Property Graph Lab: Session 1

SDM : Semantic Data Management

Djordjije Krivokapić, Karim Maatouk

March 18, 2020

Dependencies & Instructions to load the graph

Please refer to our **Github repository** or README file for more information on dependencies & how to start the project.

A. Modeling, Loading, Evolving

A.1 Modeling

The picture below represents the proposed solution for the domain in the task. Two main criteria for deriving such a model are the optimization of queries in part B, and correct semantic modeling of the domain.

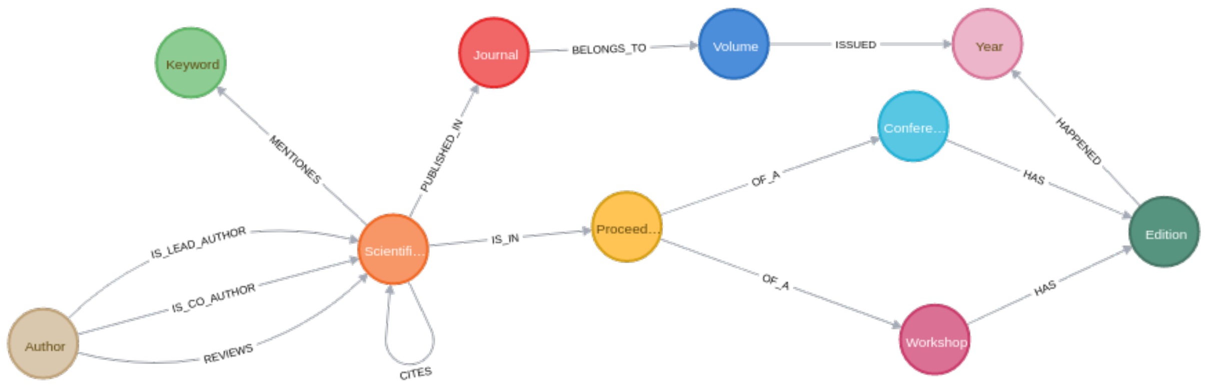


Figure 1: Graph Model

"A paper can be written by many authors, however only one of them acts as corresponding author... When a paper is submitted to a conference or a journal, the conference chair or the journal editor assigns a set of reviewers (typically three) to each paper. Reviewers are scientists and therefore they are relevant authors (i.e., published many papers in relevant conferences or journals)."

From these requirements we derived three relationships between Author(s) and Scientific-Paper(s):

- (Author)$\xrightarrow{[:IS\_LEAD\_AUTHOR]}$ of a (ScientificPaper)
- (Author)$\xrightarrow{[:IS\_CO\_AUTHOR]}$ of a (ScientificPaper)
- (Author)$\xrightarrow{[:REVIEWS]}$ other (ScientificPaper) (in this phase you can track for whom they did review by further following edges from scientific paper to either journal or proceeding of a conference or a workshop)

This modeling enables us to quick determine author's connection with the scientific paper and avoids additional query overhead in terms of filtering.

"A paper can be cited by another paper (meaning their content is related). A paper relates to one or more topics through the concept of keywords. Keywords are used by readers to quickly identify the main topics discussed in the paper."

So, a scientific paper is related to another paper by means of citations. This is a perfect candidate for a recursive relationship. In addition, the paper mentions certain keywords that we extracted as a separate vertex because of reusability (many other papers will mention the same keyword) and readability (semantic).

- (ScientificPaper)$\xrightarrow{[:CITES]}$ other (ScientificPaper)
- (ScientificPaper)$\xrightarrow{[:MENTIONS]}$ some (Keyword)

"In this domain, authors write research articles that can be published as scientific papers (papers for short) in the proceedings of a conference/workshop (a conference is a well-established forum while a workshop is typically associated to new trends still being explored), or in a journal. A proceeding is a published record which includes all the papers presented in the conference/workshop. A conference/workshop is organized in terms of editions. Each edition of a conference is held in a given city (venue) at a specific period of time of a given year. Oppositely, journals do not hold joint meeting events and, like a magazine, a journal publishes accepted papers in terms of volumes. There can be various volumes of a journal per year."

