**UNIVERSITAT POLITÈCNICA DE CATALUNYA**

FACULTAT D'INFORMÀTICA DE BARCELONA

# Knowledge Graphs

Semantic Data Management

Spring 2023

Authors:
**Abu Sbeit, Abd Alrhman Mustafa Saleem** , *email*:
abd.alrhman.mustafa.saleem.abu@estudiantat.upc.edu
**Lorencio Abril, Jose Antonio**, *email*: jose.antonio.lorencio@estudiantat.upc.edu

Professor: **Flores, Javier**

# Contents

# List of Figures



# Listings

All the code of the project can be accessed in its Github repository.

# A PreProcessing

We chose to work on the data provided by Semantic Scholar API, in alignment with the recommendation to utilize the same dataset used in SDM Lab 1, to see the pros and cons for both Knowledge Graphs and Property Graphs, we proceeded to export the data from our property graph as CSV files for both nodes and relations. And to ensure data integrity and optimize its suitability for subsequent analysis, we performed a thorough cleaning process, eliminating any unused fields and augmenting the dataset with additional fields. Furthermore, we established connections between the nodes and relations, thereby generating new, refined data files that align with our data model. This process guarantees the availability of high-quality, tailored data for our tasks.

## A.1 Synthetic Data

Since our data didn't contain all the information required for the lab, we decided to add synthetic data for all the missing fields, which are.

1. Paper: our papers didn't contain the information about the paper type, so we did a list with all the required paper types (Full Paper, Short Paper, Demo Paper, and Poster) and using the built in java Random, we been able to randomize the selection of each type to each paper.

2. Venue: our venues didn't contain both information about conference types and periodicity of each venue, so we did a list with all the required conference types and periodicity (Symposium, Workshop, Regular Conference, and Expert Group) and (Weekly, Monthly, Yearly) respectively, and using the built in java Random, we been able to randomize the selection of each conference types and periodicity to each venue.

## A.2 Final Data

After finishing the cleaning of the data and adding the missing field we end up with data in the following format.

1. Paper:

    - id: 4
    - paperTitle: Ultrasound_guided_central_venous_catheter_placement_increases_ success_rates_in_pediatric_patients:a_meta_analysis
    - area: Data Management
    - authorID: 1837,1838
    - corrAuthorID: 1837
    - publicationID: 11327
    - venueId: 11099
    - reviewer_1: Laura_Martín-Francés

- reviewer_2: J___Ceravalls
- review_1: Tree_relate_south_inside_three___Top_amount_him_section_number_series_plant_
- review_2: Public_skill_organization_final_effect_move___Tell_government_ contain_sense_action_keep_state___Mr_theory_at_marriage_TV_wall_sit_employee
- paperType: FullPaper
- paperAbstract: null
- decisions_1: Yes
- decisions_2: Yes
- year: 2016
- url: https://www.semanticscholar.org/paper/15074946c19224991386a1bb319c87b07128f2b9

2. Person:
   - id: 4970
   - name: E___Combe
   - dob: 1973-07-08
   - affiliation: Slovak University of Technology in Bratislava
3. Publications:
   - id: 13194
   - title: 2008_ICON
   - year: 2008
   - publisher: IOSPress
   - chair: Jacob_Williams
   - city: Basel
   - area: Computer Science
   - type: Proceeding
   - venueId: 13160
4. Venues:
   - id: 10878
   - area: Data Management
   - publication: null

- venueType: Journal
- conferenceType: null
- editor: Christopher_Delacruz
- url: https://bmjopen.bmj.com/
- name: BMJ_Open
- chair: null
- issn: 2044-6055
- periodicity: Weekly

# B Ontology Creation

## B.1 TBOX Definition

The TBOX is presented in Figure 1 and the corresponding file is attached to this report as *tbox.owl*, which was created using Jena, with the program called *TBOX.java*. For the creation of the diagram, we have imported this file to gra.fo, where we were able to arrange the different classes for improved readability. For instance, we have coloured all classes with common super class in the same colour. This way, we identify six different groups of classes:

1. Paper: the central concept of the ontology, with the attribute *abstract*. It has subclasses *DemoPaper*, with additional attribute *urlToDemo*, *FullPaper* with additional attribute *additionalRemarks*, *ShortPaper* with additional attribute *isPurposive* and *Poster* with additional attribute *purpose*.

