ADM HOMEWORK 4
GROUP 8

Mani Niharika Rachuri,
Gaetano Cirianni,
Livio Donnini

1) CREATE THE GRAPH:

At this point we created a graph G, using the authors as nodes and connecting them throw edges which represent the presence of (at least) one shared publication.

To obtain that firstly, we had to get “author_id” and all their publications (id_publication_int). In “list_of_authors_ID” we stored all the “author_id(s)” and in “publications_for_author_id” we get all the publications for given “author_id”.

· For example: for the given author_id: 256176 we got all these publications:
[254068, 254234, 342298, 354200, 382418, 425337, 427289, 486893,
494767, 544318, 638333, 658645, 731279, 866799, 869138, 954951,
979491]

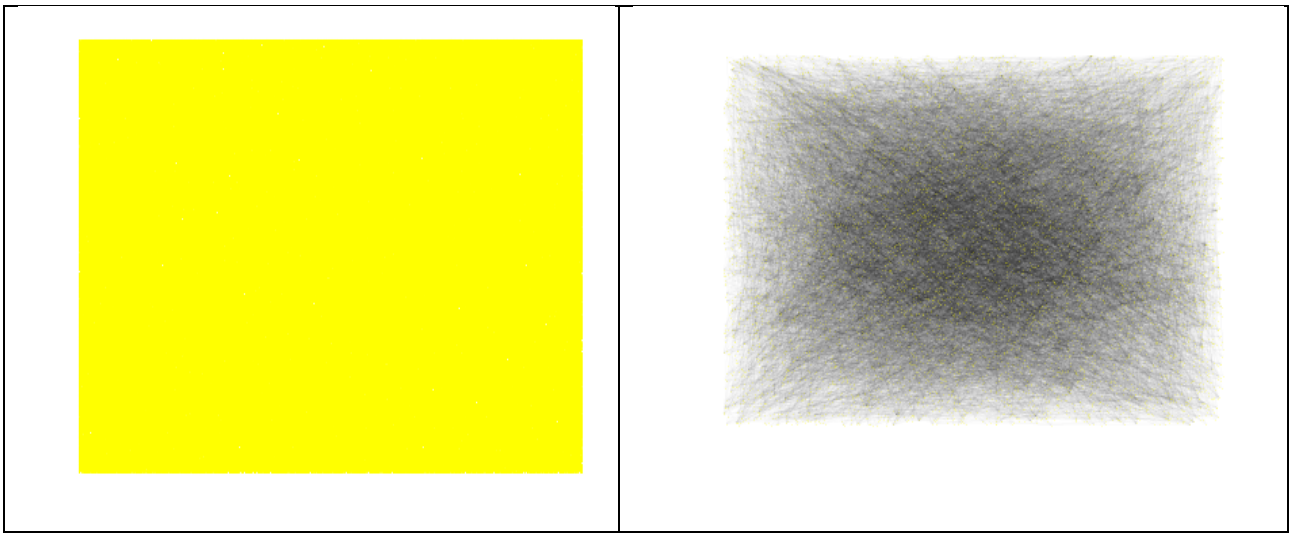
Then we created a dictionary “authorsDict” in which we stored all the pubblications for all the authors. Here “author_id” is the key and the value is rapresented by a list of all specific publications. For an easy access we loaded all the “id_publication_int” in pubIdList list. To build the edges throw itertools.combinations we created all the possible pairs for each element in the list i.e. the publications.

With the full data set we were able to build a graph of 904664 nodes and 3679331 edges weighted using the below formula:

$$weight(a1, a2) = 1 - J(p1, p2)$$

This weight is high if the graph’s vertices are different, that for how we have built the graph means they share very few publications. If its value is low, the nodes will be similar, this means they share a conspicuous number of publications.

