UNIVERSITAT ROVIRA i VIRGILI

A1. Structural descriptors of complex networks

Complex Networks (MAI)

Professor

GÓMEZ JIMÉNEZ, Sergio

Team members

BEJARANO SEPULVEDA, Edison Jair
edison.bejarano@estudiantat.upc.edu
Walzthöny Kreutzberg, Eric
Eric.Walzthony@estudiantat.upc.edu

Universitat Rovira i Virgili
Avinguda dels Països Catalans, 26, 43007 Tarragona, Spain
Master's degree in Artificial Intelligence
Tarragona, 2021

Contents

Abstract

In this assignment, we studied the most fundamental properties of networks, such as the structural descriptors. These describe their overall structure as well as having an in-depth understanding of the structural analysis of complex networks. This was done by using the Pajek files provided in the "A1-Networks" folder. This folder contained three distinct types of complex networks, mainly: Toy, Model and Real. The first two can be thought of synthetic networks which can be generated programatically whilst the later is an example of a "real-life" complex network.

To perform this analysis, we used python as a programming language and taking advantage of its open-source community and packages, used the "igraph" module. This module is specifically designed to analyze networks (as it is adapted from the R-package). For the descriptors we used a OOP approach to generate classes to store the data in, we found this approach to me more modular. Finally, the summary files were written in the specified which contained the varied descriptors as well as the Histograms (PDF and CCDF) for the model and real networks.

Overall, the approach to analyze networks can vary between the different packages (iGraph, NetworkX) or even different languages. We found that experimentation and understanding of the basic network operations are essential in having a precise understanding of what type of network we are confronted as well as its basic analytics. All the implementation of this work can be consulted in:

https://github.com/EjbejaranosAI/Complex$_N etworks.git$

Keywords: Complex networks, Descriptors, Networks structures, Real networks , Degree distribution.

1 Methodology description

For this project, the process was structured in the following parts:

- **Problem Statement:** In this step, it was necessary to understand the basic concepts in complex network structures, such as terminology and mathematical concepts, in order to develop some solutions. We took advantage of Jupyter Notebooks in order to prototype our solutions before pushing them to our final code base, this helped us understand the scope of the problem as well as rapidly sharing our results.
- **Decisions:** Consequentially, our design decisions were directed by the previous step. We found that quick prototyping was more important than optimized code — as this could be implemented a posteriori. For this reason, we chose python and its complementary library "igraph" to resolve the task at hand. Following the standard OOP approaches, we created a set of classes in order to have an abstraction of methods, file-loading, and data separate from the main code base. For each of the points to resolve we generate a specific class.
- **Software:** Apart from using python as a programming language, we used a standard set of libraries: Numpy for vectorization of expressions and handling of matrices. Pandas, for generating DataFrames and writing to csv. Alongside iGraph, these were the most used libraries apart from some small operating system management ones and file wrangler such as: Glob, OS, and Zipfile.
- **Analysis:** For the analysis of the networks, we took advantage of methods already found in the library. The main descriptors that were needed are essential metrics that the majority of the network analysis libraries have (NetworkX, GraphTool iGraph). We took advantage of these to calculate the majority of the descriptors. For some of these, we had to use these methods in custom functions in order to extract the specified descriptor.
- **Coordination:** Within the scope of this project, the organization was well defined from the beginning. Thus, the other part would be the coordination between the team-members, such that we could divide and organize the tasks. This was done by dividing the tasks by either member and having a quality control or peer-review in order to approve of the code and methods used.

2 Table of network descriptors - Part a

In this section we will include the table shown below, Table 1, which is the summary of calculations for the descriptors in the A1-Network directory.

