
Investigating collaboration within DCU through citation graph analysis

Anthony Reidy
Kian Sweeney

18369643
18306226

1 Introduction

- **Code:** <https://github.com/reidya3/CitationGraphAnalysis>
- **Video Link:** <https://drive.google.com/drive/u/0/folders/1v81OPN7zrqRXe1KVx2XWd4sqBn140gYz>

The area of citation graph analysis has always been an area of great interest for those in academia. Citation graph analysis enables the understanding of academic collaboration and modern research trends. One can identify which researchers hold great influence in a particular field and relate clusters of researchers, conferences or journals to their relevant research domains. Shi et al. (2015) notes that there have been quite a few different ways to measure the influence of individuals such as by the total number of citations or the "H-index" score.

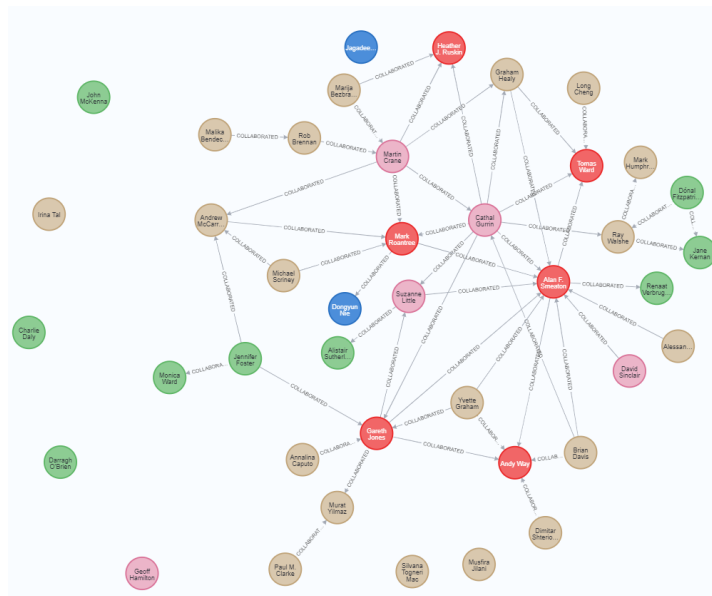


Figure 1: Graph displaying researchers that have collaborated at least once.

An initial investigation of research partnerships within DCU's school of computing revealed little collaboration. Figure 1 depicts the current current collaboration efforts among researchers at the faculty. Note in the above graph, researchers are color-coded by their position. Red denotes full professors, pink associate professors, brown assistant professors, green

lectures and blue teaching assistants. In this assignment, we plan to explore this current trends in research partnerships, exploiting the distributed computation and storage of HDFS and Graphx. Overall, we believe that the results generated from this investigation will be useful for identifying influential staff members both within DCU and further afield, which we hope will foster avenues for cooperation.

2 Related Work

Looking at related works in this area of citation graph analysis, we see that a lot of the work focuses directly on papers and not authors. In Jacovi et al. (2006), the betweenness centrality algorithm was used to detect clusters in topic trends related to all publications accepted at ACM Computer Supported Collaborative Work (CSCW) between 1986 and 2004. The data was filtered down in a lot of ways, removing papers with no citations or other important missing information which is something we could definitely look to implement in our PySpark analysis and cleaning. There was one hundred and twenty-three clusters of varying sizes detected related to topic trends in publications accepted at this conference between the aforementioned dates. In Son & Kim (2018), a recommendation system for academic papers is developed using citation analysis as a key component. Graph algorithms such as degree centrality, closeness centrality, betweenness centrality are calculated to decide the most significant papers for recommendation, or papers that hold the most influence. Co-citation analysis is undertaken in Jeong et al. (2014) which can be used to identify, trace, and visualize the intellectual structure of an academic discipline by counting the frequency with which any work of an author is co-cited with another author in the references of citing. Node weights are calculated by degree centrality for author communities here whereas the modularity algorithm is used to determine the strength of division of the author clusters. We see a different approach in Ding (2011) where this paper focuses on associating topics with authors and identifying changes in collaboration patterns over a forty year span. Different algorithms are used such as breadth first search here in contrast to other papers mentioned.

3 Dataset

For this investigation, we construct a novel dataset incorporating data relating to researcher's publications from Google scholar ¹ and DORAS² (DCU's institutional repository) and demographic information from the school of computing's website ³. The data was scraped to include data from 2011 to early November 2021 using the BeautifulSoup and Selenium python libraries. Examples of features extracted from Google Scholar include the title of paper, it's year of publication, the number of citations garnered, the journal it was found in and the authors related to the publication, among many others. Doras was scraped for its ability to add additional metadata about the publications in the form of a feature known as "Uncontrolled Keywords". Note, there are often slight differences between paper's title between Doras and Google Scholar. We employ levenshtein distance at a threshold of 90 to match papers. As DORAS's upload rate stands at approximately 50.1%, only about half of the papers have the enriched metadata. In addition, not every researcher in DCU has a Google Scholar and/or DORAS profile. A researcher's position within the faculty was retrieved from the school of computing's website. Finally, the data was processed into CSV files and saved on HDFS for further processing and analysis.