   Before going into the details of the other concepts, let's briefly describe the relationships that Paper maintains with them:

   - hasAuthor: connects a Paper to its Author(s).
   - assignedPaper: connects a review assignation to the Paper to be reviewed.
   - relatedTo: connects a Paper to the Area(s) of research of the Paper.
   - publishedAs: connects a Paper to the mean in which it was published after successful review.
   - submittedTo: connects a Paper to the Venue in which it was submitted for reviewal and publication. This property has two subproperties, *submittedToJournal* and *submittedToConference*, which are not shown in the diagram for simplicity. These are used to model the restriction that posters can only be submitted to conferences, by excluding poster from *submittedToJournal*'s domain. Note all kinds of papers can be submitted to conferences.

2. Area: simple concept that simply model the different areas of research. It has properties *areaName* and *areaDescription*. It is related to other classes via the following property:

- relatedTo: it connects Venue, Paper and Publication with the Area(s) they are related to.

3. Venue: this class represents venues, i.e., where paper are submitted for review and publication. The attributes of a venue are its *venueName* and *venueDescription*. It has two subclasses, *Journal* with attribute *ISSN* and *Conference* with attribute *periodicity*. Conference is further subdivided into *Workshop*, with added attribute *date*, *Symposium*, with added attribute *subject*, *RegularConference* with no added attributes and *ExpertGroup*, with added attribute *numberOfExperts*.

   Venue is related to other concepts through the following properties[1]:

   - submittedTo.
   - relatedTo.
   - includedIn: this property connects a Publication with the Venue which published it. Notice that it has two subproperties, *submittedToJournal*, specific from *Volume* to *Journal*, and *submittedToConference*, specific from *Proceeding* to *Conference*. This eases the modelling by restricting where each type of publication is enabled to be published.

4. Publication: this concepts represents the means of publication of a set of papers. A paper can only be included in a publication after it has received at least two positive reviews. This behavior is easily modelled by adding this class and using the appropriate logic to handle the reviewing of articles. A Publication has two attributes, *publicationDate* and *publicationWebsite*. It can be a *Volume*, with added attribute *numberOfPapersInVolume*, and *Proceeding*, with added attributes *numberOfPapersInProceeding* and *heldIn*.

   Publication is related to other concepts via the following properties:

   - publishedAs.
   - relatedTo.
   - includedIn, includedInJournal, includedInConference.

5. Person: a generic class encapsulating the different roles that people can play in the ontology. It has attribute *Name*, and a person can be an *Author*, with attribute *affiliation*, a *Reviewer*, with attribute *specialization* and a *Manager*, with attribute *yearsOfExperience*. A Manager can also be either an *Editor*, with attribute *editorUntil*, or a *Chair*, with attribute *chairUntil*.

   Person is related to other concepts through the following properties:

   - hasAuthor.
   - reviewWrittenBy: connects a review to the reviewer that is assigned to do it.
   - assignedBy: connects a review to the manager that assigned it to a reviewer.
   - isManagerOf: connects a manager to the venue that he/she manages. This property

[1] Note that we don't repeat the explanation of previously explained properties.

has two subproperties, namely *isChairOf*, connecting *Chair* and *Conference*, and *isEditorOf*, connecting *Editor* and *Journal*.

6. Review: this concept represents a review of a Paper, done by a Reviewer, and assigned by a Manager. The attributes of this class are *decision* and *reviewText*. Reviews are used to impose the logic that a paper needs to receive at least two approvals before publication.

   Review is connected to other concepts through the properties:

   - assignedBy.
   - reviewWrittenBy.
   - assignedPaper.