2.1 Results

Table 1 – Calculated descriptors of all networks A1-NETWORKS

File_Name	Folder	Edges	Nodes	Avg. Path Length	Assortativity
rb25.net	toy	66	25	2.0333	-0.1635
wheel.net	toy	16	9	1.5556	-0.3333
graph3+1+3.net	toy	8	7	2.1905	-0.6
20x2+5x2.net	toy	404	50	2.3878	0.9186
graph4+4.net	toy	13	8	1.8571	-0.0833
grid-p-6x6.net	toy	72	36	3.0857	
star.net	toy	8	9	1.7778	-1
circle9.net	toy	9	9	2.5	
zachary_unwh.net	real	78	34	2.4082	-0.4756
dolphins.net	real	159	62	3.357	-0.0436
airports_UW.net	real	14142	3618	4.4396	0.0462
PGP.net	real	24340	10680	7.4855	0.2395
SF_1000_g2.5.net	model	1905	1000	4.6149	0.02
SF_1000_g2.7.net	model	1668	1000	5.4688	-0.002
homorand_N1000_K6_0.net	model	2994	1000	4.1913	0.1919
homorand_N1000_K4_0.net	model	2000	1000	5.64	
ER1000k8.net	model	3956	1000	3.5698	-0.0168
SF_1000_g3.0.net	model	1517	1000	5.9651	-0.0085
ws2000.net	model	6000	2000	4.5111	-0.0762
BA1000.net	model	3990	1000	3.1833	-0.0542
256_4_4_4_13_18_p.net	model	4598	256	2.6511	0.0007
ER5000k8.net	model	19980	5000	4.3797	-0.0555
SF_500_g2.7.net	model	859	500	4.8759	-0.0256
256_4_4_2_15_18_p.net	model	4548	256	2.7821	0.0286
rb125.net	model	426	125	2.3032	-0.1837
ws1000.net	model	3000	1000	4.0913	-0.0999
File_Name	**Diameter**	**Transitivity**	**Min. Degree**	**Avg. Degree**	**Max. Degree**
rb25.net	4	0.9023	4	5.28	20
wheel.net	2	0.6243	3	3.5556	8
graph3+1+3.net	4	0.6667	2	2.2857	3
20x2+5x2.net	4	0.9716	4	16.16	22
graph4+4.net	3	0.875	3	3.25	4
grid-p-6x6.net	6	0	4	4	4
star.net	2	0	1	1.7778	8
circle9.net	4	0	2	2	2
zachary_unwh.net	5	0.5879	1	4.5882	17
dolphins.net	8	0.3029	1	5.129	12
airports_UW.net	17	0.6228	1	7.8176	250
PGP.net	24	0.4403	1	4.5581	206
SF_1000_g2.5.net	10	0.0096	2	3.81	30
SF_1000_g2.7.net	12	0.0067	2	3.336	24
homorand_N1000_K6_0.net	6	0.0038	5	5.988	6
homorand_N1000_K4_0.net	9	0.002	4	4	4
ER1000k8.net	6	0.0081	1	7.912	17
SF_1000_g3.0.net	13	0.0052	2	3.034	26
ws2000.net	7	0.0033	3	6	13
BA1000.net	5	0.0354	4	7.98	115
256_4_4_4_13_18_p.net	4	0.5113	20	35.9219	50
ER5000k8.net	6	0.0014	4	7.992	17
SF_500_g2.7.net	12	0.0078	2	3.436	22
256_4_4_2_15_18_p.net	5	0.7331	30	35.5312	46
rb125.net	4	0.8373	4	6.816	100
ws1000.net	6	0.0044	3	6	13

2.2 Discussion

As we can observe in Table 2, it is observed that it is not necessary to have more edges and nodes to have better transitivity, for example 20x2+5x2.net and rb25.net in Toy networks.

Table 2 – Table with the maximum values of each descriptor for each type of network

Toy networks		
	Network name	Value
Max number of Edges	20x2+5x2.net	404
Max number of nodes	20x2+5x2.net	50
Max Avg.Path length	grid-p-6x6.net	3.0857
Max Assortativity	20x2+5x2.net	0.9186
Max Diameter	grid-p-6x6.net	6
Max Transitivity	rb25.net	0.9023
Max avg degree	20x2+5x2.net	16.16
Real networks		
	Network name	Value
Max number of Edges	PGP.net	24340
Max number of nodes	PGP.net	10680
Max Avg.Path length	PGP.net	7.4855
Max Assortativity	PGP.net	0.2395
Max Diameter	PGP.net	24
Max Transitivity	airports_UW.net	0.6228
Max avg degree	airports_UW.net	7.8176
Model networks		
	Network name	Value
Max number of Edges	ER5000k8.net	19980
Max number of nodes	ER5000k8.net	5000
Max Avg.Path length	SF_1000_g3.0.net	5.9651
Max Assortativity	homorand_N1000_K6_0.net	0.1919
Max Diameter	SF_1000_g3.0.net	13
Max Transitivity	rb125.net	0.8373
Max avg degree	256_4_4_4_13_18_p.net	35.9219

Another relevant aspect of the results of the descriptors is that the model networks have more transitivity than the real networks. In addition, Real networks present networks with greater diameter and assortativity and, consequently, greater average path length.

3 Table of Airports nodes descriptors - Part b

In Table 3, shown below, we can see the descriptors which were derived for the Airports network. Since, this is an example of a real network, we can clearly see the connections between countries and their respective airports. For example, main hubs like London Heathrow, shown as LON, have a very large degree, which reflects its importance in the international travel. Without more information, such as economical or logistics, we can only assume that this can also represent an important point for international trade.

