Web Science
Week 2 - Project Report

Group 34
Cahid Arda Öz - 2872293
Carlos de Bourbon de Parme – 2153874

Question 1 - HOOD

Google has hundreds of billions of web pages indexed[1], while still being able to pick out the most relevant pages to a given query. This is even more remarkable considering that around 15% of daly queries are new to Google, so never searched / indexed before. To deal with this complex challenge simple page rankings are not always enough. In 2019 Google implemented a Natural Language Processor (NLP) named BERT[2], to help the search engine to understand context and unances within a search term. To see if google really does take the meaning of a sentence into account we looked at the influence of so-called “linking words” within the English language. Examples of these words are : to, form, in, out, firstly, also .. etc.
Tradisionaly users would leave these words out since they might confuse the search engine and lead to irrelevant results.

According to Google development BERT can “process words in relation to all the other words in a sentence, rather than one-by-one in order. BERT models can therefore consider the full context of a word by looking at the words that come before and after it” [2]

We made up a few search queries that we expect to perform differently with or without these linking words and compared the first few top ranking websites on the google front page to see which query gives more accurate information. Which query shows web pages that sufficiently answer the question, we will give these results a boolean value of pass (correct) or fail (incorrect).

Long	Short	Correct Long	Correct Short
What are good fruits for in yogurt	Good fruits yoghurt	4/5	2/4
Is the University of Twente a good university?	University Twente Good	5/5	5/5
Are Pancakes for breakfast or for dinner	Pancakes breakfast or dinner	4/5	2/5
Which chipmaker is the most innovative?	Innovative chipmakers	1/5	1/5

Conclusion

When looking at the ratio of correct to incorrect results (correct meaning the website answers the asked question) we see that the longer query with linking words score higher. 14 out of 25 results. Even though this is a small sample size, we do suspect the positive influence of natural language processing on search queries.
We also noticed that the results for both queries are mostly different, again demonstrating that NLP does have an effect on why google searches.

Question 2 - SPAM

Problem Description

In this question, we investigate how one can exploit website ranking algorithms to increase ranking of their website using a Spam Farm.

We simulate multiple graphs representing the spam farm architecture. These graphs consists of three types of pages:

- *Inaccessible Pages*: These webpages represent pages that can not be modified
- *Accessible Pages*: These webpages can be modified by the spammer. Spammer uses these websites to make his/her pages accessible to the crawlers used by the search engines.
- *Spammer Pages*: These pages are created by the spammer and they include:
    - *Target Page (T)*: This page contains the content spammer wants to make more accessible
    - *Supporting Pages*: These pages are used by the spammer to increase ranking of the Target Page.

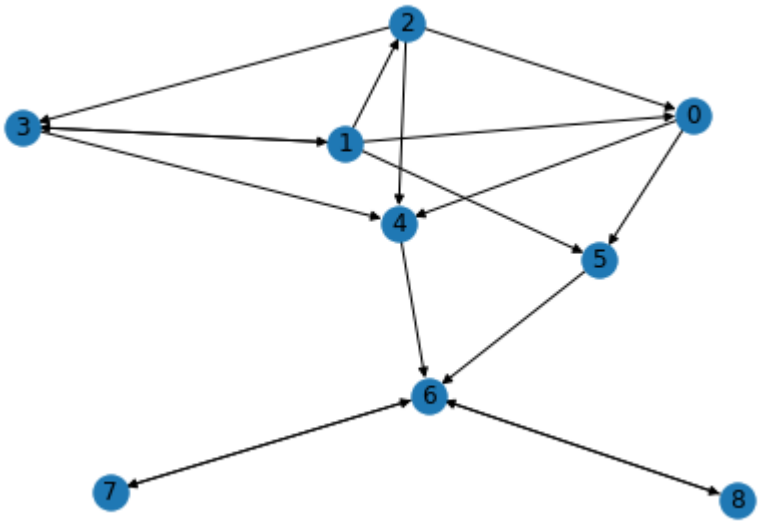
Graphs with this architecture will be investigated to find out how different parameters affect the success of the spammer.

Problem Solution

To investigate the spam farm issue, we wanted to create multiple graphs and observe how different parameters affect the ranking of the Target Page. We created the Graph class to represent these graphs. With the Graph class we are able to:

- Create adjacency matrices corresponding to graphs with spam farm architecture. These graphs are created based on several parameters which are explained in the search space table down below.
- Create transition matrices from the adjacency matrices. These matrices are used to find the PageRank equilibrium.
- Calculate PageRank equilibrium using iteration

Down below, you can see one of the graphs we created using the Graph class. Node labeled “6” is the Target Page. Nodes “7” and “8” are the other spammer pages used to support the Target Page. Nodes “4” and “5” are the accessible pages and the other nodes labeled from 0 to 3 are the inaccessible pages.