¹<https://scholar.google.com/>

²<https://doras.dcu.ie/>

³<https://www.dcu.ie/computing/people>

4 Methodology

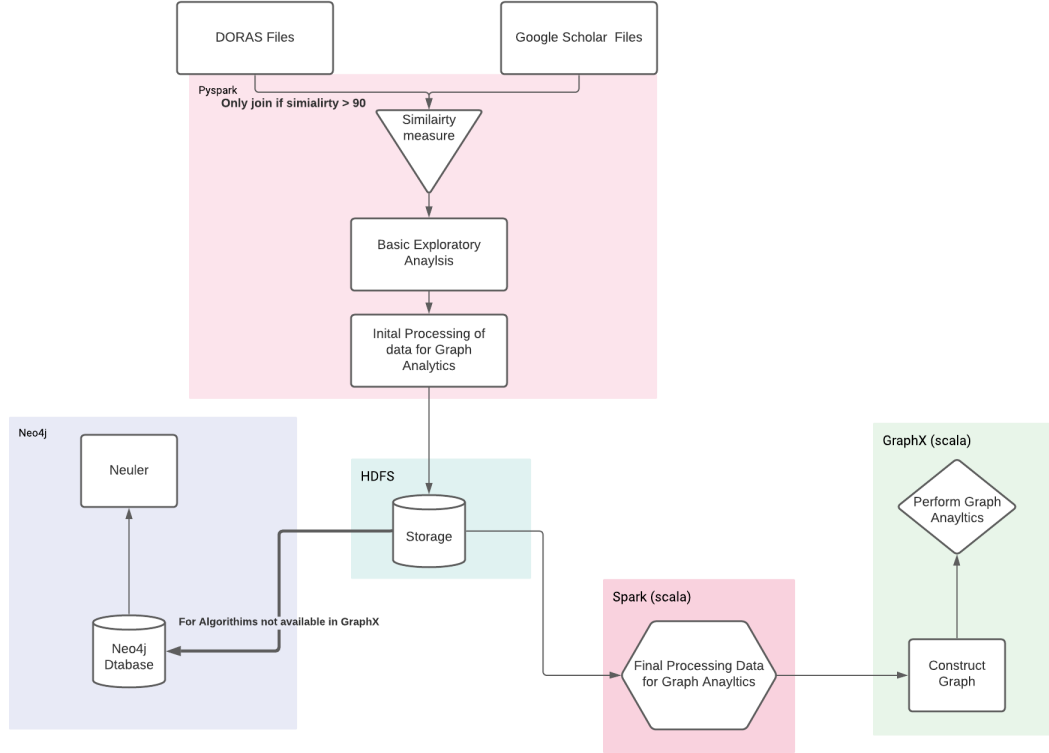


Figure 2: Technologies used

In Figure 2, the technologies used in this investigation are detailed. These differ quite considerably from the midway report and these differences are discussed in the response to peer feedback section. Pyspark is used to perform transformations on the data and perform basic exploratory analysis. This was done to improve our understanding of the data. The data files for our proposed graphs are created in pyspark and saved in HDFS. Although, it must be noted that some transformations for the graph files were still performed in Scala. Engineered features include:

- (a) the number of papers (np)
- (b) citations (c)

generated from the collaborative efforts of any two researchers. This distributed PySpark/Scala approach was utilised in an effort to enable quicker development, as this was our first time using Scala. For the graph analytic work, GraphX is primarily used. However, the open-source GraphX implementations of the Louvain Modularity⁴ and Betweenness Centrality⁵ algorithms do not work anymore. Therefore, we use Neo4j for such algorithms. As Graphx is a data processing tool, the Neuler visualisation app is used to visualise results. However, screenshots

⁴<https://github.com/Sotera/spark-distributed-louvain-modularity>

⁵<https://github.com/dmarcous/spark-betweenness>

of the shell are provided to demonstrate GraphX usage. In addition, the *graph-analytics/run-graph-algos.sh* file specifies the bash commands to run the graph algorithms, implemented in GraphX. Sbt is utilised as the building tool for the scala project.

Two undirected graphs $G(N, E)$ are constructed that consists of a finite set of nodes N (researchers) and a finite set of edges E . With each edge E of G , there will be an associated real number $w(E)$, called its weight.

$$w(E) = np + 0.5(c) \quad (1)$$

The only difference between the graphs is one **includes** collaborates **from outside DCU** (*Graph2*) and one **does not** (*Graph1*). Finally, a bipartite area of interest (paper \rightarrow keyword) is transformed into a monopartite similarity sub-graph. The graph (*Graph3*) will contain keyword \rightarrow keyword relationships where the weight, $w(E_2)$, will be:

$$w(E_2) = \text{Number of common papers} \quad (2)$$

If two keywords don't have any common papers, **there exists no relationship**. Due to the deletion of *DCU academic search* from our final project, we expand the number of algorithms used.

Graph Algorithm	Graph	Technology Used
Degree Centrality	Graph1, Graph2	GraphX, Neuler
Triangle Count and clustering coefficient (local & global)	Graph1, Graph2	GraphX, Neuler
Betweenness centrality	Graph1, Graph2	Neo4j, Neuler
Page Rank	Graph1, Graph2	GraphX, Neuler
Label Propagation	Graph3	GraphX, Neuler
Louvain Modularity	Graph1	Neo4j, Neuler

Table 1: Table showcasing the graph algorithms used and the graph it was applied to

5 Results and Challenges

5.1 Degree Centrality

Our initial graphs analysis is very simple. We compute the amount of degrees each nodes has in Graph1 and Graph2. The degree is the number of links incident upon a node. We present the results sorted by degree descending to find the researchers that have the most number of collaborators within DCU in Graph1 and the researchers who have the most number of collaborators within DCU and further afield in Graph2.

