Universiteit Leiden

Master Computer Science

Quantifying Network Complexity: An Evaluation of Measures

Name:	Simon van Wageningen
Student ID:	s2317079
Date:	17/08/2020
Specialisation:	Computer Science: Data Science
1st supervisor:	Dr. Frank Takes
2nd supervisor:	Dr. Leon Willenborg

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Quantifying Network Complexity: An Evaluation of Measures*

Simon van Wageningen[s2317079]

Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
s.van.wageningen@umail.leidenuniv.nl
https://liacs.leidenuniv.nl/

Abstract. Network complexity measures aim to quantify to what extent a real-world system has a nontrivial structure. This allows for a direct comparison of different types of networks, as well as the discovery of universal network properties. However, various network complexity measures have been proposed in the literature. Moreover, it remains debatable to what extent the overall complexity of any network can be captured using one measure. In this thesis we will implement, test and systematically evaluate seven network complexity measures and corresponding algorithms. The evaluation was done theoretically using eight axioms and empirically using four evaluation criteria. Both artificially generated as well as real-world networks, including a business network of the Netherlands, were used. It was found that two measures, namely arc symmetry and entropy, best satisfied the evaluation criteria, especially when used together with two other more basic measures. The remaining measures proved to be less useful in practice, not meeting the criteria of for example scalability and generalizability. From our analysis we conclude that using one measure is insufficient; to truly capture network complexity, depending on the network's topological properties, multiple measures need to be considered.

Keywords: Algorithms · Network Complexity · Measurement.

* Master Thesis completed at LIACS

Acknowledgments

I would like to thank my thesis supervisor Frank Takes for giving me the opportunity to work on this topic, providing incredible amounts of feedback and aiding me in general. My thanks also goes out to Leon Willenborg and Edwin de Jonge from Statistics Netherlands, for their help and feedback during my internship there. Additionally, I am grateful for the constructive criticism given by my peers in the Computational Network Science group at Leiden University. Lastly, I would like to thank my parents for supporting me and my studies.

Table of Contents

1 Introduction

Network complexity measures aim to quantify to what extent a real-world system has a nontrivial structure. In this thesis we will, attempt to conceptualize the most inherent properties of a network and evaluate the development of new network complexity measures that try to capture those properties. Note that this thesis focuses on the complex intrinsic qualities of a network's topology, not on computational complexity as used in some fields to characterize algorithms.

A network is a representation of interactions between elements. For instance, a network can model the relationships within a group of students, a university course, or even in social media. Monetary flow between banks, brain cell activity and protein interactions are examples of (non-social) processes that can also be represented by networks. The field of network science concerns itself with the network representations of diverse types of phenomena and the subsequent analysis of these networks. Since networks come in all shapes and sizes, it can be difficult to understand the differences between different networks. Additionally, the larger the network becomes the harder it becomes to visualize such a network and extract useful information. This motivates the usage of network complexity measures, capable of quantifying a network with the goal of understanding the underlying network properties. With the use of network complexity measures it becomes possible to directly compare networks, and find universal network properties. A network measure with a large value could, for example, indicate that network A is more complex than network B with a smaller value. It remains debatable, however, whether the comparison of a single measure between networks is capable of drawing such a conclusion, regarding network complexity. Even though network A is seemingly more complex according to the measure, it may very well be less complex with respect to other properties. It is therefore important to distinguish complexity measures according to the properties that they theoretically capture. All in all, simplifying a network's complexity into one or more quantities introduces a discussion surrounding the effectiveness and use of complexity measures.

The defining problem when it comes to quantifying network properties is choosing which measure(s) to use. Even though research has shown that major similarities do exist between large-scale networks, regardless of the type of network, there is no consensus of when to use which measure. Numerous measures for network complexity already exist, each measure attempts to capture the defining aspect of the network in a slightly different manner. For instance, the average degree measure, as described in Section 4, gives a description of a network based on the properties of a network's nodes. Such a measure is capable of providing a single value describing the network, yet the information given is substantially different than a measure that focuses on a network's edges.

The issue of choosing when to use which measure can be exacerbated due to requirements and implementations of complexity measures. Not all measures allow every type of network to be used as input; in other cases it is possible but the information given by the measure would then be of little use. For example,

measures that require a network to be connected without the existence of isolated nodes may not function when a disconnected network is given. Moreover, the size of the network can be problematic: depending on the implementation of a complexity measure, it may not be possible to calculate the measure within a reasonable amount of time. Even hardware issues can be a reason why certain (more favorable) measures are chosen over lesser quality measures. Take the entirety of the Facebook network as an example, with 2.5 billion active users [11] the network is immense. Some computationally expensive measures are not ideal to use, yet the expensive measure can be used effortlessly for a small high school network.

The goal of this thesis is therefore not to compare the plethora of existing complexity measures but to test and evaluate a handful of new complexity measures, where each measure attempts to determine a different aspect of a network. Willenborg [39] attempted to create multiple measures of complexity, where each measure captured separate aspects of a network's complexity. In Section 4 these measures of complexity are briefly discussed and explained. The focal point of this thesis is thus the implementation of the methods proposed by Willenborg and attempting to empirically evaluate these measures through experimental findings.

Empirically evaluating novel measures gives insight into their practical uses, as the measures could be limited by practical factors. Making use of random network models allows for extensive testing since the parameters and usage of these models have been researched extensively, such as the Barabási-Albert model [13]. The effects of slowly changing one or more simple parameters of these models are thus theorized and proven. Using these random network models allows for a better understanding of how a novel complexity measure behaves when the network properties change. However, generated networks often are not a good representations of real-world networks, only capturing one or more characteristics of a real-world network. Additional testing on real-world network data is therefore crucial, as to gain an understanding of how well a measure works, with respect to what it is supposed to measure.

So far a handful of aspects have been mentioned that each affect the performance of a measure. Whether a proposed measure is sufficient for the measuring of a network's complexity, will depend on the following criteria: scalability, explainability, generalizability and uniqueness. Each criterion plays an important role in determining the need to use a complexity measure. The scalability of a measure was already briefly mentioned, it concerns itself with how well the computation of a measure changes when the size of the problem/network is increased. How much practical information can be extracted is evaluated by the explainability criterion: what exactly does the numerical value given by the complexity measure tell about the network and how useful is it? Additionally, is there a significant difference between the practical results of a measure and the theoretical results? Generalizability ties in with the requirements of a measure, some may only be useful in small networks, real-world networks, or any other specific type of network. It could also be that a measure was specifically made

for one particular type of network, as will be seen in Section 4. Lastly, a measure may not be as unique as thought or even simply be a slightly tweaked existing measure. On top of these discussable criteria, a check-list in the form of stated axioms will be used, these axioms are given in Section 5.

In summary, the thesis at hand will attempt to answer the following question: **How can we empirically evaluate different measures of complexity, in terms of accuracy, scalability, explainability, uniqueness and generalizability, with the goal of capturing the most inherent properties of a network?**

The structure of the thesis is as follows: Section 2 will introduce some of the preliminary terms and concepts that allow the reader to better understand the content. Section 3 focuses on related work regarding inherent network similarities and how network complexity can arise from simplicity. Section 4 introduces a few familiar and novel complexity measures. These measures are classified, explained and defined with the use of equations and examples. Section 5 dives into the evaluation criteria, simulated and real-world network data, the alterations to the measures, and the expected runtime performance of said measures. The experimental setup with regards to the parameter settings, their expectations, and the hardware and software are explained in Section 6. Section 7 presents the results of the multitude of experiments performed. Section 8 dives deeper into the results by aggregating the different experimental results and discussing their practical implications and limitations, according to the evaluation criteria. Lastly, Section 9 gives final remarks as well as possible future research.

2 Preliminaries

This section will cover basic definitions related to networks, centrality measures and matrix notation, as well as some basic real-world network properties.

2.1 Definitions and terminology

In this thesis, the terms network and graph are used interchangeably. Both unweighted undirected graphs and unweighted directed graphs are used, henceforth referred to as graphs and digraphs respectively. Graphs are portrayed as $G = (V, E)$, whereas digraphs are portrayed as $G_{di} = (V, E)$. Set V indicates the nodes/vertices $V = \{1, 2, ..., n\}$ and E the set of edges in graphs or the set of arcs in digraphs. Throughout the thesis n is used to denote the number of nodes and m the number of edges.

It is common practice to use the terminology 'edge' for both graph types, yet in this thesis they are referred to as edges and arcs for graphs and digraphs, respectively. Consider two nodes v and w, an edge is defined as $\{v, w\}$ containing both an arc (v, w) and a counter-arc (w, v). Digraphs differ from graphs in that they can possibly have an arc between nodes without having a counter-arc, thus not necessarily having edges. A digraph with a counter-arc for every arc is the same as a graph with edges. The number of arcs/edges connected to a node is

called the degree, whereas the indegree and outdegree refer to the number of arcs pointing toward or from a node, respectively.

Connected graphs are graphs where every node can be reached from every other node by traversing edges. Digraphs are considered weakly connected when they can be transformed into a connected graph by converting arcs to edges. In (weakly) connected (di)graphs, there is only a single component. This thesis mainly uses the terms reachability and connectedness when referring to the state of a graph's connections. A connected graph is, by definition, fully reachable. A strongly connected digraph is considered to have full reachability, as every node can be reached from every other node.

Complexity measures are denoted by C_x with the subscript $_x$ indicating the specific measure. Each measure is a function of the graph G or digraph G_{di}, as such: $C_{measure}(G)$.

Table 1: Adjacency matrix of graph in Figure 1

node	0	1	2	3	4	5	6
0	0	1	1	1	0	0	0
1	1	0	0	0	0	0	0
2	1	0	0	0	1	1	0
3	1	0	0	0	1	0	0
4	0	0	1	1	0	0	0
5	0	0	1	0	0	0	1
6	0	0	0	0	0	1	0

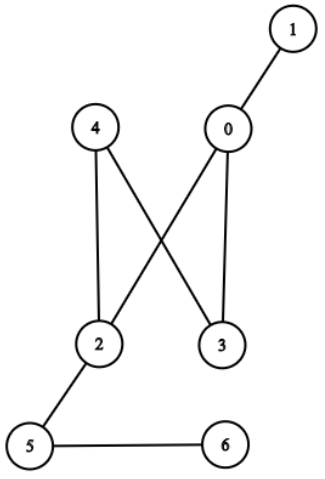

Fig. 1: Example of a graph with edges

Table 2: Adjacency matrix of digraph in Figure 2

node	0	1	2	3	4	5	6
0	0	1	1	1	0	0	0
1	0	0	0	0	0	0	0
2	0	0	0	0	0	1	0
3	1	0	0	0	1	0	0
4	0	0	1	0	0	0	0
5	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0

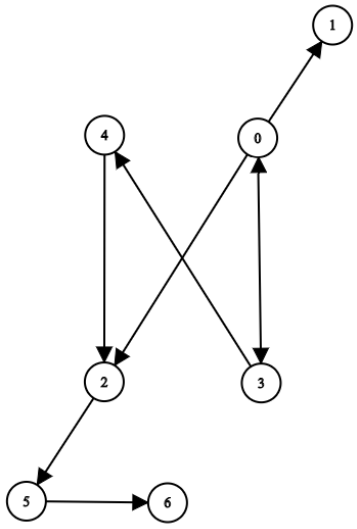

Fig. 2: Example of a digraph with arcs

Some complexity measures are more easily defined using the adjacency matrix representation of the input (di)graph. The adjacency matrix A is an $n \times n$ matrix where row indices are the source and column indices are the target. An element with a value of 1 indicates that there is an arc which points outward from a node (source) towards another node (target). The style of row and column indices being sources and targets respectively is explicitly stated, as the reverse interpretation can lead to the reverse digraph of the intended digraph. In graphs this distinction is not important due to the natural symmetry of a graph's adjacency matrix. Furthermore, ι_n refers to a column vector of all ones of size n. I is the identity matrix of size $n \times n$ with all zeros, except for the diagonal which contains all ones. The inverse and transpose of a matrix or vector M are portrayed by M^{-1} and M' respectively.

Lastly, the sentence 'if and only if' is condensed to a single term 'iff'. Most of the terms and explanations given in this section are reiterated in other sections for reading clarity.

2.2 Real-world network properties

Real-world (social) networks often share the same characteristics [27]: they are often relatively sparse meaning that the actual number of edges/arcs in a given network is much smaller than the maximum number of edges/arcs possible. Furthermore, most real-world networks have a giant component which is the largest subset of nodes of the network. When it comes to the distribution of node degrees, typical real networks tend to have the same type of degree distribution. The frequencies of degree values usually form a power law distribution with a fat tail, or a lognormal distribution. The small world phenomenon refers to the hypothesis that most, if not all, humans tend to be linked by mutual acquaintances. This phenomenon is reflected in most real-world networks where the average distance in the network is equal to six. Lastly, the number of triangles (where a subset of 3 nodes are all connected to each other) is substantially larger in real-world networks than the triangle counts in artificial networks. Most complexity measures deal with these properties in one way or another, including the measures discussed in this thesis.

Real-world networks may have multiple edges/arcs between the same nodes, metadata of nodes and edges/arcs can contain weights and timestamps. These types of additional network data can have an effect on the complexity of the network. In this thesis, however, these characteristics are not included.

3 Related Work

This section focuses on network similarities, complex graph generation rules and network complexity classification, three topics related to this thesis.

3.1 Network Similarities

One of the problems of capturing network complexity is that there is no universal framework agreed upon within the network science community, according to Butts [16]. Different formulations and definitions of network complexity are given depending on the field of interest; software, physics, biological/organic, social and finance networks are all examples of networks that can have different characteristics yet they mostly inherit similar properties.

The characteristics of social networks, for example, tend to have similar properties regardless of the type of social network. Ugander et al. [36] analyzed a large part of the Facebook network, one of the largest social networks ever analyzed, and found that even such a network has an inherent basic anatomy. Facebook has most if not all characteristics of real-world social networks, as described in the previous section. A large component of 99% exists along with a large clustering of triangles as well as the popular six degrees of separation phenomenon which has been shown in other research [31].

In the case of biological networks, Stephan et al. found that even primate cerebral cortex network structures adhere to small-world network architecture. Barabási et al. [14] took it one step further and showed that regardless of the type of biological network, large-scale biological networks mostly have universal network topology characteristics. All in all, it has become apparent that on a large scale, networks share a handful of intrinsic properties which can serve as a guide to the quantification of network complexity.

3.2 Complexity from simplicity

Even though systems and networks may appear to be incredibly complex, they are often defined by simple rules. For example, a flock of birds is visually intriguing and appears as a complex system, yet the model for creating such flight patterns is based on three simple rules that each individual bird is hardwired to follow [20]. Similarly, when it comes to the generation of complex graphs, a single simple rule can be enough to generate a complex topology. Wolfram [40] links simple models, from which complex behavior can arise, with the possibility of discovering a new fundamental theory of physics.

The following rule has the ability to create complex digraphs even after only a handful of iterations:

$$\{(v,w),(v,x)\} \rightarrow \{(v,w),(v,y),(w,y),(x,y)\}$$

The above rule states that any three nodes v, w, x with the following type set of arcs $(v,w),(v,x)$ is transformed into four nodes v, w, x, y with four arcs $(v,w),(v,y),(w,y),(x,y)$. The first transformation (iteration) of this rule is visualized in Figures 3 and 4. Figure 5 and 6 depict digraphs resulting from 2 and 15 iterations, respectively. After a handful of iterations (15) it becomes abundantly clear that even such a simple transformation rule has the ability to create a visually complex digraph.

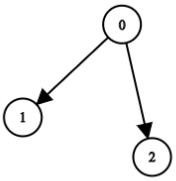

Fig. 3: Result of 0 iterations

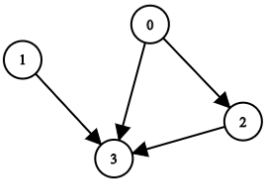

Fig. 4: Result of 1 iteration

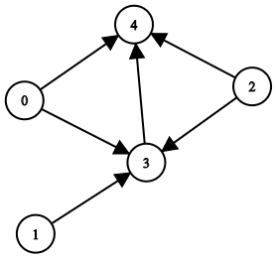

Fig. 5: Result of 2 iterations

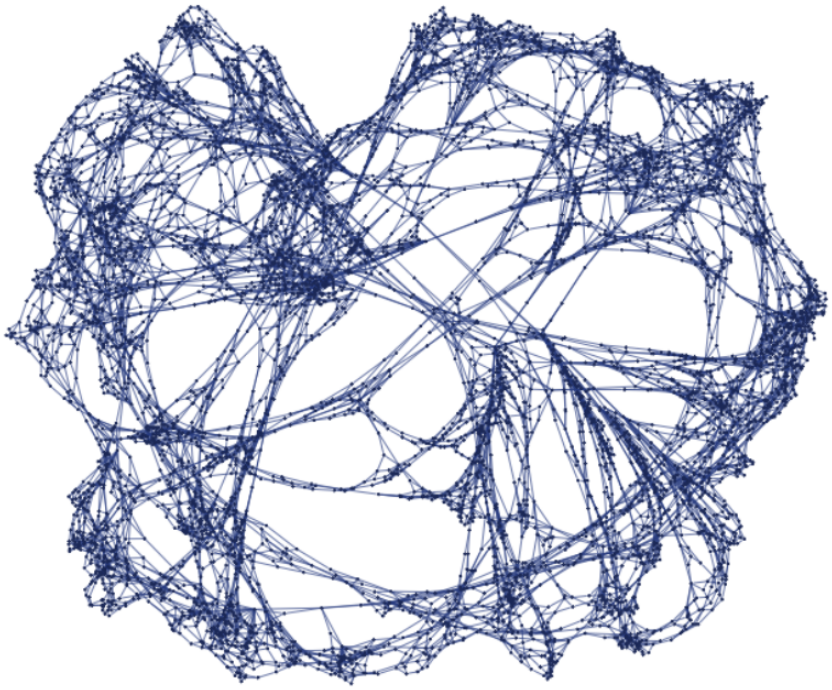

Fig. 6: Result of 15 iterations (figure from [40])

Though it is not seemingly possible to reverse engineer such a rule from a given real-world network, the idea of complexity being the result of a simplistic rule remains interesting. The question is then whether such a rule can be mini-

mized by a single measure, or rather whether the subsequent network topology resulting from the rule can be quantified.