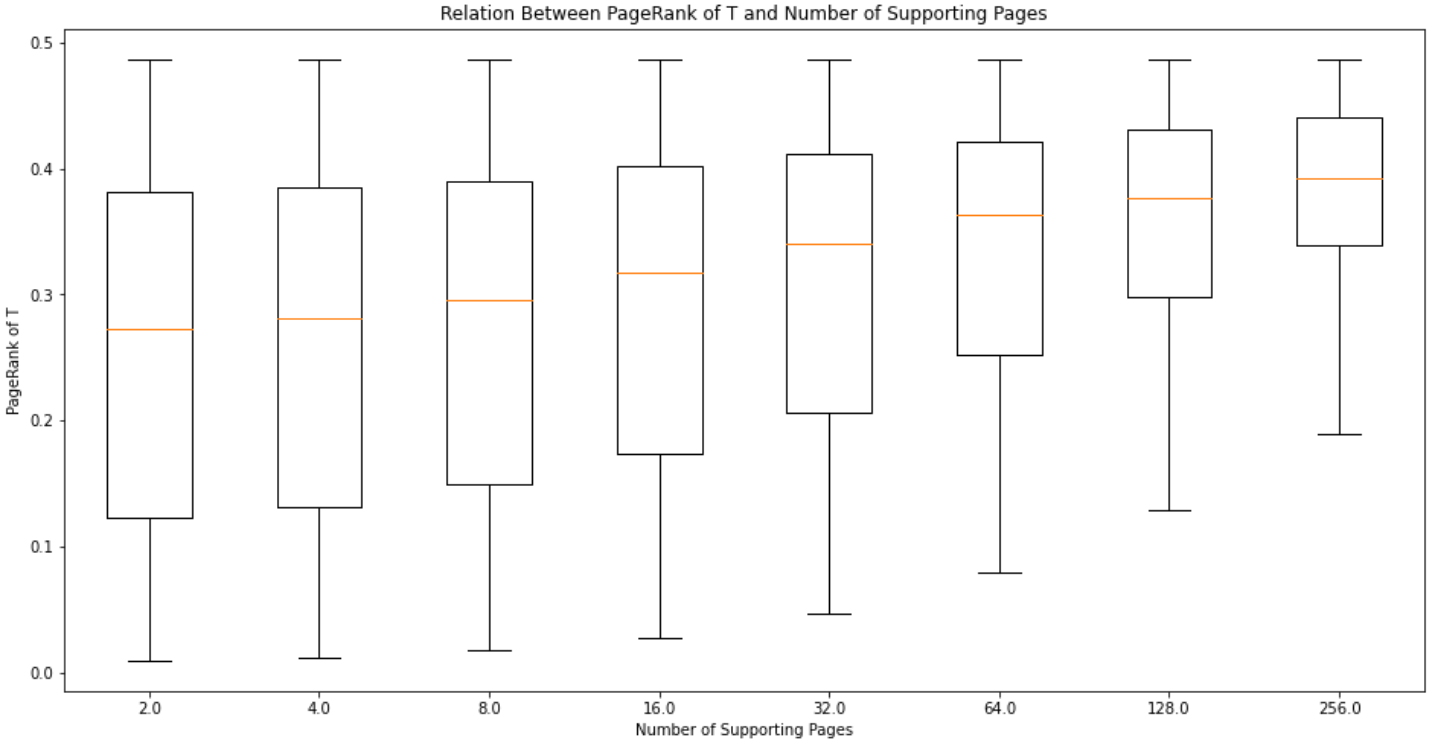
Once we created the Graph class, we used it to create thousands of random graphs from a search space. This search space has several dimensions which are explained below. You can also see the values we used in the search space:

Dimension	**Explanation**	**Search Space**	**Number of Unique Values in the Search Space**
n_inaccessible	number of inaccessible pages in the graph	4, 8, 16, 32, 64, 128, 256	7
n_accessible	number of accessible pages in the graph	1, 2, 4, 8, 16, 32, 64	7
n_supporting	number of supporting pages in the graph	2, 4, 8, 16, 32, 64, 128, 256	8
connectivity	coefficient used in the creation of edges between inaccessible pages. Higher values of connectivity correspond to more edges.	-1, 0, 1	3
beta	β parameter of spider-trap in the PageRank algorithm	0.6, 0.65, 0.7 … 0.9, 0.95	8

In the end, we ended up with 1176 (7*7*8*3) different graphs and we calculated PageRank rankings with each 8 times (There are 8 different values in the search space of the β parameter). For each ranking, we calculated the ranking of the Target Page (T) and r-score as defined in the project description. In the end, we had a dataframe with 9408 (1176*8) rows.

