Knowledge Extraction from Wikipedia Using Complex Network Analysis

Matas, Neven

UNIVERSITY OF RIJEKA
FACULTY OF HUMANITIES AND SOCIAL SCIENCES

Master's thesis

KNOWLEDGE EXTRACTION FROM WIKIPEDIA USING COMPLEX NETWORK ANALYSIS

Neven Matas

Rijeka, September 2015

UNIVERSITY OF RIJEKA
FACULTY OF HUMANITIES AND SOCIAL SCIENCES

Master's thesis

KNOWLEDGE EXTRACTION FROM WIKIPEDIA USING COMPLEX NETWORK ANALYSIS

Student ID number: 0009059992
Student: Neven Matas
Program: Informatics and English language education
Mentor: izv. prof. dr. sc. Ana Meštrović

Rijeka, September 2015

ODJEL ZA INFORMATIKU
SVEUČILIŠTE U RIJECI

Rijeka, 07.09.2015.

Zadatak za diplomski rad

Pristupnik: Neven Matas

Naziv diplomskog rada: Ekstrakcija znanja iz Wikipedie analizom kompleksnih mreža

Naziv diplomskog rada na eng. jeziku: Knowledge extraction from Wikipedia using complex networks analysis

Sadržaj zadatka: Korištenje metoda za analizu kompleksnih mreža poput pronalaska zajednica i analize centralnosti da bi se analizirala Wikipedia i kroz taj process pokazalo da je moguće ekstrahirati novonastalo znanje o njoj i konceptima prisutnima na istoj.

Mentor:

izv. prof. dr. sc. Ana Meštrović

_____

Voditelj za diplomske radove:

izv. prof. dr. sc. Ana Meštrović

_____

Zadatak preuzet: 07.09.2015.

_____

(potpis pristupnika)

I. Summary

Abstract:

In this paper I describe a complex network analysis of Wikipedia. I conduct two experiments.

In the first 10 different networks are constructed from Wikipedia entries related to the chosen domain. The goal of the first experiment is to extract domain knowledge in terms of identifying entries that are centrally positioned and entries that are strongly related. I apply complex networks analysis on all acquired networks and examine the networks' structure. Additionally, centrality measures are employed in order to find centrally positioned entries in the network. And finally, I identify communities and find which entries are densely connected according to the network structure.

In the second experiment I construct two different types of networks – superset and subset networks in which nodes are connected amongst themselves on the basis of keyword similarities between the texts of the respective entries representing nodes. These networks are then compared with the original network, of the type used in the first experiment, and through this process centrality measures are reviewed according to their ability to extract concepts semantically related to the text. I also present and analyze the extracted keywords used in the experiment.

Keywords: complex networks, Wikipedia, community detection, knowledge extraction, centrality measures, keyword extraction

II. Contents

1. Introduction

1.1. Graphs and networks

The essential component of network science is a mathematical concept which we call a graph. In order to delve further into networks we must first acquaint ourselves with some fundamentals of graph theory. A graph, generally speaking, is represented as a series of objects connected together by links. These objects are usually called vertices (also nodes or points) and they are interconnected with edges (also links or lines). A simple graph is illustrated with Figure 1.

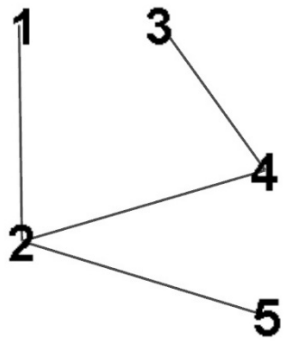


Figure 1 - A simple graph

Formally, according to [1], a graph *G* is a set consisting of two things, a non-empty finite set of vertices V and its (possibly empty) subset *E* which consists of edges. It is typically written as such:

$$G = \{V, E\}.$$

The graph *G* on Figure 1 could be formally described in the following matter:

$$V = \{1, 2, 3, 4, 5\},$$

$$E = \big\{\{1, 2\}, \{2, 4\}, \{2, 5\}, \{3,4\}\big\}.$$

V consists of all unique vertices in the set, while *E* consists of pairs (sets) of vertices indicating that these vertices are connected, i.e. an edge exists between them.

We can also define graphs in terms of an adjacency matrix, immediate adjacency being a property of vertices between which an edge is present. The following is an $n \times n$ adjacency matrix $A = [a_{ij}]$ describing the above illustrated graph G, each row i and column j being assigned to a unique vertex from 1 to 5. The value of each element in the matrix represents how many edges exist between the two vertices. In the case where multiple connections between vertices are disallowed the matrix can only contain 0s or 1s (0 representing no edge between vertices and 1 representing the existence of a connection).

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Another way to describe graphs using matrix algebra is via incidence matrices which shows which edges connect two vertices, but we will not look at them at this point.

If there is only one edge connecting two vertices the graph in question is a simple graph. In the case when there are multiple connections between two edges the graph is appropriately named a multigraph. An edge which points back to the same vertex it originated from represents a graph loop. Graphs that can contain both multiple edges and graph loops are called pseudographs [2].

In the case of undirected graphs, an edge described by {1, 3} is equivalent to {3, 1} since we regard the directedness of the edge as irrelevant. When directed graphs (digraphs) are concerned, each edge can have one or two directions going from one vertex to another. If the edges are exclusively monodirectional then we are dealing with oriented graphs. A graph containing both directed and undirected edges is called mixed [3].

Both edges and vertices can be assigned specific values, colors or names [3]. A graph carrying these properties is called a labeled graph. If a numerical value is assigned to a graph's edges we call it a weighted graph. Finally, this allows us to arrive at one specific definition of a network.