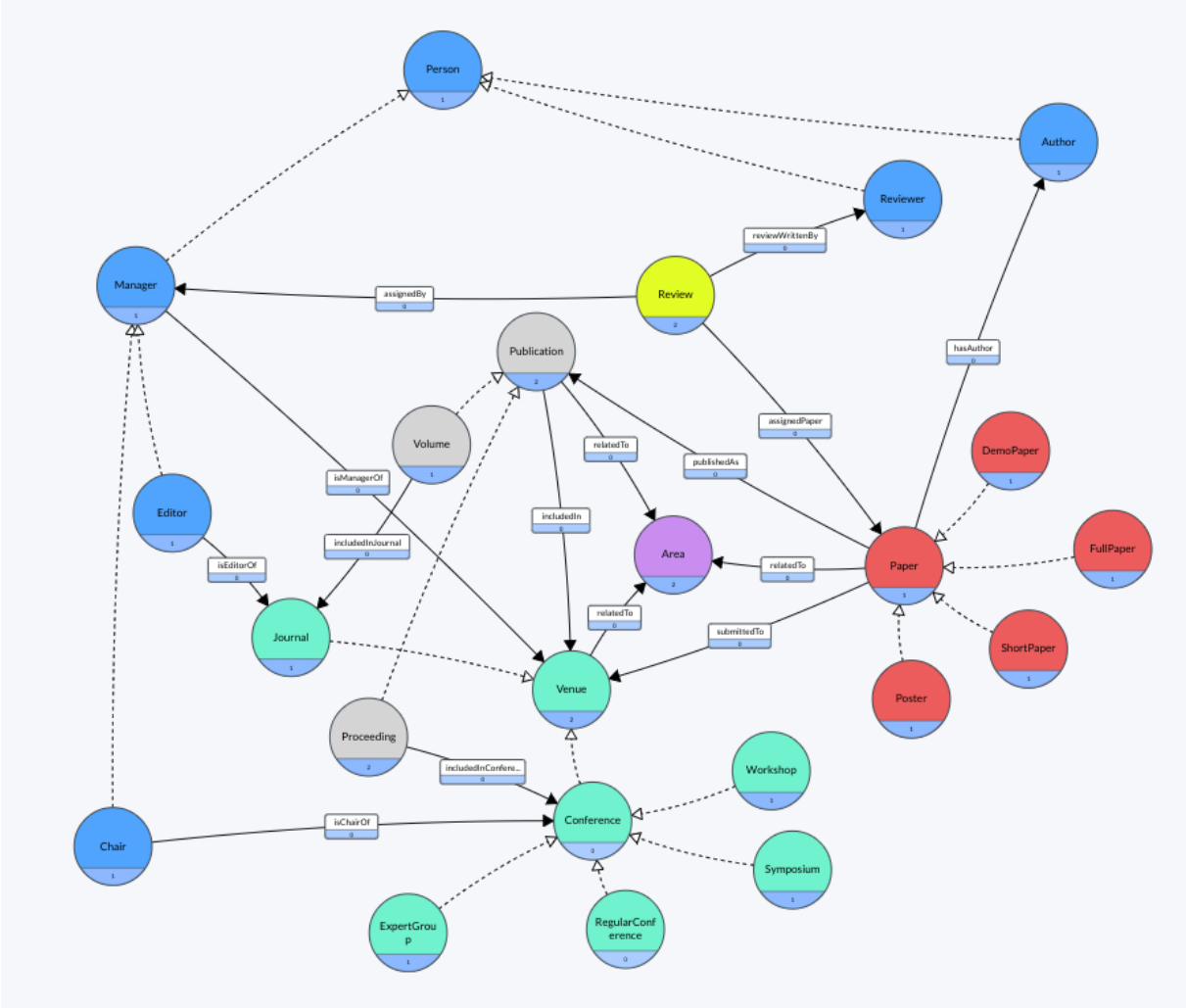


Figure 1: TBOX.

## B.2 ABOX Definition

By Using Jena, we successfully been able to create and implement the ABOX. Our process involved loading the ontology (TBOX) into the ABOX.java class and parsing the cleaned data from the CSV files (cleaned_paper.csv, cleaned_persons.csv, cleaned_publications.csv, and

cleaned_venues.csv). Subsequently, we instantiated each atomic concept and established connections within the ontology to ensure a comprehensive representation. This approach allowed us to effectively integrate the data into the ontology, enabling seamless utilization for further analysis and inference.

With OWL we have been able to load the saved ontology model and getting all the classes, subclasses, properties, and sub-properties. Then we read and parsed the cleaned data from the CSV files into HashMaps to access it easily and directly with constant time. Since the data doesn't contain all the data we needed for the properties and sub-properties, we needed to replace the missing data with synthetic data we created when filling the property such as additionalRemarks for FullPaper class, isPurposive for ShortPaper class, purpose for Poster, etc.