Researcher	Degrees
Alan Smeaton	12
Cathal Gurrin	10
Gareth Jones	8
Martin Crane	7
Mark Roantree	6

Table 2: Top 5 researchers sorted by their degrees in Graph1

Researcher	Degrees
Charlie Daly	804
Gareth Jones	763
Alan Smeaton	517
Cathal Gurrin	414
Andy Way	326

Table 3: Top 5 researchers sorted by their degrees in Graph2



Figure 3: Screenshot of GraphX degree centrality for graph2

5.2 Triangular count and clustering coefficient

Triangle count determines the number of triangles passing through a node in the graph. Clustering coefficient is the probability that the neighbours of a particular node are connected to each other and utilises triangle count within it's algorithm. The average clustering coefficient is used to determine if the community's research efforts are tight knit or sparse.

Results for Graph1: This projection has 22 unique triangles and has an average clustering coefficient of 0.245. This indicates that community research efforts are not tight knit together but rather sparse. A network with a higher average clustering coefficient is often called a small world network which we do not believe this graph shows. This is due to the presence of islands within the graph as shown in Figure 1.

Donal Fitzpatrick, Jane Kerman, Marija Bezbradica, Michael Scriney and Yvette Graham all have a clustering coefficient of 1. This means that every one of their collaborators (who are also part of the faculty) have collaborated at together at least one time (maybe all on the same paper but most likely on multiple papers). In addition, as their triangle count are all low, it seems to suggest they tend to collaborate with a small group of people, within their own "cluster" and not act as bridging points. They are also all assistant professors/ lectures which would further validate this result as they are early in their research carer and perhaps would not have the connections to collaborate with different "communities" within the faculty.

Alan Smeaton and Cathal Gurrin both have clustering coefficient less than the average but high triangle counts. They might be a structural hole in this graph i.e. a node that is well connected to nodes in different communities that aren't otherwise connected to each other. We would have thought Andy Way would have shown more of this nature since he is deputy director of the ADAPT research centre but perhaps he tends to collaborate with people who are not staff members in the school of computing or does, but does so in his own community. Additionally, PhD students and post-doctoral researchers from the school of computing and the various research centres are not included in this projection. Gareth Jones, Martin Crane, Mark Roantree and Andy Way show this behaviour although to a lesser extent. In addition, these people all occupy senior positions, either associate professor or professor, which may

indicate a correlation between seniority and the probability of a person being a structural hole. Additionally, as their research career is most likely longer than their juniors, they would have likely developed these connections over time.

People who have a clustering coefficient of zero either have not collaborated with a DCU faculty member or have but none of their neighbours have worked with each other. Examples of such people include Paul Clarke and Ray Walshe.

Results for Graph2: As the data has been scraped from faculty member's google scholar profile only, the following metrics are not representative of an external (or their place of work could be within DCU but is unknown) researcher. For example, two unknown researchers identified could have collaborated with each other, just not with a staff member of the school. 4606 extra researchers have been added to the graph.

Globally, this projection has a unique triangles count of 4712289 and has an average clustering coefficient of 0.9155642. This indicates, on average, a researcher's neighbours have a probability of 91.56% of having collaborated as well. As such, this network is tight knit and cohesive. A network with a higher average clustering coefficient is often called a small world network which I do believe this graph shows. This is very different from the previous graph's nature, but it is expected due to the nature of how the data was obtained. i.e. Every external researcher has collaborated with a member of the faculty or else they would not be in the graph! There are also no islands in this graph but there was in Graph1.

Many external researchers have a clustering coefficient of 1(3500). This could be due to the fact that they collaborate with a DCU researcher only once on one paper and as such their neighbours are all connected. Another possible reason is that certain DCU researchers and external researchers could have formed a community of their own. i.e. The research centres.

Researchers such as "A Angelis" and "B Cengiz" are favourites of the faculty to collaborate due to their high triangular count (11206, 11205). Their clustering coefficient is high so they are tight knit with the collaborators they wish to participate. Of course, such clustering coefficients cannot be taken to represent the total research efforts of that individual due to the nature of the data collected.

As in the DCU researcher only graph, Alan Smeaton, and Catha Gurrin both have clustering coefficient less than the average clustering coefficient but high triangle counts. However, unlike Graph1, Gareth Jones seems to show the qualities of a structural hole more. He has a very high triangle count (28633) and a low clustering coefficient. When taking external collaborators into account, Gareth seems to be well connected to nodes in different communities that wouldn't otherwise be connected. This might be due to the fact that he is a prolific researcher (317 papers published since 2011) and tends to collaborate with a lot of people (average per paper = 5.82) vs Alan Smeaton (average per paper = 4.62).

In addition, these people all occupy senior positions, either associate professor or professor which reaffirms the idea that there might be a correlation between seniority and the probability of a person being a structural hole.

```
(Linardos Panagiotis,(0.5,1))
(Douglas Cirqueira,(0.2575757575757575,34))
(Dimitrios Tzovaras,(0.4861111111111111,35))
(Frank Hopfgartner,(0.33516483516483514,61))
(K Ball,(0.5,28))
(Pablo Redondo,(0.5,21))
(Sérgio Nunes,(0.5,78))
(Mihael Arcan,(0.5,6))
(Marc Gorriz Blanch,(0.45,9))
(Declan O'Sullivan,(0.30555555555555556,22))
(Osman Mucuk,(0.5,1830))
(Ann Clifton,(0.5,45))
(Renan Alves,(0.5,21))
(Sravana Reddy,(0.5,45))
```

Figure 4: Local clustering coefficient values and triangle count of external researchers (Graph2)

5.3 Betweenness centrality

Betweenness centrality is used to find “bridges” within a graph. We use it to find influencers within the DCU research community and within the wider community. We correlate such people with their role so to see if powerful individuals have high seniority. In this analysis, we chose to only calculate the centrality scores for DCU researchers only as we don’t have the roles of external researchers. Sometimes the most important cog in the system is not the one with the most overt power or the highest status. It might be the “middlemen” that can be the connector of groups.