Results

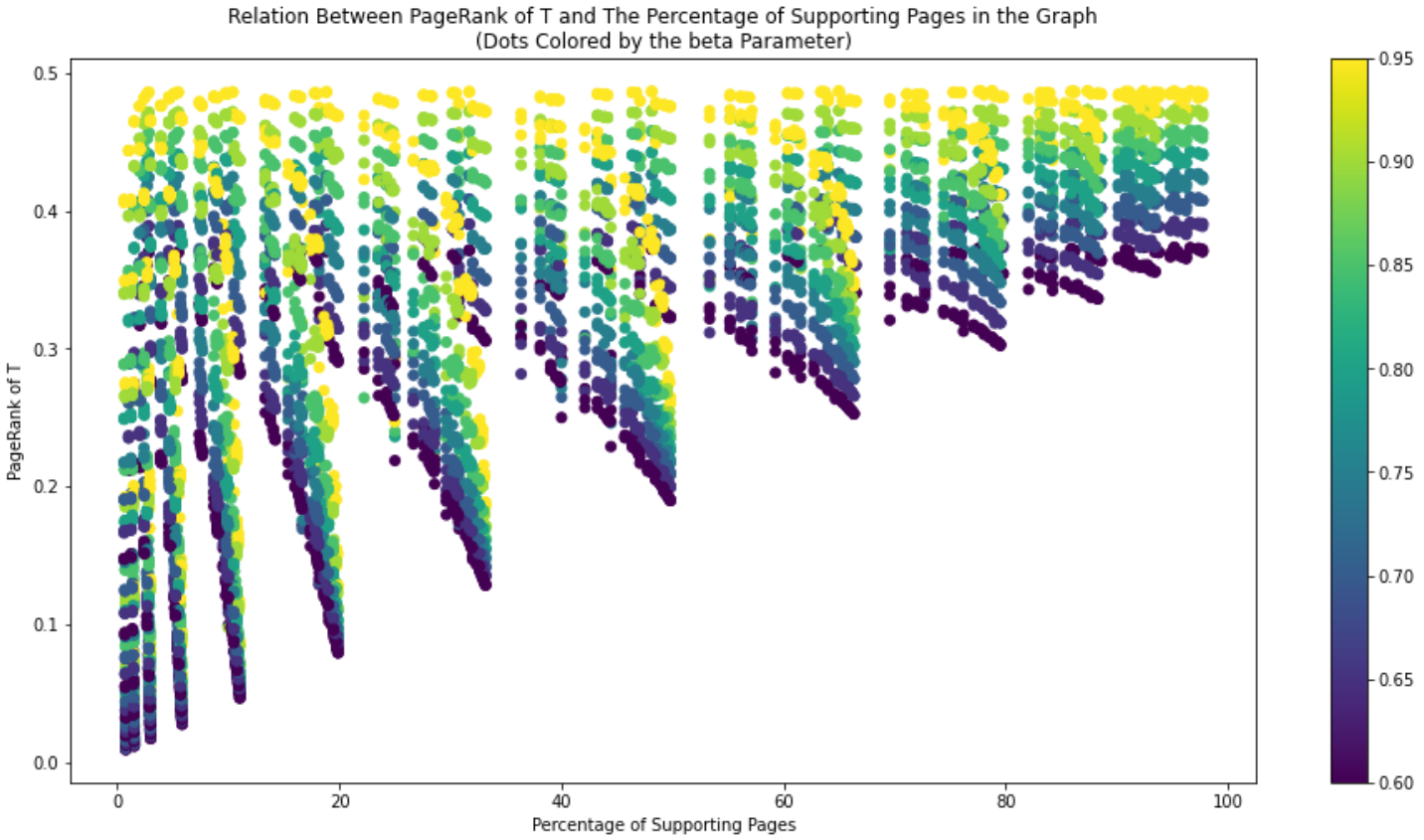
To observe how ranking of T changes with the number of supporting pages in the spam farm (*f*) and external PageRank contribution of accessible pages (*r*); we used the dataframe to create several plots. In the following graph, you can see how PageRank of T changes with number of supporting pages:



We observe that the highest value for PageRank of T is close to 0.5 for all cases. Minimum PageRank achieved in our search space increases as the number of supporting pages increases.