After reading and parsing the data into HashMaps, we created Indevidual for each class and property from the ontology using OntClass, and created the data properties and linked it using OntProperty and createTypedLiteral. Then, we outstream and created the abox.nt file with output language as N-Triple.

Note that the followed approach automatically links the ABOX to the TBOX, so that for Exercise B.3 we will only need to import the data into GraphDB.

## B.3 Create the Final Ontology

Following the recommended approach, we configured the Inference and Validation rule set to RDFS-PLUS (Optimized) to facilitate effective inference, as shown in Figure 2. This configuration enables the system to apply RDFS reasoning, allowing for enhanced data integration and inference capabilities. By using this optimized rule set, we can derive implicit knowledge from the existing ontology, resulting in a more comprehensive and meaningful representation of the data.

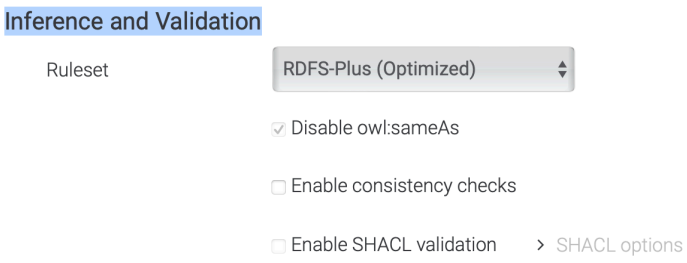


Figure 2: Inference.

Then we uploaded the abox.nt and the tbox.owl files in imported them using the IRI and Graph Name shown in Figure 3, which lead to successfully importing the data within seconds. The imported files should look as in Figure 4.

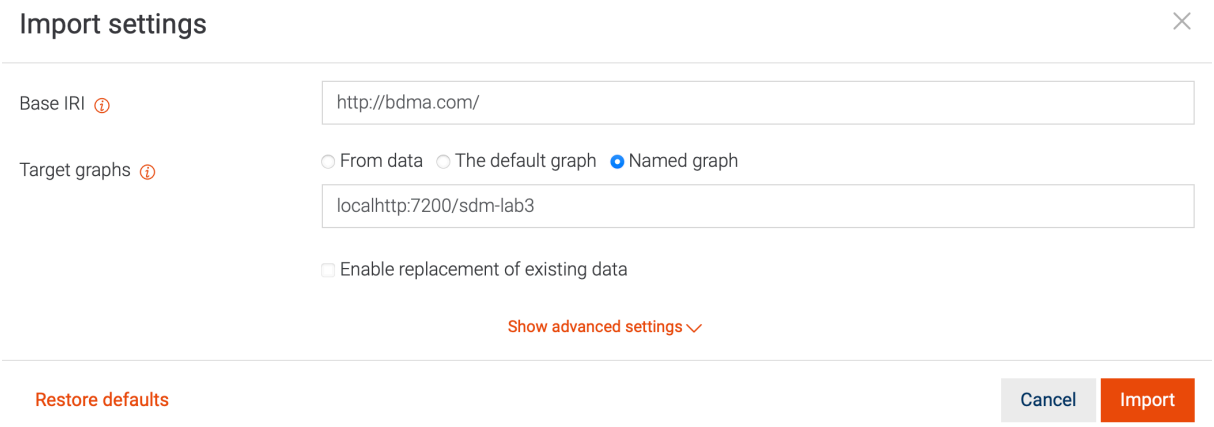


Figure 3: import settings.

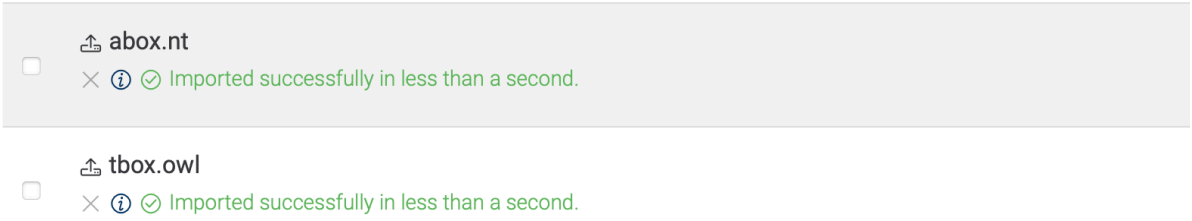


Figure 4: import.