Name	Role	centrality
"Martin Crane"	"Associate Professor"	50.0
"Cathal Gurrin"	"Associate Professor"	46.166666666666666
"Alan F. Smeaton"	"Professor"	38.0
"Gareth Jones"	"Professor"	24.5
"Mark Roantree"	"Professor"	20.666666666666668
"Rob Brennan"	"Assistant Professor"	18.0
"Ray Walsh"	"Assistant Professor"	13.0
"Suzanne Little"	"Associate Professor"	10.0
"Graham Healy"	"Assistant Professor"	5.666666666666666
"Andrew McCarren"	"Assistant Professor"	2.0
"Alessandra Mileo"	"Assistant Professor"	0.0
"Alistair Sutherland"	"Lecturer"	0.0
"Andy Way"	"Professor"	0.0
"Annalina Caputo"	"Assistant Professor"	0.0
"Brian Davis"	"Assistant Professor"	0.0
"Charlie Daly"	"Lecturer"	0.0

Figure 5: Betweenness centrality scores of researchers in Graph1 (Neo4j)

Interestingly, people with the highest centrality score (Martin Crane and Cathal Gurrin) do not occupy the most senior position. This indicates there are in high brokerage positions. Such people have connection to communities that would not otherwise be connected. Andy Way has a low centrality score and high seniority. This suggests, he tends to collaborate with people outside of the faculty and when he does collaborate within the faculty, he does so in his own community. Other than that, there is a trend that people with high centrality score occupy senior positions, people with low scores occupy junior positions. However, this could be more due to the length of their careers.

Name	Role	centrality
"Alan F. Smeaton"	"Professor"	22608.48755379998
"Gareth Jones"	"Professor"	12262.597096573276
"Cathal Gurnin"	"Associate Professor"	9796.624046720799
"Andy Way"	"Professor"	9791.066540619084
"Martin Crane"	"Associate Professor"	8788.350796508638
"Mark Roantree"	"Professor"	6804.844254423137
"Tomas Ward"	"Professor"	6405.495670995673
"Rob Brennan"	"Assistant Professor"	3668.050324675327
"Graham Healy"	"Assistant Professor"	1844.771787592776
"Suzanne Little"	"Associate Professor"	741.7208128082488
"Andrew McCann"	"Assistant Professor"	428.31347859656677
"Ray Walshe"	"Assistant Professor"	238.57021312021314
"Heather J. Ruskin"	"Professor"	235.54012467027488
"Alistair Sutherland"	"Lecturer"	154.9242424242424
"Monica Ward"	"Lecturer"	5.0
"Alessandra Mileo"	"Assistant Professor"	0.0

Figure 6: Betweenness centrality scores of DCU researchers in Graph2 (*Neo4j*)

It shows a stronger trend that people with high centrality score occupy senior positions and people with low scores occupy junior positions. Andy Way's centrality score is way higher here which confirms that he is a bridging point in the wider research community, juts not within DCU. Alan Smeaton and Gareth Jones scores are both high in the DCU community graph and this graph, indicating they are in brokerage positions in both DCU and the wider Research Community. People with high scores of centralities have a high amount of short paths running through them so they are important for the "flow of collaborations" between researchers.

5.4 Page Rank

The PageRank algorithm is designed to identify researchers with more influence over other researchers, even if they have not directly collaborated. As PageRank is influenced by the direction of a relationship, the edges will now be bi-directional. We set the damping factor and max Iterations to 0.85 and 20, respectively.

Statistic	Value
max	3.23
min	0.15
25th percentile	0.15
50th percentile	0.15
75th percentile	0.15
99th percentile	2.28
mean	0.32
standard deviation	0.45

Table 4: Statistics of PageRank values in Graph1

In graph1, 75% of researcher's now have a score less than half of the mean. The PageRank scores reflects a power law distribution with very few influential people. The presenence of islands in this graph also contributes to such results (Researchers that have no collaborations within DCU are not shown in Figure 7). Alan Smeaton has the highest rank score when restricted to only include relationships with other DCU researchers and overall. Martin Crane, Mark

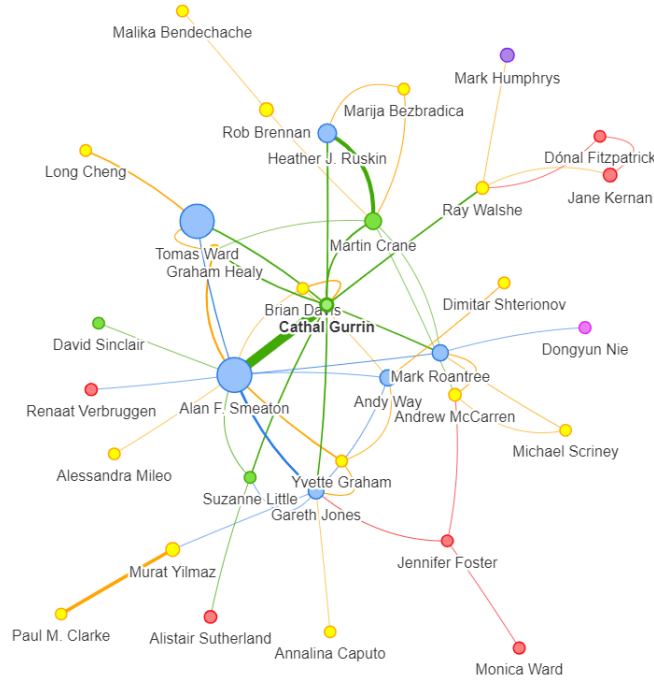


Figure 7: Pagerank of Graph 1. Note the size of the node is proportional to the researcher's score. Similarly, the thickness of the edge is proportional to it's weight (number of papers \times $(0.5 \times$ citation count)). The color of a node relects that researcher's position.