We may say that a network is a directed edge-labeled graph with edge labels being numerical values usually assigned through some kind of mathematical function [2]. We must bear in mind that there are multiple suitable definitions for a network, and, contrary to the aforementioned,

I will also be referring to unlabeled and undirected graphs as networks in order to preserve simplicity and avoid confusing terminology[1].

When we think of networks we usually focus on representing some real-world relationships. In [4], Newman notes how *many objects of interest in the physical, biological, and social sciences can be thought of as networks*. Table 1 [4] looks at a few examples of networks and how their constituent parts may model the real world.

Table 1 - Examples of real-world network models

Network	Vertex	Edge
Internet	Computer or router	Data connection
World Wide Web	Web page	Hyperlink
Citation network	Article, patent or legal case	Citation
Power grid	Generating station	Transmission line
Friendship network	Person	Friendship
Metabolic network	Metabolite	Metabolic reaction
Neural network	Neuron	Synapse
Food web	Species	Predation

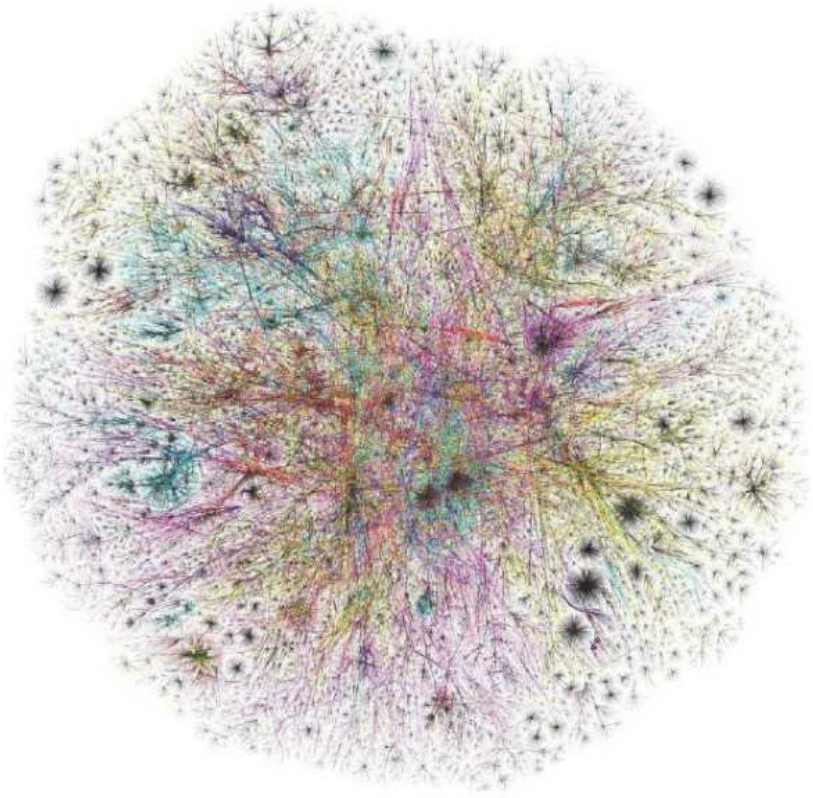


Figure 2 - The network structure of the Internet [4]

[1] In [5] Barabasí notes that the distinction has to do with the fact that graphs are usually purely mathematical entities and networks are their real-world application.

Upon the construction of our network of choice we are able to analyze it utilizing various methods and metrics in order to extrapolate information pertinent to the network which is not immediately observable through its mere visualization. For instance, we may analyze a computer network in order to deduce how tolerant it is to attacks and will the vulnerability of certain nodes result in loss of data flow. Another example is analyzing social networks in order to reason about the influence of certain agents. For a final out of many possible uses of network analyses, I would like to point out a recent breakthrough approach in finding the origin of an epidemic within a network of disease outbreak [6]. These and many other examples enable us to conclude that using networks as models and employing network analysis can produce significant results with real-world consequences.

Since we now have enough introductory information to describe and understand the main principles behind networks, we may move onto the topic of complex networks.

1.2. Complex networks

When we talk about real-world networks we often talk about complex networks. Complex networks differ from regular or random networks[2] in that they exhibit some specific features such as community structure, giant components, hierarchical structure, power law degree distributions, short average path lengths and high clustering coefficients. We'll look at all of these features in detail later in section 2.

Two main classes of complex networks are small-world and scale-free networks.

Small-world networks, as described by Duncan Watts and Steven Strogatz [7], have a defining characteristic of a small average path length, i.e. the distance between two nodes which are not neighbors is proportional to the logarithm of the number of nodes in the network:

$$L \propto \log N.$$

This network model echoes Stanley Milgram's famous small-world experiment wherein he found that any two people in the United States are averagely separated by as much as six steps (*"six degrees of separation*") or six jumps between connected vertices representing people. Some other real-world occurrences of the small world phenomenon as mentioned by Watts and

[2] Referring to the Erdős-Rényi model for generating random graphs.

Strogatz is the neural network of the *C. elegans* roundworm and the layouts of electrical grids. Other examples include the Internet, brain networks, word adjacency networks, etc. [8]. Along with the short average path length property *L*, small-world networks also exhibit a high clustering coefficient *C* – tightly interconnected clusters of nodes. Figure 3 shows the differences in structure between regular, small world and randomly connected networks according to these two criteria [7].