3.3 Quantifying Complexity

The similarities of real-world networks and the complex patterns resulting from simple rules have been discussed. The focus now lies on quantifying a network's complexity and evaluating the strength of this quantification. Dehmer and Pickl [17] reviewed the uncertainty of when to use which complexity measure. They stressed the importance of a measure having unique values for each unique (di)graph: a complexity measure of a graph, that focuses on the topology of the network, should not have the same value for a non-isomorphic graph G_i. Figures 7 – 9 depict two isomorphic graphs and a non-isomorphic graph.

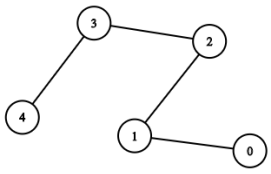

Fig. 7: Graph with 5 nodes and 4 edges

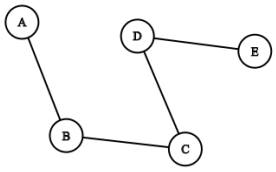

Fig. 8: Graph with 5 nodes and 4 edges

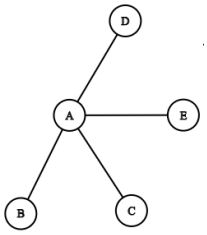

Fig. 9: Graph with 5 nodes and 4 edges (non-isomorphic)

Even though the labels or orientation of the nodes may be different in Figures 7 and 8, the topology remains the same. The same cannot be said for the graph in Figure 9. Intuitively, it then makes sense that a complexity measure attempting to quantify the structure of the graph should have the same values when comparing a graph with its isomorphic brother. Additionally, the non-isomorphic counter-part of a graph is expected not to have the exact same complexity measure value as the graph. Dehmer and Pickl did find that the most popular complexity measures used at that time were not immune to having similar values in non-isomorphic graphs. It does not necessarily mean that a measure that does not defy the non-isomorphic dissimilarity is objectively bad. A categorization of complexity measure classes is simply needed and more importantly a good set of criteria for the evaluation of measures. The two classes of complexity measures used in this thesis are connectedness and distance, which are touched upon in Section 4.

4 Complexity Measures

The network complexity measures, as proposed by Willenborg [39], could potentially be used to characterize (di)graphs based on their topology and other inherent properties. The following sections focus on measures for unweighted (di)graphs. Section 4.1 focuses on measures based on subgraph identification and analysis, whereas Section 4.2 describes measures that are interested in the connection between nodes in terms of distance .

4.1 Connectedness

The first class is based on the concept of graph connectedness [24], the identification and analysis of subgraphs within a network. This concept can be applied to both undirected and directed graphs.

4.1.1 Average Degree The most basic graph complexity measure is the average degree measure.

Undirected graphs The degree of a random node v in a graph is equal to the number of edges connected to node v. The average degree of graph G can be calculated by using the following formula:

$$C_{deg}(G) = \frac{2m}{n} \tag{1}$$

As mentioned in Section 2, m and n represent the number of edges and nodes, respectively.

Directed graphs Equation (1) holds for digraphs.

4.1.2 Routing complexity The routing complexity is a measure specifically tailored for routing digraphs [38]. Routing digraphs are acyclic digraphs with a single source node and a single sink node, these can also be referred to as starting and ending nodes. The topology of routing digraphs can for example be found in questionnaires where there is a predefined order of questions, in Figure 10 such a digraph is shown.

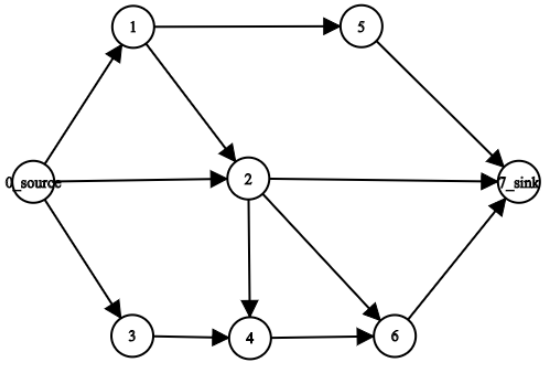

Fig. 10: **Example of a routing digraph**

Though this complexity measure deals with paths it does not bother with path lengths but solely focuses on the number of paths. The intuition behind the measure is that a routing digraph is, arguably, more complex the more paths it has from source to sink.

Undirected graphs Many algorithms and complexity measures exist that find the total number of paths between two nodes in graphs, yet for the proposed complexity measure these are not worth exploring in detail as the focus of the measure is purely on routing digraphs.

Directed graphs Routing digraphs from questionnaire have the interesting property that the sink can be reached from any node in G_{di}. Additionally, the inverse digraph $G_{di}^{\leftarrow}$ of G_{di} exists if the arcs are reversed and the source and sink are swapped.

Even in small routing digraphs the number of paths from the source to the sink can become quite large due to all possible combinations. Arguably, a routing digraph with more possible ways of reaching the sink is more complex. The complexity measure $C_{rou}(G_{di})$ deals with the number of paths from the source to the sink and is defined as:

$$C_{rou}(G_{di}) = ln(\pi(G_{di})) = ln((I_n - A)^{-1})_{1,n} \tag{2}$$

Here, π represents the total number of unique paths from source to sink in G_{di}. Using the adjacency matrix A and the identity matrix I it is possible to calculate this measure algebraically.

4.1.3 Arc symmetry Arc symmetry or asymmetry is a problem unique to digraphs, since by definition an undirected graph has a counter-arc (w, v) for every arc (v, w).

Directed graphs Of a digraph G_{di} the level of arc asymmetry can be used as a measure for complexity. The more arc asymmetry present in a digraph the more complex it will be, arguably. The arc symmetry can be quantified in the following manner:

$$\Theta(A, A') = max\{A, A'\} - min\{A, A'\}$$

Here, $\Theta(A, A')$ is an $n \times n$ matrix where each entry(i, j) is equal to max$\{a_{i,j}, a_{j,i}\}$ - min$\{a_{i,j}, a_{j,i}\}$. Iff the matrix $\Theta(A, A')$ is equal to 0 then perfect arc symmetry exists in the digraph and it can therefore be treated as an undirected graph. With Θ the complexity measure C_{sym} can be defined:

$$C_{sym}(G_{di}) = \frac{\iota_n' \Theta(A, A') \iota_n}{n(n-1)} \tag{3}$$

Here, ι_n represents a column vector of all ones of length n. Since the property $\iota_n' \Theta(A, A') \iota_n \leq n(n-1)$ holds true, the following also holds true $0 \leq C_{sym} \leq 1$. A value of 1 indicates the digraph has no saturated arcs whereas a value of 0 indicates that the digraph is fully symmetric.

However, the arc symmetry complexity is quite plain, it disregards all information of the digraph except for the arc symmetry. Two separate digraphs will have the same C_{sym} iff they have the same number of nodes and same number of arcs without counter-arcs, yet their topologies and reachability, for example as measured using the measures in Section 4.1.4, may vary enormously. Two simple examples of such an occurrence are depicted in Figure 11 and 12.

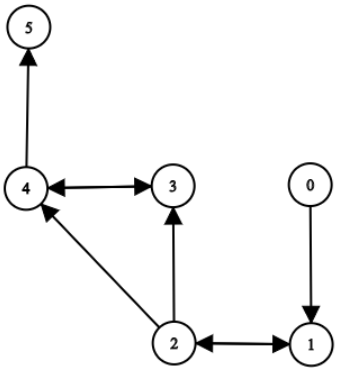

Fig. 11: $C_{sym} = 0.27$

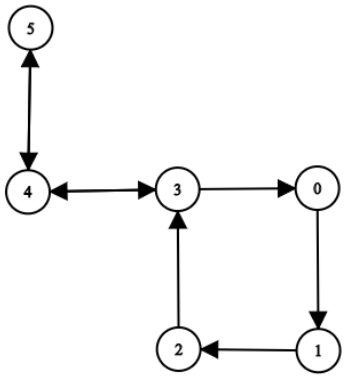

Fig. 12: $C_{sym} = 0.27$

The digraph in Figure 11 is not fully reachable and has a C_{sym} value of 0.27. Similarly, the digraph in Figure 12 also has a C_{sym} value of 0.27, it is, however, fully reachable and has a topology which is structurally different from the digraph portrayed in Figure 11.

4.1.4 Reachability When dealing with undirected graphs, the terms connectedness and reachability are interchangeable. However, in digraphs the term reachability is solely used due to the presence of arc asymmetry. In digraphs an arc from node v to node w may or may not have a counter-arc from w to v. Additionally, the existence of a path between two nodes (v, w) does not guarantee the existence of a path (w, v). A digraph is called a fully reachable digraph when there is a path which connects nodes v and w, for each existing pair of nodes v and w. Determining how far removed a digraph is from its fully reachable counterpart is worth exploring as a measure of complexity.

Directed graphs Three procedures of determining the complexity of a digraph in terms of reachability are considered, namely a probabilistic approach C_{pro} and two deterministic approaches, C_{det} and C_{ent}. The probabilistic approach C_{pro} directly augments the digraph by adding counter-arcs and the deterministic approaches compute how far removed from reachability the digraph is. Pairs of nodes v, w that have an edge $\{v, w\}$ (and thus an arc and a counter-arc) will be referred to as saturated pairs; whereas pairs of nodes that only have a single arc (v, w) or (w, v) will be named unsaturated pairs. If a given digraph G_{di} has u unsaturated pairs of nodes, the number of possible different augmentations is equal to $2^u - 1$.

The proposed complexity measures use full reachability as a criterion. Therefore it is necessary to be able to verify whether a digraph is fully reachable or not. Warshall's algorithm [21] can be used to calculate the transitive closure matrix A^* of the adjacency matrix A. The criterion of full reachability can then be tested by checking if $A^* = J$.

First, the deterministic approach is considered. The required minimum number of arcs needed to make a given digraph G_{di} fully reachable is used, the deterministic digraph complexity C_{det} is then easily defined but challenging to compute.

$$C_{det}(G_{di}) = \frac{v(G_{di})}{m} \tag{4}$$

Here, $v(G_{di})$ in (4) is the minimum number of arcs added to G_{di}. Determining $v(G_{di})$ is the challenging part, the largest possible number of arcs added, the upper bound, would be equal to adding counter-arcs in the digraph wherever possible, which at most would be m but could potentially be smaller based on the topology of the digraph.

Second, arcs can be actively added to a digraph in a probabilistic manner. The idea is to repeatedly generate augmented digraphs with a varying probability parameter p, where p is the probability of adding a counter-arc to the digraph. In random generated graph generation algorithms, where p would be the probability of adding an arc between two nodes, there exists a probability p for which the generated graph is connected with probability 1. The same concept can be applied to define a probabilistic reachability complexity measure

C_{pro} where p^c is the minimum probability for which every augmented digraph of a digraph G_{di} becomes fully reachable.

$$C_{pro}(G_{di}) = p^c \tag{5}$$

Estimating p^c can be done with the use of experiments: for a specific probability p repeatedly draw samples and denote the fraction of samples that attain full reachability. Starting from 0, p^c can be slowly increased up until the aforementioned fraction becomes 1 or very close to 1. This probabilistic approach is computationally very intensive, however.

The third, and less computationally expensive, measure is the proposed entropy complexity. Using the transitive closure matrix A^*, the distributions of how often a node appears in a reachability set of another node (for all nodes) can be calculated as such:

$$\Delta = \frac{\iota' A^*}{\iota' A^* \iota}$$

Here, Δ can be used to find the proposed complexity measure entropy C_{ent}

$$C_{ent}(\Delta) = -\sum_{v \in V} \Delta(v) ln \Delta(v) \tag{6}$$

The more reachable a digraph G_{di} is, the smaller the value of entropy will be.

4.2 Distance

The second class concerns the connection between two nodes in a (di)graph, also called a path. A path can be represented by a list of all the nodes starting from node v and ending in node w, the length of the path is then equal to the length of this list minus 1. A path from node v to node w can have a length in the interval $[1, \infty]$ where a length of ∞ indicates that there is no path between the two nodes (or that the length is equal to ∞ which occurs in infinite graphs).

4.2.1 Average Distance The average distance complexity measure is relatively straightforward, it deals with finding the distance matrix

$$D = (d_{vw})$$

of a connected (di)graph, where d_{vw} is the minimum length of the path in G connecting nodes v and w. This measure is known as one of the most robust techniques used to quantify a network's topology [18]. For example, the distance matrix D of the example graph depicted in Figure 1 can be seen in Table 3.

Table 3: Distance Matrix D, $C_{dis} = 1.81$

node	0	1	2	3	4	5	6
0	0	1	1	1	2	2	3
1	1	0	2	2	3	3	4
2	1	2	0	2	1	1	2
3	1	2	2	0	1	3	4
4	2	3	1	1	0	2	3
5	2	3	1	3	2	0	1
6	3	4	2	4	3	1	0

Undirected graphs The average distance over all different pairs of nodes in a graph G is defined as C_{dis}:

$$C_{dis}(G) = \frac{1}{n(n-1)} \sum_{v,w \in V} d(v,w) \tag{7}$$

Directed graphs The complexity measures as described above can also be applied to digraphs:

$$C_{dis}(G_{di}) = \frac{1}{n(n-1)} \sum_{v,w \in V} \sum_{w,v \in V} d(v,w) \tag{8}$$

In this case, however, both paths of the node pairs (v,w) and (w,v) are of interest. Due to arc asymmetry these paths may not be the same length.

4.2.2 Route search The route search problem, which is related to the Chinese postman problem, deals with the issue of finding the most efficient path in a connected graph. The goal is to find the shortest closest path or circuit that visits every edge in the graph. The unit of interest is the length of the resulting path. For connected graphs, an adaptation of the Chinese postman problem is used.

Undirected graphs The following constraints must hold true for the computation of the route search complexity C_{sea}:

1. The search is continuous.
2. The search covers all edges of the graph
3. The path associated with the search should be of minimal length.

The route search complexity is defined as:

$$C_{sea}(G) = \frac{\lambda_G}{\tau_G} \tag{9}$$

λ_G refers to the search of the graph with the constraints as described above, in this case the starting node does not have to be the ending node, although that

may occur in some graphs. τ_G considers a tour of the graph. τ_G has the same constraints as above but with an additional rule that the starting node should always be equal to the ending node. Additionally, the edges may be visited more than once.

These variables are closely related to Eulerian paths and Eulerian tours/cycles. A Eulerian path visits every edge in a graph exactly once whereas the Eulerian tour/cycle (if it exists) visits every edge exactly once while starting and ending in the same node. Iff the existence of an Eulerian cycle in the graph can be proven, C_{sea} is equal to 1, as both terms are equal to each other. Euler's Formula [12] states the following: a connected graph has an Eulerian cycle iff every node has an even degree whereas a connected graph has an Eulerian path iff every node has an even degree *or* every node has an even degree except for two nodes with an odd degree. These types of graphs are called Eulerian and semi-Eulerian graphs, respectively. Figures 13 and 14 depict a Eulerian and semi-Eulerian graph. The solution to finding the Eulerian graph from the semi-Eulerian graph is to add a single parallel edge at the optimal spot, thereby effectively retracing/revisiting that edge.

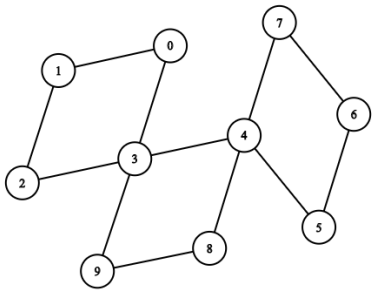

Fig. 13: Eulerian graph $\lambda_G = \tau_G = 12$, $C_{sea} = 1$

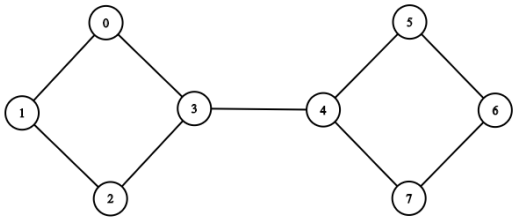

Fig. 14: Semi-Eulerian graph example, $\lambda_G = 9, \tau_G = 10$, $C_{sea} = 0.9$

τ_G can be calculated by finding the length of the Eulerian cycle whereas λ_G is equal to the length of the Eulerian path. In Figure 14 λ_G is equal to 9 as the most optimal search starts in node 3 and ends in node 4 per the following sequence of edges: ({3, 2}, {2, 1}, {1, 0}, {0, 3}, {3, 4}, {4, 5}, {5, 6}, {6, 7}, {7, 4}). τ_G in Figure 14 is then equal to 10, it has the same sequence of edges plus the addition of {4, 3}. Theoretically, the difference between these two variables, τ_G and λ_G, becomes smaller and smaller as the graph increases in size, since it is only a single path that makes the difference.

Directed Graphs The route search complexity can also be used in digraphs though different approaches will have to be explored. Similarly, if the digraph is Eulerian and thus contains a Euler cycle then $C_{sea}(G_{di}) = 1$. However, for

digraphs finding a solution to this problem becomes much more difficult than for graphs as digraphs are required to be strongly connected to find a solution.

5 Methods

The methods section is divided into four subsections: the criteria for complexity measures, the simulated network data, the real-world network data, and the computation of measures, each with their own subsections.

The criteria for complexity measures subsection goes over the axioms and criteria used to evaluate the complexity measures. The simulated and real-world network data subsections discuss the artificial networks and real-world networks used, respectively. Lastly, the computation of measures subsection describes the measures that are to be evaluated, as well as their adaptations, algorithms and time complexity.

5.1 Criteria for Complexity Measures

Two methods of evaluating the network complexity measures are presented. A theoretical evaluation in the form of eight axioms, as presented in Section 5.1.1. In Section 5.1.2 four empirical evaluation criteria are given.