From this description, we derived two paths for a scientific paper:

- (ScientificPaper) is $\xrightarrow{[:PUBLISHED\_IN]}$ a (Journal) which $\xrightarrow{[:BELONGS\_TO]}$ (Volume) which is further $\xrightarrow{[:ISSUED]}$ in a certain (Year).
- (ScientificPaper) $\xrightarrow{[:IS\_IN]}$ a Proceeding $\xrightarrow{[:OF\_A]}$ (Conference or Workshop) that $\xrightarrow{[:HAS]}$ (Edition) that $\xrightarrow{[:HAPPENED]}$ in a certain (Year).

From the description we extracted all relevant entities and relationships between them. Even though most of the queries do not touch certain nodes (volumes, proceeding etc.) with this design we denormalized data and avoided filtering in order to find some simple information (e.g finding all volumes of a journal in certain year).

A.2 Instantiating/Loading: Design Choices & Methodology:

Extracting journal papers:
For extracting journal papers we used output_article.csv file. From this file we are deriving new CSV file with header: author, title, pages, key, url, journal, volume, year.
We are parsing the initial file line by line and when we encounter journal/* in a key attribute then we know that that row corresponds to the article that is published in journal. Rest of the attributes is trivially extracted. Additionally, we needed to add some extra attributes in order to completely align with task requirements, in this regard we tried to be minimalistic and we avoided randomizing the data except when that was unavoidable.
So, in post-processing step, we are making a distinction between lead author and co-authors. Then using *textblob* library we derived the keywords based on the paper's title. Using *lorem* library we generated random text for paper's abstract.
Finally, we are doing post data cleaning to make sure that we are not missing key data features.
Extracting conference papers: For extracting confernece papers we used two files: output_inproceeings.csv and output_proceedings.csv. The key part of extracting is joining those

two files based on the **crossref** attribute. After joining, we basically have extended information about every paper like conference or workshop where it was published, information about inproceeding, authors, edition of a conference, city, country, part of the year etc. After this step we are applying post-processing steps similar to ones we already explained above.
Generating reviews: every pair (Author)$\xrightarrow{[:REVIEWS]}$ (ScientificPaper) we are generating comment, final decision and additional information about author like university or company where he/she works in order to evolve our graph.

A.3 Evolving the graph

In this part we are using reviews.csv generated in the previous step. Finally, evolving the graph we obtain:

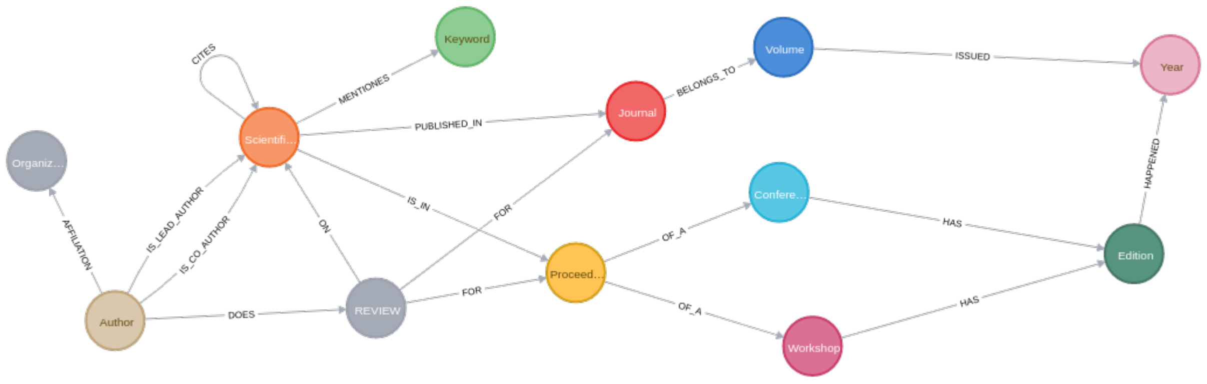


Figure 2: Evolved Graph Model

We destroyed(Author)$\xrightarrow{[:REVIEWS]}$ (ScientificPaper) relationship and evolved it to (Author)$\xrightarrow{[:DOES]}$(Review)$\xrightarrow{[:ON]}$ (ScientificPaper)$\xrightarrow{[:FOR]}$ (Journal or Inproceeding) $\xrightarrow{[:OF\_A]}$(conference or workshop).
Additionally, we generated relationship between (Author) is $\xrightarrow{[:AFFILIATED]}$ is to an (Organization) and this edge contains information whether organization is company or university.

B. Querying

1. Find the h-indexes of the authors in your graph

```
MATCH (author:Author)-[:IS_LEAD_AUTHOR|:IS_CO_AUTHOR]->(scientificPaper:ScientificPaper)
<-[:CITES]-(citatingPaper:ScientificPaper)
WITH author, scientificPaper, count(citatingPaper) as numberOfCitations
ORDER BY author, numberOfCitations DESC
WITH author, collect(numberOfCitations) as orderedCitations
UNWIND range(0, size(orderedCitations)-1) as arrayIndex
WITH author, arrayIndex as key, orderedCitations[arrayIndex] as value, size(
orderedCitations) as arrayLength
WITH
CASE
WHEN key > value THEN key-1
ELSE arrayLength END AS result, author
RETURN author, min(result) as hindex
```

2. *Find the top 3 most cited papers of each conference.:*

```
MATCH (scientificPaper:ScientificPaper)-[:IS_IN]->(:Proceeding)-[:OF_A]->(conference:
Conference),
(scientificPaper)<-[:CITES]-(citingPaper:ScientificPaper)
WITH conference, scientificPaper, count(citingPaper) AS numberOfCitations
ORDER BY conference, numberOfCitations DESC
WITH conference, collect(scientificPaper) as allPapers, collect(numberOfCitations) as
allCitations
WITH conference, allPapers[0..3] as a, allCitations[0..3] as b
UNWIND(range(0, 2)) as indexKey
RETURN conference, a[indexKey] as paper, b[indexKey] as numOfCitations
```

3. *For each conference find its community*

```
MATCH (author:Author)--(scientificPaper:ScientificPaper)-[:IS_IN]->(:Proceeding)-[:OF_A]->(
conference:Conference),(conference)-[:HAS]->(edition:Edition)
WITH author, conference, count(DISTINCT edition) as appearance
ORDER BY appearance DESC
WHERE appearance > 4
RETURN author, conference, appearance
```

4. *Find the impact factors of the journals in your graph*

```
MATCH (scientificPaper:ScientificPaper)-[:PUBLISHED_IN]->(journal:Journal)-[:BELONGS_TO
]->(:Volume)-[:ISSUED]->(year:Year), (scientificPaper)<-[:CITES]-(citingPaper:
ScientificPaper)
WHERE year.year = "2018.0" OR year.year = "2019.0"
WITH scientificPaper, journal, COUNT(citingPaper) as citations
WITH count(scientificPaper) as totalPublications, journal, sum(citations) as totalCitations
RETURN journal, toFloat(totalCitations / totalPublications) as impactfactor
```