Rountree, Suzanne little Ray, Walshe and Graham Healy are in the top 11 but do not appear in the top 11 when we examined these researchers in Graph2. This indicates they are more influential within the faculty than in the wider research community. Tomas Ward has a relatively low pagerank score (1.15) despite his high overall number of citations (1937). When retrieving his score (15.5) in graph2, it seems like he is likely collaborated with influential people just not as much within DCU. The following stats detail the page rank score of every node in Graph2 (DCU and external researchers).

Statistic	Value
max	33.91
min	0.15
25th percentile	0.55
50th percentile	0.97
75th percentile	1.07
99th percentile	3.19
mean	0.94
standard deviation	1

Table 5: Statistics of PageRank values in Graph2

The following stats detail the page rank score of every node. Half of researchers have score

less than 1, meaning they don't possess much influence. However, this could be due to the way in which the data is collected. Two external researchers could have collaborated at one time, but we just don't know since a researcher in DCU was not involved. It is interesting to note 50% of researchers have a score less than or near the mean. Normally such data reflects a power law distribution with very few influential people. Again, we must take this with a pinch of salt as we do not have the full picture.

```
(233,(Researcher(Darragh O'Brien,Lecturer),0.3952629548551194))
(1105,(Researcher(Venkatesh Balavadhani Parthasarathy,Unkown),1.19414555589742))
(991,(Researcher(Sheila Castilho,Unkown),2.3737799727979296))
(537,(Researcher(Jurgen Mulsow,Unkown),0.5362860780455434))
(880,(Researcher(Plamen Petkov,Unkown),0.9075648424172797))
(604,(Researcher(Liqiang Nie,Unkown),0.7288030134779208))
(377,(Researcher(Graham Healy Akanksha Rajpute,Unkown),0.4266782181329561))
(384,(Researcher(Guodong Xie,Unkown),0.43679152851970754))
(522,(Researcher(Jonathan Fiscus,Unkown),1.0016437190934997))
(315,(Researcher(Eva Vanmassenhove,Unkown),1.1012042483266171))
```

Figure 8: Pagerank values of external researchers (Graph2)

5.5 Label Propagation

This algorithm is applied to Graph3. In the data, there are 1268 unique keywords. Examples include game theory, software developers' personalities and language pairs. With hundreds of thousands of publications being written every year, clustering such keywords in "Super keywords" attempts to alleviate information overload, allowing researchers to access publications faster. Weighted label propagation created 514 different "super keywords". We will now evaluate these groupings, by examining three examples.

The first example describes a "superkeyword" which contains the keywords data analytic, data warehousing, ETL and data mining. We believe this is an effective clustering as such keywords all relate to the storage and analysis of data i.e. They are similar in semantics and are basically the content the data warehousing and OLAP module. Such a super keyword could also provide a mechanism for researchers to easily identify other researchers in the school of computing who already have experience in this topic.

The second shows stipulates a superkeyowrds which contains the keywords logic gates, gaultiy of service, packet loss, throughput and performance evaluation. These keywords all got "trapped" in the same community label. All of these keywords relate to software systems and its performance. Throughput refers to the rate of production. Packet loss refers to errors in data transmission. High packet loss would indicate low performance evaluation and quality of service. Interestingly, this superkeyword comes from the keywords of just one paper.

The last example describes a super keywords (c) which relate to data anaylis in the retail industry. This is perhaps the best community we have seen. This superkeyword contains the keywords consumer, ecommerance, digital retail,behaviour anayslits and purchase prediction. Such a keyword would provide an opportunity for a faculty member to collaborate with others if their idea related to retail data analysis.

```
(Cloud Migration Processes,140)
(Crisis Management,514)
(Query Service,119)
(Dialogue translation,373)
```

Figure 9: Screenshot of shell detailing label propagation communities of keywords

5.6 Louvain Modularity

People who are islands in the Graph1 form their own community and provide no new insight. Therefore, they were not included in my visualizations and analysis. The following visualizations show the results when no weight was used and when a weight was used. These communities represent the final communities of the louvain algorithm i.e. the maximally coherent subnetworks.