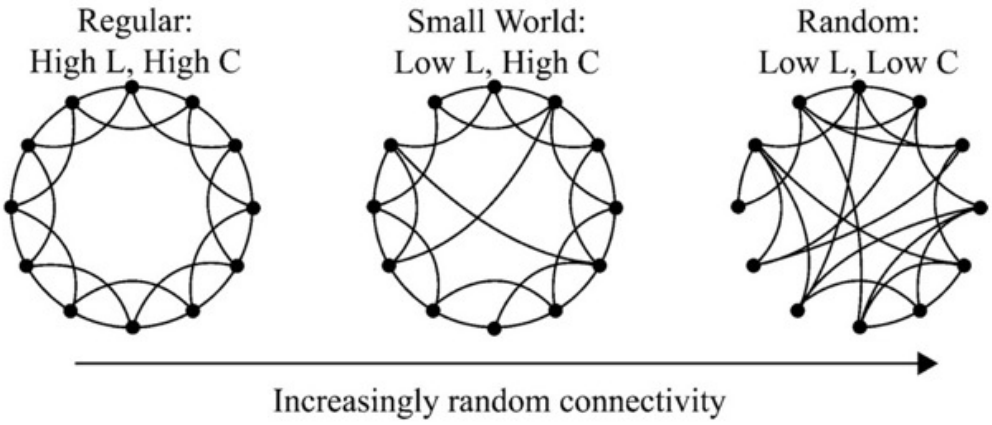


Figure 3 - Regular, small world and random networks

Another major class of complex networks are scale-free networks, described by Barabasí and Albert in 1999 [9] by inspecting the structure of the World Wide Web network. They found that this network had several hubs with a much larger number of connections than the average node. Scale-free networks exhibit a power-law degree distribution, i.e. the relationship between number of nodes and their respective degrees[3] tends to be inversely proportional (see Figure 4) and power-law approximate. One of the explanations for why some networks exhibit scale-free characteristics is preferential attachment which is the basis for the Barabási-Albert model for the generation of scale-free networks. Preferential attachment, also called cumulative advantage [4], theorizes that there is a tendency for strongly connected vertices to gain even more connections. For instance, in a citation network, the probability that a new randomly chosen paper will cite a previously existing one *is proportional to the number of citations those previous papers already have* [4]. This can explain how densely connected nodes get even

[3] The degree of a node is its number of connections with other nodes.

more densely connected, according to the *rich-get-richer* principle[4]. The especially well connected vertices are often called hubs.

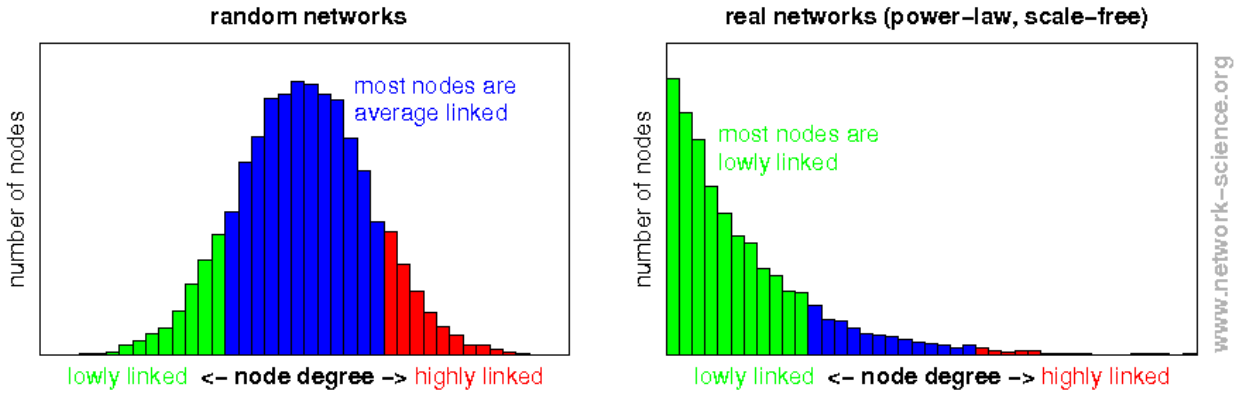


Figure 4 - Degree distributions in random and scale-free networks [10]