Full data set:	Reduced data set:
Name: my graph Type: Graph Number of nodes: 904664 Number of edges: 3679331 Average degree: 8.1341	Name: my graph Type: Graph Number of nodes: 7771 Number of edges: 16489 Average degree: 4.2437



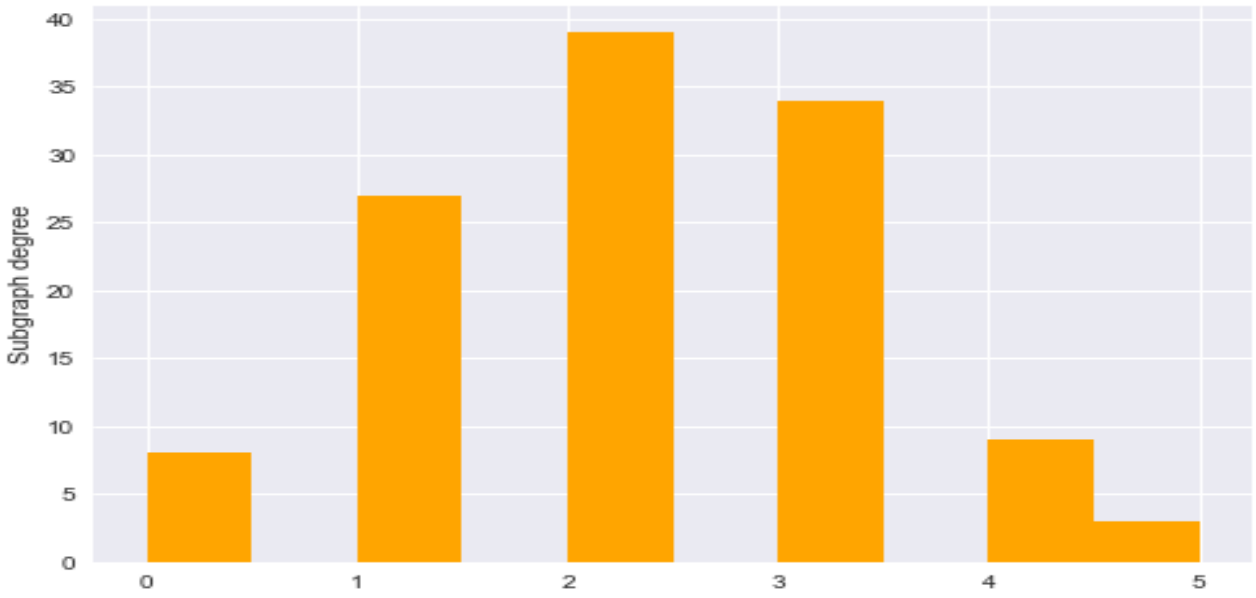
2) STATISTICS AND VISUALIZATIONS:

*a) Subgraph:*Here we will give “id_conference_int” as an input and we get a subgraph induced in wich authors are connected if they share at least one publication at a given input conference.

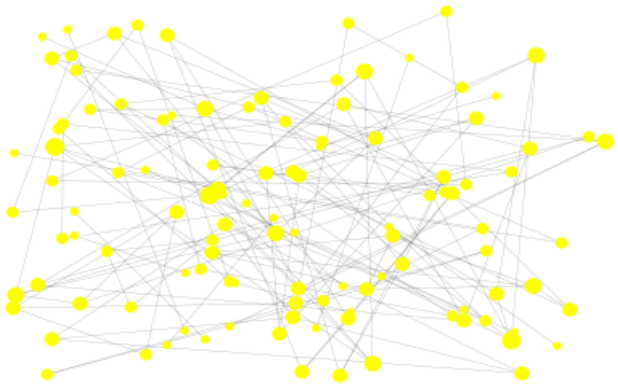
· For the input conference “3052” we obtained:

Name: my graph
Type: Graph
Number of nodes: 120
Number of edges: 129
Average degree: 2.1500

Subgraph Histogram:

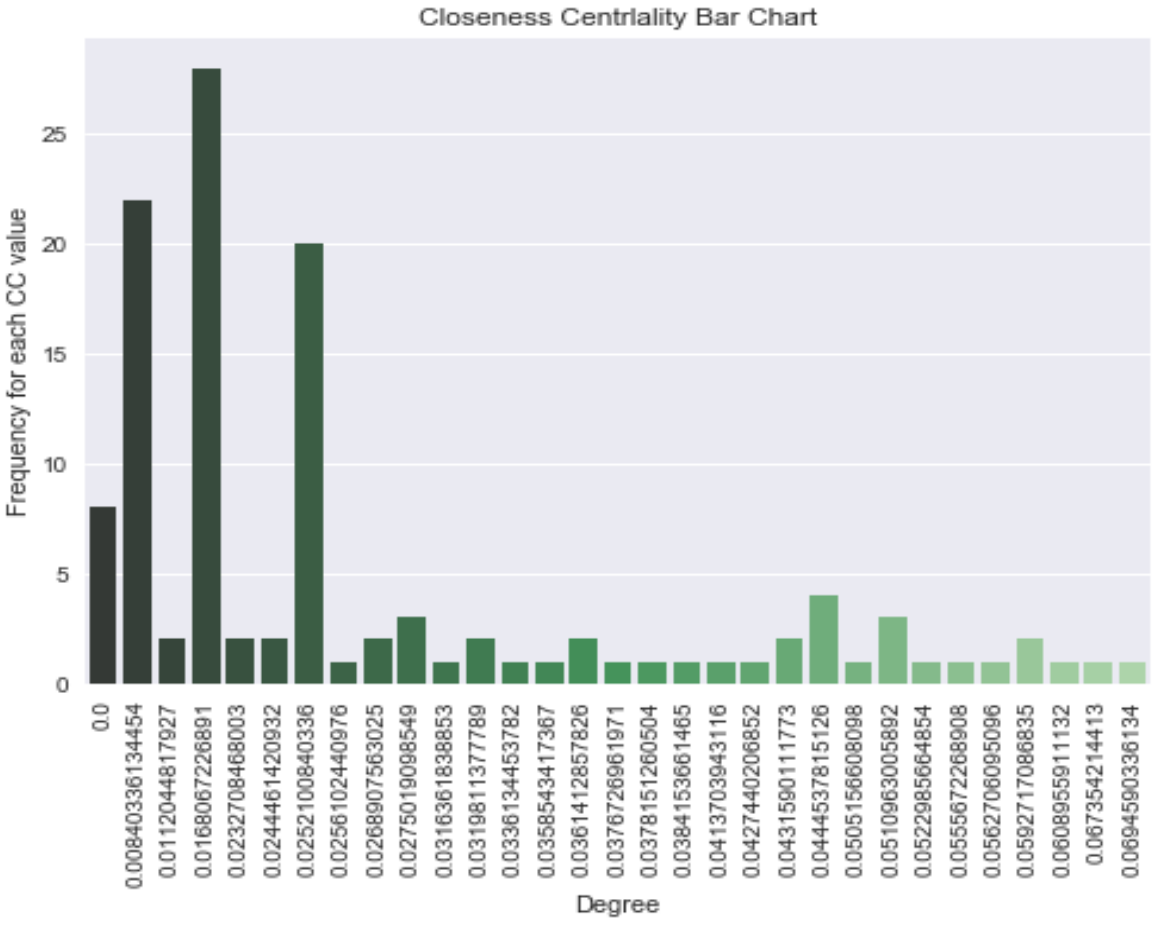


Subgraph Plot:



Centralities Measures:
Here we applied these measure to our data to find out the most influenced node (author) in this network.

· *Closeness Centrality:* represents the mean distance between a given node and all the others. For this we have to import "statistics" package in which we have a function called closeness, which returns this measure.

