LAB 1: Property Graphs

Semantic Data Management

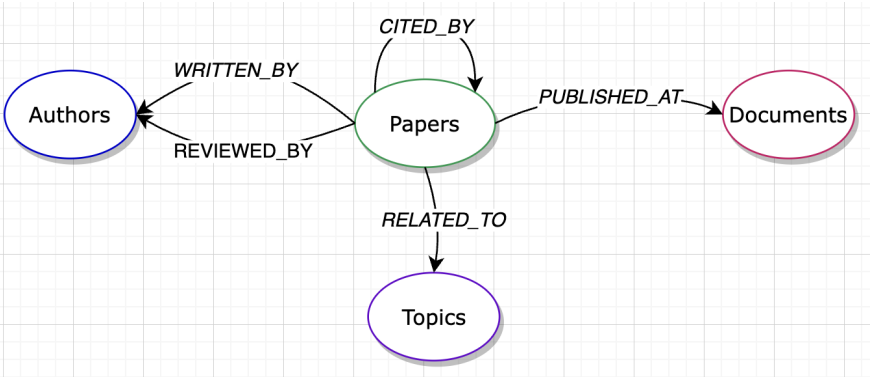
Luis Alfredo León Villapún
Liliia Aliakberova

Section A. Modeling, Loading, Evolving

A.1 Modeling

A.1.1 Create a visual representation of the graph you would create. Use different colours to distinguish data (instances) from metadata (schema) in your graph.

The schema of the graph for the lab consists of 5 nodes and 7 edges. The visual representation of the graph is given below.



A.1.2 Also, justify your design decisions. Besides maintenance or reusability issues, also consider the performance of queries in Part B when creating your solution.

Nodes and and corresponding attributes:

1. Authors. The node represents all information about authors who wrote at least one scientific paper (article). The node includes the AuthorName attribute.
2. Papers. The node consists of the scientific articles that were written by authors and subsequently were published at journals or presented at conferences/workshops. Presented below Title, Abstract, Content (url to body of the paper), Year attributes constitute the node.
3. Documents. The node consists of a proceeding of a conference/workshop or journals that were published. DocumentType, Title, ConferenceName (NULL for journals), Volume (applied for journal and conference), Year attributes constitute the node:
4. Topics. The node represents the information about a keyword that was used in one scientific paper (article). There may be multiple keywords for an article. A keyword in our case is a topic of a paper, so the node was also named topics. The node includes Keyword attribute:

Edges and corresponding attributes:

1. Written_by. Current edge connects “Authors” and “Papers” nodes and represents an author who wrote a specific scientific article. There might be multiple authors connected to one corresponding paper. Reviewed by. The edge connects “Authors” and “Papers” nodes and represents an author who reviewed a specific scientific article. The limitation for the current edge is that the authors can not review their own articles.
2. Cited_by. Current edge connects different “Papers” nodes. The limitation for the “Cited by” edge is that the paper can not cite itself.
3. Published_at. The current edge “Published at” edge connects “Papers” and describes the relation of the papers that were published in a journal or proceeding.
4. Related_to. The “related to” edge connects “Papers” and “Topics” describing which paper is related to specific keywords (topics).

5. Reviewed_by. The current edge “Reviewed_by” connects “Authors” and “Papers” and describes the relation of the papers that were reviewed by authors different from the paper’s writer.

There are two main constraints that were implemented:

1. The authors can not review their own articles
2. The paper cannot cite itself

Assumptions:

During the implementation, we assumed that the process of assigning a set of reviewers to each paper by conference chair or the journal editor is not documented in the database. The reviewing is presented via the “Reviewed by” edge in our project.

A.2 Instantiating / Loading

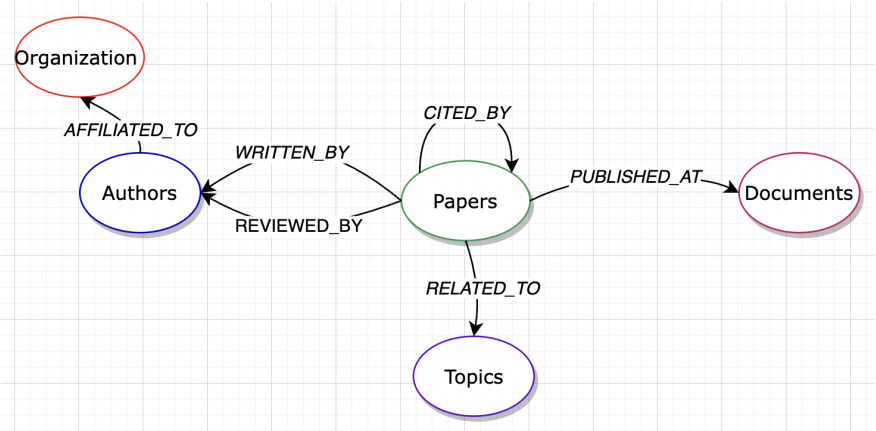
The code used for instantiating and loading this part of the lab, as well as the description of the steps performed can be found in the PartA.2_LeonAliakberova script.