3.1 Results

Table 3 – Results for the Airports nodes descriptors - Part b

ID Node	Airport	Degree	Strength	Cluster Coefficient	Average Length
33	ADA	7	7	0.71428571	5.34221848
62	AGU	7	7	0.76190476	5.17638879
115	AMS	192	192	0.14283377	4.2099746
173	ATL	172	172	0.1378349	4.52885451
239	BCN	80	80	0.32848101	5.29630424
526	CHC	20	20	0.25263158	5.22628257
725	DJE	20	20	0.7	5.3745885
925	FRA	237	237	0.11696346	4.35164967
1712	LON	242	242	0.11234183	4.19376168
1953	MOW	186	186	0.09584423	4.40847869
2160	NYC	179	179	0.15755445	4.59452185
2268	PAR	250	250	0.08915663	4.28330769
2876	TBO	2	2	1	6.18249901
3233	WAW	55	55	0.45858586	5.27124608
3613	ZVA	1	1		9.1875381
ID Node	**Airport**	**Max Length**	**Eigen Vector Centrality**	**Betweenness**	**PageRank**
33	ADA	12	0.05336155	86.13209062	0.00018651
62	AGU	12	0.02559141	37.68371295	0.00019367
115	AMS	11	0.85577995	264799.7485435	0.00401098
173	ATL	12	0.60864673	162809.45400881	0.00381899
239	BCN	12	0.44507124	12636.35135909	0.00160535
526	CHC	11	0.02089875	22021.5322141	0.00084125
725	DJE	12	0.15892481	953.83921258	0.00041071
925	FRA	11	0.97602259	428847.8032452	0.00496923
1712	LON	11	1	555788.52171987	0.00526564
1953	MOW	11	0.5823272	341434.48646073	0.00511005
2160	NYC	12	0.80100266	453081.88267624	0.00389207
2268	PAR	11	0.89986318	610925.94208172	0.00614411
2876	TBO	13	0.00061431	0	0.00012092
3233	WAW	12	0.37541514	10181.73242408	0.00115741
3613	ZVA	16	0.00000001	0	0.00013213

3.2 Discussion

4 PDF and CCDF of the degree distributions - Part c

In this section, we will show the graphs for the 5 different models (3 from the toy and 2 from the real Dataset), with their respective PDF and CCDF, as well as showing in which graph the log-log scale is used and preferred.

For the following graphs the log-log scale would be prefered: ER5000k8, and ws1000. We chose to use this scaling here since there can be a skewness towards larger values, and changing to the log-log scale gives a better representation of the data.

The remaining graphs, were used in a normal scale, i.e. linear scaling.