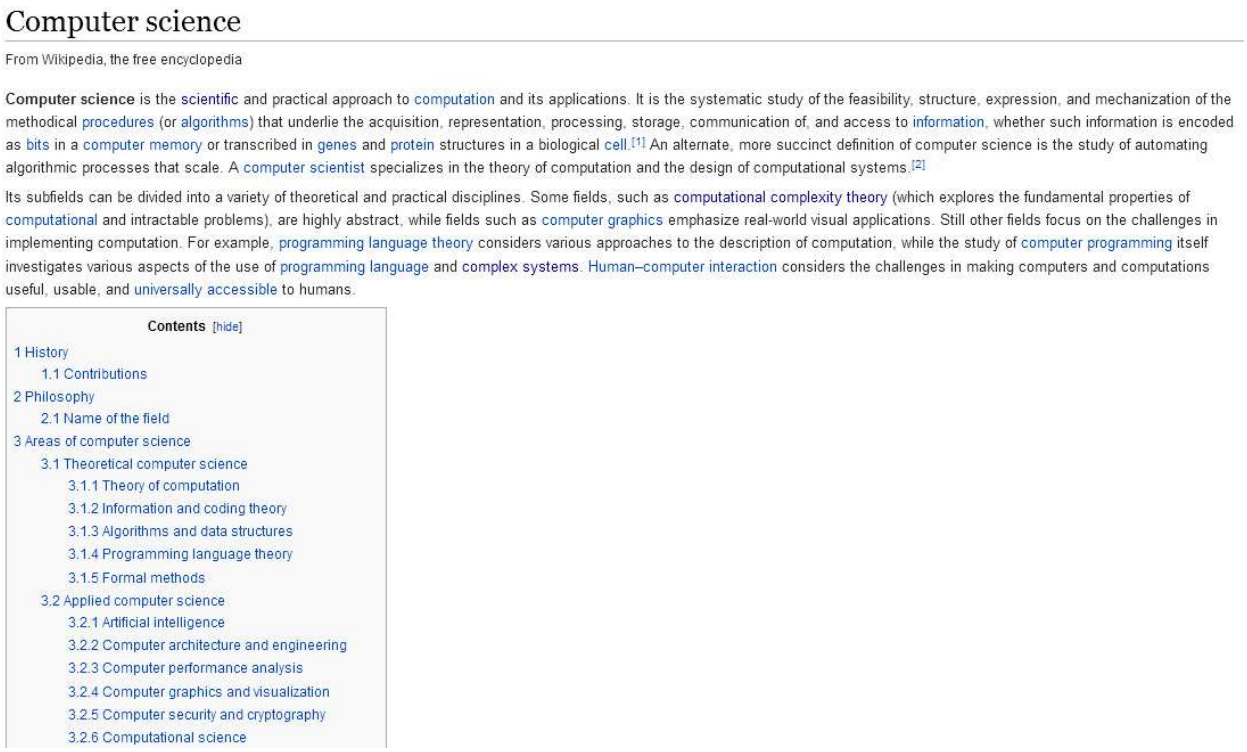
This concludes a very general overview of what complex networks are and what crucially distinguishes them from other types of networks. Since here we apply complex network methods to Wikipedia, the following section will briefly look at what makes it an interesting area of research.

1.3. Wikipedia as a complex network

Wikipedia is a free, online, collaborative general knowledge encyclopedia. It was launched in 2001 and is currently available in 288 different languages. It is among the 10 most popular websites in the world and its English language variety includes over 4.9 million unique articles (the entire encyclopedia offering 35,689,754 articles on August 4, 2015). [11]. As such, Wikipedia is one of the largest open access compendiums of human knowledge which is updated daily by a workforce of over 118,788 regular volunteer editors.

As far as the validity and quality of Wikipedia articles are concerned, a 2005 study published by Nature [12] showed that Wikipedia averages 3.86 errors per article, contrasted with the 2.92 errors per article average of the de facto standard which is Encyclopedia Britannica, thus cementing its status as a valuable resource.

[4] Also known as the Matthew effect as described by sociologist Robert K. Merton.



Computer science

From Wikipedia, the free encyclopedia

Computer science is the scientific and practical approach to computation and its applications. It is the systematic study of the feasibility, structure, expression, and mechanization of the methodical procedures (or algorithms) that underlie the acquisition, representation, processing, storage, communication of, and access to information, whether such information is encoded as bits in a computer memory or transcribed in genes and protein structures in a biological cell.[1] An alternate, more succinct definition of computer science is the study of automating algorithmic processes that scale. A computer scientist specializes in the theory of computation and the design of computational systems.[2]

Its subfields can be divided into a variety of theoretical and practical disciplines. Some fields, such as computational complexity theory (which explores the fundamental properties of computational and intractable problems), are highly abstract, while fields such as computer graphics emphasize real-world visual applications. Still other fields focus on the challenges in implementing computation. For example, programming language theory considers various approaches to the description of computation, while the study of computer programming itself investigates various aspects of the use of programming language and complex systems. Human–computer interaction considers the challenges in making computers and computations useful, usable, and universally accessible to humans.

Contents [hide]

1 History
1.1 Contributions
2 Philosophy
2.1 Name of the field
3 Areas of computer science
3.1 Theoretical computer science
3.1.1 Theory of computation
3.1.2 Information and coding theory
3.1.3 Algorithms and data structures
3.1.4 Programming language theory
3.1.5 Formal methods
3.2 Applied computer science
3.2.1 Artificial intelligence
3.2.2 Computer architecture and engineering
3.2.3 Computer performance analysis
3.2.4 Computer graphics and visualization
3.2.5 Computer security and cryptography
3.2.6 Computational science

Figure 5 - Example of a Wikipedia article

Figure 5 shows a typical Wikipedia article. The entry in question is "*computer science*". As is typical for WWW documents, it is a hypertext wherein normal text is interspersed with hyperlinks pointing towards other related Wikipedia articles. Since an encyclopedia of this type strives to have its articles mutually well connected in order to facilitate the traversal of relevant topics, the number of hyperlinks is usually rather high (see Figure 5).

This connectedness of Wikipedia articles is the most basic principle following which I constructed complex networks from its structure. The model to construct a network relied on taking a starting entry as a seed vertex and then building edges according to the appearance of hyperlinks, each new hyperlinked entry being as a new vertex within the network (See Figure 6).

Figure 6 - One edge of a Wikipedia network

Having a methodology for constructing networks out of knowledge embedded in Wikipedia's articles, I was able to extrapolate new knowledge pertaining to the chosen networks of concepts and Wikipedia at large. The detailed methodology behind the experiments conducted on Wikipedia will be given later, but first we must get back to complex networks and explain some basic network properties and what measures we may use in order to analyze them.

2. Complex network analysis

2.1. Complex network measures

The following is a list of network measures relevant in my analysis of Wikipedia networks. All of these help us ascribe certain numerical properties related to the structure of the network, i.e. how big it is, how well connected it is, what nodes play important roles in relation to others, are there any clusters of nodes and how densely are they connected, etc.