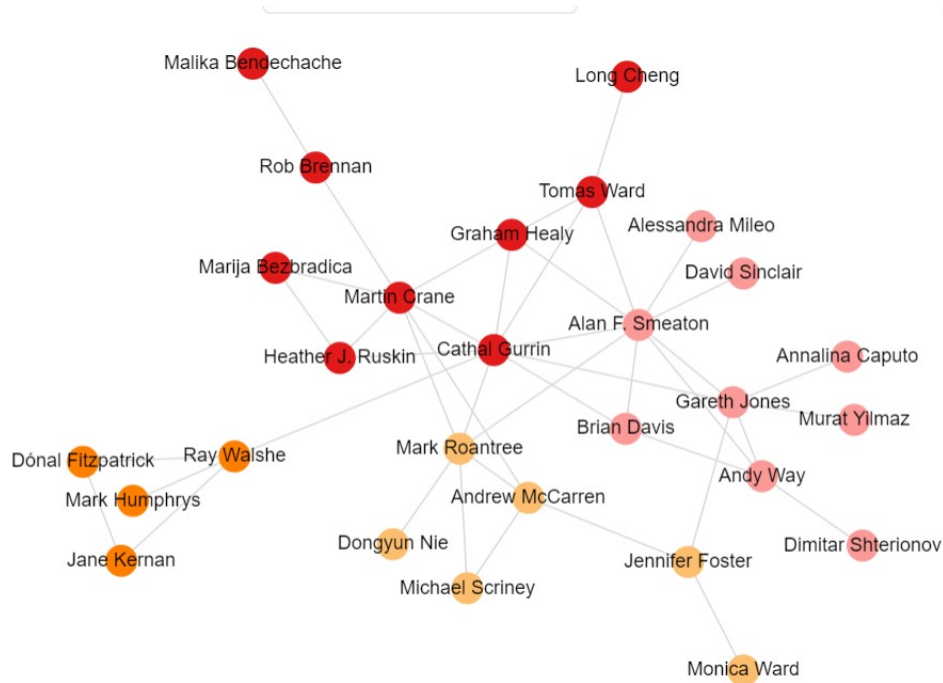


Figure 10: Unweighted Louvain communities of Graph1

Louvain modularity compare relationship densities in given clusters to densities between clusters. i.e. we want a lot of connections in the set and few connections outside the set. It is interesting that Andy way, Brain Davis and Brain Davis were grouped into a cluster with insight centres members based just purely on relationships but are in their own separate community when0ighted Louvain modularity was used. It would make sense that such a grouping would form as all 3 have affiliations with the Adapt research centre and are likely to have higher citations merely due to the fact that they have collaborated more. In the Louvain modularity algorithm, we perform a modularity optimization process to determine the number of communities, then the communities become nodes with the internal relationship contributing a weight of 2 on that node. External relationships between clusters merge into a single relationship with a weight equal to the number of relationships between those two clusters. We then repeat the process until the process stops merging. The increased weights caused the grouping to be viewed as a distinct cluster in the modularity optimisation process. This also caused the increase in the number of communities between the unweighted and weighted modularity as more distinct communities were found. This also could be due to the fact that Louvain algorithm is an approximation and thus relies on some randomness. The amount of communities discovered increased 2-fold.

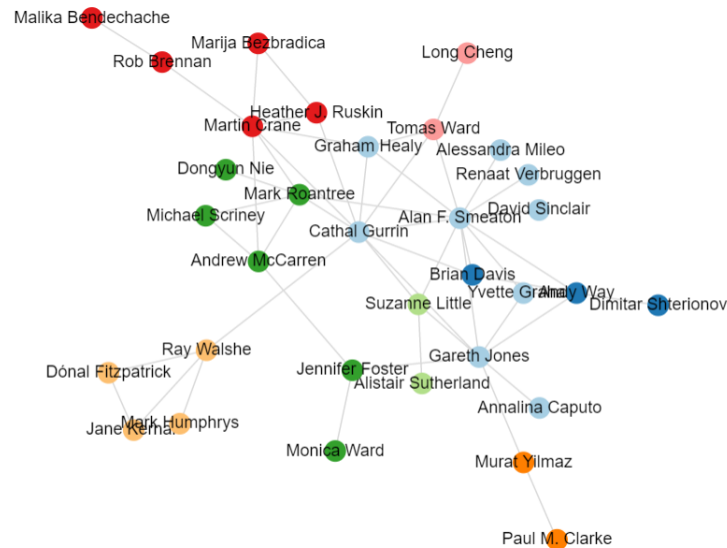


Figure 11: Weighted Louvain communities of Graph1

5.7 Challenges

Our biggest challenge involved the use of Scala and Graphx. A lot of the documentation and tutorials for GraphX was written pre-2017 and does not reflect the latest version of GraphX. For instance, we wasted a lot of time trying to implement an open source version of Betweenness centrality, only to discover that it relies on an external library that has changed considerable since then. We also had no prior experience of Scala. Learning the syntax slowed us down quite a bit. In addition, although we had some experience running scala files via the spark shell, we had no prior experience of running a self-Contained scala app via the spark submit command. Sbt has a deep learning curve to retrieve the functionality we desired i.e. the ability of running of different main classes desperately. Finally, we found resolving dependency issues in sbt was not as intuitive as in python.

6 Responsibility Statement

As per our midway report, we had planned to distribute our work in an even manner. Our intentions were to split the data scraping in a 50:50 split which we achieved successfully. Kian scraped the relevant data from each relevant academic profile's Google Scholar, while Anthony scraped the data off Doras also. We also intended at the start to split our work up evenly regarding our data analysis. It must be said Anthony covered the majority of work concerning graph analysis but Kian covered the majority of work considering our ground work PySpark analysis. This report has also been split in an equal manner with both of us taking different sections and covering an equal distribution of the work at hand for the report. Obviously, we dropped our development of our flask app that was mentioned in our mid-way report also. Overall, we felt we stuck to the originally intended distribution of work fairly well and our grades are as follows below.

- ANTHONY: 50%
- KIAN: 50%

7 Response to Peer Feedback

7.1 Group A Feedback

Section 1: Regarding our feedback from group A, we feel they have made a valid point regarding time constraints. We are already near the end of November and trying to implement our flask app outlined regarding our retrieval system would be challenge with our assignment due on the 10th of December. Both of us have limited knowledge in front-end technologies, with Kian only really covering it on his intra work placement covering JavaScript. Although we might be able to get a basic app functioning it may not be to the level we would like. As a result of reading this feedback we think we are going to adapt our approach. We will look to drop our app development and focus more on our graph analysis and PySpark analysis of our data. We feel if we spend more time on this then we will get a more complete analysis of our data. We also feel this is more in line with the requirements outlined for the project (big data analysis) and would be more worthwhile. As a result, we definitely found this feedback here useful and the different opinions made us realise we may have undertaken too much work in a short timeframe.