C. Graph algorithms

Algorithm 1 : PageRank

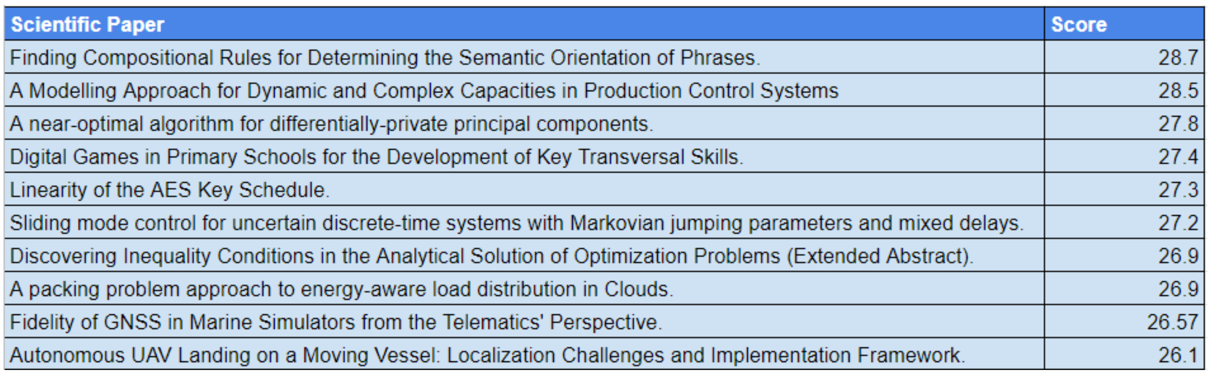
The first algorithm we ran was PageRank algoritm, one of the centrality algorithms. We chose the node **ScientificPaper** and the relationship **CITES** to simulate the algorithm on.

It is important to note some of the algorithms configuration :

- iterations: 20, nbr of runs of the algorithm

- direction : **’INCOMING’**, relationship direction

The number of iterations is set to 20 to have robust results. The direction of the edge of interest ’CITES’ is set to ’INCOMING’ because the paper being cited is more important in the research community than if the paper is citing other sources.
Running the algorithm and returning 10 papers with the highest score assigned by PageRank, the results look as follows :



Scientific Paper	Score
Finding Compositional Rules for Determining the Semantic Orientation of Phrases.	28.7
A Modelling Approach for Dynamic and Complex Capacities in Production Control Systems	28.5
A near-optimal algorithm for differentially-private principal components.	27.8
Digital Games in Primary Schools for the Development of Key Transversal Skills.	27.4
Linearity of the AES Key Schedule.	27.3
Sliding mode control for uncertain discrete-time systems with Markovian jumping parameters and mixed delays.	27.2
Discovering Inequality Conditions in the Analytical Solution of Optimization Problems (Extended Abstract).	26.9
A packing problem approach to energy-aware load distribution in Clouds.	26.9
Fidelity of GNSS in Marine Simulators from the Telematics' Perspective.	26.57
Autonomous UAV Landing on a Moving Vessel: Localization Challenges and Implementation Framework.	26.1

Figure 3: Page Rank Results

Therefore, observing the results, we can identify the most reputable papers in the community, i.e the ones being cited the most by other scientific papers in the research community. This is actually important because the worth of the paper is determined by how many papers in the research community cite that paper, which is how PageRank calculates the score by the frequency it hits a certain paper which in turn depends on how many papers cite that paper. We can go a further step and use the results for suggesting papers for certain topics. For instance, if someone searches for ’Semantic Orientation of Phrases’, which paper should the search engine suggest given the abundance of research paper on the topic ? The search engine should list the scientific papers related to that in order of their PageRank score because they are more likely to be more significant on that topic between other same theme papers. Therefore, in the current Graph DB , the suggestion would be the paper ’Finding Compositional Rules for Determining the Semantic Orientation of Phrases.’

Algorithm 2 : Louvain

The second algorithm of choice was Louvain, one of the community detection algorithms. We chose the node ScientificPaper and the relationship CITES to simulate the algorithm on. We specify **direction : ’BOTH’** for the relationship ’CITES’ since both directions are important in detecting citation communities. Therefore, our aim from running Louvain is to detect communities of scientific papers that heavily cite each other significantly more respect to other papers in other communities.
The aglorithm as is gives respective community of a paper. Therefore we extend it, and group by communities in Cypher query and return the 5 largest communities with the community size and the members, i,e papers, belonging to that community. We obtain 5 largest communities with varying sizes 1352, 1286, 1282, 1212, 1154 scientific papers in each. To that extent, the data means these papers are actively involved in citations within that community. However, other further usages of those communities includes finding the underlying reasons and the

commonalities between those papers such as keywords, authors. Another use case of the data is suggestions of reads, i.e a user reads a paper that belongs to community A, we can suggest that they read another paper that belongs to the same community as they might be of interest. Moreover, the communities obtained from Louvain can be combined with the PageRank top paper to identify top papers within each community to identify best suggestions for a researcher.