5.1.1 Axioms A method of evaluating complexity measures was developed by Butts [16] in the form of eight axioms. When a complexity measure satisfies the most important axioms, then it can be said that the measure is able to provide some useful information regarding the network. The relaxation and/or violation of one ore more axioms is what distinguishes measures from one another and allows for a check-box-like manner of evaluation and comparison. The eight axioms are as follows:

1. The given complexity measure should return a cardinal real number.
2. Random variables should not be part of the measure.
3. Given *any* finite (di)graph, it is expected that the (di)graph measure C is able to supply some value.
4. A complexity measure's result should not be unbounded.
5. A complexity measure's value can never be smaller than the complexity of a complete (di)graph of size 1.
6. The labeling of nodes should not have an effect on the complexity of the (di)graph.
7. A (di)graph's components should not be more complex than the (di)graph itself.
8. Changing the representation of a (di)graph should not have an effect on the complexity measure.

The first four Axioms 1 – 4 are related to basic yet non-trivial statements which should hold mostly true for (almost) all complexity measures. Axioms 5 – 8

are considered to be more reasonable to violations; the measure may, for example, specifically relax an axiom to differ itself from other measures.

All of the axioms described above can and will be used as a check-list for new measures. Yet for the creation of *new* measures some other criteria, as mentioned in the introduction, have to be introduced and discussed rather than following a simple check-list.

5.1.2 Criteria The purpose of the developed complexity measures is to characterize separate aspects of networks. To evaluate the usefulness of each measure four criteria are considered. Each criterion focuses on an important quality of the measure.

1. Explainability, to what extent does the measure's value explain the network's complexity? Is there a disparity between what each measure theoretically should capture, versus what its practical results show? As each measure has its own characteristics, the explainability of a measure concerns itself with the practical information that can be extracted from a measure's results.
2. Generalizability of a measure is an important consideration, whether the usage of the measure is applicable and/or useful in every type of graph.
3. Scalability refers to how much more difficult the computation of measure becomes as the (di)graph grows. Thus, the scalability of a measure goes hand-in-hand with the algorithm to compute the measure. If a measure has been proven to be incredibly useful with regards to other criteria, possible developments and improvements can be made to how well the measure scales. Section 5.4.5 touches upon this criterion through expectations.
4. Uniqueness, how unique is complexity measure? The research behind complexity measures in the field of computation network science is extensive; certain (similar) measures may be created independently from each other. Moreover, some measures may have been inspired by others or developed with minor improvements to the original.

5.2 Simulated network data

In order to evaluate each measure based on the aforementioned criteria, they will be tested on various generated (di)graphs before real-world networks are tested. Generated networks are not good indicators of real-world networks but are useful in that the parameters can be tweaked and the effects of those parameter settings can be explored. For instance, the number of edges can be varied with a fixed number of nodes thus allowing for testing on graphs with different levels of density.

Five different random graph models are considered: Complete, Barabási-Albert, Scale-Free, Erdős-Rényi (di)graphs and routing digraphs, each with its own unique characteristics and parameters, thereby creating new testing environments for every measure.

5.2.1 Complete (di)graphs The interesting property of a complete (di)graph is that every node is directly connected to every other node. Hence, there is only a single parameter, *the number of nodes n*. With n nodes in a graph every node has $n-1$ edges and therefore the total number of edges m is equal to $\frac{n*(n-1)}{2}$. The resulting (di)graph is incredibly dense and can be considered the worst case scenario for the computation time of numerous complexity measures (though not necessarily every measure). The information regarding computation time is incredibly insightful when it comes to complete (di)graphs whereas the values of the proposed measures should remain relatively constant with respect to their boundaries and may not be that interesting. The time results of experiments performed on complete (di)graphs are therefore incredibly helpful for the scalability criterion.

5.2.2 Barabási-Albert graphs Barabási-Albert graphs are randomly generated graphs with two simple parameters, *the number of nodes n* and *the number of edges each added node forms* m_b. Each graph starts with m_b number of nodes; nodes are then added continuously and connected to existing nodes by m_b number of edges where new nodes prefer connecting to nodes with a high degree. As a result, the number of edges in the graph will always be equal to $m_b * n - (m_b^2)$. Since newer nodes prefer connecting to nodes with a high degree, the degree distribution will be scale free.

5.2.3 Scale-Free digraphs The Barabási-Albert model does not exist for directed graphs and therefore a suitable replacement had to be found. Since the degree distribution of the Barabási-Albert model is scale-free, the best alternative was the scale-free digraph from Bollobás et al. [15]. This new model is similar to the Barabási-Albert model in that the probability of connecting a new node to an existing node is proportional to the degree of the existing node; in this digraph case it is proportional to the in and out-degree. The model has five parameters: a *probability* β of adding an arc between existing nodes, a *probability* α of adding a new node connected to an existing node based on the in-degree distribution, a *probability* γ of adding a new node connected to an existing node based on the out-degree distribution, a *bias* for choosing the nodes from the in-degree distribution, and finally *the number of nodes n*.

5.2.4 Erdős-Rényi (di)graphs Erdős-Rényi (di)graphs are similar to Barabási-Albert graphs in that they both allow for the manipulation of the number of edges created in the network and *the number of nodes n*. However, the parameter of the Barabási-Albert model fixes the number of edges whereas the parameter of the Erdős-Rényi model sets a *probability p of connecting each and every node to each other*. Thus, as n goes to ∞ the expected number of edges will get closer to $\binom{n}{2}p$.

5.2.5 Routing digraphs As explained in Section 4.1.2, routing digraphs are a special type of acyclic digraphs. A routing digraph does not contain any cycles and is therefore not fully reachable/strongly connected. It is, however, always weakly connected with a path from every node to the sink node, and a path from the source node to every other node. Algorithms for the random generation of routing digraphs, specifically, do not exist. Luckily, it is not difficult to create an algorithm for creating acyclic digraphs. With a few adjustments the constraints of a single source and sink node can be added. Algorithm 1 in Appendix A shows how a routing digraph can be generated, the newly created algorithm is similar to the Erdős-Rényi model in that it has two parameters: a *probability p of connecting nodes to each other* independent of other nodes' distributions, and *the number of nodes n.*

5.3 Real-world network data

It is important to see how informative the complexity measures are when it comes to real-world networks. Therefore, open-source (di)graphs were extracted and tested upon. Additionally, Statistics Netherlands provided a large real-world network for testing purposes.

5.3.1 Open-source (di)graphs Six real-world graphs were reused from previous work, their sources are depicted in Table 19. These specific graphs were chosen since they had no multiple edges between the same nodes nor did they allow for self-loops. Some graphs had weighted edges but weights were ignored in the computation of the measures. Most importantly, for each graph the largest connected component was taken and the operations were subsequently performed on these largest connected components. Graph density was not a criterion for the graph selection process. Similarly, seven open-source digraphs, as portrayed in Table 20, were used that were also unweighted and contained no self-loops. The largest weakly connected component from each digraph was taken and used.

5.3.2 Statistics Netherlands (di)graphs One network was provided by Statistics Netherlands, a network depicting the business relations between companies and buyers in the Netherlands. The arcs in the business network indicated a relationship between a provider and a buyer. The original network was far too large (∼200,000,000 arcs) to analyze. Therefore, samples were repeatedly taken using a simple edge sampling technique where a sample with m arcs was taken with probability $\frac{m_{sample}}{m_{tot}}$. It is debatable whether the simple edge sampling technique can create representative samples; Hu and Lau [25] found that edge sampling reduces the average degree and also does not sample existing network neighborhoods well. Additionally, Leskovec and Faloutsos [30] concluded that the Random Walk and Forest Fire sampling algorithms perform best when a sample of 25% was taken; however, they also mentioned that a 15% sample is usually large enough to match the properties of the real graph. Multiple samples

in the range of 0.1% and 0.3% were taken from the business network; for these small samples all seven measures, as seen in Section 5.4, could be calculated. The difference in sampling techniques would thus have been minimal with such small samples.

5.4 Computation of measures

5.4.1 Original Measures Even though numerous measures are proposed in Section 4, not every measure is fit to be tested on every type of graph. The measures that were chosen to be tested on both graphs and digraphs: the average degree C_{deg} (1) and the average distance C_{dis} (7). The search complexity C_{sea} (9) is only tested on graphs, whereas the following measures are tested on digraphs and routing digraphs: the routing complexity C_{rou} (2), the arc symmetry C_{sym} (3), entropy C_{ent} (6) and the probabilistic reachability C_{pro} (5).

5.4.2 Measures Adaptations Three measures, C_{sym}, C_{ent} and C_{pro}, were slightly changed with respect to their definitions. Additionally, new algorithms had to be developed for C_{pro} and C_{sea} as their definitions did not indicate how to calculate their elements.

Intermediate experiments for C_{sym} showed that even with a low number of saturated arcs, C_{sym} would still be incredibly close to 0, and thus indicating a large amount of arc symmetry, due to the $n(n-1)$ term. Therefore (3) was changed to the following, where s indicates the number of saturated arcs:

$$C_{sym}(G_{di}) = 1 - \frac{s}{m} \tag{10}$$

Furthermore, the calculation of C_{ent}, as shown in (6), was changed into (11) so that the resulting values would always be between 0 and 1 as such $0 \leq C_{ent} \leq 1$.

$$C_{ent}(\Delta) = 1 + \frac{\sum_{v \in V} \Delta(v) ln \Delta(v)}{ln(n)} \tag{11}$$

A C_{ent} value of 0 would indicate that the digraph is fully reachable whereas a value of 1 would mean that the topology of the digraph is incredibly far away from being fully reachable.

Additionally, the probabilistic reachability complexity C_{pro} was altered. The original calculation would calculate the minimum probability in which every augmented digraph would become fully reachable, after adding counter-arcs. The problem with this method is that nodes with an outdegree of 0 or an indegree of 0 set the theoretical minimum probability. Figure 15 shows a very simple digraph where this is the case.

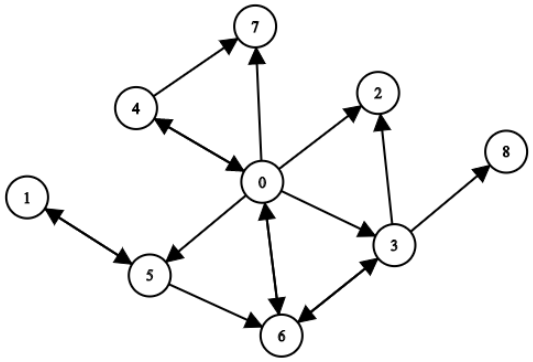

Fig. 15: **Example graph of problematic nodes for the original** C_{pro}

Node 8 only has a single arc pointing towards it and thus must always acquire a counter-arc in order to achieve full reachability in the digraph. Augmenting the digraph (by adding a counter-arc with probability p) MC times, will lead to a $p * MC$ fully reachable digraphs due to the presence of node 8. Removing nodes such as node 8 is possible but the problem still occurs for nodes 2 and 7. Therefore, a different approach to tackle this problem was created. The definition of C_{pro} only changes in notation:

$$C_{pro}(G_{di}) = p^{change}(p, MC, \delta, \epsilon) \tag{12}$$

In (5) the minimum probability p^c, for which each augmented digraph will become fully reachable with probability p, was calculated. Now $p^{change}(p, MC, \delta, \epsilon)$ is calculated, the probability for which a δ change in the probability p does not lead to an increase of the size of the largest strongly connected component by at least ϵ.

5.4.3 Measures Algorithms An algorithm to calculate C_{pro} was created and is displayed in Algorithm 2 in Appendix A. The algorithm terminates when increasing the probability does not lead to a substantial increase in the size of the largest strongly connected component.

Additionally, Section 4.2.2 described how C_{sea} is defined in (9). The two components, λ_G and τ_G, both find a continuous search of minimal length of all edges in the graph, though τ_G specifies that the search has to start and end in the same node. The solution to this problem is to convert the graph into semi-Eulerian and Eulerian graphs to calculate λ_G and τ_G, respectively. The algorithm that implements said solution can be seen in Algorithm 3 in Appendix A.

5.4.4 Measures Overview A quick overview of the measures to be used and their possible ranges of values can be seen in Table 4

Table 4: Complexity Measured to be used in experiments, their value ranges and time complexities

Overview Measures

Name Graph Measure	Notation	Range	Time Complexity
Average Degree	C_{deg}	$(0, n-1]$	$\mathcal{O}(1)$
Average Distance	C_{dis}	$(0, m]$	$\mathcal{O}(n(m + nlog_2(n))$
Search Complexity	C_{sea}	$(0, 1]$	$\mathcal{O}(n(m + nlog_2(n)) + n^2log(n))$
Name DiGraph Measure	**Notation**	**Range**	**Time Complexity**
Average Degree	C_{deg}	$(0, n-1]$	$\mathcal{O}(1)$
Average Distance	C_{dis}	$(0, m]$	$\mathcal{O}(n(m + nlog_2(n))$
Symmetry Complexity	C_{sym}	$[0, 1]$	$\mathcal{O}(m)$
Entropy	C_{ent}	$[0, 1)$	$\mathcal{O}((n + m))$
Probabilistic Reachability	C_{pro}	$[0, 1]$	$\mathcal{O}(i_{max} * m)$
Routing complexity	C_{rou}	$(0, ln(2^{(n-2)})]$	$\mathcal{O}(n^2m)$

5.4.5 Time complexity When discussing the performance of some algorithm, with respect to the execution time, the term time complexity or big-O notation is used. For the measures, as described above, the theoretical worst case time complexities are displayed in Table 4.

The average case time complexities of C_{sea} and C_{rou} will most likely be much better than their worst case time complexities. The calculation of C_{sea} will take the longest when all nodes in the graph have an odd degree which is a highly unlikely real-world scenario. Additionally, both C_{dis} and C_{sea} make use of Dijkstra's algorithm [18] which has $\mathcal{O}(n(m + nlog_2(n)))$ but can perform at $\Theta(n(m + log_2(n)))$. Moreover, the costly part of computing C_{rou} is taking the inverse of a matrix, which can have a cubic time complexity of $\mathcal{O}(n^3)$ if the matrix only has non-zero elements. Since the matrix used in the calculation of C_{rou} will only have m elements the big O will at most be $\mathcal{O}(n^2m)$. The experiments will shed light on the Θ of the other measures.

6 Experimental Setup

6.1 Experimental settings

Two types of experiments were used in combination with differing random (di)graph models. First, the number of nodes varied (from 50, 100, 200, 400, 800, 1000, 1500, 2000, 2500 to 3000 nodes) and the parameter(s) of the models were fixed. Secondly, the number of nodes was fixed (1000 nodes) and the parameter(s) of the models varied for 50 (di)graphs.

6.1.1 Complete (di)graphs Complete (di)graphs only have a single parameter, the number of nodes. Therefore only the first experimental setup could be used with the aforementioned varying numbers of nodes.

6.1.2 Barabási-Albert graphs The number of nodes and the number of edges each node acquires m_b are the two parameters in Barabási-Albert graphs. The first setup kept m_b fixed at 2 and the second setup kept n fixed at 1000 with m_b ranging from $[1, 50]$ with steps of 1.

6.1.3 Scale-Free digraphs The default parameters were used to create the scale-free digraphs in the first experimental setup: A 0.54 probability β, a 0.41 probability α, a 0.05 probability γ, a bias of 0.20 and the similar number of nodes parameter settings as Barabási-Albert graphs. The second experimental setup had the same bias and γ but kept n fixed at 1000. Moreover, β started at 0.20 and increased by 0.01 whereas α started at 0.75 and decreased by 0.01. The ranges for β and α are thus as follows: $[0.20, 0.69]$ and $[0.75, 0.26]$, respectively.

6.1.4 Erdős-Rényi (di)graphs Erdős-Rényi (di)graphs, with a certain minimum probability, will almost always be (weakly) connected. Most of the experiments played with this threshold of connectedness which lies around $\frac{ln(n)}{n}$ [19]. In all cases, the largest connected component was taken and analyzed if by any chance the probability failed to create a single (weakly) connected component. In the first setup, the probability p was kept constant relative to the number of nodes i.e. $2\frac{ln(n)}{n}$ with only the number of nodes as the changing parameter. In the second setup, the number of nodes was fixed at 1000 but the probability varied; the maximum probability was set to $2\frac{ln(n)}{n}10$ where the minimum probability and step size were both equal to $\frac{2\frac{ln(n)}{n}10}{50}$. Thus the probabilities ranged from $[0.0028, 0.1399]$ with steps of 0.0028.

6.1.5 Routing digraphs Since the algorithm for the generation of routing digraphs was inspired by the Erdős-Rényi model, the minimum probability of weakly connectedness of $\frac{ln(n)}{n}$ was reused. The parameter values used in the routing digraph experiments were thus kept exactly similar to the ones of the Erdős-Rényi (di)graphs experiments.

6.2 Measure expectations

Up until this point in Section 5, the measures have been summarized, the models used to generate the random (di)graphs have been explained and the two experimental settings have been discussed. The final step is to write down the expectations of how a measure value will change when the parameter(s) of a model increase(s) or decrease(s). Table 5 depicts these expectations.

Table 5: Models measure expectations

Model		n	m	$\mathbf{m_b}$	$\boldsymbol{p}$	$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	C_{deg}	C_{dis}	C_{sea}	C_{sym}	C_{ent}	C_{pro}	C_{rou}
Barabási-Albert	*Setting*1	⇑	⇑	⇔				⇔	⇑	⇑				
	*Setting*2	⇔	⇑	⇑				⇑	⇓	⇑				
Scale-Free model	*Setting*1	⇑	⇑			⇔	⇔	⇔	⇔		⇔	⇓	⇓	
	*Setting*2	⇔	⇑			⇓	⇑	⇑	⇑		⇓	⇓	⇓	
Erdős-Rényi	*Setting*1	⇑	⇑		⇔			⇑	⇔	⇑	⇔	⇓	⇓	
	*Setting*2	⇔	⇑		⇑			⇑	⇓	⇑	⇓	⇓	⇓	
Routing model	*Setting*1	⇑	⇑		⇔			⇑	⇔		⇔	⇓	⇓	⇑
	*Setting*2	⇔	⇑		⇑			⇑	⇓		⇔	⇓	⇓	⇑

The rows in these tables denote the experimental setting. The bolded column header stresses that the changes in that column's parameter cause the changes in the measure values. The left to right arrow ⇔ shows that the parameter value remains the same. The Erdős-Rényi model had a constant p relative to the number of nodes, therefore in *Setting*1 in these models, the p is depicted as unchanging even though the actual values do change.