With inference ratio of 22%, shown in Figure 5, and class relationships diagram as depicted in Figures 6 and 7.

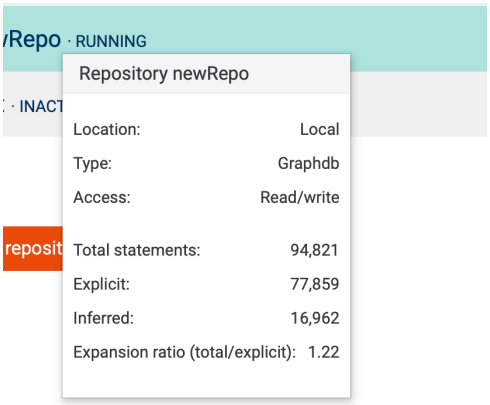


Figure 5: inference ratio.

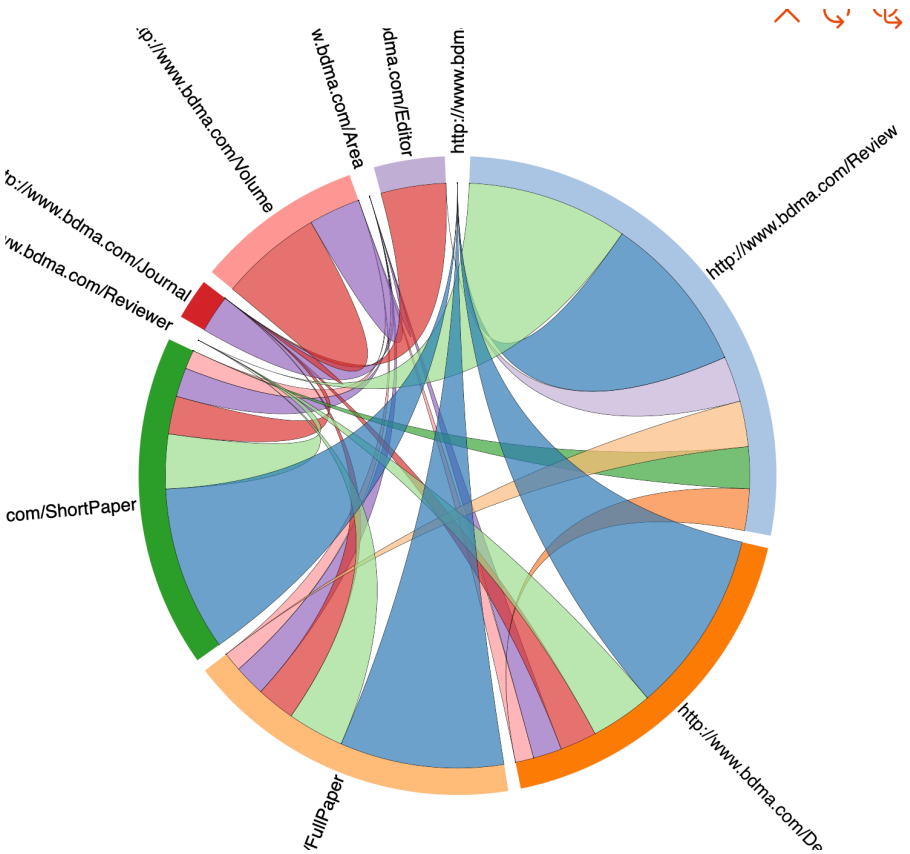


Figure 6: class relationships.