A.3 Evolving the graph

Task A.3.1

Changes:

1. Evolve the edge “Reviewed_by” by adding attributes “Content” and “Suggested decision”. This makes sense because we can make use of the edge to save some content, without needing to spare more space creating a “Review” node, for example. This will also help when creating the queries to see if a paper was approved for revision, as long as the number of edges with “Suggested decision” as “Approve” are bigger than the number of required reviews by a journal.
2. Add a node of type “Organization” and connect it with the node “Authors” via “Affiliated_to” edge. The edge will contain a “type” attribute to indicate if the organization is a company or a university. This allows us to keep the general structure of the graph, and also keeps the flexibility of adding organizations that have different attributes and characteristics, even though they all belong to the same class of node.



Task A.3.2 and A.3.3

The code used to evolve the graph as well as the description of the steps performed can be found in the PartA.3_LeonAliakberova script.

Section B. Querying

1. Find the top 3 most cited papers of each conference.

```
MATCH (p:Paper)←[c:CITED_BY]-(a:Paper)-[r:PUBLISHED_AT]→(d:Document)
        WHERE d.DocumentType ="Conference"
        WITH d.ConferenceName AS name, a, count(c) AS citations
        ORDER BY name, citations DESC
        WITH name, collect({paper: a.Title, cited: citations}) AS papers
        RETURN name AS Conference, [p IN papers[..3] | p.paper] AS Papers, [p IN
papers[..3] | p.cited] AS Cited
```

"Conference"	"Papers"	"Cited"
"2018 International Conference on High Performance Computing and Simulation, HPCS"	["FleCSPH: A parallel and distributed smoothed particle hydrodynamics framework based on FleCSI"]	[8]
"AAAI Workshop"	["Semantic style creation"]	[3]
"ACM-BCB-International Conference on Bioinformatics, Computational Biology, and Health Informatics"	["Feedback regulation of immune response to maximum exercise in Gulf war illness","The PepSeq Pipeline: Software for Antimicrobial Motif Discovery in Randomly-Generated Peptide Libraries"]	[9,8]
"ACM/IEEE International Conference on Human-Robot Interaction"	["Haptic Shape-Based Management of Robot Teams in Cordon and Patrol","Design and Evaluation of Adverb Palette: A GUI for Selecting Tradeoffs in Multi-objective Optimization Problems","Swarm Transparency"]	[6,6,5]
"AIAA Aerospace Sciences Meeting"	["Analysis of distortion transfer and generation through a compressor using the harmonic balance	[6,6,3]

2. For each conference find its community: i.e., those authors that have published papers on that conference in, at least, 4 different editions.

```
MATCH (a:Author)←[w:WRITTEN_BY]-(p:Paper)-[r:PRESENTED_AT]→(e:Event)
WITH a,e.ConferenceName as Conference_Name, count(DISTINCT e.Edition) as w
WHERE w > 3
RETURN Conference_Name, collect(a.AuthorName) as Community_member
```

"Conference_Name"	"Community_member"
"AIAA/CEAS Aeroacoustics Conference"	[" Neilsen T.B.","James M.M.","Wall A.T.","Gee K.L."]
"ASME Design Engineering Technical Conference"	[" Mattson C.A.","Magleby S.P.","Howell L.L."]
"ASME Turbo Expo"	[" Gorrell S.E."]
"Fall Technical Meeting of the Western States Section of the Combustion Institute, WSSCI"	[" Fletcher T.H."]
"Geotechnical Special Publication"	["Rollins K.M.","Franke K.W."]
"IEEE Photonics Conference, IPC"	[" Schmidt H.","Hawkins A.R."]
"International Conference on Engineering and Product Design Education"	[" Howell B."]
"International Telemetering Conference"	[" Rice M.","Saquib M.","Afran M.S.","Rice M."]
"Optics InfoBase Conference"	[" Hawkins A.R.","Schmidt H."]
"SPIE - The International Society for Optical Engineering"	[" Smalley D."]