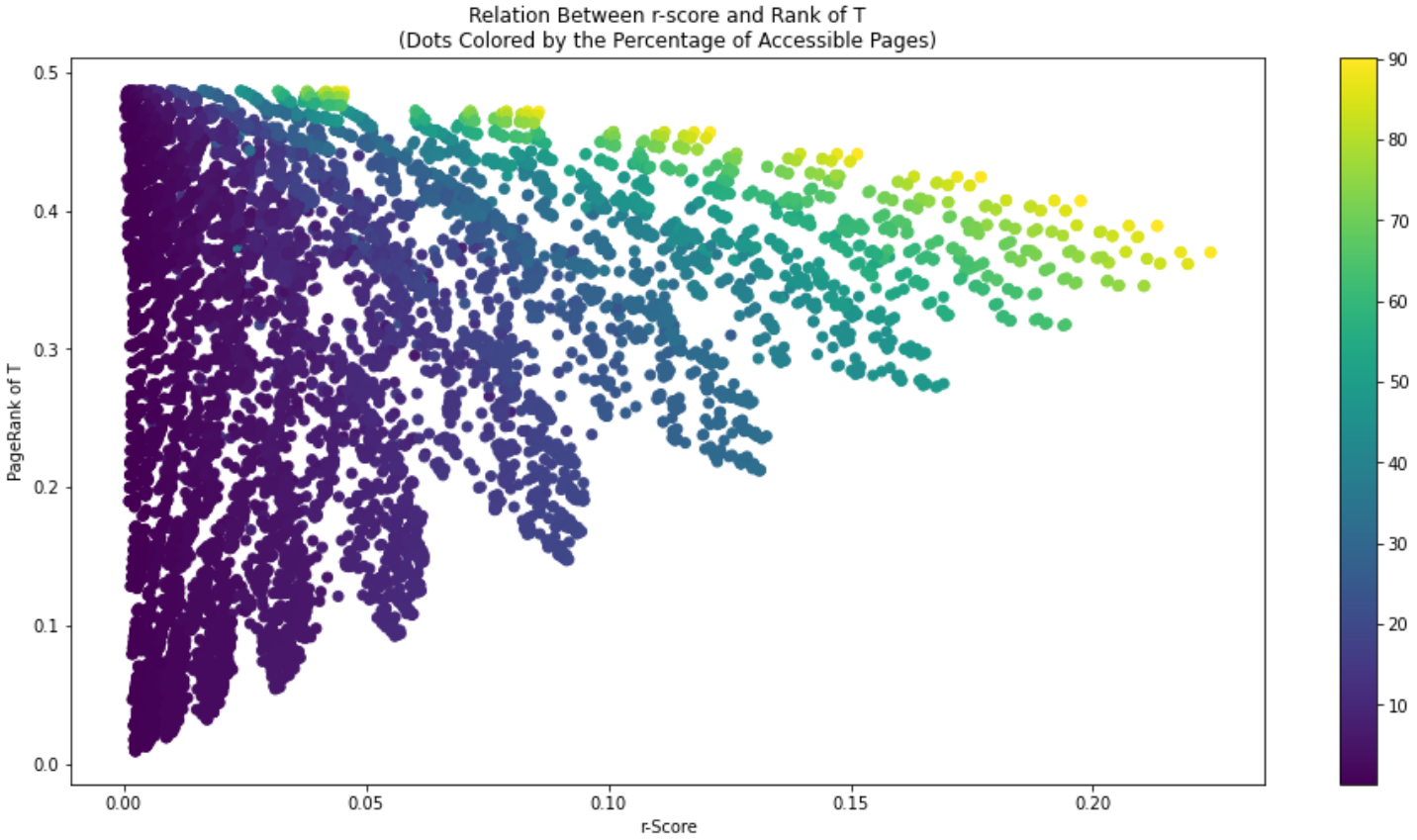
Before moving on to *r-score* however, we wanted to investigate the relation between PageRank of T and number of supporting pages further. We wanted to make sure that what we observed in the previous plot wasn't simply the result of spammer network overwhelming the rest of the graph when the number of supporting pages is set to a high value such as 256.

To look at this relationship from another perspective, we changed the x-axis from "Number of Supporting Pages" to the "Percentage of Supporting Pages". Resulting plot can be seen below:



With this plot, we can clearly see that as the percentage of supporting pages in the graph increases, the minimum value of PageRank of T in the search space increases. An interesting observation we made is that when the β parameter is increased while the percentage of supporting pages is kept constant, PageRank of T increases.

To investigate the relationship between the PageRank of T and the *r-score,* we generated the following plot:



We observe that as the r-score increases, the minimum value for PageRank of T increases in our search space. When we colored the data points by the percentage of accessible pages, we made an additional observation: Higher percentages of accessible pages are more likely to result with high r-score and high PageRank of T.

Question 3 - RANK

Problem Description

In this problem, we investigate how similar ranking algorithms are and what shapes of graphs result in more different or similar rankings for different ranking algorithms.

We will look at three ranking algorithms:

- *InDegree*: Pages are ranked based on the number of in degree edges
- *PageRank*: Pages are ranked based on the PageRank algorithm with spider-trap (β=0.85)
- *HITS*: Pages are ranked based on hubs and authorities

After rankings are calculated with each algorithm, the rankings are converted to ordinal page rankings. Page with the highest ranking gets an ordinal ranking of 1. Then ordinal rankings of each algorithm are compared with Manhattan distance (sum of absolute distance between two vectors).

Several graphs are generated and results are investigated.