4.1 Results

- **model/ER5000k8.net**

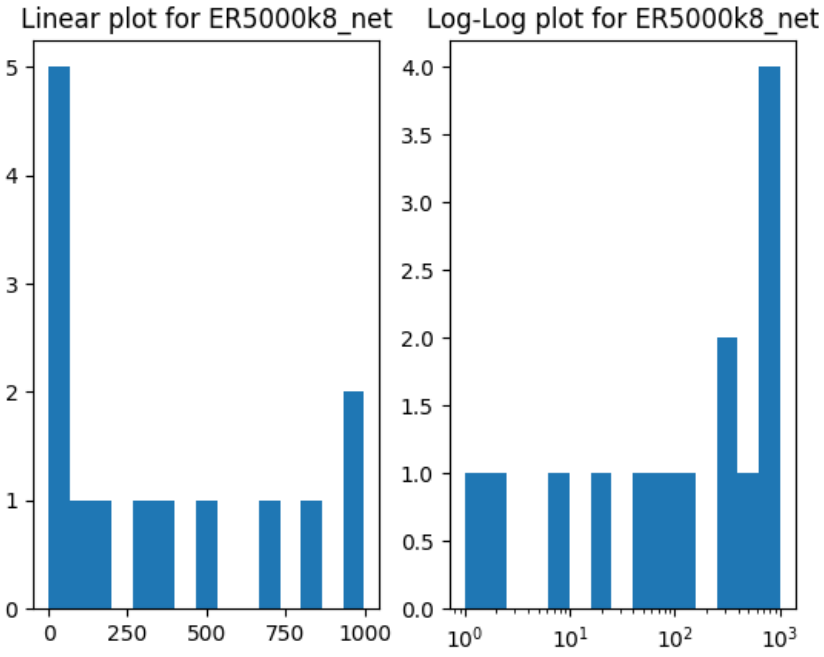

Figure 1 – Histograms of the degree distribution (PDF) for ER5000k8.net

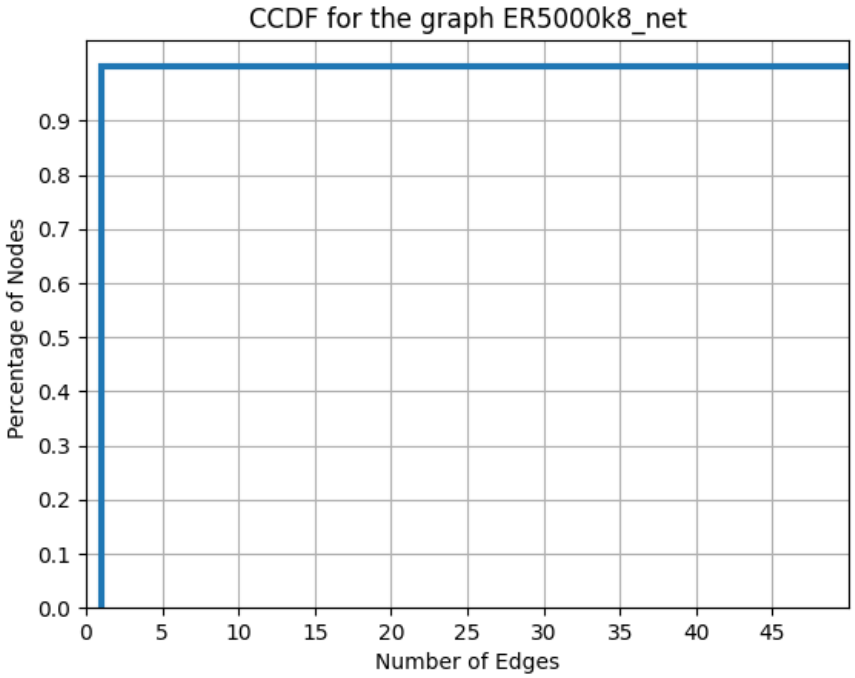

Figure 2 – CFDF plot for ER5000k8.net

- model/SF_1000_g2.7.net

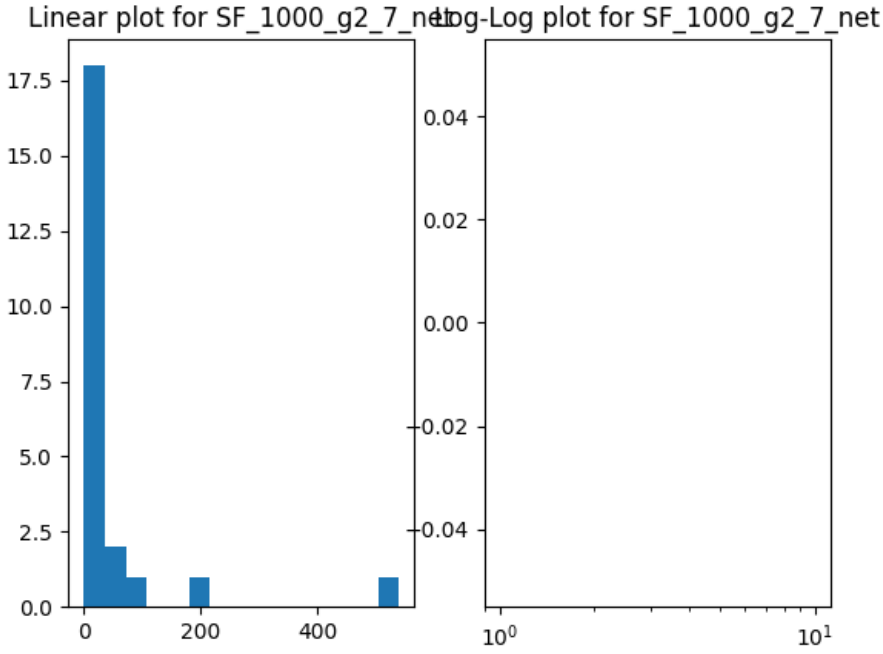

Figure 3 – Histograms of the degree distribution (PDF) for model/SF_1000_g2.7.net