D. Recommender

1. Stage one

```
MATCH (sp:ScientificPaper)-[:MENTIONES]->(keyw:Keyword)
WHERE keyw.name = "data management" or keyw.name = "indexing" or keyw.name = "data modeling
" or keyw.name = "big data" or keyw.name = "data processing" or keyw.name = "data storage"
or keyw.name = "data querying"
SET sp.topic = "Databases"
```

2. Stage two

```
call apoc.cypher.run('
MATCH (scientificPaper:ScientificPaper)-[:IS_IN]->(proceeding:Proceeding)
WITH proceeding, COUNT(DISTINCT scientificPaper) AS totalPapers
MATCH (scientificPaper:ScientificPaper {topic: "Databases"})-[:IS_IN]->(proceeding)
WITH proceeding, totalPapers, COUNT(DISTINCT scientificPaper) as databasePapers
WHERE toFloat(databasePapers) / totalPapers > 0.9
RETURN proceeding AS JournalWorkshopConference
UNION
MATCH (scientificPaper:ScientificPaper)-[:PUBLISHED_IN]->(journal:Journal)
WITH journal, COUNT(DISTINCT scientificPaper) AS totalPapers
MATCH (scientificPaper:ScientificPaper {topic: "Databases"})-[:PUBLISHED_IN]->(journal)
WITH journal, totalPapers, COUNT(DISTINCT scientificPaper) as databasePapers
WHERE toFloat(databasePapers) / totalPapers > 0.09
RETURN journal AS JournalWorkshopConference', {}) yield value
WITH value.JournalWorkshopConference as jwc
MATCH (sp:ScientificPaper)--(jwc)
WITH DISTINCT jwc
SET jwc.community = "Database"
```

3. Stage three

```
CALL algo.pageRank.stream('ScientificPaper', 'CITES', {iterations:20})
YIELD nodeId, score
CALL apoc.cypher.run('
MATCH (scientificPaper:ScientificPaper {topic: "Databases"})-[:IS_IN]->(proceeding:
Proceeding {community:"Database"})
RETURN scientificPaper, proceeding as jwc
UNION
MATCH (scientificPaper:ScientificPaper {topic: "Databases"})-[:PUBLISHED_IN]->(journal:
Journal {community: "Database"})
RETURN scientificPaper, journal as jwc', {}) yield value
WITH value.scientificPaper as paper, value.jwc as jwc, score
WHERE id(paper) = nodeId
WITH paper, jwc, score
LIMIT 100
RETURN paper
```

4. Stage four

```
MATCH (author:Author)-[:IS_LEAD_AUTHOR|:IS_CO_AUTHOR]->(sp:ScientificPaper {Top100: "YES"})
WITH author, COUNT(sp) as total
WHERE total >=2
RETURN author
UNION
MATCH (author:Author)-[:IS_LEAD_AUTHOR|:IS_CO_AUTHOR]->(sp:ScientificPaper {Top100: "YES"})
RETURN DISTINCT author
```

Submission Notes:

In the submission you will find the following files :

- PartA.2_KrivokapicMaatouk : Instantiate/load
- PartA.3_KrivokapicMaatouk : Evolve Graph
- PartB_KrivokapicMaatouk : Queries
- PartC_KrivokapicMaatouk.py : Algorithms
- PartD_KrivokapicMaatouk: Recommender
- Queries.txt : Containing all the Cypher Queries