- Number of nodes
- Number of edges
- Average degree
- Average shortest path length
- Diameter
- Average clustering coefficient
- Density
- Modularity
- Degree assortativity coefficient
- Centralities[5]:
    - Degree centrality
    - Betweenness centrality
    - Closeness centrality
    - Eigenvector centrality
    - Current-flow betweenness centrality (random flow betweenness centrality)
    - Current-flow closeness centrality (information centrality)
    - Communicability centrality

[5] Centralities usually point towards the most prominent (central) nodes in a network according to some principle.

2.1.1. Number of nodes (*N*)

This measure simply specifies the total sum number of nodes present in a network, usually given as *N*. It corresponds to the cardinality of the set *V*, the set of all nodes present in a graph. If we look back at Figure 1, we can deduce its number of nodes *N* is equal to 5.

2.1.2. Number of edges (*K*)

Equivalently to the previous measure, number of edges, usually given as *K*, corresponds to the total sum number of edges present in a network. It corresponds to the cardinality of the set E which contains two-member subsets of *V*, representing edges. The graph visualized on Figure 1 has a *K* equal to 4. We may even check if this is so by counting the number of edges on the resulting graph.

2.1.3. Average degree (<*k*>)

The degree of a node *i* is the number of links with which it is connected, specified as k_i. When we are working with directed networks, we specify two types of degrees: the in-degree, k_i^{in}, corresponding to the number of incoming links and the out-degree, k_i^{out}, equal to the number of outgoing links for any particular node *i*. The average degree $<k>$ is an average degree of all nodes in a network. For undirected networks, it is given as:

$$< k > = \frac{2K}{N}.$$

And in the case of directed networks we do not have to multiply the number of edges by two (since we have both edges that go in and edges that go out of a certain node), so we arrive at the following, slightly modified equation:

$$< k > = \frac{K}{N}.$$

2.1.4. Average shortest path length (*L*)

The shortest path length[6] measure represents the minimum number of node-to-node jumps (via incident edges) one must make in order to arrive at another node in the network. In the case of weighted networks, the shortest path is the one for which the sum of the traversed

[6] Shortest path lengths are also referred to as graph geodesics.

edges is the smallest. Finding a shortest path in a network is not a trivial problem[7], and many algorithms were constructed in order to tackle it (Dijkstra's algorithm being the most well-known). It is related to both the travelling salesman problem and mathematical problem originally introduced by the mathematician Leonhard Euler dubbed the *Seven Bridges of Königsberg*. Both problems can be reduced to finding shortest paths in graphs. The constraint in the first problem are passing through each city only once and returning to the origin. In the second one it is crossing each bridge only once whilst visiting all the islands. Figure 7 pictures Euler's problem:

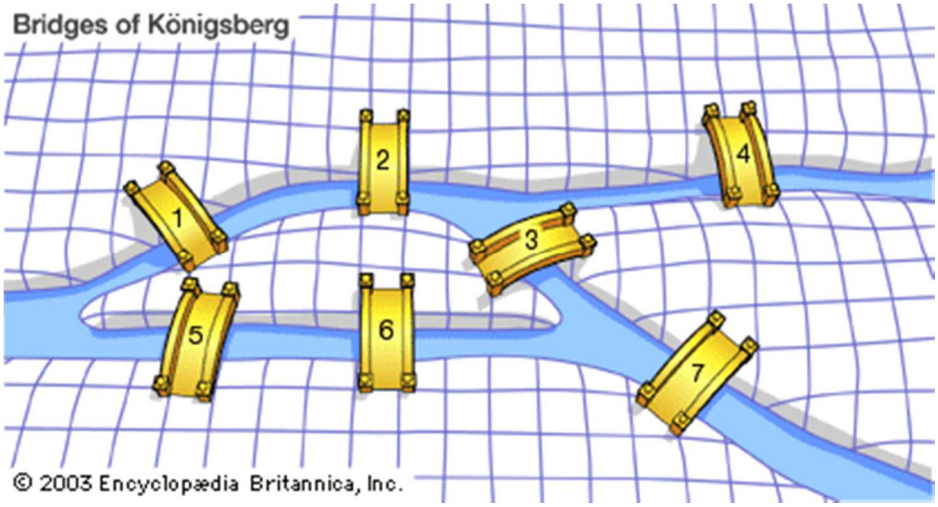


Figure 7 - Seven Bridges of Königsberg [13]

A relatively small average shortest path length (also referred to just as *average path length*) for an entire network is indicative of the mentioned small-world effect, making it possible to traverse wide regions of the network with a small number of jumps. In order to give an equation for the average shortest path length we must first look at the average distance of a single node from all other nodes, and it is given as:

$$d_i = \frac{\sum_j d_{ij}}{N}.$$

The total possible number of links in a directed network is also relevant and given as:

$$N = N(N-1).$$

[7] It belongs to the NP-complete complexity class for algorithms, meaning its solution cannot always be found in non-polynomial time.