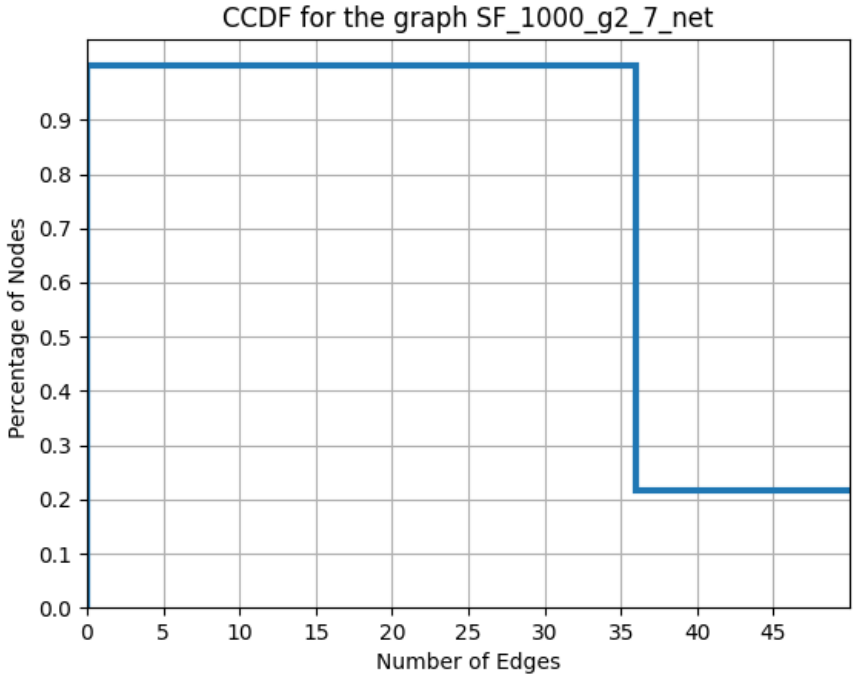

Figure 4 – CFDF plot for model/SF_1000_g2.7.net

- model/ws1000.net

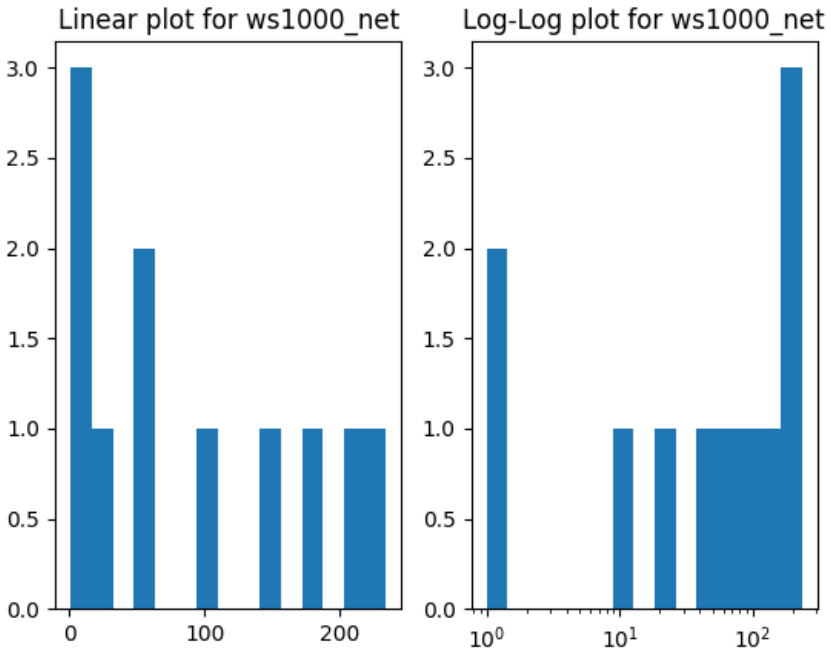

Figure 5 – Histograms of the degree distribution (PDF) for model/ws1000.net

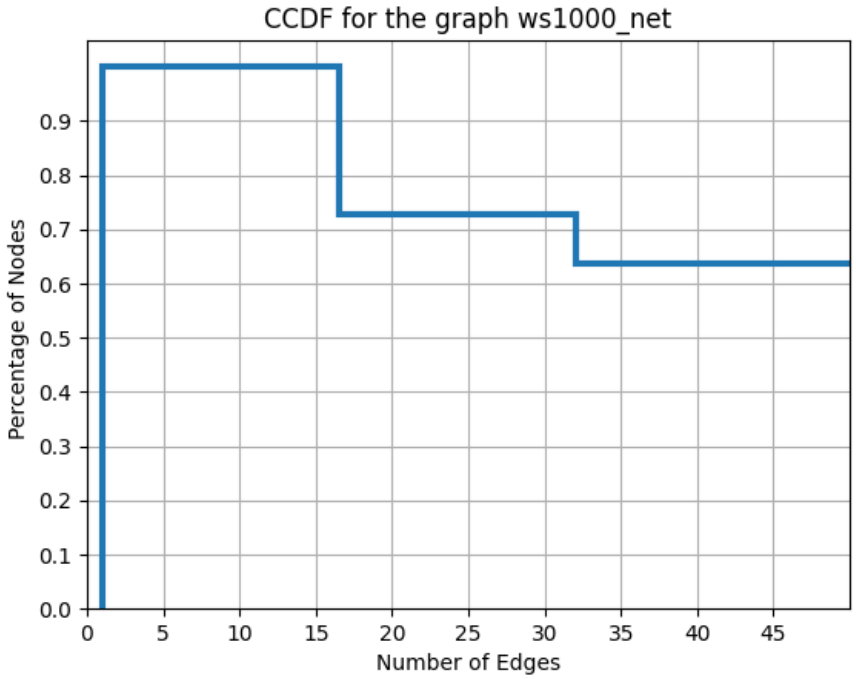

Figure 6 – CFDF plot for model/ws1000.net

- real/airports_UW.net

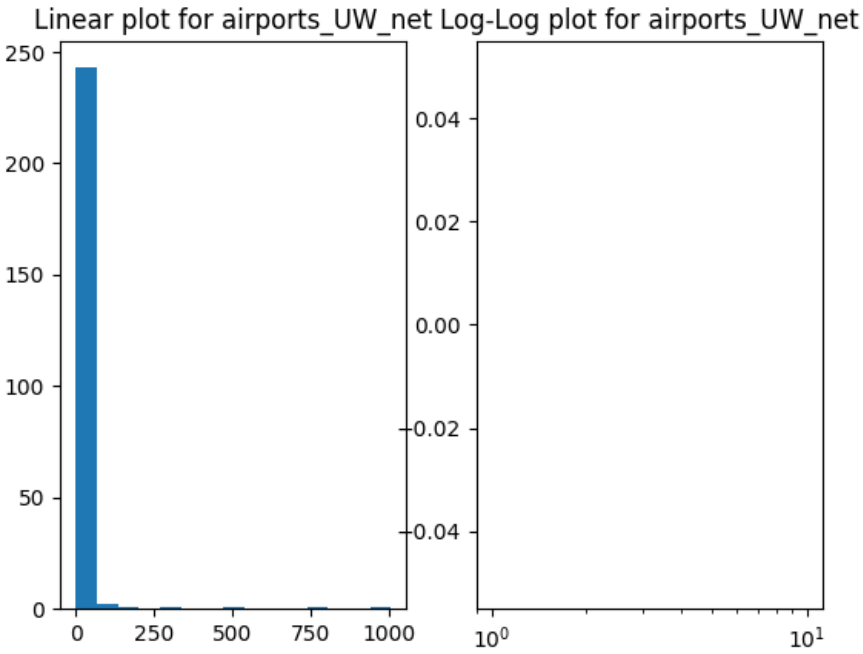

Figure 7 – Histograms of the degree distribution (PDF) for model airports_UW.net

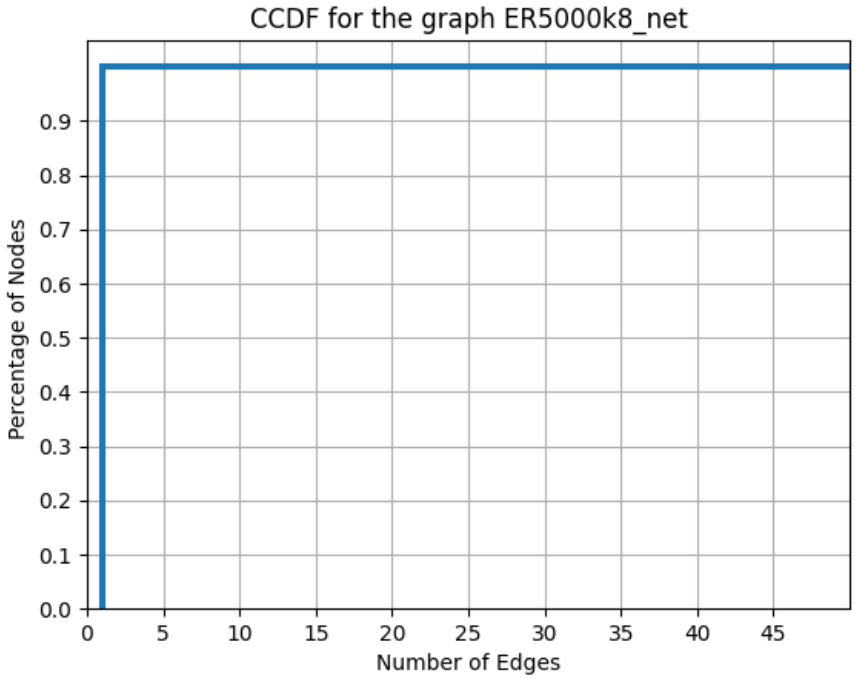

Figure 8 – CFDF plot for for model airports_UW.net

- real/PGP.net

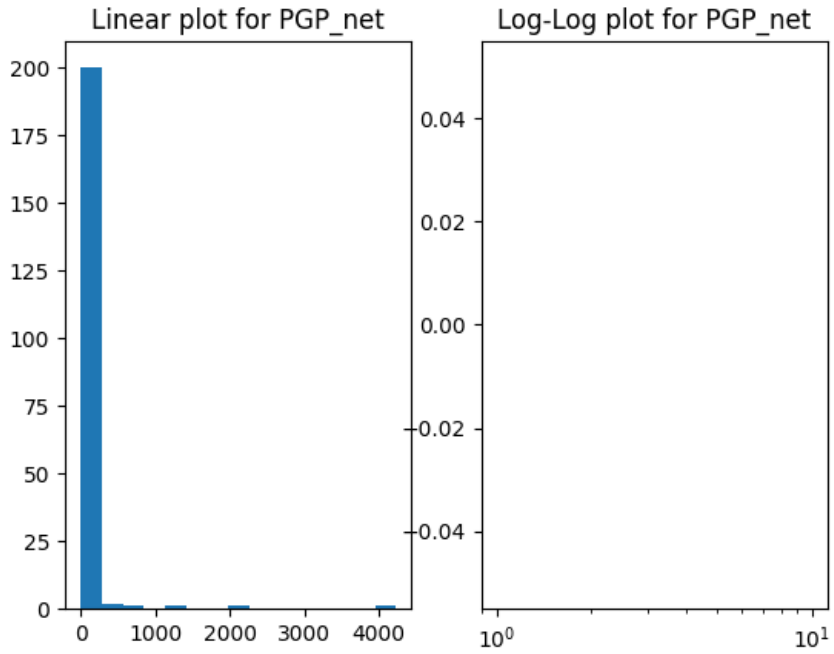

Figure 9 – Histograms of the degree distribution (PDF) for forreal/PGP.net

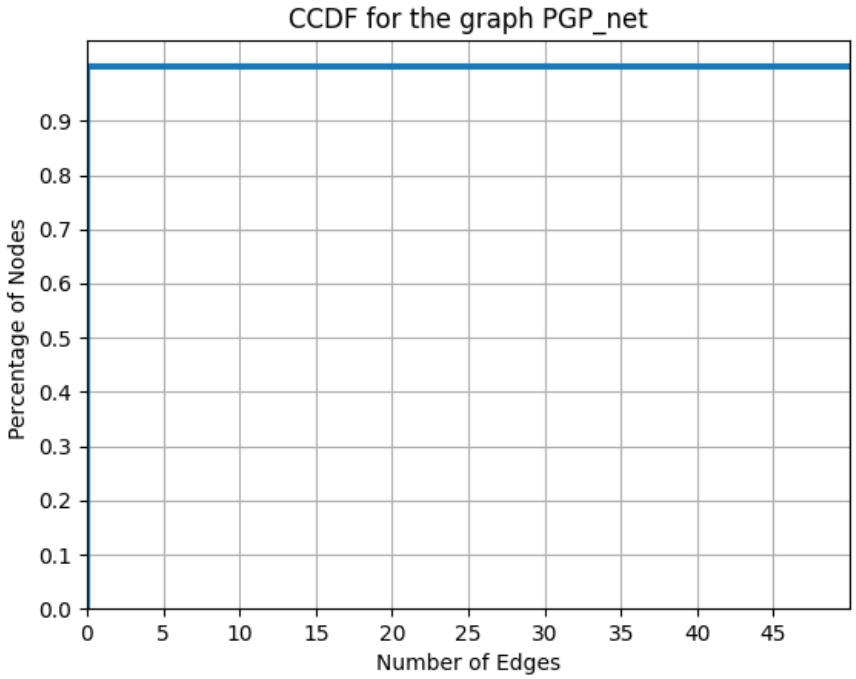

Figure 10 – CFDF plot for forreal/PGP.net model

4.2 Discussion

During the design, implementation and resolution of this assignment we faced a variety of challenges stemming from the existing methodologies, lack of experience with networks, as well as assumptions of the nature and structure of the graphs. One of the challenges that we faced in the beginning was choosing the appropriate tool for the job. Because, there can be a variety of approaches to this problem,