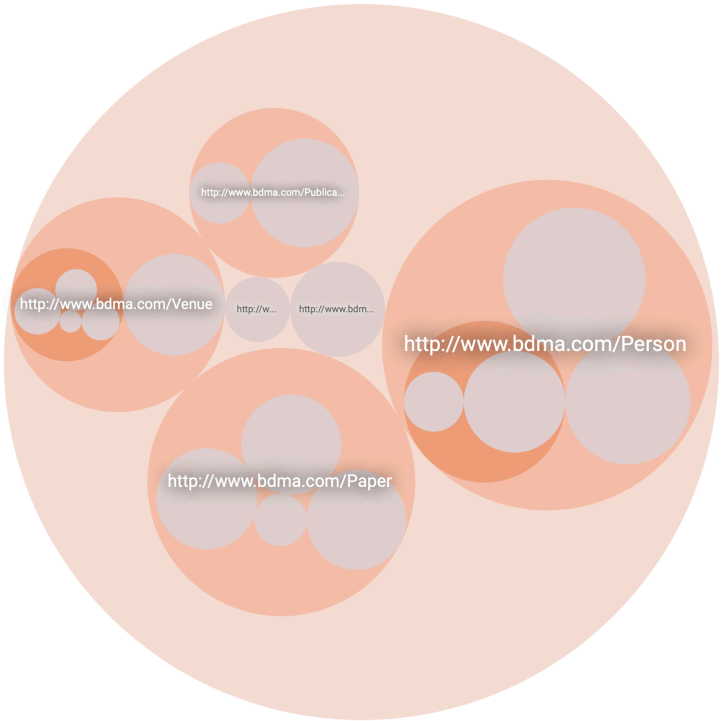


Figure 7: class hierarchy.

## B.4 Querying the Ontology

In Listing 1 we show the first query, which finds all authors in the database. The query is pretty straightforward: we select all distinct instances declared of type Author. Note also the prefixes defined, which will be a part of all of our queries. A partial result of the query is shown in Figure 8.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://www.bdma.com/>

SELECT DISTINCT ?author
WHERE {
  ?author rdf:type :Author .
}
```

Listing 1: Query 1.


Figure 8: Partial result of query 1.

In listing 2 we can see the second query, where we retrieve all properties with Author in their domain. According to the TBOX[2], there are no object properties with this characteristic, and only the data property *affiliation* has Author in its domain. This is indeed the result of the query, as can be seen in Figure 9.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://www.bdma.com/>

SELECT DISTINCT ?property
WHERE {
  ?property rdfs:domain :Author .
}
```

Listing 2: Query 2.


Figure 9: Result of query 2.

The third query is really similar to the second one, as it asks to find all properties with Conference or Journal in their domain. We can do this easily with the `UNION` clause, as in Listing 3. The

[2]See Section B.1.

expected result is, following the TBOX definition, just their corresponding data properties, since they are not in the domain of any object property. These properties are *periodicity* from Conference and *ISSN* from journal. The result is shown in Figure 10.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://www.bdma.com/>

SELECT DISTINCT ?property
WHERE {
  {
    ?property rdfs:domain :Conference .
  } UNION {
    ?property rdfs:domain :Journal .
  }
}
```

Listing 3: Query 3.


Figure 10: Result of query 3.

Finally, query 4 asks to find all papers written by a given author that were published in database conferences. In our case, this maps to papers that were published in a proceeding included in a conference which is related to the area with *areaName* 'Databases'. In our data, this does not happen (all papers related to 'Databases' were published in journals), but we show the query for the area 'Computer Science'. Note that the query is exactly the same. The code is as in Listing 4, where {authorID} has to be replaced with the ID of the author whose papers we want to find. In Figure 11, we can see the result when we instantiate the {authorID} with value `John_Daniel_Bossér`.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://www.bdma.com/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT DISTINCT ?paper
WHERE {
  ?paper :hasAuthor :{authorID} .
  ?paper :publishedAs ?proceeding .
  ?proceeding :includedInConference ?conference .
  ?conference :relatedTo ?area .
  ?area :areaName "Computer Science"^^xsd:string .
}
```

Listing 4: Query 4.

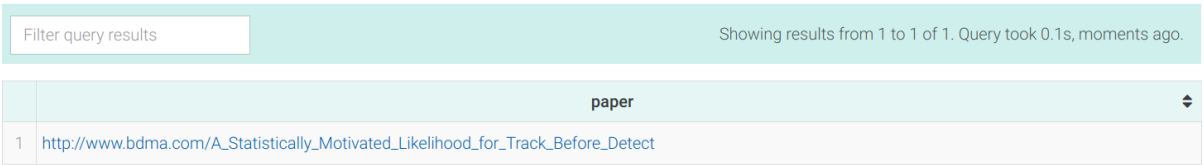


Figure 11: Result of query 4 for author `John_Daniel_Bossér`.