Problem Solution

In the solution of this problem, we extended the Graph class we used in question 2 (SPAM). We added the following functionalities:

- Represented graph can be changed by passing a new adjacency matrix
- InDegree and HITS rankings can be calculated
- Represented graph can be drawn

Using these new functionalities, we create hundreds of random graphs and calculate rankings. We also added several helper methods:

- *convert_to_ordinal_rank*: Converts any ranking to ordinal ranking
- get_ranking_manhattan_distance: Calculates Manhattan distance between two ordinal rankings
- *create_random_graph*: Creates a randomly connected graph. This method has a seed parameter so that we will be able to recreate the graphs for further investigation.
- *plot_sample_graphs*: Plots several graphs based on the values in a dataframe

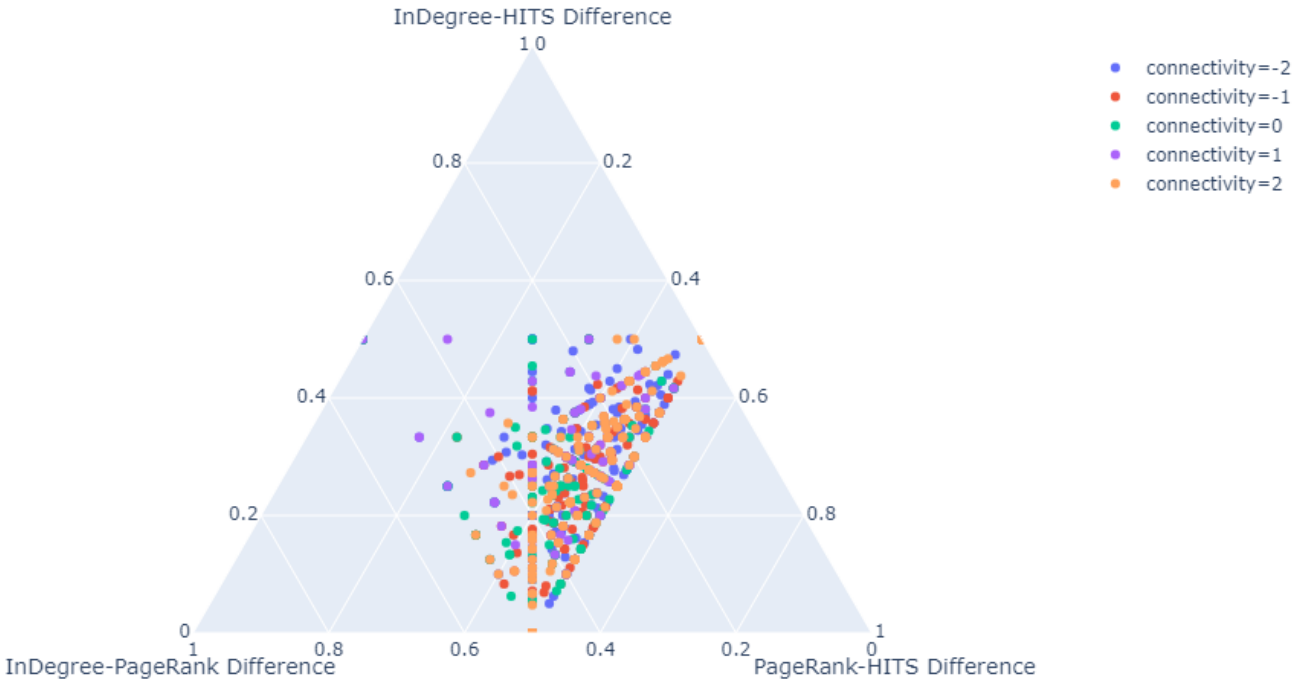
When creating the random graphs, we took a similar approach to question 2 (SPAM) and used a search space. The parameters can be seen in the following table:

Dimension	**Explanation**	**Search Space**	**Number of Unique Values in the Search Space**
n_nodes	Number of nodes in the graph	4, 6, 8, 10	4
connectivity	coefficient used in the creation of edges between inaccessible pages. Higher values of connectivity correspond to more edges.	-2, -1, 0, 1, 2	5

For each point (n_nodes, connectivity) in the search space, we generated 64 random graphs. This resulted in 1280 (4*5*64) graphs. For each graph, we calculated Manhattan distances between the rankings and saved the results in a dataframe together with the parameters used to generate the graph.

Results

Since we have 3 different ranking algorithms, we have 3 different results. We wanted to visualize these results in a ternary plot seen below. Triangle has three axes, each representing Manhattan distance between ordinal rankings of ranking algorithms. Each data point represents a graph. For each graph, Manhattan distances are normalized so that the total of Manhattan distance differences are 1.