6.3 Hardware and Software

The proposed complexity measures and node ranking measures were implemented in Python 3.7.1 64-bit using the Spyder 3.3.2 IDE. The Python programming language was chosen due to the wide variety of public tools (packages) available that immensely aid in visualizing networks and allowing easy computations. The 'networkx' package [23] was a huge asset to the production of these new measures. The package removed the necessity of the usage of adjacency matrices and matrix calculations in most cases. Additionally, it allowed for easy acquisitions with respect to basic network information, such as the number of nodes, as well as network alterations. Moreover, the networkx package has built-in complexity measures and random graph generators which were used in the experiments of this thesis. Appendix C contains the packages used in these experiments as well as their (advised) versions.

The experiments performed on generated (di)graphs and open-source (di)graphs were executed on a Windows 10 64-bit machine with 8 GB of RAM and an i5-5200 CPU processor with 2.20 GHz. Due to the sensitivity of the data provided by Statistics Netherlands, the experiments performed on their data were executed on their own machines, via a virtual machine connection, with the following specifications: Windows 10 Enterprise 64-bit, 8 GB RAM and an Intel Xeon Gold 6146 CPU with 3.20GHz.

7 Results

The results section will contain the measure outcomes as well as the execution times of the experiments performed on generated (di)graphs, open-source (di)graphs and large digraphs provided by Statistics Netherlands.

7.1 Generated (di)graphs

7.1.1 Complete graphs Even though the properties of complete graphs and digraphs are well known, experiments are incredibly helpful as to determine how well any given measure scales.

Table 12 depicts the results of the experiments performed on complete graphs. The three complexity measures C_{deg}, C_{dis} and C_{sea} were computed, their computation times are portrayed in each measure's column to the right with the header 't (sec)' which indicates the execution time taken in seconds.

As previously mentioned, the resulting measure values are not that interesting due to the nature of complete graphs. What is, however, interesting to note is that the complexity measure C_{sea} approaches the maximum value of 1. A C_{sea} value of 1 can be attained when there already is an Eulerian cycle in the graph or due to rounding as it approaches 1. Each node's degree is uneven thus there is no Eulerian cycle present. The experiments in the other sections will shed light on the behavior of C_{sea}.

The time complexity of C_{deg} is simply a constant and is therefore $\Theta(1)$, since this holds true for all other experiments this will not be reiterated. The measure C_{dis} has been tried and tested many times over and has a worst case time complexity of $\mathcal{O}(n(m + nlog_2(n))$, the results from Table 12 indicate that the measure performs better than expected and adheres to $\Theta(n(m + log_2(n)))$.

The measure C_{sea} was not computed for complete graphs with 800 nodes and more, due to the drastic increase in computation time. It is therefore difficult to estimate the time complexity of this measure. Theoretically, the worst case time complexity of C_{sea} is equal to $\mathcal{O}(n^2(m+nlog_2(n))+n^2log(n)))$, this occurs in the event that each and every node has an odd degree on top of the worst case time complexity of Dijkstra's Algorithm. Since the implementation of C_{sea} makes use of C_{dis}, a performance of $\mathcal{O}(n^2(m + log_2(n)) + n^2log(n)))$ can be expected. The results indicate that the algorithm performs much better than $\mathcal{O}(n^2(m+log_2(n))+n^2log(n)))$ though there is not enough data to fully support that argument.

7.1.2 Complete digraphs In Table 13 the results of the experiments performed on complete digraphs are portrayed. Five complexity measures were evaluated, namely: C_{deg}, C_{dis}, C_{sym}, C_{ent} and C_{pro}.

Similarly to complete graphs, C_{dis} is 1. The arc symmetry complexity C_{sym} is equal to 0 when there is perfect arc symmetry which only occurs when a graph is treated as a digraph or in the case of complete digraphs, as shown by the matching results of Table 13. Since complete digraphs are fully reachable/strongly

connected the entropy values should, and are, equal to 0. Additionally, the algorithm for C_{pro} starts with a default probability of 0.3 which leads to each result having a value of 0.3.

The algorithm for C_{dis} follows $\Theta(nm)$ almost perfectly. The algorithm for the C_{sym} measure only makes use of a single pass over the edge list resulting in a theoretical time complexity of $\mathcal{O}(m)$, the data support the theory as the algorithm for C_{sym} performs slightly better than $\Theta(m)$. The measure C_{ent} has a worst case time complexity of $\mathcal{O}((n+m))$. The findings in Table 13 indicate that the computation of C_{ent} performs slightly better than its expected big O of $\mathcal{O}(n+m)$. The number of iterations for C_{pro} stay fixed, since there are no unsaturated arcs, therefore $\mathcal{O}(i_{max} * m)$ is equal to $\Theta(m)$. All data points, except the graph with 3000 nodes, support this claim.

7.1.3 Barabási-Albert graphs Two experimental setups were used. Firstly, the results of the first setup are given. The average degree C_{deg} approaches $2m$ as n goes to ∞. The average distance C_{dis} increases as the number of nodes and edges increase, in fact this trend follows the following equation $C_{dis} \sim ln(n)$ for small n but for large n this becomes $C_{dis} \sim \frac{ln(n)}{lnln(n)}$ according to the inventor of the Barabási-Albert model[13]. Similarly to the complete graph results, C_{sea} approaches 1 as the size of the graph increases. These findings match what was theorized in Section 4.2.2.

In the second setup, the average degree goes up as the number of edges increases with a fixed n; in fact the number of edges m follows the following rule $m = n * m_b - m_b^2$. C_{dis} goes down as the number of edges increases, which is as expected. Once more, the trend of C_{sea} approaching 1 is visible. The difference between the expected changes and the actual changes are given in Table 6.

Table 6: Barabási-Albert model measure expectations comparison

	$\boldsymbol{n}$	m	$\mathbf{m_b}$	C_{deg}	C_{dis}	C_{sea}
$Setting1_{expected}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow$
$Setting1_{actual}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Uparrow\!\!\Uparrow$	$\Uparrow$	$\Uparrow$
$Setting2_{expected}$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$
$Setting2_{actual}$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$

The time complexity C_{dis} appears to follow the pattern of $\Theta(nm)$. The big O of C_{sea} $\mathcal{O}(n^2(m + nlog_2(n)) + n^2log(n)))$ was not attained. C_{sea} appears to perform better than $\Theta(nm + n * log(n))$.

7.1.4 Scale-Free digraphs The results in Table 15 indicate that the average degree C_{deg} slowly increases as the number of nodes increase. Though, theoretically, C_{deg} should remain the same irregardless of the number of nodes due to the

same probabilities being used. The average distance C_{dis} results are deceiving, a C_{dis} value should give the average path length between all nodes; a value of 0.2392 says that, on average, any node can reach another node in 0.2932 steps which is not possible. The density is the culprit for the deception: if the digraph is not fully reachable then some nodes cannot reach other nodes, the distance between those nodes is then measured as 0. This is a limitation of the implementation of the average distance measure and may skew results. Interestingly, C_{sym} increases as the network grows in size which is unexpected as the density of the digraphs increases slightly. The C_{ent} show a small decreasing trend. It was expected that the networks with low C_{pro} values, such as the networks with 50, 200 and 400 nodes, would have the lowest Entropy values, but this appears not to be the case. The C_{pro} results show that most of the scale-free digraphs are incredibly far removed from being fully reachable.

In the second experimental setting, the probability of adding an arc between a new node and an existing node based on the in-degree distribution (α) started out high and ended low. The probability of adding an arc between existing nodes (β) started out low but ended high. As mentioned in the measure expectations section, these changes would lead to more interconnectivity in the digraph and thus higher values of C_{deg} and C_{dis} and lower values of C_{sym}, C_{ent} and C_{pro}. These expectations are mirrored in the experiments' findings with the exception of the results of the probabilistic reachability complexity C_{pro}. Table 7 portrays these found differences.

Table 7: Scale-Free model measure expectations comparison

	$\boldsymbol{n}$	m	$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	C_{deg}	C_{dis}	C_{sym}	C_{ent}	C_{pro}
$Setting1_{expected}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Downarrow$	$\Downarrow$
$Setting1_{actual}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\mathbf{\Uparrow}$	$\Leftrightarrow$	$\mathbf{\Uparrow}$	$\Downarrow$	$\mathbf{\Leftrightarrow}$
$Setting2_{expected}$	$\Leftrightarrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$Setting2_{actual}$	$\Leftrightarrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Downarrow$	$\mathbf{\Leftrightarrow}$

The time complexities of C_{dis} and C_{sym} cannot be estimated with the findings in Table 15 since the computation times are incredibly low making it too inconsistent. The same holds true for the time complexity of C_{ent}. The C_{pro} results indicate that the algorithm performs much better than $\Theta(i_{max} * m)$.

7.1.5 Erdős-Rényi graphs The results of both experimental settings, as depicted in Table 16 show an increasing trend in C_{deg} as expected. As the number of nodes increases and the probability remains constant with respect to the number of nodes, C_{dis} increases. The findings are rather interesting due to the fact that the average distance increases rather than remaining constant or decreasing. In the case with fixed n the number of edges constantly increase due to the increase in probability, hence C_{dis} goes down while the average degree goes up

simultaneously. As seen in the other results tables, C_{sea} gets closer and closer to 1 as the number of nodes and edges increases. Table 8 visually shows the difference between the expected and actual value changes.

Table 8: Erdős-Rényi model measure expectations

	$\boldsymbol{n}$	m	$\boldsymbol{p}$	C_{deg}	C_{dis}	C_{sea}
$Setting1_{expected}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Uparrow$	$\Leftrightarrow$	$\Uparrow$
$Setting1_{actual}$	$\Uparrow$	$\Uparrow$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow\!\!\Uparrow$	$\Uparrow$
$Setting2_{expected}$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$
$Setting2_{actual}$	$\Leftrightarrow$	$\Uparrow$	$\Uparrow$	$\Uparrow$	$\Downarrow$	$\Uparrow$

The time complexity of both experimental settings follow $\Theta(nm)$ for C_{dis}. The average time complexity for C_{sea} performs much better than $\Theta(nm + n * log(n))$

7.1.6 Erdős-Rényi digraphs Table 17 depicts the experimental results of Erdős-Rényi digraphs. Likewise to their graph counterparts, C_{deg} increases as n increases and the probability is fixed relative to n, yet C_{dis} increases in this setting. The arc symmetry measure C_{sym} appears to be increasing in the fixed probability setting as n and m increase. The digraphs get closer and closer to becoming fully reachable, according to the C_{ent} results. The C_{pro} results are mirrored by the entropy findings, as the number of iterations needed becomes smaller and smaller. With a fixed probability p and larger number of nodes, the number of unsaturated arcs, relative to the number of nodes, decreases leading to higher values of C_{sym}. Even though there are hardly any arcs with counter-arcs present, the digraphs still become (almost) fully reachable.

In the experimental setting where n is fixed and the probability increases, the trends for C_{deg}, C_{ent} and C_{pro} remain similar to the previous setting. Similarly to Erdős-Rényi graphs, the average distance C_{dis} decreases with higher probabilities. The relationship between C_{sym} and the probability p is an inverse linear relationship, which is only logical; as the probability of 2 nodes being connected increases, the number of edges/arcs with symmetry increases and thus C_{sym} decreases. In fact, if n were to go to ∞ the following rule would hold true: $p = 1 - C_{sym}$.

Table 9 summarizes the difference between the expected value changes of the digraph measures versus the actual value changes.

Table 9: Erdős-Rényi model measure expectations

	$\boldsymbol{n}$	m	$\boldsymbol{p}$	C_{deg}	C_{dis}	C_{sym}	C_{ent}	C_{pro}
$Setting1_{expected}$	⇑	⇑	⇔	⇑	⇔	⇔	⇓	⇓
$Setting1_{actual}$	⇑	⇑	⇔	⇑	**⇑**	**⇑**	⇓	⇓
$Setting2_{expected}$	⇔	⇑	⇑	⇑	⇓	⇓	⇓	⇓
$Setting2_{actual}$	⇔	⇑	⇑	⇑	⇓	⇓	⇓	⇓

As for time complexities, C_{dis} performs slightly better than $\Theta(nm)$. The time complexity of C_{sym} remains $\Theta(m)$. The worst case time complexity of $\mathcal{O}(n+m)$ for the C_{ent} measure is not attained with the experimental results from the first setting, as these results indicate that it performs better than expected. In the second experimental setting the number of nodes is a non-factor and $\Theta(n+m)$ holds true for C_{ent}. The probabilistic reachability C_{pro} results show that the algorithm performs slightly better than $\Theta(im)$.

7.1.7 Routing digraphs Since the generation of routing digraphs is similar to the Erdős-Rényi model, the results are expected to be somewhat similar. Due to the nature of routing digraphs, the arc symmetry complexity C_{sym} was not calculated as its value would always be equal to 1.0 as a result of perfect asymmetry. Table 18 depicts these results. In the first experimental setting, with a fixed probability p relative to n, the average degree goes up as well as C_{dis}. The average distance C_{dis} remains under 1 since, by definition, the digraph cannot be fully reachable; meaning that there are plenty of nodes that cannot reach other nodes leading to a C_{dis} below 1. As expected, C_{sym} is equal to 1 due to no arc symmetry being present in routing digraphs. The values of C_{ent} slightly decrease as the number of nodes increases, the digraph thus becomes closer to becoming fully reachable: in theory, less arcs need to be added in the reversed direction in order to make the digraph fully reachable. The trend spotted in C_{pro} supports the same argument. The newest measure, C_{rou}, is an indicator of how complex a routing digraph is when it comes to the number of paths from source to sink. As n increases C_{rou} also increases.

The results from the second setup show that C_{deg} increases and C_d decreases, similarly to the Erdős-Rényi digraph results. Once more, the digraphs become closer to being fully reachable, though it can never acquire full reachability, as shown in the C_{ent} and C_{pro} results. As p increases, and subsequently m, so does the measure C_{rou}. Once more, a table is provided that summarizes the expected versus actual differences, as depicted in Table 10.

Table 10: Routing model measure expectations

	$\boldsymbol{n}$	m	$\boldsymbol{p}$	C_{deg}	C_{dis}	C_{ent}	C_{pro}	C_{rou}
$Setting1_{expected}$	⇑	⇑	⇔	⇑	⇔	⇓	⇓	⇑
$Setting1_{actual}$	⇑	⇑	⇔	⇑	⇑	⇓	⇓	⇑
$Setting2_{expected}$	⇔	⇑	⇑	⇑	⇓	⇓	⇓	⇑
$Setting2_{actual}$	⇔	⇑	⇑	⇑	⇓	⇓	⇓	⇑

The average time complexities of C_{dis}, C_{ent} and C_{pro} remain the same as the Erdős-Rényi results, namely: $\Theta(nm)$, $\Theta(n+m)$ and $\Theta(im)$ respectively. C_{rou} was expected to have a worst case time complexity of $\mathcal{O}(n^2m)$. The time results of C_{rou} in Table 18 support a slightly faster Θ, though the execution times are too small to credibly support that notion.

7.2 Real-world networks

7.2.1 Open source graphs Table 19 shows that the jazz and the astroph graphs are incredibly dense compared to the four other real-world graphs. Interestingly, the values for the measure C_{dis} are not necessarily the largest for the four sparse graphs. The caida and pgp graphs are relatively sparse compared to the jazz and astroph graphs but their C_{dis} does not reflect their sparsities. Similarly to the generated network findings, on average C_{sea} gets closer and closer to 1 though the density of the network has a tiny effect on small-scale graphs.

The average time complexity of C_{dis} is slightly worse than the previously found $\Theta(nm)$ yet better than $\mathcal{O}(n(m+nlog_2(n))$. It becomes clear that the computation of C_{sea} is incredibly expensive as the execution time of C_{sea} for the powergrid network took more than 40 hours. Hence why the search measure was not computed for the other larger graphs. The Erdős-Rényi experiment results suggested that the time complexity of C_{sea} lies below $\Theta(nm+n*log(n))$. The results in Table 19 indicate that the actual Θ lies between $\Theta(nm+n*log(n))$ and $\mathcal{O}(n(m+nlog_2(n))+n^2log(n)))$, though there are too few data points to fully support that argument. What can be seen, however, is that the execution time of C_{sea} is a realistic barrier and hazard for this complexity measure.

7.2.2 Open source digraphs Table 20 contains the results and specifications of seven real-world open-source digraphs. The google-plus network as well as both the small and large versions of the p2p networks are quite sparse compared to the first four smaller graphs. This is reflected in their C_{deg} values. Similar to the graph results, the average distance C_{dis} of each graph is not directly correlated to the density of each graph. The C_{sym} results for the p2p networks indicate that there is barely any arc symmetry present. Interestingly, the googleplus network has a decent amount of arc symmetry, yet the average distance C_{dis} is incredibly small. Due to long computation times, C_{ent} was only calculated for the first three

dense graphs. Their entropy values do indicate that they are very close to being strongly connected matching the C_{pro} results where the number of iterations needed were less than the maximum number of iterations.

The C_{dis} algorithm performs better than $\Theta(nm)$ in most cases, most likely due to the density of the larger graphs, though the googleplus network seems to be a heavy outlier. The arc symmetry measure C_{sym} still performs at the previously proven time complexity of $\Theta(m)$, whereas C_{ent} performs at its worst $\mathcal{O}(n+m)$. For most results, the C_{pro} algorithm performs with a time complexity of $\Theta(im)$.