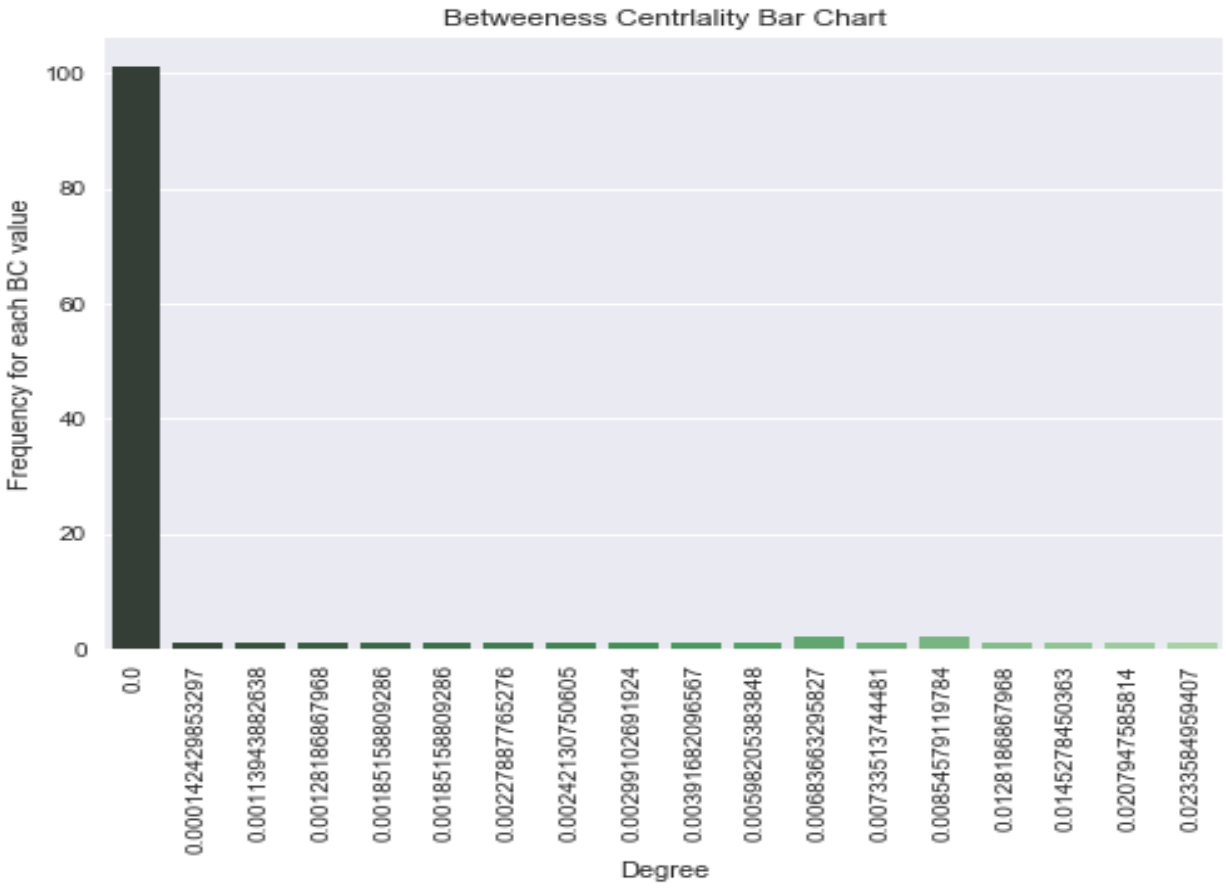


From the above the 0.0168067226891 degree has the highest frequency, more than 25 times. That means all the nodes which have this degree are “most influenced” and they are closest to all the other nodes. Closest implies they share lots of publications/conferences.

```
for author, measure in closenessDict.items():
    if measure == 0.01680672268907563:
        print(author)
```

- 256124
- 256276
- 9772
- 114626
- 114625
- 114483
- 202882
- 256135
- 9451
- 255280
- 255281
- 3326
- 255339
- 205380
- 115228
- 210891
- 255394
- 255395
- 176994
- 73799
- 24151
- 255902
- 124828
- 239007
- 189237
- 256123
- 256124

· *Betweenness Centrality:* is a measure of centrality based on the shortest paths. For every pair of vertices in a connected graph, there exists at least one shortest path between the vertices such that the sum of the weights of the edges, for a weighted graph is minimized.