Finally, we can combine everything into our equation for the average shortest path length L for an entire given network:

$$L = \frac{1}{N(N-1)} \sum_{i \neq j} d_{ij} .$$

2.1.5. Diameter (*D*)

The diameter (also called the *longest graph geodesic*) of a network is the biggest shortest path length in a network, i.e. highest number of jumps one must make in order to arrive from one node to another. It is given as $D = max_i d_i$. Figure 8 shows the diameter (which is equal to 5) of the famous Zachary Karate Club network[8]:

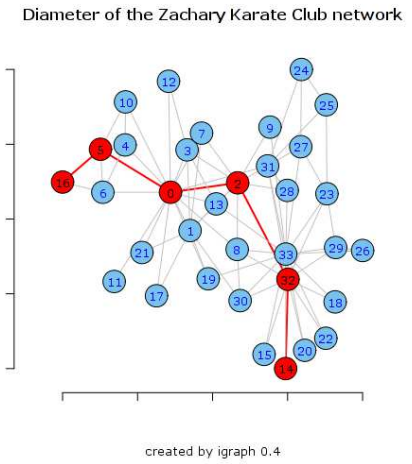


Figure 8 - Diameter of the Zachary's Karate Club network [15]

2.1.6. Average clustering coefficient (*C*)

According to [4], the clustering coefficient *measures the average probability that two neighbors of a vertex are themselves neighbors*, effectively measuring the density of triangles (three interconnected vertices) in a network. We can differentiate between the local clustering coefficient pertaining to a single node and a global average cluster coefficient which covers the entire network.

The local clustering coefficient c_i (for an undirected graph) of a node k_i is a fraction of the number of existing edges E_i between k_i and its adjacent nodes and the total possible number of edges:

[8] First used and analysed by W. W. Zachary in [14].

$$c_i = \frac{2E_i}{k(k-1)}.$$

The average (global) clustering coefficient of a network is defined as the average value of the clustering coefficients of all nodes in a network:

$$C = \frac{1}{N}\sum_i c_i\,.$$

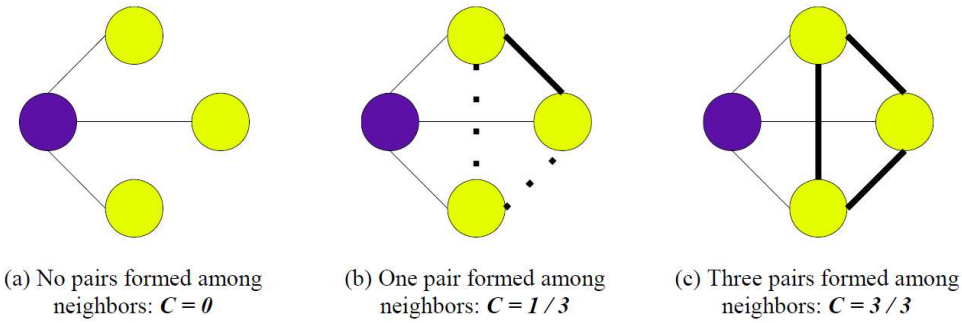


Figure 9 - Global clustering coefficients for a small network [16]

2.1.7. Density (*d*)

The density of a network is a measure of network cohesion defined as the number of observed edges (*K*) divided by the number of total possible links. It is represented by the following formula:

$$d = \frac{K}{N(N-1)}.$$

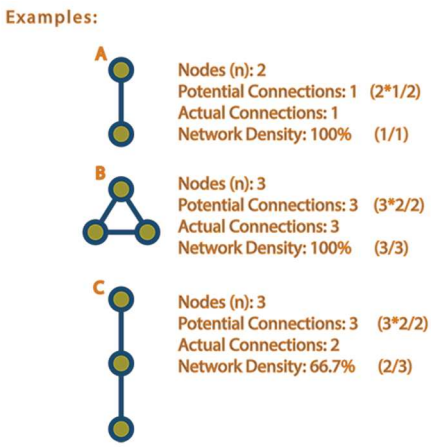


Figure 10 - Network density explained [17]

Figure 10 shows a simple visualization of network density and the difference between realized and potential edges.

2.1.8. Modularity (*Q*)

The measure of modularity has to do with the extent to which similar groups of nodes (clusters or communities or *modules*) are interconnected [4]. We'll look at community structure in more detail in the section on complex network structure, but as a working definition we can state that communities are densely interconnected groups of nodes which have fewer connections to the rest of the network (or towards other communities) than they exhibit internally. In [18], Newman states how modularity represents *the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random.* As a measure of community structure, Newman also notes that it *can be either positive or negative, with positive values indicating the possible presence of community structure.*

So modularity measures the quality of network partitioning into communities. The modularity of a network partition is a scalar value between -1 and 1 that measures the density of links inside communities as compared to links between communities. Let e_{ij} be the fraction of edges in the network that connect vertices in group i to those in group j, and let $a_i = \sum_j e_{ij}$. Then modularity can be calculated using following equation:

$$Q = \sum_{i=1}^{k} (eii - a_i^2).$$

Figure 11 shows both global and local modularity within a biological network.

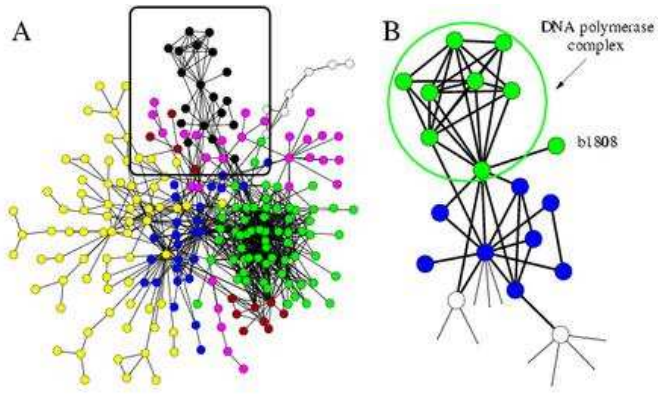


Figure 11 - Modularity in biological networks [19]

2.1.9. Degree assortativity coefficient (*r*)

The degree assortativity coefficient measures the tendency of nodes in a network to connect to nodes similar to themselves. The coefficient lies in the $[-1,1]$ interval and is quantified via the Pearson correlation coefficient[9]. Assortative mixing, quantified by the assortativity coefficient, is observed in nodes which preferentially attach to nodes of a similar degree. The obverse to assortative mixing is disassortative mixing, where, for instance, high degree nodes may tend to connect to nodes of a lower degree.