3. Find the impact factors of the journals in your graph (see https://en.wikipedia.org/wiki/Impact_factor, for the definition of the impact factor).

```
neo4j$ MATCH (d:Document)←[w:PUBLISHED_AT]-(p:Paper)-[b:CITED_BY]→(p1:Paper) WHERE d.…
```

	Journal	Citation_Year	Impact_factor
1	"AIAA Journal"	2020	1.4
2	"Chinese Journal of Mechanical Engineering (English Edition)"	2020	2.0
3	"Fire Safety Journal"	2020	1.0
4	"IEEE Journal of Biomedical and Health Informatics"	2020	1.0
5	"IEEE Journal of Oceanic Engineering"	2020	2.0
6	"IEEE Journal of Quantum Electronics"	2020	1.0

4. Find the h-indexes of the authors in your graph (see https://en.wikipedia.org/ wiki/H-index, for a definition of the h-index metric).

```
MATCH (a:Author)←[w:WRITTEN_BY]-(p:Paper)-[b:CITED_BY]→(p2:Paper)
WITH a, p, count(b) AS citations
WITH a, p, citations ORDER BY citations DESC
WITH a, count(p) AS total, collect(citations) AS list
WITH a, total, list, [x in range(1, size(list)) WHERE x ⩽ list[x - 1] | [list[x - 1], x] ] AS list_hindex
WITH *, list_hindex[-1][1] AS h_index
ORDER BY h_index DESC
RETURN a.AuthorName, h_index
```

"a.AuthorName"	"h_index"
" Howell L.L."	8
" Magleby S.P."	8
" Hawkins A.R."	7
" Schmidt H."	7
" Rice M."	6
" Beard R.W."	6
" Maynes D."	6
" Goodrich M.A."	6
" Ning A."	6
" Warnick S."	5
" Harrison W.K."	5

Section C. Recommender

1. The first thing to do is to find/define the research communities. A community is defined by a set of keywords. Assume that the database community is defined through the following keywords: data management, indexing, data modeling, big data, data processing, data storage and data querying.

```
neo4j$ MATCH (a:Author)←[w:WRITTEN_BY]-(p:Paper)-[r:RELATED_TO]→(t:Topic) WHERE t.Key…
```

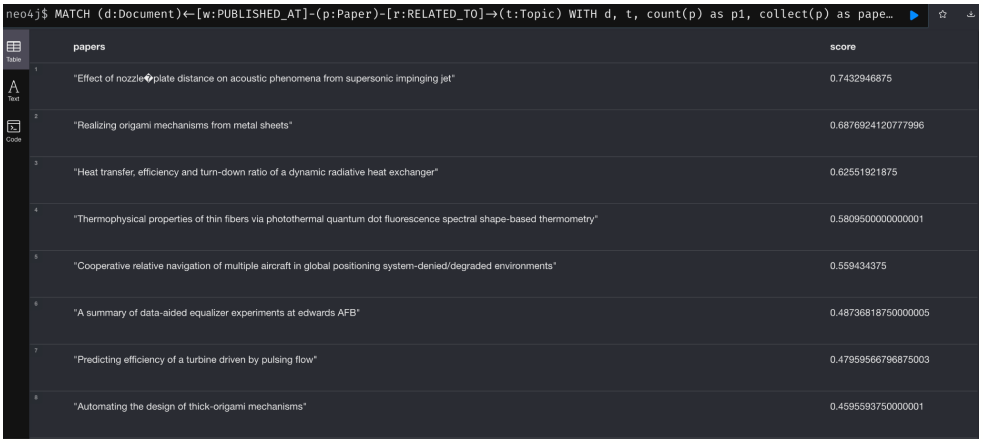
	Author	Keywords
1	" Crockett J."	["Data processing", "Data processing", "Data processing", "Big data", "Big data", "Big data", "Indexing", "Indexing", "Data
2	" Maynes D."	["Data processing", "Data processing", "Data processing", "Big data", "Big data", "Big data", "Indexing", "Indexing", "Data
3	"Searle M."	["Data processing", "Data processing", "Data processing", "Data management"]
4	" Emerson P."	["Data processing"]
5	" Howell L.L."	["Data processing", "Data processing", "Data processing", "Data storage", "Data storage", "Data storage", "Data storage",
6	" Magleby S.P."	["Data processing", "Data processing", "Data processing", "Data processing", "Data storage", "Data storage", "Data storag

2. Next, we need to find the conferences and journals related to the database community (i.e., are specific to the field of databases). Assume that if 90% of the papers published in a conference/journal contain one of the keywords of the database community we consider that conference/journal as related to that community.