one can built the methods and functions in order to solve this by implementing different graph traversal algorithms such as DFS or BFS. Or one can also choose a specific library, or software in order to solve this. We experimented with NetworkX as well as GraphTool for python, but ultimately decided to stick with igraph, since we found it more comfortable to be working with. On the other hand, once that the tool was identified, there needed to be a sound decision on which methods to use, for example, iGraph has about 3-4 different methods to calculate transitivity, this required us to look at the literature as well as match the problem statements from the assignment in order to understand which method is the appropriate not only for the task at hand but also for the type of network. Another issue that was encountered, was for Figure 7 and 8 respectively, that the log-log graph did not show any data, this was tried out with different methods of plotting the log-log scale and it did not seem to change the result. We suppose that there was an error in our methodology, yet were unable to identify the bug.

Lastly, it is important to note that there were a number of assumptions that were done on these graphs, such as that they are all undirected graphs, as well as being unweighted (except for the calculation of the strength descriptor). We have seen in our experimentation that changing the these assumption also change the final result for the descriptor.

5 Conclusions

In conclusion the analysis of Complex Networks needs to be done with a a foundation of their descriptors. In this assignment we faced three different types of models: Toy, Model, Real. These were with increasing complexity and computational cost. During this we have seen that there are "standard" descriptors of networks which need to be taken into consideration, as we have seen in parts (a) and (b) of the assignment. Moreover, we also used a graphical approach here, not by plotting the graph, but by calculating the PDF and CCDF for 5 different models (3 from the Model Dataset, and 2 from the Real Dataset).