Positive r values indicate a correlation between similar-degree nodes. Let q_k and q_j be the distribution of the degree of out-edges that do not connect to the other node in question, e_{jk} the joint probability distribution of q_k and q_j, and σ_q^2 the variance of the distribution. Then we can calculate the assortativity coefficient using the following equation:

$$r = \frac{\sum_{jk} jk(e_{jk} - q_j q_k)}{\sigma_q^2}.$$

2.1.10. Degree centrality (*dc*)

Centralities in general try to discover the most important nodes in a network. They are often applied in social networks to extrapolate key actors, but often prove useful in all manner of different networks. There are several distinct varieties, the simplest of which is degree centrality.

The degree centrality of a node is determined according to its degree (in- and out-degree in the case of directed networks) – the number of nodes with which it is connected. When normalized by dividing it by the maximum possible degree N-1 we get the following equation:

$$dc_i = \frac{k_i}{N-1}.$$

Figure 12 illustrates degree centrality on sample friendship network where the node labelled "Diana" clearly exhibits the highest degree centrality (being connected to all other nodes in the network).

[9] The Pearson correlation coefficient is a measure of the strength of the linear relationship between two variables.

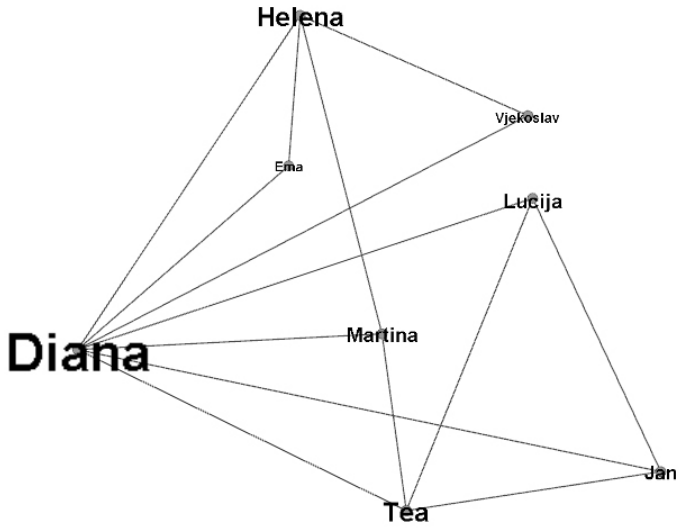


Figure 12 - An illustration of degree centrality

2.1.11. Betweenness centrality (*bc*)

Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes, i.e. it is the measure of how many time it is found on the network's geodesics. According to [4], *vertices with high betweenness centrality may have considerable influence within a network by virtue of their control over information passing between others.* He also notes that *it differs from the other centrality measures … in being not principally a measure of how well-connected a vertex is. Instead it measures how much a vertex falls "between" others.* Figure 13 clearly illustrates an example of betweenness centrality, where node *C* lies on both shortest paths between *A* and *B* – thus being most centrally positioned.

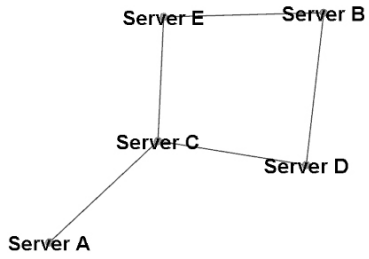


Figure 13 - An illustration of betweenness centrality[10]

[10] Modified example from [4].

Let σ_{jk} be the number of shortest paths from node j to node k and let $\sigma_{jk}(i)$ be the number of those paths that pass through the node i. The normalised betweenness centrality of a node i is then given as:

$$bc_i = \frac{\sum_{i \neq j j \neq k} \frac{\sigma_{jk}(i)}{\sigma_{jk}}}{(N-1)(N-2)}.$$

2.1.12. Closeness centrality (*cc*)

As per [4], closeness centrality is defined as *the mean geodesic distance from a vertex to all other reachable vertices*. In other words, it is the inverse of farness, i.e. the sum of the shortest distances between a node and all other nodes. Let d_{ij} be the shortest path between nodes i and j. The normalized closeness centrality of a node i is then given by:

$$cc_i = \frac{N-1}{\sum_{i \neq j} d_{ij}}.$$

So the *closer* a node is, the lesser is its distance to all other nodes in a network. The following figure place the three centralities we have mentioned so far into context, comparing them in order to make it easier to appreciate the difference in a node's influence given by degree centrality, betweenness centrality and closeness centrality.

