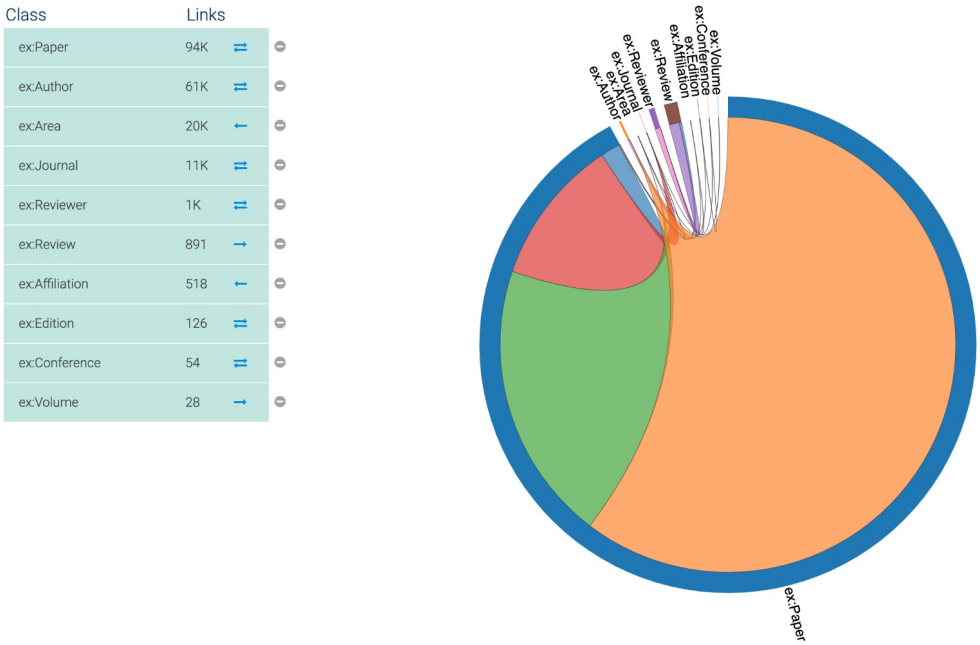

Figure 3: The data ingested in GraphDB

5.5 Generating TBOX and ABOX

1. **TBOX**
   The TBOX defines the schema or ontology for the graph database. This includes defining classes, properties, and relationships. For example:

   - Classes: Author, Paper, Institution
   - Properties: title, abstractm publicationDate...etc.
   - Relationships: Define how entities are related, e.g., an Author writes a Paper, a Paper cites another Paper.

   The code for this file is found under `TBOX.ttl`

2. **ABOX**
   The ABOX consists of individual instances of the entities defined in the TBOX. This involves:

   - Creating nodes for each Author, Paper, Journal..,etc.

- Establishing relationships between these nodes based on the processed data, such as linking an author to their papers, papers to their citations, and authors to their affiliations.

For each of our data sources we have a python script generating the ABOX file. These scripts are `generate_abox_semantic.py` and `generate_abox_arxiv.py`

6 Exploiting the Graph (M5)

6.1 Analytics

We present some analytic queries that we need for our application and give the according `SparQl` queries.

- **Total Number of Papers:** Count all nodes of type Paper.

```
PREFIX ex: <http://www.gra.fo/schema/untitled-ekg#>

SELECT (COUNT(?paper) AS ?totalPapers)
WHERE {
  ?paper a ex:Paper .
}
}
```

Listing 1: Find Papers by Author in a Specific Research Area

The total number of papers is 21,566.

- **Distribution of Papers Across Fields:** Aggregate papers by their Area or related categories.

```
gf: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?AreaNode (COUNT(DISTINCT ?Paper) AS ?NumberOfPapers)
WHERE {
  ?Paper rdf:type gf:Paper .
  ?Paper gf:published_in_edition ?Edition .
  ?Edition gf:published_in_conference ?Conference .
  ?Conference gf:conf_related_to ?AreaNode .
}
GROUP BY ?AreaNode
ORDER BY DESC(?NumberOfPapers)
```

Listing 2: Distribution of Papers Across Fields

AreaNode	NumberOfPapers
gf:Area/NetworkSecurity	78
gf:Area/Databases	71
gf:Area/ArtificialIntelligence	68
gf:Area/MachineLearning	40

Table 1: Distribution of Papers Across Different Areas

- **Distribution of Papers Across Years:** Aggregate papers by publication year.

```
PREFIX ex: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?year (COUNT(?paper) AS ?count)
WHERE {
  ?paper a ex:Paper .
  ?paper ex:paper_year ?year .
}
GROUP BY ?year
ORDER BY ?year
```

Listing 3: Count Papers Across Years

The distribution of papers across years is as follows:

Year	Number of Papers
1978	9
1986	9
1987	18
1988	9
1989	9
1990	9

Table 2: Snippet of distribution of Papers Across Years

- **Author Statistics:** Analyze nodes of type Author to determine the number of papers each author has written, co-author relationships, and collaboration networks.

```
PREFIX ex: <http://www.gra.fo/schema/untitled-ekg#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?author (COUNT(?paper) AS ?numPapers)
WHERE {
  ?paper rdf:type ex:Paper;
         ex:writtenBy ?author.
}
GROUP BY ?author
ORDER BY DESC(?numPapers)
```

Listing 4: SPARQL Query to Count Papers per Author

The results from executing this query are presented in the table below:

Author URI	Number of Papers
ex:Author/Damien_Chablat	53
ex:Author/Philippe_Wenger	42

Table 3: Snippet of number of papers published by each author

These statistics are highly valuable for inclusion in our application.

6.2 Search

The graph serves as the backbone for the search functionality in our application, enabling advanced search capabilities through the following methods:

- **Textual Data Search:** Title and Abstract Matching: Search for target papers based on keywords in the title and abstract. This utilizes full-text search capabilities within the graph database.
- **Relationship-Based Search**
  - Paper/Area Traversal: Traverse the graph to find papers related to specific area, leveraging the `Area` and `published_in` relationships.
  - Author Relationships: Identify papers and researchers connected to a particular author, enabling users to explore the research network and find potential collaborators or influential papers.

6.3 Community Detection and Centrality

In order to identify groups of papers or authors that are densely connected, we can utilize algorithms such as Louvain or Girvan-Newman. These methods can detect communities within the graph, thereby helping to uncover hidden research groups, thematic clusters, and potential collaborations.

Centrality is also a crucial concept in our graph analysis, aimed at identifying the most influential authors within a specific research area. This information is valuable and can be presented to the end user in our application.

After some research on implementing these techniques in `GraphDB` we quickly discovered that it is not as straightforward as with platforms like `Neo4j`. This is primarily because SPARQL is not designed to handle complex graph algorithms, and currently, there is no available module within `GraphDB` that supports such operations. Given that this report is an exploration of Knowledge Graphs for our use case, we did not pursue further research or attempt to implement these techniques.

7 Conclusion

To conclude we present the advantages and disadvantages of Knowledge graphs that we found for our use-case.

Advantages	**Disadvantages**
The standardized format of Knowledge Graphs simplifies the integration of data	It is more complex to set up
We can express complex relationships and taxonomies	We don't have the built in modules that we need to perform complex graph algorithms.

Table 4: Advantages and Disadvantages of Knowledge Graphs