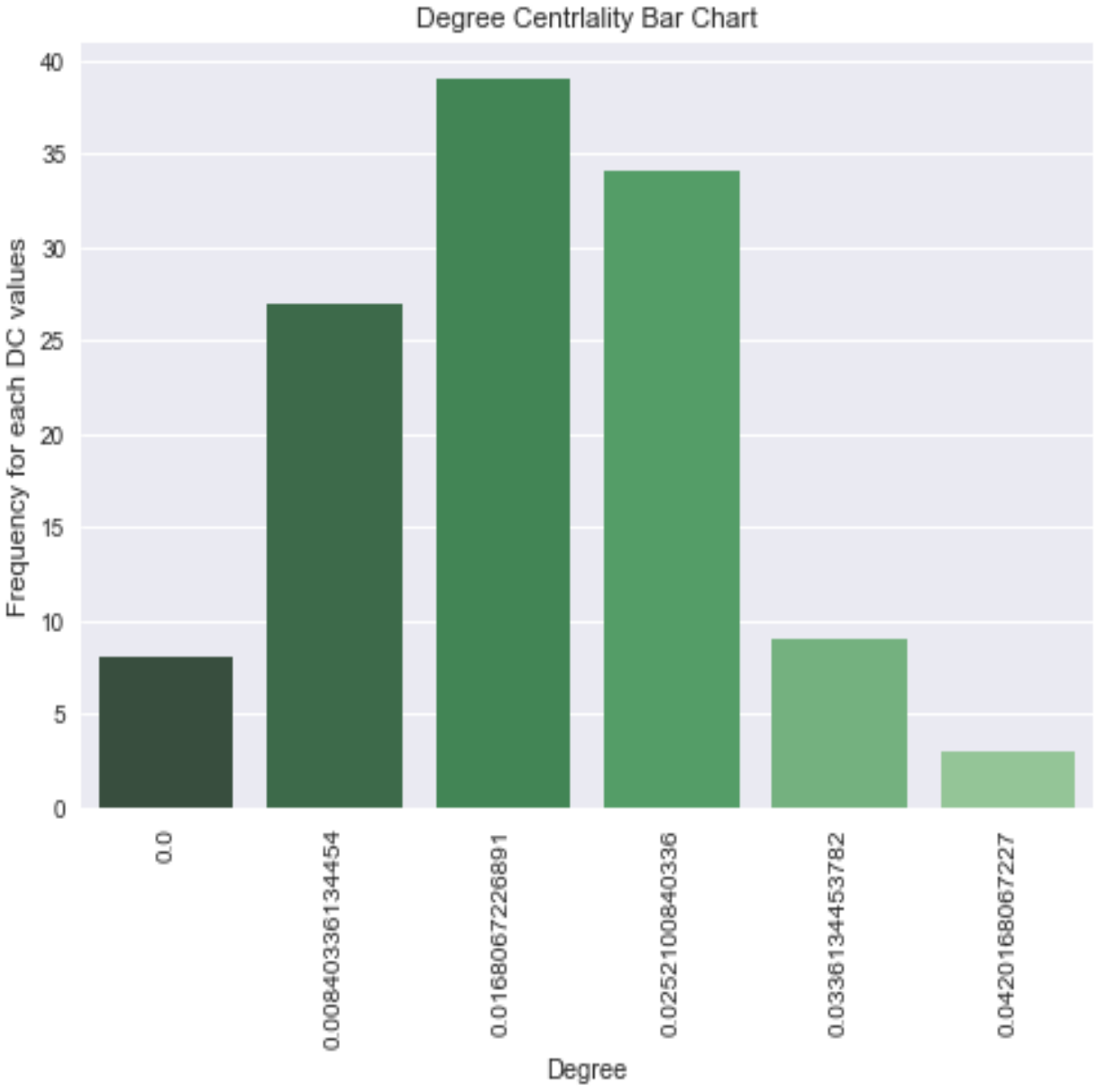


As we know these measures returns some value(degrees) to the node. From the above graph we see that the nodes with the First degree in the graph are the influenced nodes which acted like a bridge and presented in the shortest part, for this we suppose it lies in the middle of the graph. This part helps us in finding the measure for quantifying the control of a node on the communication between the other node in a network.