7.2.3 Statistics Netherlands (di)graphs The entire business network contained 199,193,030 arcs and therefore samples had to be drawn with a certain edge sampling probability $\frac{m_{sample}}{m_{tot}}$. The column m_{sample} in Table 21 portrays how large each sample was intended to be, with respect to the number of edges. The actual size of the sample graph can then depend on the size of the largest weakly connected component. Table 21 shows the samples and the complexity measures results.

From the difference between the number of arcs m in the largest weakly connected component, and the number of arcs desired in the sample m_{sample}, it is clear that the edge sampling technique works better when a large enough sample is taken which starts around 150,000 arcs. This means that the smaller graphs may not be as representative as expected and therefore the interpretations of these results are heavily biased. Additionally, not all measures could be calculated either due to long execution times (C_{dis} and C_{pro}) or memory restrictions C_{ent}.

As expected, the larger the sample the larger the values of C_{deg} and C_{dis}. Since these samples are incredibly small relative to the entire network they are expected to be very sparse and highly asymmetric when it comes to the arcs. This is also reflected in the C_{sym} results; only for the largest samples with 20,000,000 and 25,000,000 arcs does C_{sym} become slightly smaller than 1. In conjunction with the large average degree results, in these large samples, it can be said that the business network has a decent amount of one-to-many relationships between providers and customers with only a few customers being providers themselves. The results of C_{ent} indicate that even though these samples are incredibly sparse, they are still relatively close to being fully reachable. The C_{pro} values do not necessarily agree with the C_{ent} results which is quite an interesting behavior.

The average distance C_{dis} algorithm performs almost at its worst, and thus the big O of $\mathcal{O}(n(m + nlog_2(n))$. The measure C_{sym} still performs at $\Theta(m)$. Additionally, the time results of C_{ent} and C_{pro} perform slightly better than $\Theta((n + m))$ and $\Theta(i * m)$ respectively.

8 Discussion

Through the use of experiments a lot of information has been gathered regarding the novel complexity measures ($C_{sea}, C_{sym}, C_{ent}, C_{pro}$ & C_{rou}). Two

separate experimental settings were capable of giving good indications of how each measure behaved, as the change in parameters lead to (di)graph topology changes. The extensive testing performed on various generated (di)graphs has led to a better understanding of how these measures react to structural changes. Additionally, the real-world (di)graphs indicate the usefulness of the measures and what their combined values can tell about the given network. The enormous amount of data need to be summarized and evaluated. This is done with the use of the given axioms as well as the four created criteria.

8.1 Axiom Evaluation

Firstly, the violation of or adherence to the axioms are presented. Table 11 shows whether an axiom was adhered, with the letter T, or whether it was violated, with the letter F.

Table 11: Eight Axioms 1 - 8 to be used to judge new measures

Overview Axioms

Axiom	C_{deg}	C_{dis}	C_{sea}	C_{sym}	C_{ent}	C_{pro}	C_{rou}
Axiom 1	T	T	T	T	T	T	T
Axiom 2	T	T	T	T	T	F	T
Axiom 3	T	T	F	T	T/F	F	F
Axiom 4	T	T	T	T	T	T	T
Axiom 5	T	T	T	T	T	T	T
Axiom 6	T	T	T	T	T	T	T
Axiom 7	F	F	F	F	F	F	T
Axiom 8	T	T	T	T	F	T	F

All measures did not violate Axiom 1, meaning that each measure only ever outputs a real single number.

Based on the implementation and the theoretical foundations of the Probabilistic Reachability measure, it is no surprise that this measure was the only one that violated the Axiom 2. All six other measures contain no probabilistic aspect.

Axiom 3 states that the measure should always be able to supply the user with a value given *any* finite (di)graph. Even though all testing was done on connected graphs and weakly connected digraphs, some measures such as the average degree C_{deg}, the average distance C_{dis} and the symmetry complexity C_{sym} do not require the graph inputs to be in a specific form. The Entropy measure C_{ent} is capable of calculating values when given non-weakly connected digraphs, though it is generally more useful to use Entropy for purely weakly connected digraphs. The same cannot be said for the probabilistic reachability C_{pro} and routing complexity C_{rou} measures: both measures require the given network input to be either a digraph or routing digraph, respectively.

All measures have a theoretical upper bound, therefore Axiom 4 concerning the finiteness of measures is not violated. Moreover, the floor values of applying the measures to a complete (di)graph of size 1, will always be the smallest value attainable for all measures, thereby each measure adheres to Axiom 5. Additionally, labelling the nodes and will not have an effect on the resulting complexity values and thus all measures do not violate Axiom 6.

Axiom 7 was already quite controversial, it stated that any (di)graph component's complexity should not be larger than the entire (di)graph's complexity. The only measure that is capable of following that statement is the routing complexity C_{rou}. For all the other measures, from average degree C_{deg} to the probabilistic reachability C_{pro} measure, components of the given network are capable of having a higher complexity. Such violations do not indicate that all of the aforementioned measures are subpar, even the author of the axioms [16] proposed a relaxation of this assumption.

The concept of complementarity is mentioned in Axiom 8. If the representation of the adjacency matrix would change, for example, then the measures that explicitly make use of adjacency matrix computations would cease to provide the exact same values. In this case, the entropy C_{ent} and routing complexity C_{rou} measures both make use of adjacency matrices and are not immune to changes in these matrices. It is worthwhile to note, however, that all Pythonic implementations of the measures initially make use of adjacency matrices to represent the input network. Yet only C_{ent} and C_{rou} truly rely on matrix computations, whereas the other measures are implemented in a way that does not require such computations.

The instability of the measures can be told by the violations of these axioms. If any measure were to violate all given axioms, then clearly it is not a useful measure to begin with. If, on the other hand, all axioms were not violated then the measure becomes less informative [16] and tends to be the same as others. The existing methods such as the average degree C_{deg} and the average distance C_{dis} tend to be relatively correlated, they also only ever violate Axiom 7. That does not mean, however, that two measures quantify the same thing if their axiom violation is the same. From Table 11 it is visible that more violations lead to less stable measures so to say. For instance, the probabilistic reachability measure C_{pro} has some specific requirements and may not yield the same results.

8.2 Criteria Evaluation

The four evaluation criteria are as follows: Explainability, generalizability, scalability and uniqueness. Each (novel) measure is discussed.

8.2.1 Search Complexity

Explainability The search complexity was theorized to make valid comparisons between the topology of different graphs. Through the use of of search-like graph

traversal style, where the visit of *every* edge was of importance, distinctions between graph structures could be made. In Section 4.2.2, it was already mentioned that, in theory, as the graphs become bigger then the value of C_{sea} would automatically get closer and closer to 0, regardless of heavy changes to the graph topology. The results agreed with the theorized increments, as the graphs became larger and larger, C_{sea} approached its maximum value of 1, even if the topology of the network remained relatively similar (as no other parameters of the graph were changed). When C_{sea} was calculated for the open-source graphs, tiny differences could be found with a magnitude of 0.01, which was attributed to the difference in density. However, such a density difference was already easily recognized through the use of the average degree C_{deg}. The practical information that the measure value gives is therefore extremely limited.

Generalizability There are a few requirements for the graph input of the search complexity. The network has to be a graph that is connected. Disconnected graphs or digraphs are not usable as input in this measure. Moreover, the previous paragraph touched upon the measure value's convergence to 1, as the size of the graph increases: by increasing the number of nodes and edges or just the number of edges, C_{sea} approaches 1. Therefore, the search complexity may only be useful for extremely small graphs ($n < 100$).

Scalability With an incredibly large potential big O of $\mathcal{O}(n(m + nlog_2(n)) + n^2log(n))$, it was expected that the computation of C_{sea} would be incredibly taxing. The experiments performed on generated graphs indicated that the computation was lengthy and in some cases problematic, while the algorithm tended to perform slightly better than its big O. The results from the real-world open-source graphs, however, showed that even small sparse graphs with less than 5000 nodes were subject to the high computation costs, with one particular graph requiring over 40 hours of computation. Scalability is, by default, related to the implementation and its quality, which might give this criterion slightly less weight. Though in the case of this measure's implementation, the techniques used to calculate the measure's value is thought to be the current best solution (Euclidean graph techniques).

Uniqueness The search complexity is the brother of the travelling-salesman problem, where the most efficient route of visiting every node is computed. While the components of C_{sea} may not be as unique, as they are both related to the Chinese-postman problem, the idea of combining the components as a method of measuring graph complexity is new. Thus, even though the measure may have been inspired by existing solutions to other problems, it remains relatively novel.

8.2.2 Arc Symmetry

Explainability The arc symmetry complexity measure captures a very simple and specific aspect of a given digraph. It simply measures the ratio between

arcs that have a counter-arc and total arcs. Therefore topological properties are not captured, as the measure is only capable of providing information about how many direct back and forth relations there are between nodes. Due to its simplicity, the experiments do show that this measure captures exactly what it intends to measure. On its own, the measure values are not particularly useful as it only tells how many counter-arcs are present. However, when it is used alongside other measures such as the average degree C_{deg} or the entropy measure C_{ent}, more characteristics of the given digraph can be uncovered. For example, a high value of $C_{sym} = 0.9$ combined with a high value of $C_{deg} = 30$ would indicate that on average a node A connects with 30 other nodes but a very small number of those 30 other nodes directly connect back to node A.

Generalizability The only restriction of the arc symmetry measure is that the given networks have to be directed graphs. Disconnected, weakly and strongly connected digraphs could all be used for the calculation of C_{sym}. The measure can be used for routing digraphs, but is not useful as by definition routing digraphs do not have any counter-arcs. Though routing digraphs themselves are a specific type of digraph that are not often used, therefore the arc symmetry measure can be used in almost all digraph cases.

Scalability The implementation was incredibly simple and the worst possible scaling factor of $\mathcal{O}(m)$ is small relative to the other seen measures. The experimental results also showed that the algorithm of C_{sym} performed slightly better than its big O. Moreover, the computation times were incredibly low in general. The business network, from which samples of 25 million edges were drawn, showed that even incredibly large digraphs could be used as input for this measure without acquiring unrealistically long execution times.

Uniqueness Since the measure is very basic, it is expected that it may already exist in one form, though it may not be as popular or widely known. It could, therefore, be possible that such a measure may have been overlooked or dismissed by others, due to its simplicity.

8.2.3 Entropy and Probabilistic Reachability These two measures were grouped together as they serve similar purposes.

Explainability Both entropy C_{ent} and the probabilistic reachability C_{pro} attempt to calculate how far a given digraph is from becoming fully reachable. The former does this through the use of reachability sets, whereas the latter attempts to find a roof probability where further increments to the probability do not aid in increasing the digraph reachability. Each measure has their own strengths and weaknesses. The experimental results have shown that the entropy measure is far easier to interpret than the probabilistic reachability measure. It is clear that lower entropy values correlate with more digraph reachability, as the experiments of generated digraphs have indicated. The interpretation of C_{pro} is more

difficult however. The difference between a C_{pro} value of 0.5 and 0.6 is much more vague. More importantly, the usage of the probabilistic reachability measure is heavily subjected to the algorithm's parameters. Much more information could be acquired just to determine how useful C_{pro} can be, by experimenting with its parameters. In this case, it appears that the entropy measure C_{ent} is more interpretive and gives a clear indication of far any given digraph is from becoming fully reachable.

Generalizability Weakly connected digraphs are recommended to be used as input for both measures. The entropy measure does give information when used on disconnected digraphs yet the interpretation of the value becomes less valid, since it could be skewed due to some components. C_{pro} will always give the maximum probability value of 1 when any disconnected digraph is given and is therefore a useless measure in that situation. Both measures should thus only be used whenever weakly connected digraphs are considered. However, since C_{pro} has tweakable parameters it may be more difficult to generalize the results of such a measure.

Scalability Even though the difference between worst-case time complexities is rather large, $\mathcal{O}(m + n)$ and $\mathcal{O}(i_{max} * m)$ for C_{ent} and C_{pro}, respectively, the density of the network did have a major effect on the performance of both measures. In most experimental results, the entropy implementation performed better than its big O, with the digraph's density having a visible effect on the computation time, most visible in the differences in the open-source digraphs and companies network results. Non-sparse digraphs tend to be more reachable by default which leads to the C_{pro} algorithm to require less iterations to complete, and thus scaling better compared to the entropy algorithm, with respect to their computation times. However, in other cases the C_{pro} did require the maximum number of iterations which could lead to incredibly large computation times. Due to the usage of matrix computations it is, however, important to note that the entropy measure could and did run into hardware memory issues, thus making it less ideal for large digraphs ($n > 200,000$).

Uniqueness Even though the entropy name has been introduced in other research, most notably in the field of information theory, the method of calculating C_{ent} and the manner of capturing the reachability aspect has not been seen in other work, as far as the author of the thesis is aware. The same can be said for the probabilistic reachability measure C_{pro}.

8.2.4 Routing Complexity

Explainability The characteristic encapsulated by the routing complexity C_{rou} is very straightforward. It captures how complex a routing digraph is through the calculation of the number of paths from the source to the sink. For instance, when comparing two questionnaires, each represented as a routing digraph with

the same number of nodes, the amount of individual ways of getting to the last question directly compares how complex each questionnaire is. The routing digraphs' results have shown that C_{rou} is a good indicator of how complex any given routing digraph is, when it comes to the aspect of total paths. Though both experimental setups played with the density of the routing digraph, additional experiments that would tweak the internal structure while keeping n and m constant, would show that the routing complexity is a good indicator when directly used to compare routing digraphs of similar size. The measure is, however, less useful when it comes to comparing networks with different sizes, as the value of C_{rou} would increase as n increases, regardless of the number of paths from source to sink. Since the theoretical upper limit of C_{rou} is known for each routing digraph ($(0, ln(2^{(n-2)})]$), the measure could be improved by using the upper limit in its implementation.

Generalizability The generalizability criterion is the weakest point of the routing complexity. The network *has* to be a routing digraph, no other types of digraphs are allowed.

Scalability Since the routing complexity was specifically designed for the analysis of routing digraphs, the scalability is not much of an issue, as routing digraphs tend to be small in practice. Nonetheless, if more experiments were performed on larger networks the worst case scaling factor of $\mathcal{O}(n^2m)$ could become a potential problem.

Uniqueness Finding the number of paths from one node to another node is a classical network science problem. In that regard, the measure at hand is not unique. What does differentiate the routing complexity measure is that is specifically used as a measure to quantify the complexity of routing digraphs. In that aspect, the measure is relatively unique.

9 Conclusion

Five novel measures were implemented to try and find the most important properties of a network. These complexity measures were theorized to capturing the complexity of a network, and through the combined use of them accurately quantify a network's complexity. Through the use of systematic empirical evaluation, in the form of four criteria and eight axioms, the performance of the proposed measures was evaluated. Generated (di)graphs and various real-world networks allowed for intimate testing and experimenting. It was found that some measures performed better than others.

The search complexity should have been able to quantify certain topological differences, but in practice it was not able to. When comparing graphs with the search complexity, the characteristics of a graph's topology were not reflected in the difference of the measure's values. Additionally, the measure scaled drastically. Other and more simple methods of quantifying a network have proven

to be more useful, the search complexity only appears to have a distinctive role when used for small graphs.

Though the arc symmetry measure has a simplistic design, it was proven to be quite useful. It was capable of dealing with any type of digraph, scaled incredibly well, was uncomplicated in its interpretation, and allowed the user to make a lot of inferences about the network when put next to the average degree. The business network provided by Statistics Netherlands, for example, had an enormous size. Yet, it was impossible to calculate some measures in a given time-span. The simple average degree and arc symmetry complexity measures were capable of providing information about the network where other more sophisticated measures could not.

The usefulness of the entropy and probabilistic reachability measures remains disputable. The measures may not be as interpretative and straightforward as others. Moreover, the probabilistic reachability is heavily dependent on the parameters used. Even though both measures have some drawbacks, including their not so optimal scaling, they do provide the user some information that could not be attained with other measures. While established methods of quantifying the connectedness of a digraph were capable of locating the weakly connected components and the strongly connected/fully reachable components, the manner of how far removed these individual components were from each other is new. Improvements could therefore be made to the computations and implementations of these methods, as the idea behind capturing this specific network aspect has shown to be relatively new and useful. For example, as described in Section 5.4.2, a new measure could focus on problematic nodes that must always acquire a counter-arc.

As expected, no single measure is capable of capturing all inherent properties of a network. However, through the use of two measures coming from different classes one is capable of telling a lot more of what the network looks like. Taking a measure's result from the connectedness class alongside a measure's result from the distance class makes it possible to infer a (di)graph's topology. For instance, having a low average distance in a digraph with a large amount of arc symmetry infers that there are a lot of cycles present in the digraph. Even when measures from the same class are used, conclusions about the given network can still be made: a large average degree with a low probabilistic reachability or low entropy would mean that the given digraph is well-connected and would consequently have a low average distance, as most other nodes can be reached quite quickly.

In this thesis we have shown that it remains, and most likely will remain, improbable to quantify the essence of a network with a single complexity measure. Even though complex networks are capable of being created by simple graph generation rules, finding a measure or a group of measures that attempts to reverse engineer such a rule remains tough. The results do indicate, however, that the combination of measures from different classes allow us to infer the properties of the network, similar to findings of other research. Additionally, we have improved the existing measure evaluation techniques with the extension of four evaluation criteria. With these techniques we have shown that the novel arc

symmetry, the entropy, the probabilistic reachability and the routing complexity measures are capable of capturing inherent network properties in a reasonable manner.

Future research could look into the optimization and improvements of some of the measures implemented for this paper. The development, classification and evaluation of novel network complexity measures should be encouraged and more universal, so that perhaps one day a network's complex characteristics could be described by a single unifying measure. Moreover, the reduction of networks and the subsequent behavior of various network complexity measures could also be explored using the evaluated measures.

References

[1] Arxiv astro-ph network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/ca-AstroPh.

[2] Caida network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/as-caida20071105.

[3] Euroroad network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/subelj_euroroad.

[4] Jazz musicians network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/arenas-jazz.

[5] Pretty good privacy network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/arenas-pgp.

[6] Us power grid network dataset – KONECT, Sept. 2016. http://konect.uni-koblenz.de/networks/opsahl-powergrid.