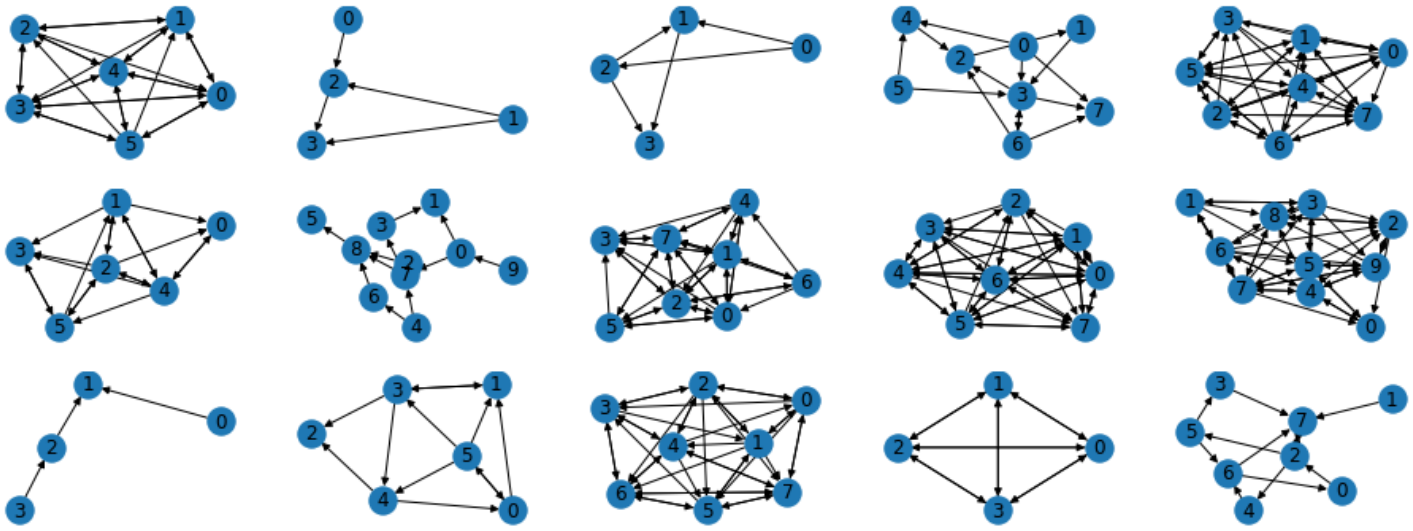


Quick way to interpret the plot above is as follows: If a data point is close to the corner labeled “InDegree-HITS Difference”, then the Manhattan difference between InDegree and HITS is high.

We observe that the data points form a triangle. Here is how we interpret this: Manhattan distance between ranking algorithms A-B is never higher than the sum of Manhattan differences between A-C and B-C. This is because A, B and C are points in an n dimensional space and with the Manhattan Distances, a triangle is formed with A, B and C as corners. The A-B edge of this triangle can never be longer than the sum of A-C and B-C.

We also observe that there are more data points closer to the “PageRank-HITS Difference” corner. This tells us that the difference between PageRank and HITS algorithm tends to be higher compared to the other two axes.

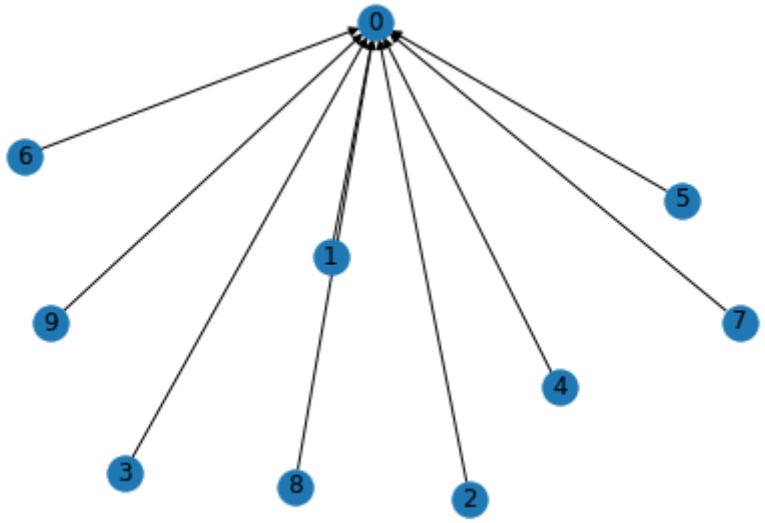
Next, we wanted to visualize some of the graphs. In the following plots, you can see some of the graphs we generated:



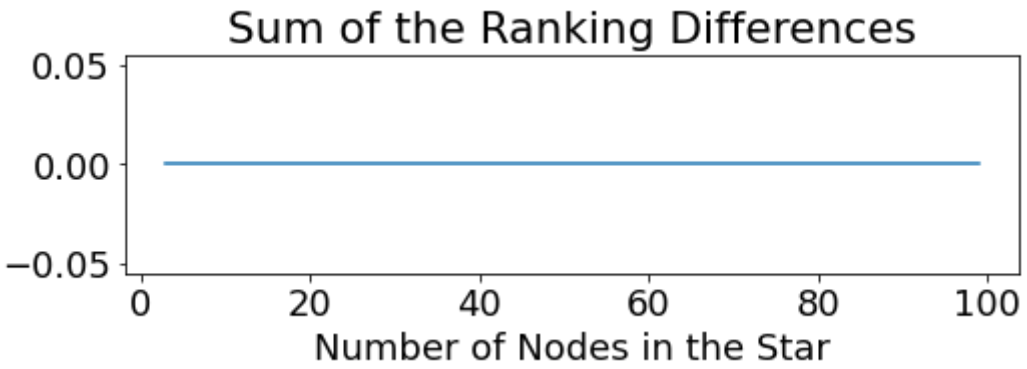
We looked at hundreds of graphs and also looked at the examples provided to investigate the following types of graphs further:

- Directed Bipartite Graph ("The inward star" shape defined in the project description is a subset of this group)
- Fully connected

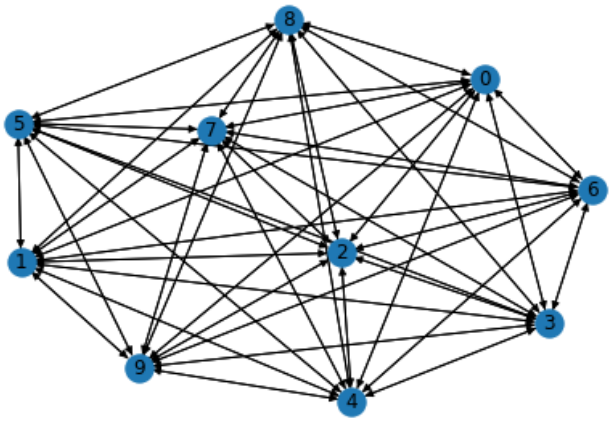
When we generated tens of directed bipartite graphs, we didn't observe any conclusive result. Some had no difference between rankings and some did. We then investigated "The Inward Star". Following is an inward star example:

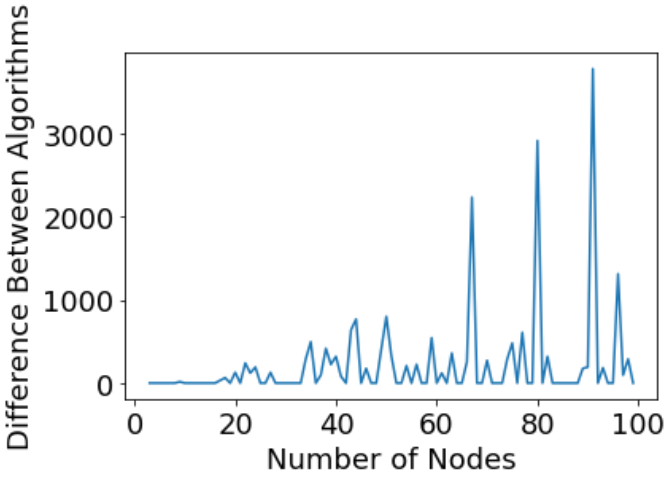


When we created inward stars graphs, we observed that all ranking algorithms were producing the same rankings.