References

[1] Garnham, Alan. "Introduction". *Artificial intelligence: an introduction.* (Routledge Kegan Paul. 1988): 01–24, 24–44.

[2] Newman, Mark. 'Networks'. *Second Editon: chapter 1 to 8.* (Newman, Mark. 2018): 13–262.

[3] Haykin, Simon. "Introduction". *Neural networks and learning machines.*(Prentice Hall/Pearson): 11–17.(2009)

[4] Documentation igraph-Python– for version 0.9.9 https://igraph.org/python/

Appendix A. Implementation details

- **Airport network graphs**

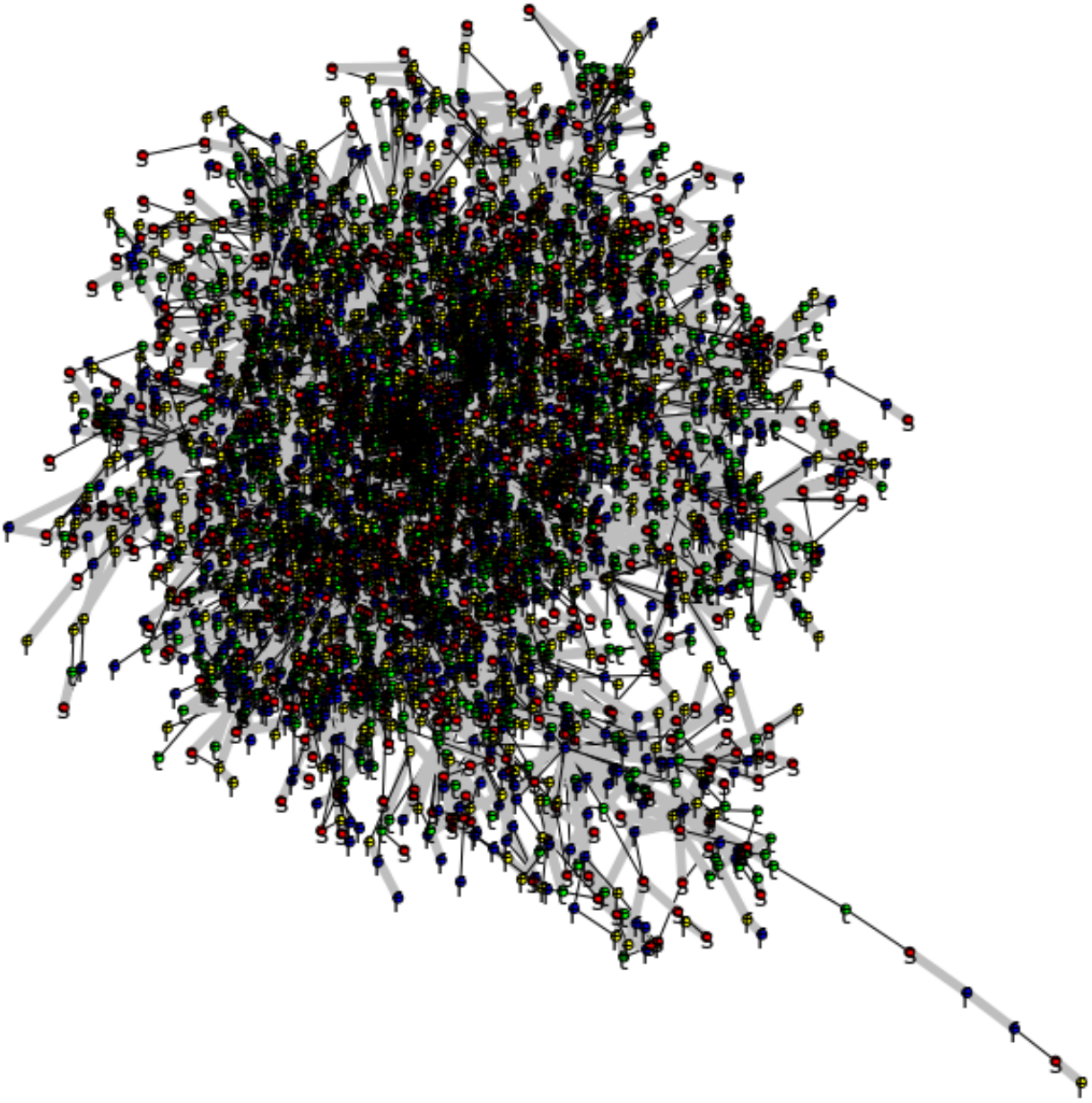

Figure 11 – Airports network graph