[7] Google+ network dataset – KONECT, Apr. 2017.

[8] Openflights network dataset – KONECT, Apr. 2017. http://konect.uni-koblenz.de/networks/opsahl-openflights.

[9] Residence hall network dataset – KONECT, Apr. 2017. http://konect.uni-koblenz.de/networks/moreno_oz.

[10] Us airports network dataset – KONECT, Apr. 2017. http://konect.uni-koblenz.de/networks/opsahl-usairport.

[11] Facebook reports fourth quarter and full year 2019 results 2020, Jan 2020. https://investor.fb.com/investor-news/press-release-details/2020/Facebook-Reports-Fourth-Quarter-and-Full-Year-2019-Results/default.aspx.

[12] M. Aigner and G. Ziegler. *Three applications of Euler's formula*, pages 75–80. 01 2010.

[13] A.-L. Barabási. *The Barabási-Albert Model*, chapter 5, pages 8–15.

[14] A.-L. Barabási, Z. N. Oltvai, and S. Wuchty. Characteristics of biological networks. In *Complex networks*, pages 443–457. Springer, 2004.

[15] B. Bollobás, C. Borgs, J. Chayes, and O. Riordan. Directed scale-free graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '03, page 132–139, USA, 2003. Society for Industrial and Applied Mathematics.

[16] C. T. Butts. An axiomatic approach to network complexity. *Journal of Mathematical Sociology*, 24(4):273–301, 2000.

[17] M. Dehmer and S. Pickl. *NETWORK COMPLEXITY MEASURES. AN INFORMATION-THEORETIC APPROACH.* Citeseer.

[18] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[19] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[20] T. Feder. Statistical physics is for the birds. *Physics today*, 60(10):28, 2007.

[21] R. W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, June 1962.
[22] Z. Galil. Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surv.*, 18(1):23–38, Mar. 1986.
[23] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.
[24] J. M. Hernández and P. Van Mieghem. Classification of graph metrics. *Delft University of Technology: Mekelweg, The Netherlands*, pages 1–20, 2011.
[25] P. Hu and W. C. Lau. A survey and taxonomy of graph sampling. *arXiv preprint arXiv:1308.5865*, 2013.
[26] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95, 2007.
[27] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, 04 2001.
[28] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 333–341. ACM, 2018.
[29] S. Kumar, F. Spezzano, V. Subrahmanian, and C. Faloutsos. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 221–230. IEEE, 2016.
[30] J. Leskovec and C. Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2006.
[31] J. Leskovec and E. Horvitz. Planetary-scale views on an instant-messaging network, 2008.
[32] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2–es, 2007.
[33] W. McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.
[34] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.
[35] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *arXiv preprint cs/0209028*, 2002.
[36] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503*, 2011.
[37] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen,

E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[38] L. Willenborg. Computational aspects of survey data processing. 1988. CWI Tract 54, Centre for Mathematics and Computer Science, Amsterdam.

[39] L. Willenborg. Complexity and simplification of networks. 2019. https://www.cbs.nl/en-gb/background/2019/31/complexity-and-simplification-of-networks and https://www.researchgate.net/publication/334971508_Complexity_and_simplification_of_networks. Date accessed: 12-11-2019.

[40] S. Wolfram. A class of models with the potential to represent fundamental physics. *arXiv*, pages arXiv–2004, 2020.

A Appendix - Algorithms

Algorithm 1 Routing Digraph Generation

See Section 5.2.5

Input:
number of nodes n
Probability of connecting p
Output: Routing Digraph G
A = (0...n-1)(0...n-1) ▷ Empty Adjacency Matrix
A[0][1] = 1 ▷ Set the source node
for i in 0:(n-1) **do**
    for j in (i+1):n **do**
        if $p >$ Uniform(0, 1) **then**
            A[i][j] = 1
        end if
    end for
end for
G = DiGraph(A)
for k in 0:(n-2) **do**
    if G.path_exists(0, (k)) == False **then** ▷ (0) is source node
        G.add_edge(0, (k))
    end if
    if G.path_exists((n - k - 1), (n-1)) == False **then** ▷ (n-1) is sink node
        G.add_edge((n - k - 1), (n-1))
    end if
end for
return G

Algorithm 2 Probabilistic Reachability C_{pro}

See Section 5.4.3

Input:
Weakly Connected Directed graph G with E as set of arcs and n total nodes
Starting probability p
Step Size δ
Precision ϵ
Maximum number of iterations max
Number of Monte Carlo Simulations MC
Output: Probability p^{change}
$U \leftarrow \{\}$
for e in E **do** ▷ Find unsaturated pairs
    r = ReverseDirection(e)
    if r not in G **then**
        U.append(r)
    end if
end for
$Prop = SLSCC(G)/n$ ▷ SLSCC finds the size of the largest strongly connected component
$I = 0$
while $I < max$ **do**
    $RProp \leftarrow \{\}$
    for i in 0:MC **do**
        $G_{aug} = G$.copy()
        for r in U **do**
            if $p >$ Uniform(0, 1) **then**
                G_{aug}.add_arc(r)
            end if
        end for
        $RProp$.append($SLSCC(G_{aug})/n$)
    end for
    $CProp$ = mean($RProp$)
    $step$ = abs($CProp - Prop$)
    if $step < \epsilon$ **then** ▷ If change in proportion is $< \epsilon$ break while loop
        BREAK
    else
        $Prop = CProp$
        $p \mathrel{+}= \delta$
        $I \mathrel{+}= 1$
    end if
end while
p^{change} = p
return p^{change}

Algorithm 3 Search Complexity C_{sea}

See Section 5.4.3

Input: Connected graph G with E as set of edges and m total edges
Output: Search Complexity value C_{sea}
$OD = G$.get_odd_degree() ▷ Only select nodes with odd degree
$P = OD$.get_all_pairs() ▷ Acquire ALL possible pairs
$P_l \leftarrow \{\}$
$G_{temp} \leftarrow \{\}$
for p in P **do**
    $l =$ Dijkstra(p) ▷ Dijkstra [18] shortest path
    P_l.append(l)
    G_{temp}.add_edge(p, weight $= 1/l$)
end for
$BE = G_{temp}$.max_weight_matching() ▷ Find best edges from all pairs [22]
$BE_r =$ ReverseSort(BE) ▷ Put best edges up front
$\tau_G = m$
$\lambda_G = m$
$cnt = 0$
for i in BE_r **do**
    $CP_l = P_l$.get(i) ▷ Get path length of edge i
    if $cnt < (|BE_r| - 1)$ **then**
        $\lambda_G \mathrel{+}= CP_l$
    end if
    $\tau_G \mathrel{+}= CP_l$
    $cnt \mathrel{+}= 1$
end for
$C_{sea} = \lambda_G / \tau_G$
return C_{sea}

B Appendix - Tables

B.1 Generated (di)graphs results

Table 12: Values and computation times of complexity measures performed on Complete Graphs

Complete Graphs

$\boldsymbol{n}$	m	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sea}	t (sec)
50	1225	49	0.0000	1.0000	0.0469	0.9992	1.4530
100	4950	99	0.0000	1.0000	0.2982	0.9998	21.9415
200	19900	199	0.0000	1.0000	2.6092	1.0000	348.9377
400	79800	399	0.0000	1.0000	22.2541	1.0000	5197.2807
800	319600	799	0.0000	1.0000	384.7493	NaN	NaN
1000	499500	999	0.0000	1.0000	350.3124	NaN	NaN
1500	1124250	1499	0.0000	1.0000	1075.2023	NaN	NaN
2000	1999000	1999	0.0000	1.0000	2759.4075	NaN	NaN
2500	3123750	2499	0.0000	1.0000	5336.4636	NaN	NaN
3000	4498500	2999	0.0000	1.0000	8835.7128	NaN	NaN

Table 13: Values and computation times of complexity measures performed on Complete DiGraphs

Complete DiGraphs

$\boldsymbol{n}$	m	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sym}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)
50	2450	98	0.0	1.0	0.0781	0.0	0.0000	0.0	0.4218	0.3	0.5937
100	9900	198	0.0	1.0	0.3750	0.0	0.0156	0.0	11.2614	0.3	4.2029
200	39800	398	0.0	1.0	2.4998	0.0	0.0312	0.0	65.5550	0.3	11.0396
400	159600	798	0.0	1.0	18.8617	0.0	0.1719	0.0	378.8512	0.3	43.3929
800	639200	1598	0.0	1.0	151.4199	0.0	0.6719	0.0	2469.2863	0.3	157.4676
1000	999000	1998	0.0	1.0	287.9932	0.0	0.9843	0.0	4679.9970	0.3	248.8862
1500	2248500	2998	0.0	1.0	958.3062	0.0	2.4256	NaN	NaN	0.3	587.5515
2000	3998000	3998	0.0	1.0	2264.4300	0.0	4.5622	NaN	NaN	0.3	1053.0050
2500	6247500	4998	0.0	1.0	4413.5200	0.0	8.1869	NaN	NaN	0.3	1825.6667
3000	8997000	5998	0.0	1.0	7600.5040	0.0	11.4130	NaN	NaN	0.3	3571.8504

Table 14: Values and computation times of complexity measures performed on Barabási-Albert Graphs

Barabási-Albert Graphs

n	m	$\mathbf{m_b}$	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sea}	t (sec)
50	96	2	3.8400	0.0000	2.6922	0.0156	0.9808	0.0156
100	196	2	3.9200	0.0000	2.9305	0.0312	0.9860	0.0937
200	396	2	3.9600	0.0000	3.3251	0.1406	0.9955	1.0468
400	796	2	3.9800	0.0000	3.6138	0.6093	0.9977	10.0644
800	1596	2	3.9900	0.0000	3.9490	2.4373	0.9989	63.4909
1000	1996	2	3.9920	0.0000	4.0969	3.9216	0.9982	137.6870
1500	2996	2	3.9947	0.0000	4.3075	9.0981	0.9988	463.5864
2000	3996	2	3.9960	0.0000	4.3681	16.5134	0.9989	1206.9140
2500	4996	2	3.9968	0.0000	4.4871	27.5150	0.9993	2332.8538
3000	5996	2	3.9973	0.0000	4.6129	38.7881	0.9992	3782.9594
1000	999	1	1.998	0.0000	7.2684	5.5947	0.9910	2381.4023
1000	1996	2	3.992	0.0000	4.0927	4.0779	0.9986	129.2407
1000	2991	3	5.982	0.0000	3.4582	4.5466	0.9991	821.5797
1000	3984	4	7.968	0.0000	3.1794	5.1868	0.9995	364.4400
1000	4975	5	9.950	0.0000	2.9423	5.6746	0.9996	971.3843
1000	5964	6	11.928	0.0000	2.8532	6.2834	0.9998	540.3077
1000	6951	7	13.902	0.0000	2.7070	6.7362	0.9997	1150.9461
1000	7936	8	15.872	0.0000	2.6637	7.4852	0.9999	682.6565
1000	8919	9	17.838	0.0000	2.6054	8.0317	0.9999	1343.1095
1000	9900	10	19.800	0.0000	2.5511	8.5784	0.9998	928.9402
1000	10879	11	21.758	0.0000	2.5027	9.2520	0.9999	1558.2670
1000	11856	12	23.712	0.0000	2.4600	10.0801	0.9999	1280.3858
1000	12831	13	25.662	0.0000	2.4068	10.3922	0.9999	1874.8925
1000	13804	14	27.608	0.0000	2.3633	11.1572	0.9999	1486.5687
1000	14775	15	29.550	0.0000	2.3314	11.4863	0.9999	1856.7379
1000	15744	16	31.488	0.0000	2.2893	12.4682	0.9999	1770.8343
1000	16711	17	33.422	0.0000	2.2585	13.0810	0.9999	2106.6168
1000	17676	18	35.352	0.0000	2.2224	13.9854	0.9999	1497.8181
1000	18639	19	37.278	0.0000	2.1932	14.4389	0.9999	2341.5603
1000	19600	20	39.200	0.0000	2.1642	15.0471	0.9999	1982.5939
1000	20559	21	41.118	0.0000	2.1407	15.5614	1.0000	2780.5087
1000	21516	22	43.032	0.0000	2.1262	16.2043	1.0000	2344.0458
1000	22471	23	44.942	0.0000	2.1001	16.8781	1.0000	2858.2788
1000	23424	24	46.848	0.0000	2.0795	17.4877	1.0000	2315.7737
1000	24375	25	48.750	0.0000	2.0598	18.1108	1.0000	2917.5491
1000	25324	26	50.648	0.0000	2.0460	18.6744	1.0000	3113.0626
1000	26271	27	52.542	0.0000	2.0339	19.0358	1.0000	3197.0886
1000	27216	28	54.432	0.0000	2.0220	20.0544	1.0000	2606.3860
1000	28159	29	56.318	0.0000	2.0087	20.4628	1.0000	3691.8199
1000	29100	30	58.200	0.0000	1.9978	21.7478	1.0000	3282.7960
1000	30039	31	60.078	0.0000	1.9883	22.5516	1.0000	3812.2071
1000	30976	32	61.952	0.0000	1.9801	22.8676	1.0000	3135.9840
1000	31911	33	63.822	0.0000	1.9719	23.4778	1.0000	3687.3431
1000	32844	34	65.688	0.0000	1.9672	24.6931	1.0000	3515.9538
1000	33775	35	67.550	0.0000	1.9599	24.2828	1.0000	3790.1972
1000	34704	36	69.408	0.0000	1.9541	25.1761	1.0000	3498.4871
1000	35631	37	71.262	0.0000	1.9484	25.3168	1.0000	4787.4515
1000	36556	38	73.112	0.0000	1.9435	26.4093	1.0000	3746.6489
1000	37479	39	74.958	0.0000	1.9386	27.6590	1.0000	4413.5404
1000	38400	40	76.800	0.0000	1.9364	27.6762	1.0000	4572.6367
1000	39319	41	78.638	0.0000	1.9314	28.1603	1.0000	4065.3385
1000	40236	42	80.472	0.0000	1.9282	28.6304	1.0000	4472.1091
1000	41151	43	82.302	0.0000	1.9248	29.7221	1.0000	4904.8862
1000	42064	44	84.128	0.0156	1.9218	29.9106	1.0000	4823.2017
1000	42975	45	85.950	0.0000	1.9190	31.2854	1.0000	5115.8566
1000	43884	46	87.768	0.0000	1.9165	31.8191	1.0000	4706.7512
1000	44791	47	89.582	0.0000	1.9139	31.6903	1.0000	5553.7049
1000	45696	48	91.392	0.0000	1.9117	33.0856	1.0000	5021.6125
1000	46599	49	93.198	0.0000	1.9096	33.5684	1.0000	5310.1188
1000	47500	50	95.000	0.0000	1.9069	34.8487	1.0000	5521.5978

Table 15: Values and computation times of complexity measures performed on Scale-Free DiGraphs