To find them:

```
for author, measure in betweenessDict.items():
    if measure == 0.0:
        print(author)
```

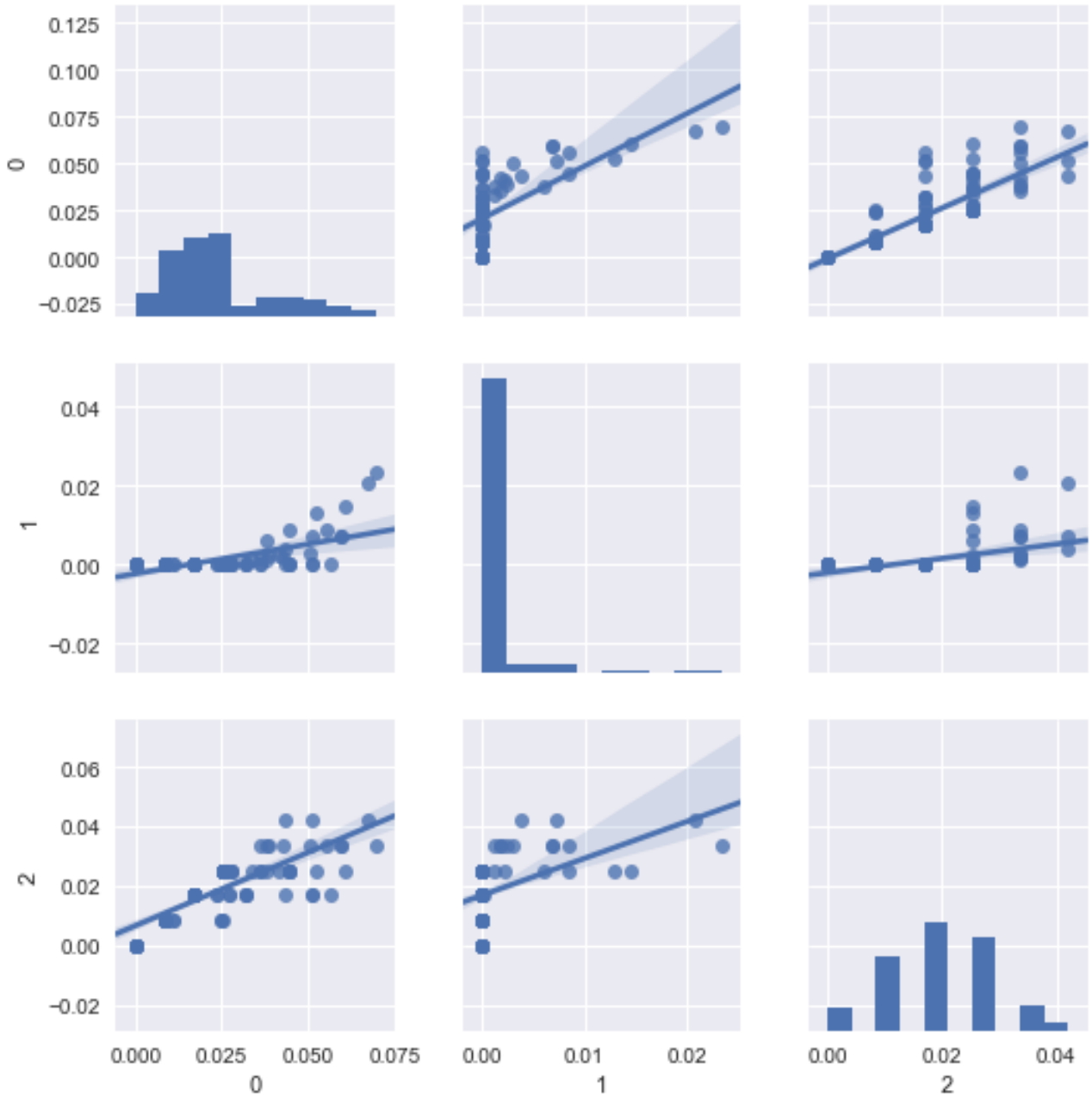
Degree Centrality: is a simple centrality measure that counts how many neighbors a node has.