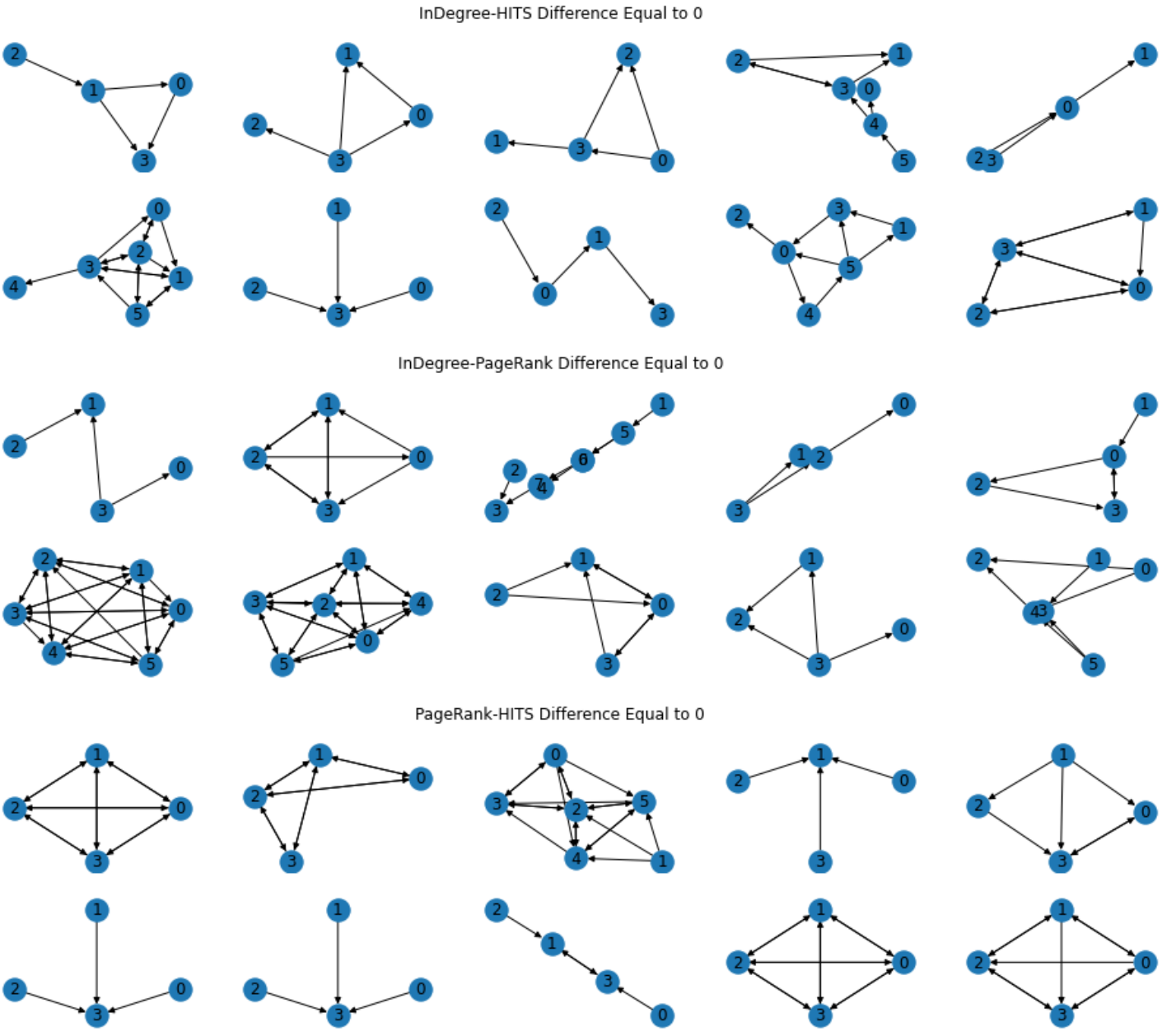


Then we looked at the fully connected graphs. We made the following observation: InDegree and HITS algorithms result with the same rankings while PageRank differs. Following are two plots; an example graph and the ranking difference between PageRank and the other two algorithms based on number of nodes:

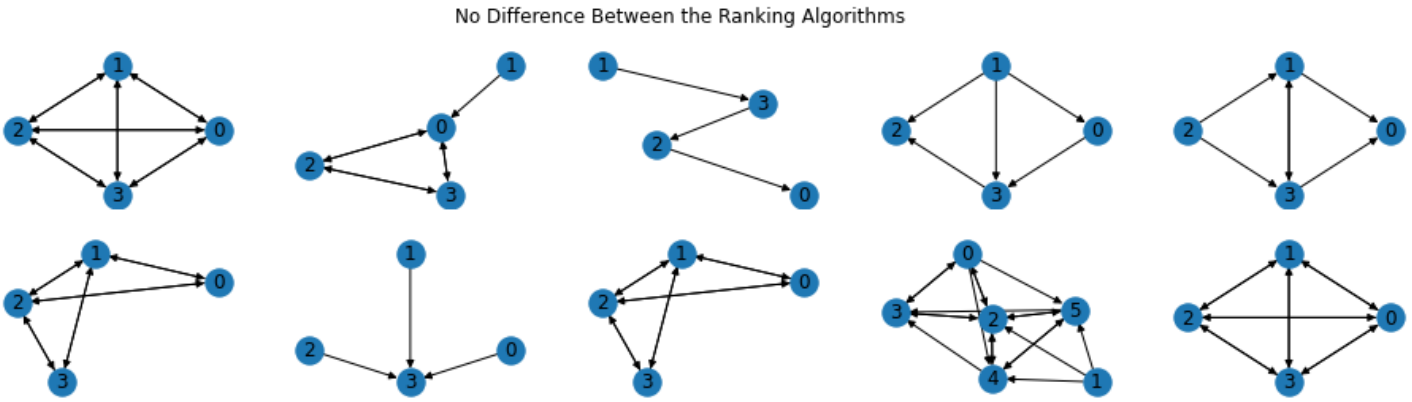




Last, we wanted to visualize the graphs that resulted with the same rankings for two algorithms:



Next, you can see some of the graphs were all the algorithms agreed on the results:



On the matter of generalizing to larger graphs, we believe that what we observed in inward star and fully connected graphs will generalize to larger graphs because we ran tests with multiple parameters.

References:

[1] https://blog.google/products/search/search-language-understanding-bert/
[2] https://blog.google/products/search/search-language-understanding-bert/