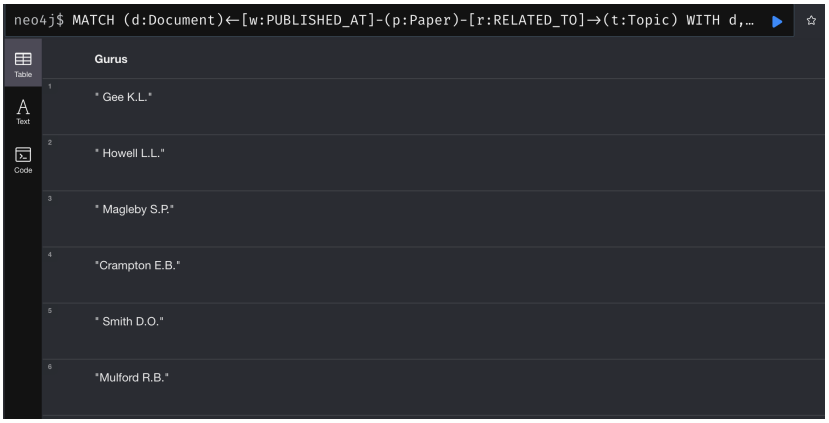
```
neo4j$ MATCH (d:Document)←[w:PUBLISHED_AT]-(p:Paper)-[r:RELATED_TO]→(t:Topic) WITH d, t, count(p) as p1, collect(p) as pape…
```

	Paper_Title	Publication_Type
1	"AIAA Scitech 2019 Forum"	"Conference"
2	"Proceedings of the ASME Design Engineering Technical Conference"	"Conference"
3	"ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE)"	"Conference"
4	"Proceedings of the ASME Turbo Expo"	"Conference"
5	"Proceedings - IEEE International Conference on Data Mining, ICDM"	"Conference"
6	"IEEE International Conference on Intelligent Robots and Systems"	"Conference"
7	"2019 IEEE Milan PowerTech, PowerTech 2019"	"Conference"
8	"Proceedings of the 19th International Conference on Engineering and Product Design Education: Building Community: Design Education for a Sustainable Future, E and PDE 2017"	"Conference"

3. Next, we want to identify the top papers of these conferences/journals. We need to find the papers with the highest page rank provided the number of citations from the papers of the same community (papers in the conferences/journals of the database community). As a result we would obtain (highlight), say, the top-100 papers of the conferences of the database community.



4. Finally, an author of any of these top-100 papers is automatically considered a potential good match to review database papers. In addition, we want to identify gurus, i.e., very reputated authors that would be able to review for top conferences. We identify gurus as those authors that are authors of, at least, two papers among the top-100 identified.



Section D. Graph Algorithms

Task D.1

The implementation for Task D.1 can be found in the PartD_LeonAliakberova script.

Task D.2

<u>Betweeness</u>

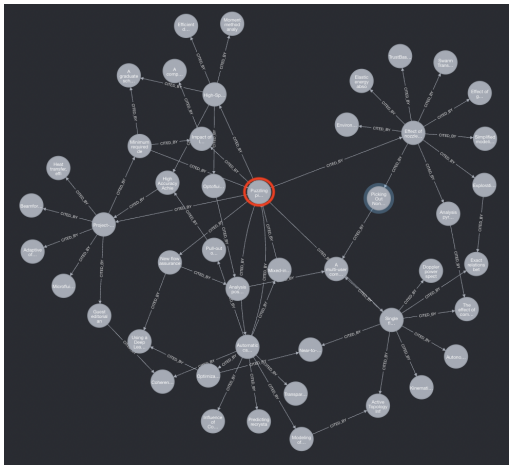
The first algorithm we decided to analyse is betweeness. This algorithm helps to find areas of "connection bridges" on a graph. This is useful because it can help us determine nodes that are crucial in interconnecting the whole graph, or nodes that give the graph network more significance.
In our case, we are calculating betweeness for the Paper nodes, with the CITED_BY edge. This is useful since it will let us know which papers are cited more often, and are influentiating the work of

others. For instance, let's say paper A is cited by paper B, and paper B is cited by paper C, therefore we know that paper A is indirectly influenced by paper C even though it's not cited directly.