From the above graph we see that the authors with third degree (in the x-axis of the graph) has many neighboring nodes. That is those nodes are connected to most of the total nodes in this network.
To find them:

```
for author, measure in degreeDict.items():
    if measure == 0.0:
        print(author)
```

Now by plotting all these three centralities by using regression function we get:

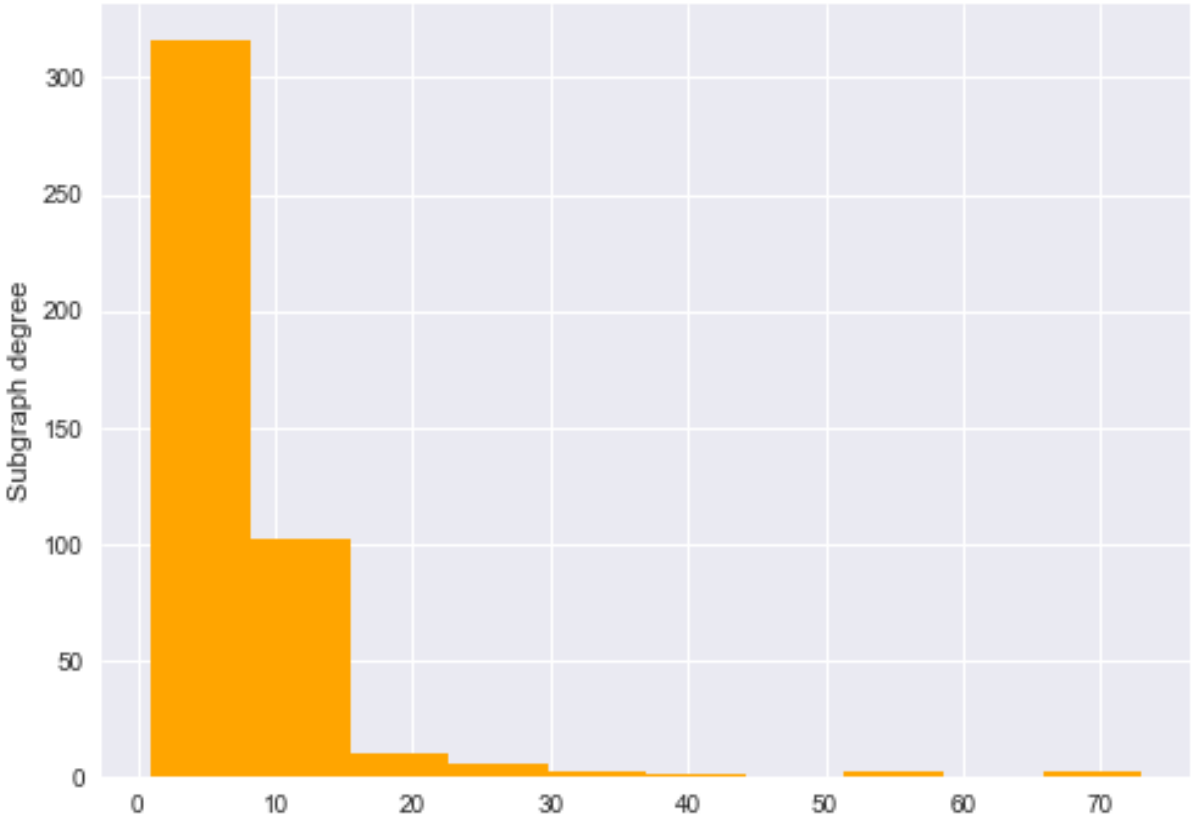


b) Neighbors: We have defined a function called “neighborhood” function which finds for each node its neighbors until a defined distance “d” from the starting node.