Section 2: - Technology 1 – We feel that group A try to make a valid point regarding web scraping but that it is probably not as relevant due to us already having our data scraped and processed. The web scraping admittedly took a sufficient amount of time but that is to be expected. We feel even if we used Scrapy it would have taken a long enough amount of time to scrape our data anyway. Taking our scraping of google scholar profiles there were over 4,800 papers to scrape with their corresponding information. Between iterating through different author pages, going in and out of different papers and also taking the necessary time for re rendering of pages and not scraping too quickly it was always going to be a slower process. Selenium was used here and very accurate with its data scraped. It was also already outlined on the Gantt chart in our midway report as something we already hoped to have done to save time before reviews were due. - Technology 2 – We feel group A probably make a valid point in some respects here when talking about Scala using Spark. To clarify our approach using this, our graph analysis will be performed using Scala as the needs of GraphX dictate. We will also conduct some data cleaning and analysis our scraped data using PySpark using a python based Jupyter notebook. This would be initial analysis looking at averages, number of papers published by an author and so forth while also looking for faulty values in the data. This would be conducted using a PySpark dataframe with our CSV data loaded in. Once this is done, we feel could easily use Scala Spark to conduct our graph analysis. We feel that working PySpark and Scala using spark in tandem can allow us to gain the best and most accurate results in our data.

Section 3 It is hard to argue with group A's recommendations here. We felt we had fairly distributed our work and thus far Anthony has taken leadership roles within the group. The two of us have worked together before in groups and typically Anthony has taken on leadership roles in these groups. With some of the past results obtained with this structure when we have worked together, it is unnecessary to change leadership roles here as a result.

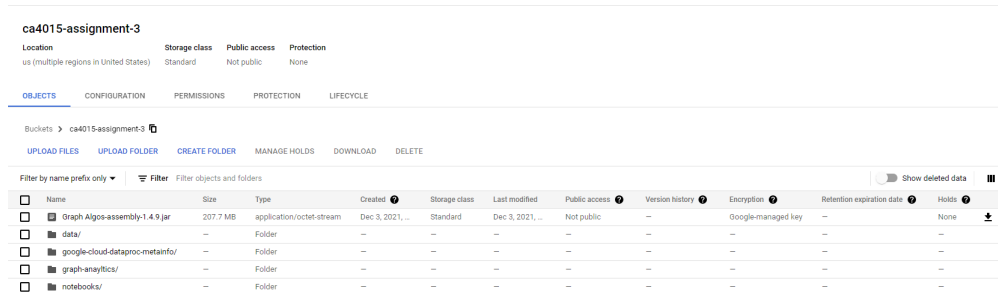
Grade: 7/7

8 Cloud Deployment

8.1 Pre-requisites

We utilise Google cloud Platform to run Assignment 3 on the Cloud. First, we create a project called assnignment-3 .Next, we have to enable the Dataproc, Compute Engine, and Cloud Storage APIs. Next, services account is created. Google Cloud SDK is also installed locally. We create a standard cluster called cluster-0d18 where both master and worker nodes have the

following hardware configurations: n1-standard-2 (2cpu, 7,5GB). We enable also enable the jupyter component. We also create a cloud storage bucket called ca4015-assignment-3 to store all of our files, currently on HDFs, Jupyter notebooks and our scala app. To upload the wanted files and folders, we click the Upload Files button.



ca4015-assignment-3											
Location	Storage class	Public access	Protection								
us (multiple regions in United States)	Standard	Not public	None								
<div> <div>OBJECTS</div> <div>CONFIGURATION</div> <div>PERMISSIONS</div> <div>PROTECTION</div> <div>LIFECYCLE</div> </div>											
<div> <div>Buckets > ca4015-assignment-3</div> <div> <div>UPLOAD FILES</div> <div>UPLOAD FOLDER</div> <div>CREATE FOLDER</div> <div>MANAGE HOLDS</div> <div>DOWNLOAD</div> <div>DELETE</div> </div> </div>											
<div> <div>Filter by name prefix only</div> <div>Filter objects and folders</div> <div>Show deleted data</div> </div>											
<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention expiration date	Helps
<input type="checkbox"/>	Graph Algos-assembly-1.4.9.jar	207.7 MB	application/octet-stream	Dec 3, 2021, ...	Standard	Dec 3, 2021, ...	Not public	—	Google-managed key	—	None
<input type="checkbox"/>	data/	—	Folder	—	—	—	—	—	—	—	⋮
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—	—	⋮
<input type="checkbox"/>	graph-analytics/	—	Folder	—	—	—	—	—	—	—	⋮
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—	—	⋮

Figure 12: Files in our bucket

8.2 Run pyspark transformations

For running the pyspark transformations, we utilise the jupyter componet of our cluster. We follow this tutorial ⁶. We also enable the component gateway so that we can utilise the web interface of the jupyter UI. The major difference in our jupyter notebook in the repo and the one located on the cloud is that we had to change the file paths so that they point to the files located in the bucket.