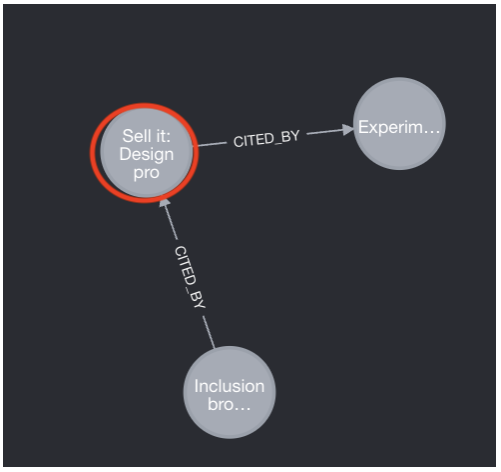
```
CALL gds.betweenness.stream('betweeness')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).Title AS name, score
ORDER BY score DESC
```

"name"	"score"
"Puzzling the pieces: Conceptual blocks of engineering student ideas in a service learning project"	6930.68936874896
"Three-way spectral decompositions of high-performance military aircraft noise"	6816.906757232212
"Analysis of pyrolysis products from live shrub fuels"	6435.607106084784
"Identification and Prioritization of Critical Subject Matter within Mechanical Systems Curriculum in Construction Management Education"	5967.193138962931
"Space-Time Coded ARTM CPM for Aeronautical Telemetry"	5955.785119984186
"Process responses and resultant joint properties of friction stir welding of dissimilar 5083 and 6061aluminum alloys"	5762.645295191435

Graph Query 1 Results



High betweeness node



Low betweeness node

In Graph Query 1 results, we can clearly see the most influential paper is titled "Puzzling the pieces", we can see the neat "bridges" it generates, which is indicative of this paper being influential to several other works from different authors. On the other hand we pick the paper title "Sell it", which has a very low betweeness score, and we can see how it only influenced two other papers.

Node similarity

To determine node similarity, Neo4j implements what is known as Jaccard coefficient, which is defined as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

This can be of interest since this formula can tell us the level of similarity between two nodes sharing a particular edge. For example, in this lab we ran this algorithm between Papers sharing a Topic over the edge RELATED_TO.

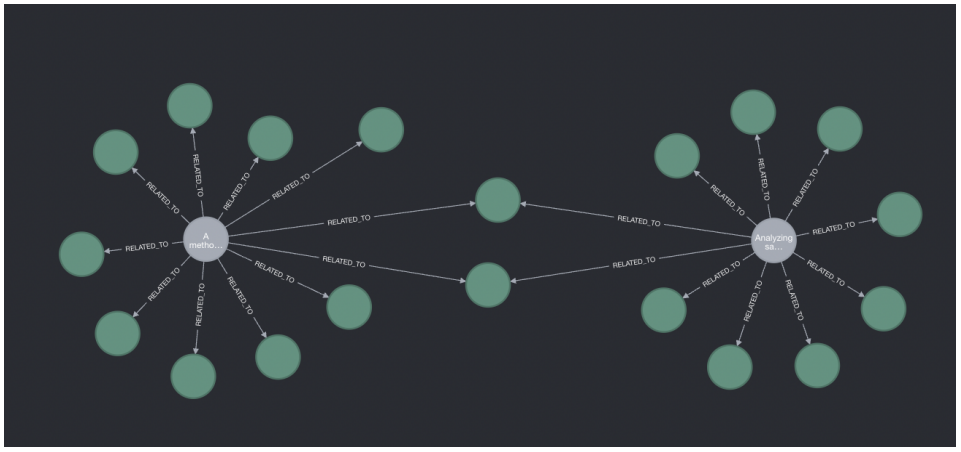
```
1 CALL gds.nodeSimilarity.stream('NSPaperTopic', {bottomK:1})
2 YIELD node1, node2, similarity
3 RETURN gds.util.asNode(node1).Title AS Paper1, gds.util.asNode(node2).Title AS
  Paper2, similarity
4 ORDER BY similarity DESC
```

Table | Text | Code

"Paper1"	"Paper2"	"similarity"
"Analyzing the safety impacts of raised medians"	"A methodology for analyzing intersection safety"	0.10526315789473684
"Hybrid power system state estimation with irregular sampling"	"Probabilistic extension of flexible hybrid state estimation for cyber-physical systems"	0.0869565217391304 3
"Influence of Communication Irregularities and Co-simulation on Hybrid Power System State Estimation"	"Probabilistic extension of flexible hybrid state estimation for cyber-physical systems"	0.0869565217391304 3

Graph Query 2 Results

If we review our graph, we can observe that the two first papers share a total of 2 topics (through the edge RELATED_TO) from a total of 10 topics for Paper1, and 11 topics for Paper2. Since the shared topics is 2, then the union is 19 because (10+11)-2 = 19, this means P1 U P2.
Therefore, J(P1, P2) = 2/19 = 0.10526. As you can observe, we can see that this graph's papers are not really similar in terms of topic keywords, as the maximum similarity found was the one from the example above. A higher similarity would mean that many papers on the graph belong to similar topics, hinting at a stronger "community" of keywords.



Highest similarity nodes over RELATED_TO edge.