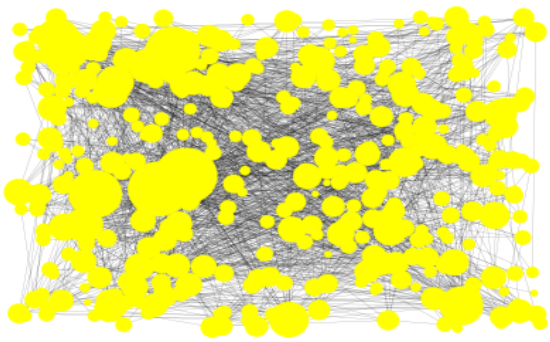
· For input: “author_id” = “43” with the distance d = “2” we got the following plot:

Name: my graph
Type: Graph
Number of nodes: 440
Number of edges: 1548
Average degree: 7.0364

Subgraph Histogram:



Subgraph Plot:



3) SOME GENERALIZED VERSION OF THE ERDOS NUMBER:

a) Verifying the presence of a path between Aris and the input author and finding the weight:
Firstly, we've to check if the input author has any path connected to "aris". If yes then we'll find the shortest path between them. Here We used the defined function "shortest path" for finding.

· Let us give the input author id:256133

· We got yes and then now we calculated the shortest path.
"0.15554478489909918" is the shortest path between them.

a) Group number:
Here we took in input a subset of nodes (cardinality smaller than 21) and returns for each node of the graph, its groupNumber, defined as the minimum Shortest path(SP) between the nodes. Mostly the group number lies between the 0 to 1.

Let us take the input subset of nodes as: "256176 255207 255394 20199 176994"
Output:
The Group Number for 1 with 256176 is: 0.37068398095740773
The Group Number for 2 with 256176 is: 0.12068398095740773
The Group Number for 3 with 256176 is: 0.6206839809574077
The Group Number for 4 with 256176 is: 0.04044427226946723
The Group Number for 5 with 256176 is: 0.10711093893613388
The Group Number for 6 with 256176 is: 0.03298063214861979
The Group Number for 7 with 256176 is: 0.5329806321486198
The Group Number for 8 with 256176 is: 0.5329806321486198
The Group Number for 9 with 256176 is: 0.06539311443578955
The Group Number for 10 with 256176 is: 0.03313504991966054
The Group Number for 11 with 256176 is: 0.032131679039622774
The Group Number for 12 with 256176 is: 0.046530923726896045
The Group Number for 13 with 256176 is: 0.546530923726896
The Group Number for 14 with 256176 is: 0.046530923726896045
The Group Number for 15 with 256176 is: 0.09202218590225575
The Group Number for 16 with 256176 is: 0.08939060695488743
The Group Number for 17 with 256176 is: 0.039390606954887386
The Group Number for 18 with 256176 is: 0.18468289731949006
The Group Number for 19 with 256176 is: 0.07424737695749906
The Group Number for 20 with 256176 is: 0.11585500125544157
The Group Number for 21 with 256176 is: 0.08252166792210824
The Group Number for 22 with 256176 is: 0.39679507898826083
The Group Number for 23 with 256176 is: 0.2467950789882608
The Group Number for 24 with 256176 is: 0.5967950789882608
The Group Number for 25 with 256176 is: 0.03659086364928066
The Group Number for 26 with 256176 is: 0.07131038692460356
The Group Number for 27 with 256176 is: 0.07131038692460356
The Group Number for 28 with 256176 is: 0.041898622218721204
The Group Number for 29 with 256176 is: 0.07131038692460356
The Group Number for 30 with 256176 is: 0.07131038692460356

The Group Number for 31 with 256176 is: 0.10638132835090541
The Group Number for 32 with 256176 is: 0.4484865915088001
The Group Number for 33 with 256176 is: 0.1326971178245896
The Group Number for 34 with 256176 is: 0.1326971178245896
The Group Number for 35 with 256176 is: 0.1326971178245896
The Group Number for 36 with 256176 is: 0.07554501175504069 … etc.