Scale-Free DiGraphs

n	m	α	β	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sym}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)
50	73	0.41	0.54	2.9200	0.000	0.2392	0.0000	0.8312	0.0000	0.5131	0.0156	0.55	2.2186
100	144	0.41	0.54	2.8800	0.000	0.3762	0.0156	0.9427	0.0000	0.3901	0.0156	1.00	12.8780
200	329	0.41	0.54	3.2900	0.000	0.1931	0.0000	0.9599	0.0000	0.3814	0.0625	0.52	8.4714
400	666	0.41	0.54	3.3300	0.000	0.0318	0.0156	0.9577	0.0000	0.5250	0.0469	0.65	29.3346
800	1336	0.41	0.54	3.3400	0.000	0.2362	0.1719	0.9697	0.0000	0.3331	0.9531	1.00	137.0022
1000	1861	0.41	0.54	3.7220	0.000	0.1293	0.1406	0.9812	0.0000	0.3603	0.6719	1.00	182.5142
1500	2572	0.41	0.54	3.4293	0.000	0.0554	0.1562	0.9760	0.0000	0.4688	0.7812	1.00	292.4130
2000	3381	0.41	0.54	3.3810	0.000	0.2319	0.8593	0.9761	0.0000	0.3273	4.4384	1.00	412.7940
2500	4383	0.41	0.54	3.5064	0.000	0.2521	1.6249	0.9906	0.0000	0.3063	8.4712	1.00	513.2149
3000	5100	0.41	0.54	3.4000	0.000	0.2750	2.2512	0.9923	0.0156	0.2884	11.7046	1.00	655.3780
1000	1254	0.75	0.20	2.5080	0.0000	0.3290	0.2188	0.9968	0.0156	0.3794	0.9699	1.00	150.0330
1000	1248	0.74	0.21	2.4960	0.0000	0.2434	0.1406	0.9920	0.0000	0.3591	0.6542	1.00	144.7362
1000	1244	0.73	0.22	2.4880	0.0000	0.1826	0.1250	0.9928	0.0000	0.4230	0.6094	1.00	141.2465
1000	1332	0.72	0.23	2.6640	0.0000	0.2723	0.1875	0.9887	0.0000	0.3704	0.9687	1.00	143.3398
1000	1308	0.71	0.24	2.6160	0.0000	0.4118	0.2500	0.9855	0.0000	0.3179	1.2499	1.00	140.3167
1000	1332	0.70	0.25	2.6640	0.0000	0.4650	0.2656	0.9872	0.0000	0.3398	1.2499	1.00	142.6505
1000	1338	0.69	0.26	2.6760	0.0000	0.2828	0.1719	0.9925	0.0000	0.3602	0.7845	1.00	140.1544
1000	1353	0.68	0.27	2.7060	0.0000	0.3281	0.2031	0.9904	0.0000	0.3545	1.0624	1.00	144.8253
1000	1389	0.67	0.28	2.7780	0.0000	0.6420	0.3437	0.9942	0.0000	0.2894	1.6561	1.00	142.9744
1000	1390	0.66	0.29	2.7800	0.0000	0.6773	0.2955	0.9950	0.0156	0.3093	1.4036	1.00	147.4111
1000	1397	0.65	0.30	2.7940	0.0000	0.3385	0.2344	0.9928	0.0000	0.3300	1.1562	1.00	144.0601
1000	1386	0.64	0.31	2.7720	0.0000	0.1176	0.0937	0.9928	0.0000	0.4367	0.5937	1.00	146.3909
1000	1444	0.63	0.32	2.8880	0.0000	0.7093	0.3437	0.9938	0.0000	0.2941	1.6874	1.00	147.7752
1000	1445	0.62	0.33	2.8900	0.0000	0.4786	0.3281	0.9882	0.0000	0.3062	1.5624	1.00	147.8863
1000	1447	0.61	0.34	2.8940	0.0000	0.7070	0.3437	0.9938	0.0000	0.2894	1.7030	1.00	151.6839
1000	1510	0.60	0.35	3.0200	0.0000	0.4869	0.2669	0.9934	0.0156	0.3194	1.3562	1.00	151.0195
1000	1532	0.59	0.36	3.0640	0.0000	0.7531	0.4843	0.9798	0.0000	0.2527	2.5936	1.00	153.6451
1000	1552	0.58	0.37	3.1040	0.0000	0.5568	0.4531	0.9839	0.0000	0.2986	2.2811	1.00	153.8100
1000	1592	0.57	0.38	3.1840	0.0000	0.8432	0.4687	0.9899	0.0000	0.2457	2.4530	1.00	155.9653
1000	1604	0.56	0.39	3.2080	0.0000	0.5186	0.3906	0.9763	0.0000	0.2745	1.9530	1.00	156.8909
1000	1654	0.55	0.40	3.3080	0.0000	0.7346	0.5000	0.9794	0.0000	0.2440	2.5623	1.00	162.8647
1000	1615	0.54	0.41	3.2300	0.0000	0.6992	0.4844	0.9839	0.0000	0.2373	2.6561	1.00	158.9794
1000	1665	0.53	0.42	3.3300	0.0000	0.6538	0.4687	0.9868	0.0000	0.2462	2.4690	1.00	158.0678
1000	1670	0.52	0.43	3.3400	0.0000	0.8244	0.5000	0.9934	0.0000	0.2250	2.6261	1.00	162.7665
1000	1649	0.51	0.44	3.2980	0.0000	0.7134	0.4375	0.9927	0.0000	0.2633	2.1730	1.00	162.3578
1000	1735	0.50	0.45	3.4700	0.0000	1.0215	0.6250	0.9890	0.0000	0.2237	3.3604	1.00	200.9182
1000	1779	0.49	0.46	3.5580	0.0000	0.9118	0.6093	0.9843	0.0000	0.2194	3.3219	1.00	168.8769
1000	1755	0.48	0.47	3.5100	0.0000	0.5855	0.4531	0.9795	0.0000	0.2488	2.5453	1.00	166.5654
1000	1820	0.47	0.48	3.6400	0.0000	0.8404	0.6718	0.9764	0.0000	0.2121	3.6091	1.00	170.6782
1000	1821	0.46	0.49	3.6420	0.0000	0.9219	0.6562	0.9769	0.0000	0.1987	3.6404	1.00	169.1197
1000	1875	0.45	0.50	3.7500	0.0000	0.7820	0.6406	0.9675	0.0000	0.1961	3.4060	1.00	176.1851
1000	1841	0.44	0.51	3.6820	0.0000	0.7113	0.6093	0.9707	0.0000	0.2071	3.3904	1.00	171.9177
1000	1866	0.43	0.52	3.7320	0.0000	0.8745	0.7043	0.9737	0.0000	0.1907	3.7497	1.00	174.1362
1000	1989	0.42	0.53	3.9780	0.0000	0.9453	0.7812	0.9708	0.0000	0.1807	4.2184	1.00	177.9378
1000	1889	0.41	0.54	3.7780	0.0000	0.6440	0.6094	0.9576	0.0000	0.2136	3.4207	1.00	175.4218
1000	2001	0.40	0.55	4.0020	0.0000	1.0772	0.8606	0.9755	0.0000	0.1636	4.6403	1.00	181.1657
1000	2093	0.39	0.56	4.1860	0.0000	1.0812	0.9231	0.9780	0.0000	0.1795	5.0153	1.00	184.7020
1000	2064	0.38	0.57	4.1280	0.0000	0.9527	0.8593	0.9671	0.0000	0.1774	4.6665	1.00	186.0080
1000	2152	0.37	0.58	4.3040	0.0000	0.8880	0.7656	0.9670	0.0000	0.1814	4.2979	1.00	188.7221
1000	2097	0.36	0.59	4.1940	0.0000	0.8299	0.7812	0.9390	0.0000	0.1767	4.4841	1.00	183.7324
1000	2174	0.35	0.60	4.3480	0.0000	0.9384	0.9062	0.9512	0.0000	0.1677	5.1416	1.00	190.2234
1000	2250	0.34	0.61	4.5000	0.0000	0.9037	0.8437	0.9573	0.0000	0.1688	5.0019	1.00	190.7252
1000	2101	0.33	0.62	4.2020	0.0000	0.8615	0.9062	0.9343	0.0156	0.1583	5.3314	1.00	185.9894
1000	2263	0.32	0.63	4.5260	0.0000	0.8274	0.8593	0.9456	0.0000	0.1672	5.0488	1.00	193.2310
1000	2345	0.31	0.64	4.6900	0.0156	0.9138	0.9687	0.9407	0.0000	0.1582	5.5321	1.00	199.0206
1000	2334	0.30	0.65	4.6680	0.0000	1.0354	0.9531	0.9610	0.0000	0.1529	5.4684	1.00	199.7444
1000	2383	0.29	0.66	4.7660	0.0000	0.9507	1.0468	0.9291	0.0000	0.1537	6.5321	1.00	200.2672
1000	2371	0.28	0.67	4.7420	0.0000	0.9406	0.9687	0.9393	0.0000	0.1442	6.0707	1.00	196.8793
1000	2478	0.27	0.68	4.9560	0.0000	1.1455	1.2187	0.9431	0.0000	0.1337	7.3299	1.00	207.3351
1000	2509	0.26	0.69	5.0180	0.0000	0.8701	0.9687	0.9418	0.0000	0.1476	5.9371	1.00	204.1463

Table 16: Values and computation times of complexity measures performed on Erdős-Rényi Graphs

Erdős-Rényi Graphs

n	m	p	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sea}	t (sec)
50	184	0.1565	7.3600	0.0000	2.1012	0.0156	0.9900	0.0469
100	449	0.0921	8.9800	0.0000	2.3396	0.0625	0.9979	0.5937
200	974	0.0530	9.7400	0.0000	2.4652	0.2812	0.9991	4.8248
400	2465	0.0300	12.3250	0.0000	2.6783	1.3437	0.9996	39.6729
800	5261	0.0167	13.1525	0.0000	2.8410	6.5791	0.9998	365.8848
1000	6862	0.0138	13.7240	0.0000	2.8849	7.3758	0.9997	732.6500
1500	11062	0.0098	14.7493	0.0000	2.9654	17.8477	0.9999	2864.6365
2000	15059	0.0076	15.0590	0.0000	3.0596	33.1422	0.9999	6630.0010
2500	19690	0.0063	15.7520	0.0000	3.1261	57.1702	0.9999	13640.9391
3000	24213	0.0053	16.1420	0.0000	3.1637	81.0407	0.9999	24368.2329
1000	1321	0.0028	2.6420	0.0156	6.5429	10.4680	0.9939	779.2049
1000	2689	0.0055	5.3780	0.0156	4.2103	17.1310	0.9987	1489.2746
1000	4253	0.0083	8.5060	0.0000	3.5408	22.4086	0.9991	1880.8010
1000	5435	0.0111	10.8700	0.0156	3.1104	29.3119	0.9995	2275.3437
1000	6983	0.0139	13.9660	0.0156	2.9053	33.4250	0.9999	2471.6413
1000	8399	0.0167	16.7980	0.0156	2.7546	40.0663	NaN	NaN
1000	9810	0.0195	19.6200	0.0312	2.6556	46.7547	NaN	NaN
1000	11159	0.0223	22.3180	0.0312	2.5601	52.3645	NaN	NaN
1000	12690	0.0251	25.3800	0.0156	2.4993	56.2922	NaN	NaN
1000	14028	0.0279	28.0560	0.0312	2.4111	63.0969	NaN	NaN
1000	15299	0.0307	30.5980	0.0312	2.3422	68.6731	NaN	NaN
1000	16866	0.0335	33.7320	0.0312	2.2886	76.2593	NaN	NaN
1000	17974	0.0363	35.9480	0.0312	2.2217	79.8788	NaN	NaN
1000	19385	0.0391	38.7700	0.0313	2.1691	86.1126	NaN	NaN
1000	20794	0.0419	41.5880	0.0313	2.1269	91.5605	NaN	NaN
1000	22390	0.0447	44.7800	0.0469	2.0868	99.6640	NaN	NaN
1000	23523	0.0475	47.0460	0.0313	2.0512	103.2895	NaN	NaN
1000	25065	0.0503	50.1300	0.0468	2.0256	110.1101	NaN	NaN
1000	26602	0.0531	53.2040	0.0625	2.0056	118.0778	NaN	NaN
1000	28278	0.0559	56.5560	0.0469	1.9838	124.1404	NaN	NaN
1000	29340	0.0587	58.6800	0.0469	1.9691	129.5378	NaN	NaN
1000	30635	0.0615	61.2700	0.0625	1.9624	132.0498	NaN	NaN
1000	32227	0.0643	64.4540	0.0469	1.9513	139.2867	NaN	NaN
1000	33797	0.0671	67.5940	0.0625	1.9427	144.4329	NaN	NaN
1000	35199	0.0699	70.3980	0.0625	1.9359	150.1942	NaN	NaN
1000	36405	0.0727	72.8100	0.0625	1.9330	153.8217	NaN	NaN
1000	37388	0.0755	74.7760	0.0781	1.9275	161.8510	NaN	NaN
1000	39125	0.0783	78.2500	0.0781	1.9241	164.7483	NaN	NaN
1000	40076	0.0811	80.1520	0.0781	1.9207	171.4660	NaN	NaN
1000	41729	0.0839	83.4580	0.0781	1.9170	178.9258	NaN	NaN
1000	43378	0.0867	86.7560	0.0781	1.9138	183.8620	NaN	NaN
1000	44807	0.0895	89.6140	0.0781	1.9101	191.3422	NaN	NaN
1000	46195	0.0923	92.3900	0.0781	1.9078	198.1982	NaN	NaN
1000	47428	0.0951	94.8560	0.0781	1.9053	203.8149	NaN	NaN
1000	49062	0.0979	98.1240	0.0937	1.9014	208.9661	NaN	NaN
1000	50249	0.1007	100.4980	0.0781	1.8992	214.7062	NaN	NaN
1000	51098	0.1035	102.1960	0.0781	1.8973	217.7236	NaN	NaN
1000	53273	0.1063	106.5460	0.1094	1.8938	224.4809	NaN	NaN
1000	54846	0.1091	109.6920	0.1094	1.8911	227.3912	NaN	NaN
1000	56000	0.1119	112.0000	0.1094	1.8888	235.5589	NaN	NaN
1000	57609	0.1147	115.2180	0.1250	1.8854	241.6562	NaN	NaN
1000	58705	0.1175	117.4100	0.0938	1.8818	246.8472	NaN	NaN
1000	60151	0.1203	120.3020	0.1094	1.8794	251.9692	NaN	NaN
1000	61545	0.1231	123.0900	0.0938	1.8764	258.3795	NaN	NaN
1000	63201	0.1259	126.4020	0.1094	1.8746	262.8071	NaN	NaN
1000	64403	0.1287	128.8060	0.1250	1.8714	269.7343	NaN	NaN
1000	66051	0.1315	132.1020	0.0937	1.8683	275.1362	NaN	NaN
1000	67373	0.1343	134.7460	0.1094	1.8657	280.5666	NaN	NaN
1000	68601	0.1371	137.2020	0.1250	1.8628	287.3298	NaN	NaN
1000	70004	0.1399	140.0080	0.1093	1.8609	291.2518	NaN	NaN

Table 17: Values and computation times of complexity measures performed on Erdős-Rényi DiGraphs

Erdős-Rényi DiGraphs

n	m	p	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sym}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)
50	371	0.1565	14.8400	0.0000	2.9335	0.0156	0.9087	0.0000	0.0052	0.0781	0.40	1.6874
100	949	0.0921	18.9800	0.0156	3.1723	0.0313	0.9356	0.0000	0.0044	0.3906	0.39	3.5636
200	2107	0.0530	21.0700	0.0000	3.2208	0.1719	0.9661	0.0000	0.0029	1.6249	0.32	2.6873
400	4805	0.0300	24.0250	0.0000	3.5057	0.9398	0.9884	0.0000	0.0004	8.6950	0.33	7.8292
800	10641	0.0167	26.6025	0.0000	3.7252	4.5622	0.9907	0.0156	0.0000	47.0400	0.31	8.3142
1000	13924	0.0138	27.8480	0.0000	3.7702	5.3683	0.9940	0.0000	0.0001	80.5648	0.31	10.9381
1500	21983	0.0098	29.3107	0.0000	3.8999	12.0491	0.9947	0.0156	0.0000	235.4659	0.31	17.2854
2000	30241	0.0076	30.2410	0.0000	3.9771	22.1135	0.9967	0.0156	0.0001	480.9503	0.31	23.7038
2500	39403	0.0063	31.5224	0.0000	4.0318	36.0554	0.9969	0.0156	0.0001	827.5908	0.31	37.8228
3000	48146	0.0053	32.0973	0.0000	4.0543	69.3322	0.9975	0.0312	0.0000	1572.5826	0.31	39.3146
1000	2798	0.0028	5.5960	0.0000	5.6762	12.3001	0.9957	0.0309	0.0096	46.1950	0.64	152.3140
1000	5394	0.0055	10.7880	0.0000	4.2043	18.5483	0.9963	0.0468	0.0007	80.0133	0.32	19.2527
1000	8402	0.0083	16.8040	0.0156	3.5009	24.4601	0.9892	0.0469	0.0001	111.2263	0.31	17.7072
1000	11052	0.0111	22.1040	0.0156	3.1085	30.8953	0.9897	0.0781	0.0000	133.8620	0.30	11.0176
1000	13955	0.0139	27.9100	0.0156	2.8918	36.8835	0.9840	0.1094	0.0000	154.9203	0.30	13.3479
1000	16726	0.0167	33.4520	0.0156	2.7534	42.6350	0.9844	0.1093	0.0000	186.1621	0.30	15.8307
1000	19606	0.0195	39.2120	0.0156	2.6591	48.6334	0.9816	0.1250	0.0000	205.5570	0.30	17.8845
1000	22140	0.0223	44.2800	0.0312	2.5706	54.7290	0.9796	0.1719	0.0000	231.8333	0.30	20.3472
1000	25099	0.0251	50.1980	0.0313	2.4940	62.5929	0.9737	0.1718	0.0000	253.4466	0.30	23.7690
1000	28026	0.0279	56.0520	0.0156	2.4125	67.1567	0.9707	0.2031	0.0000	281.8988	0.30	25.0606
1000	30750	0.0307	61.5000	0.0312	2.3451	72.1847	0.9688	0.2188	0.0000	307.9022	0.30	27.0960
1000	33501	0.0335	67.0020	0.0312	2.2834	78.6151	0.9653	0.2656	0.0000	331.2841	0.30	29.1313
1000	36358	0.0363	72.7160	0.0312	2.2194	86.6107	0.9634	0.2500	0.0000	355.4264	0.30	31.6092
1000	38817	0.0391	77.6340	0.0469	2.1712	90.7571	0.9601	0.2812	0.0000	380.3953	0.30	33.6719
1000	42019	0.0419	84.0380	0.0312	2.1234	96.5035	0.9561	0.2812	0.0000	404.1020	0.30	36.0541
1000	44781	0.0447	89.5620	0.0312	2.0839	103.7393	0.9560	0.2969	0.0000	428.3961	0.30	37.6522
1000	47376	0.0475	94.7520	0.0468	2.0521	108.5122	0.9510	0.3437	0.0000	456.1846	0.30	40.6181
1000	50566	0.0503	101.1320	0.0469	2.0243	114.6351	0.9498	0.3750	0.0000	483.0516	0.30	44.1690
1000	53296	0.0531	106.5920	0.0469	2.0018	122.1398	0.9454	0.4062	0.0000	496.8530	0.30	44.3016
1000	56162	0.0559	112.3240	0.0469	1.9856	126.2011	0.9451	0.3906	0.0000	524.8561	0.30	46.4771
1000	58960	0.0587	117.9200	0.0469	1.9705	131.5938	0.9426	0.3749	0.0000	553.8024	0.30	48.9500
1000	61946	0.0615	123.8920	0.0468	1.9583	141.3847	0.9420	0.5468	0.0000	577.2576	0.30	50.6478
1000	64059	0.0643	128.1180	0.0469	1.9495	144.6296	0.9353	0.5156	0.0000	596.3216	0.30	53.0886
1000	66986	0.0671	133.9720	0.0625	1.9429	151.7814	0.9319	0.4687	0.0000	624.1873	0.30	56.6828
1000	69941	0.0699	139.8820	0.0781	1.9369	156.0782	0.9303	0.4844	0.0000	643.4944	0.30	56.4554
1000	72597	0.0727	145.1940	0.0625	1.9318	163.7047	0.9268	0.6889	0.0000	669.9760	0.30	58.5589
1000	75702	0.0755	151.4040	0.0781	1.9271	170.5891	0.9224	0.7812	0.0000	692.3521	0.30	60.3873
1000	78160	0.0783	156.3200	0.0625	1.9240	173.6046	0.9231	0.5630	0.0000	713.0085	0.30	61.8815
1000	80779	0.0811	161.5580	0.0625	1.9201	182.8216	0.9179	0.5468	0.0000	750.5691	0.30	66.4640
1000	84131	0.0839	168.2620	0.0625	1.9169	186.3268	0.9142	0.7031	0.0000	758.5826	0.30	67.0267
1000	86590	0.0867	173.1800	0.0781	1.9141	194.6446	0.9149	0.6718	0.0000	790.5235	0.30	68.7337
1000	89497	0.0895	178.9940	0.0781	1.9108	202.0619	0.9108	0.6093	0.0000	814.0444	0.30	70.5760
1000	92016	0.0923	184.0320	0.0625	1.9077	205.8473	0.9052	0.7636	0.0000	830.1134	0.30	72.3901
1000	94673	0.0951	189.3460	0.0937	1.9053	213.3686	0.9035	0.7343	0.0000	851.0961	0.30	75.6881
1000	98403	0.0979	196.8060	0.0938	1.9021	218.8441	0.9017	0.7499	0.0000	874.8928	0.30	76.2983
1000	101326	0.1007	202.6520	0.0781	1.8991	224.4748	0.8987	0.7031	0.0000	894.5061	0.30	78.2482
1000	103608	0.1035	207.2160	0.0781	1.8963	232.3633	0.8979	0.7187	0.0000	928.4583	0.30	79.4368
1000	106374	0.1063	212.7480	0.0781	1.8932	238.6161	0.8961	0.7343	0.0000	957.7443	0.30	83.9687
1000	108940	0.1091	217.8800	0.0938	1.8908	244.5538	0.8913	0.8124	0.0000	974.5113	0.30	83.9559
1000	112013	0.1119	224.0260	0.0938	1.8880	249.0088	0.8846	0.7968	0.0000	1000.5811	0.30	85.7534
1000	115159	0.1147	230.3180	0.0937	1.8855	252.6703	0.8854	0.9135	0.0000	1011.8938	0.30	91.2679
1000	117774	0.1175	235.5480	0.1094	1.8827	259.0367	0.8823	0.8906	0.0000	1044.7777	0.30	90.0000
1000	120689	0.1203	241.3780	0.0937	1.8795	265.7714	0.8815	0.9062	0.0000	1059.3067	0.30	91.6030
1000	123655	0.1231	247.3100	0.0937	1.8765	273.8081	0.8754	0.8749	0.0000	1083.2222	0.30	94.1719
1000	125631	0.1259	251.2620	0.1094	1.8739	280.1252	0.8737	1.0168	0.0000	1109.6884	0.30	97.4387
1000	128914	0.1287	257.8280	0.0938	1.8712	283.9339	0.8711	0.9062	0.0000	1133.0100	0.30	97.6459
1000	131086	0.1315	262.1720	0.1094	1.8684	289.9489	0.8710	0.9999	0.0000	1143.4898	0.30	99.8460
1000	134071	0.1343	268.1420	0.1094	1.8658	296.3890	0.8675	0.9218	0.0000	1184.6930	0.30	103.2430
1000	136831	0.1371	273.6620	0.1094	1.8628	302.6849	0.8644	1.0012	0.0000	1196.1281	0.30	102.6690
1000	140127	0.1399	280.2540	0.1406	1.8599	310.3431	0.8617	0.9531	0.0000	1211.6052	0.30	105.7742