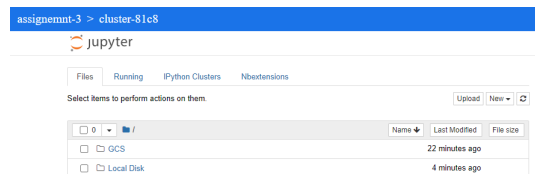


Figure 13: Our jupyter notebook files are located in the GCS folder

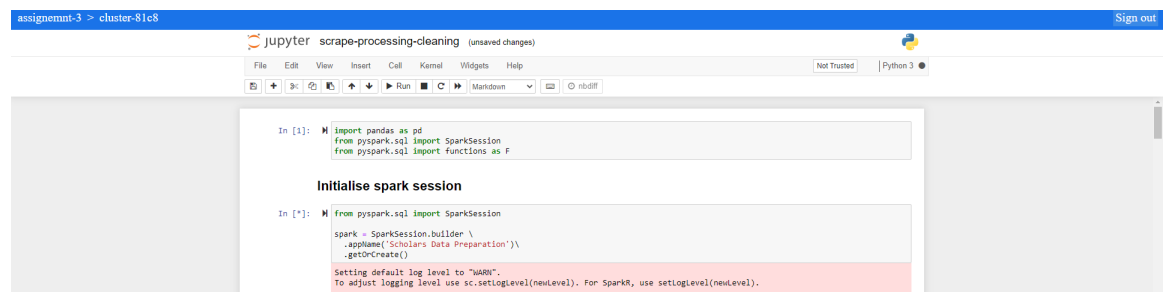


Figure 14: The scrape-processing-cleaning jupyter notebook on the cloud

⁶<https://cloud.google.com/dataproc/docs/tutorials/jupyter-notebook>

8.3 Run spark jobs

As mentioned previously, we use sbt as our building tool. We use sbt to create a jar file. Again, we change our file paths to . We use sbt to create the jar file and we follow the following tutorial to submit a job ⁷. Currently, in the scala app, we have four entry points, one for every graph algorithm. We use these different endpoints to construct four different spark jobs as follows:

The screenshot shows the 'Submit a job' form in Google Cloud Dataproc. The 'Job ID' field is filled with 'degree-centrality'. The 'Region' dropdown is set to 'us-central1'. The 'Cluster' dropdown is set to 'cluster-81c8'. The 'Job type' dropdown is set to 'Hadoop'. The 'Main class or jar' field is filled with 'DegreeMain'. The 'Jar files' field is filled with 'gs://ca4015-assignment-3/Graph-Algos-assembly-1.4.9.jar'. Below the 'Jar files' field, there is a note: 'Jar files are included in the CLASSPATH. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix.'

Figure 15: Google cloud job for degree centrality

The screenshot shows the 'Submit a job' form in Google Cloud Dataproc. The 'Job ID' field is filled with 'triangle-clustering-coefficient'. The 'Region' dropdown is set to 'us-central1'. The 'Cluster' dropdown is set to 'cluster-81c8'. The 'Job type' dropdown is set to 'Spark'. The 'Main class or jar' field is filled with 'LocalGlobalClusteringMain'. The 'Jar files' field is filled with 'gs://ca4015-assignment-3/Graph-Algos-assembly-1.4.9.jar'. Below the 'Jar files' field, there is a note: 'Jar files are included in the CLASSPATH. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix.'

Figure 16: Google cloud job for triangle count and clustering coefficient

The screenshot shows the 'Submit a job' form in Google Cloud Dataproc. The 'Job ID' field is filled with 'label-propagation'. The 'Region' dropdown is set to 'us-central1'. The 'Cluster' dropdown is set to 'cluster-81c8'. The 'Job type' dropdown is set to 'Spark'. The 'Main class or jar' field is filled with 'LabelPropagationMain'. The 'Jar files' field is filled with 'gs://ca4015-assignment-3/Graph-Algos-assembly-1.4.9.jar'. Below the 'Jar files' field, there is a note: 'Jar files are included in the CLASSPATH. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix.'

Figure 17: Google cloud job for label propagation

⁷https://cloud.google.com/dataproc/docs/tutorials/spark-scalausing_sbt

← Submit a job

Job ID *
page-rank

Region *
us-central1
Specifies the Cloud Dataproc regional service, which determines what clusters are available.

Cluster *
cluster-01c8

Job type *
Spark

Main class or jar *
PageRankMain
The fully qualified name of a class in a provided or standard jar file, for example, com.example.wordcount, or a provided jar file to use the main class of that jar file.

Jar file
gs://c42015-assignment-3/GraphAlgo-assembly-1.4.9.jar
Public file with file extension: Public (Access to anonymous: Yes)

Figure 18: Google cloud job for page rank

References

- Ding, Y. (2011), ‘Scientific collaboration and endorsement: Network analysis of coauthorship and citation networks’, *Journal of informetrics* **5**(1), 187–203.
- Jacovi, M., Soroka, V., Gilboa-Freedman, G., Ur, S., Shahar, E. & Marmasse, N. (2006), The chasms of cscw: a citation graph analysis of the cscw conference, in ‘Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work’, pp. 289–298.
- Jeong, Y. K., Song, M. & Ding, Y. (2014), ‘Content-based author co-citation analysis’, *Journal of Informetrics* **8**(1), 197–211.
- Shi, L., Tong, H., Tang, J. & Lin, C. (2015), ‘Vegas: Visual influence graph summarization on citation networks’, *IEEE Transactions on Knowledge and Data Engineering* **27**(12), 3417–3431.
- Son, J. & Kim, S. B. (2018), ‘Academic paper recommender system using multilevel simultaneous citation networks’, *Decision Support Systems* **105**, 24–33.