Table 18: Values and computation times of complexity measures performed on routing DiGraphs

Routing DiGraphs

n	m	p	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)	C_{rou}	t (sec)
50	194	0.1565	7.7600	0.0000	0.7082	0.0000	0.0953	0.0312	0.66	7.2977	8.20	0.0000
100	430	0.0921	8.6000	0.0000	0.7078	0.0156	0.1039	0.0625	0.44	6.3433	7.26	0.0000
200	1119	0.0530	11.1900	0.0000	0.8210	0.0469	0.0834	0.2812	0.46	18.0613	12.02	0.0156
400	2343	0.0300	11.7150	0.0000	0.6853	0.1406	0.0845	0.8906	0.35	13.2359	10.04	0.0312
800	5369	0.0167	13.4225	0.0000	0.8273	0.6718	0.0708	3.9685	0.40	56.2255	13.64	0.1094
1000	6968	0.0138	13.9360	0.0000	0.8045	0.9687	0.0789	5.7198	0.32	21.4758	13.09	0.2031
1500	11397	0.0098	15.1960	0.0156	0.8861	2.3918	0.0761	15.4094	0.32	31.7436	15.70	0.3750
2000	15410	0.0076	15.4100	0.0000	0.8675	3.9997	0.0722	25.4603	0.33	56.9937	14.07	0.7500
2500	19433	0.0063	15.5464	0.0000	0.8741	6.1584	0.0737	39.2947	0.33	72.4774	15.16	1.2523
3000	24479	0.0053	16.3193	0.0000	0.9189	9.3757	0.0677	63.6250	0.33	89.9202	16.71	2.0468
1000	6960	0.0138	13.9200	0.0000	0.8250	1.0155	0.0753	5.8101	0.31	12.8116	13.83	0.1406
1000	8266	0.0163	16.5320	0.0000	0.9096	1.2812	0.0667	8.0945	0.31	14.7561	15.95	0.1562
1000	9320	0.0188	18.6400	0.0000	1.0167	1.5481	0.0613	10.3616	0.31	16.2382	18.51	0.1562
1000	10511	0.0213	21.0220	0.0000	1.0037	1.6405	0.0580	11.5474	0.31	17.7793	18.89	0.1562
1000	12099	0.0238	24.1980	0.0000	1.0365	1.9426	0.0536	14.6092	0.31	20.3290	21.73	0.1562
1000	13235	0.0262	26.4700	0.0000	1.0779	2.1874	0.0497	18.8291	0.31	22.1951	25.73	0.1719
1000	14362	0.0287	28.7240	0.0000	1.0913	2.3123	0.0466	19.2005	0.31	23.3918	30.06	0.2187
1000	15679	0.0312	31.3580	0.0000	1.0874	2.5623	0.0446	21.2844	0.31	25.4094	30.94	0.2031
1000	16767	0.0337	33.5340	0.0000	1.0896	2.6883	0.0429	23.5173	0.31	26.7842	33.49	0.2344
1000	18178	0.0362	36.3560	0.0000	1.0734	2.9998	0.0423	25.2859	0.31	29.0301	34.57	0.1875
1000	19281	0.0387	38.5620	0.0000	1.0778	3.0154	0.0423	26.7495	0.31	30.4219	35.80	0.2031
1000	20220	0.0412	40.4400	0.0000	1.0755	3.0936	0.0409	28.2790	0.31	33.2393	38.31	0.2031
1000	21932	0.0437	43.8640	0.0000	1.0779	3.4841	0.0395	32.1378	0.31	34.3944	42.71	0.2344
1000	22926	0.0461	45.8520	0.0000	1.0495	3.5154	0.0394	32.8967	0.31	35.6686	44.37	0.2031
1000	24442	0.0486	48.8840	0.0000	1.0425	3.6749	0.0392	35.5516	0.31	37.6363	44.67	0.2343
1000	25770	0.0511	51.5400	0.0000	1.0521	3.9241	0.0374	38.4890	0.31	42.2202	49.71	0.2500
1000	26721	0.0536	53.4420	0.0000	1.0538	3.9381	0.0366	40.0998	0.31	41.1432	52.96	0.2187
1000	28184	0.0561	56.3680	0.0000	1.0437	4.1912	0.0359	42.1152	0.31	43.0062	54.32	0.2500
1000	29124	0.0586	58.2480	0.0000	1.0296	4.3591	0.0362	45.9602	0.31	44.4431	55.16	0.2344
1000	30635	0.0611	61.2700	0.0000	1.0239	4.5480	0.0352	47.9212	0.31	46.6148	57.50	0.2343
1000	31958	0.0636	63.9160	0.0000	1.0194	4.8447	0.0350	48.7855	0.31	48.4759	58.06	0.2500
1000	33039	0.0660	66.0780	0.0000	1.0128	4.8278	0.0344	51.8778	0.31	50.3260	62.19	0.2656
1000	34153	0.0685	68.3060	0.0000	1.0109	5.0357	0.0353	52.7725	0.31	51.7958	66.95	0.2656
1000	35487	0.0710	70.9740	0.0000	1.0004	5.1247	0.0343	54.2800	0.31	53.2105	64.05	0.2500
1000	36823	0.0735	73.6460	0.0000	0.9960	5.2832	0.0342	56.5714	0.31	56.8581	66.54	0.2656
1000	38148	0.0760	76.2960	0.0000	0.9912	5.4894	0.0335	59.7882	0.31	57.5163	70.91	0.2500
1000	39284	0.0785	78.5680	0.0000	0.9891	5.6036	0.0331	63.0853	0.31	58.9005	72.58	0.2656
1000	40149	0.0810	80.2980	0.0000	0.9887	5.7340	0.0323	62.8253	0.31	60.3376	76.67	0.2812
1000	42128	0.0834	84.2560	0.0000	0.9748	6.0322	0.0329	65.7725	0.31	62.9272	76.64	0.2826
1000	43150	0.0859	86.3000	0.0000	0.9690	6.2352	0.0328	67.6640	0.31	65.9251	78.54	0.2812
1000	44155	0.0884	88.3100	0.0156	0.9779	6.1691	0.0325	71.2186	0.31	65.9506	83.10	0.3437
1000	45331	0.0909	90.6620	0.0156	0.9703	6.3590	0.0323	71.2624	0.31	70.3192	86.25	0.2812
1000	47035	0.0934	94.0700	0.0000	0.9720	6.6558	0.0323	75.1978	0.31	70.4701	90.02	0.2969
1000	48066	0.0959	96.1320	0.0000	0.9637	6.7339	0.0323	75.8914	0.31	72.9875	88.38	0.2969
1000	48998	0.0984	97.9960	0.0000	0.9620	6.8121	0.0318	77.1207	0.31	72.9139	94.89	0.2982
1000	50750	0.1009	101.5000	0.0000	0.9577	7.0933	0.0317	80.8509	0.31	76.7141	93.52	0.3281
1000	51523	0.1033	103.0460	0.0000	0.9606	7.2363	0.0314	85.4124	0.31	76.3831	101.03	0.3281
1000	52906	0.1058	105.8120	0.0000	0.9539	7.3459	0.0313	85.3046	0.31	77.9159	101.46	0.3125
1000	54302	0.1083	108.6040	0.0000	0.9504	7.3745	0.0314	87.6339	0.31	80.0538	96.82	0.3437
1000	55247	0.1108	110.4940	0.0000	0.9500	7.5464	0.0315	87.8853	0.31	81.6733	103.39	0.3442
1000	56431	0.1133	112.8620	0.0000	0.9468	7.9057	0.0308	90.9838	0.31	86.1031	103.83	0.3281
1000	57611	0.1158	115.2220	0.0000	0.9440	8.0009	0.0313	92.4521	0.31	85.3300	103.54	0.4375
1000	58817	0.1183	117.6340	0.0000	0.9426	8.0022	0.0310	94.6731	0.31	87.0751	107.13	0.3437
1000	60110	0.1207	120.2200	0.0000	0.9404	8.2338	0.0309	99.1379	0.31	90.3286	112.37	0.3593
1000	61663	0.1232	123.3260	0.0000	0.9361	8.2206	0.0307	99.4845	0.31	91.0006	111.02	0.3593
1000	62864	0.1257	125.7280	0.0000	0.9359	8.2831	0.0309	99.5018	0.31	94.1533	117.37	0.4218
1000	64129	0.1282	128.2580	0.0000	0.9343	8.6424	0.0308	105.1412	0.31	94.5517	118.55	0.4375
1000	65122	0.1307	130.2440	0.0000	0.9336	8.6836	0.0305	107.6357	0.31	95.2904	120.18	0.4218
1000	66527	0.1332	133.0540	0.0000	0.9330	8.8612	0.0304	107.6874	0.31	97.8937	121.34	0.4009
1000	67900	0.1357	135.8000	0.0000	0.9251	9.0165	0.0304	109.3971	0.31	101.6253	118.43	0.3796

B.2 Open-source (di)graphs

Table 19: Values and computation times of complexity measures performed on Open-Source Graphs

Open-Source Graphs

network	n	m	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sea}	t (sec)
jazz[4]	198	2742	27.6970	0.0156	2.2350	5.5616	0.9989	58.6558
euroroad[3]	1039	1305	2.5120	0.0313	18.3951	28.2511	0.9863	487.9313
powergrid[6]	4941	6594	2.6691	0.1094	18.9892	467.2417	0.9979	145400.7716
pgp[5]	10680	24316	4.5536	0.2812	7.4855	1893.3436	NaN	NaN
astroph[1]	17903	196972	22.0044	1.1718	4.1940	17948.8601	NaN	NaN
caida[2]	26475	53381	4.0326	0.6802	3.8756	10554.6425	NaN	NaN

Table 20: Values and computation times of complexity measures performed on Open-Source DiGraphs

Open-Source DiGraphs

network	n	m	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sym}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)
moreno[9]	217	2672	24.6267	0.0000	2.7269	1.4062	0.3765	0.0312	0.0017	4.4997	0.31	3.8591
usairport[10]	1572	28235	35.9224	0.0312	2.8669	97.2368	0.2193	0.2656	0.0065	454.5889	0.67	685.2410
openflights[8]	2905	30442	20.9583	0.0312	4.0924	237.3847	0.0279	0.3150	0.0008	2006.0641	0.34	105.7157
bitcoin[29][28]	5875	35587	12.1147	0.0625	2.9834	562.5677	0.2077	0.3102	NaN	NaN	1.00	2441.6999
p2p[32][35]	10876	39994	7.3545	0.0781	2.6936	699.6089	1.0000	0.4375	NaN	NaN	1.00	4216.4074
googleplus[7]	23613	39230	3.3227	0.1562	0.0078	11.5652	0.9976	0.5625	NaN	NaN	1.00	8975.7015
p2p[32][35]	36646	88303	4.8192	0.2969	1.8797	3825.4008	1.0000	1.0937	NaN	NaN	1.00	11599.7948

B.3 Statistics Netherlands (di)graphs

Table 21: Values and computation times of complexity measures performed on a large company digraph provided by Statistics Netherlands

Companies Network

n	m	$\mathbf{m_{sample}}$	C_{deg}	t (sec)	C_{dis}	t (sec)	C_{sym}	t (sec)	C_{ent}	t (sec)	C_{pro}	t (sec)
32	31	10000	1.9375	0.0000	0.0312	0.0030	1.0000	0.0000	0.0092	0.0120	0.48	0.6930
70	69	20000	1.9714	0.0000	0.0151	0.0030	1.0000	0.0030	0.0260	0.0120	1.00	7.1401
109	108	30000	1.9817	0.0000	0.0125	0.0030	1.0000	0.0000	0.0499	0.0120	0.75	6.3633
322	321	40000	1.9938	0.0000	0.0066	0.0030	1.0000	0.0000	0.0769	0.0150	0.77	21.4771
5154	5157	50000	2.0012	0.0000	0.0005	0.0690	1.0000	0.0090	0.0582	0.3210	0.49	250.5047
15399	15531	60000	2.0171	0.0030	0.0002	0.2310	0.9999	0.0360	0.0556	0.7860	0.37	314.4845
22710	23155	70000	2.0392	0.0061	0.0001	0.4036	0.9998	0.0530	0.0569	1.3136	0.31	109.8377
30078	31078	80000	2.0665	0.0086	0.0001	0.5232	0.9999	0.1087	0.0591	2.2110	1.00	5798.1626
37728	39536	90000	2.0958	0.0119	0.0001	0.8130	0.9998	0.1499	0.0607	3.0990	1.00	7940.2146
45408	48178	100000	2.1220	0.0150	0.0001	0.8010	0.9999	0.1230	0.0627	2.7240	1.00	8288.3652
82309	93202	150000	2.2647	0.0240	0.0001	1.7880	0.9998	0.2490	0.0763	6.4950	NaN	NaN
117828	141158	200000	2.3960	0.0300	0.0002	3.7410	0.9998	0.3750	0.0964	13.4010	NaN	NaN
151485	190474	250000	2.5148	0.0420	0.0008	10.1490	0.9997	0.4560	0.1396	37.5452	NaN	NaN
183474	240929	300000	2.6263	0.0630	0.0078	67.6110	0.9997	0.8160	0.1670	336.7430	NaN	NaN
213571	292113	350000	2.7355	0.0604	0.0530	460.9782	0.9996	0.7459	NaN	NaN	NaN	NaN
241599	343677	400000	2.8450	0.0758	0.2595	2833.1184	0.9996	0.9460	NaN	NaN	NaN	NaN
268026	395247	450000	2.9493	0.1255	0.5230	6808.8575	0.9996	1.5252	NaN	NaN	NaN	NaN
293039	446950	500000	3.0504	0.0749	0.8376	13423.6949	0.9996	1.0212	NaN	NaN	NaN	NaN
478974	965380	1000000	4.0310	0.1165	NaN	NaN	0.9995	2.0254	NaN	NaN	NaN	NaN
707310	2487708	2500000	7.0343	0.3339	NaN	NaN	0.9992	6.9335	NaN	NaN	NaN	NaN
858488	9998657	10000000	23.2936	0.3540	NaN	NaN	0.9979	24.7680	NaN	NaN	NaN	NaN
872281	20000302	20000000	45.8575	0.3500	NaN	NaN	0.9963	85.2316	NaN	NaN	NaN	NaN
873667	24997546	25000000	57.2244	0.4912	NaN	NaN	0.9955	115.9346	NaN	NaN	NaN	NaN

C Appendix - Packages

C.1 Packages Versions

In order to make use of the implemented techniques the version of Python has to be 3.7.1 or higher. Setups with Python versions below 3.7.1 were not tested.

The following Python libraries were used with their recommended versions:

1. Networkx 2.3 [23]
2. Numpy 1.17.2 [34]
3. Scipy 1.3.1 [37]

Though not necessary for making use of the complexity measures, the following libraries are useful for plotting and creating tables:

1. Pandas 0.25.1 [33]
2. Matplotlib 3.1.1 [26]

C.2 Implementations availability

The Python implementations of the complexity measures and additional (unused) functions are made available and can be found here: https://github.com/simonvw95/NetworkX-Complexity-Measures-Additions/tree/master/SVW_CBS_pkg

Instructions on how to install the Python package as well as a documentation of all the available functions are included.