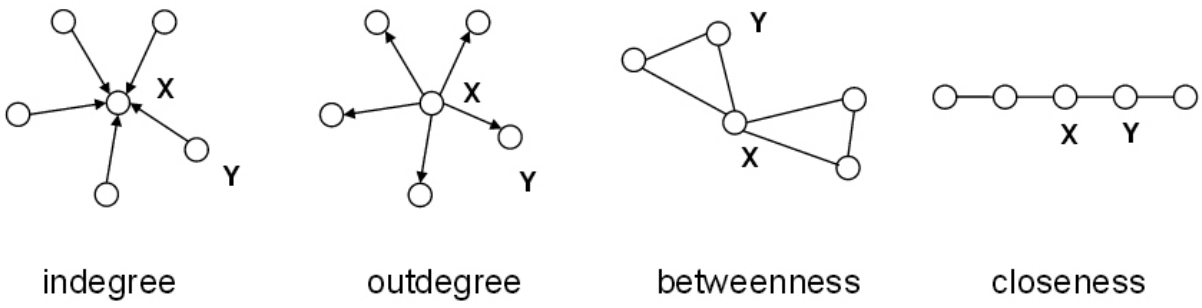


Figure 14 - A simple comparison of dc, bc and cc [20]

2.1.13. Eigenvector centrality (*ec*)

Eigenvector centrality[11] can be thought of as an improvement upon standard degree centrality. Degree centrality measures only the amount of connections a node has but disregards towards which nodes these connections are established. Eigenvector centrality modifies this approach by gives a higher centrality score those connections which are made towards those nodes which are themselves central. Thus, it measures influence within a network.

So, according to [4], a node's eigenvector centrality has *the nice property that it can be large either because it has many neighbors or because it has important neighbors (or both).* Also, according to [4], the centrality x_i of vertex i is proportional to the sum of the centralities of i's neighbors. So the equation for calculating eigenvector centrality can be written down as the following:

$$x_i = k_1^{-1} \sum_j A_{ij} x_j .$$

Finally, Figure 15 compares the previous four centralities on the same network. The hotter (orange and red colored) nodes or regions signify higher observed centrality. In contrast, the colder (dark and light blue) regions signify nodes or cluster with lower centrality within the scope of the network.

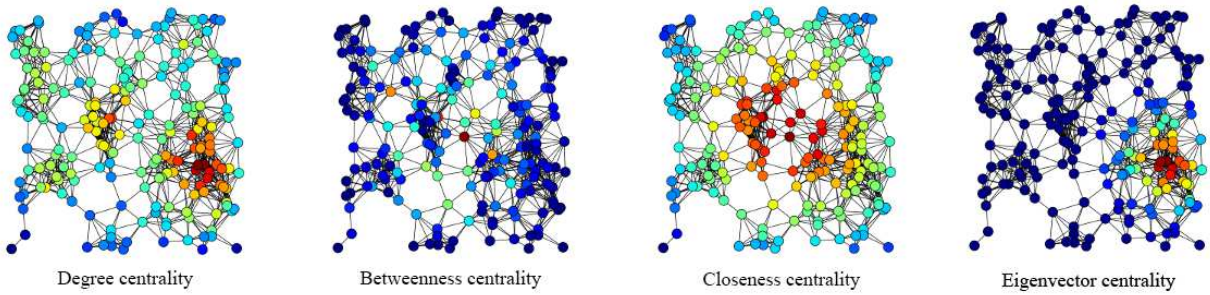


Figure 15 - A comparison of dc, bc, cc and ec [22]

2.1.14. Current-flow centralities (*cfbc*, *cfcc*)

Current-flow centralities are variations on the classical betweenness and closeness centralities in which information spread is calculated not via shortest paths (geodesics) but via

[11] Google's PageRank algorithm for ranking websites (i.e. nodes in a network) is one example of eigenvector centrality (see [21]).

the assumption that it spreads as efficiently as an electrical current (hence current-flow) [23]. In the second experiment that will be described here we have used both current-flow betweenness centrality[12] and current-flow closeness centrality[13], among others, to analyze centralities of certain concepts in a language network and also compare the centralities amongst themselves.

However, due to the complex mathematical nature steeped in physical descriptions of electrical current, we will not be going deep into explaining the exact model and behavior of these centralities. For the sake of completeness I'll provide the formulae for calculating both (see [23]).

Current-flow betweenness centrality:

Let $N = (G; c)$ be an electrical network. Current-flow betweenness centrality $c_{CB} : V \to \mathbb{R}_{\geq 0}$ *is defined by*

$$c_{CB}(v) = \frac{1}{n_B} \sum_{s,t \in V} \tau_{st}(v) \qquad \text{for all } v \in V \ ,$$

where $n_B = (n-1)(n-2)$.

Current-flow closeness centrality:

Let $N = (G; c)$ be an electrical network. Current-flow closeness centrality $c_{CC} : V \to \mathbb{R}_{>0}$ *is defined by*

$$c_{CC}(s) = \frac{n_C}{\sum_{t \neq s} p_{st}(s) - p_{st}(t)} \qquad \text{for all } s \in V \ .$$

2.1.15. Communicability centrality (*comc*)

Communicability centrality is another measure closely tied to betweenness centrality. Instead of looking just at paths passing through nodes in a network, communicability centrality introduces scaling so that not all paths are seen to be of equal worth (longer paths obviously having a lower value). As such, it measures how easy it is to pass messages between nodes in a network.

[12] Also known as random flow betweenness centrality.
[13] Also known as information flow.

We can look at the local communicability of a node to measure its well-connectedness and the total global communicability of the entire network to, for instance, discover bottlenecks [24].

According to Estrada and Hatano (see [24], [25]), the communicability between two nodes can be calculated as the weighted sum C of walks between node i and j. Then the total communicability of a node is given as:

$$TC(i) \coloneqq \sum_{j=1}^{N} C(i,j) = \sum_{j=1}^{N} [e^{\beta A}]_{ij} .$$

2.2. Complex network structure

We have already looked at complex network structure when the small-world and scale-free models and how they rely on small average path lengths and power-law degree distributions, respectively. In this section I plan to expand the discussion with some new terminology to describe network structure and provide some additional insights into what makes the structure of complex networks different from regular or random networks.

It is often said that real-world networks present us with some non-random *topological features*[14] not usually detected elsewhere. The previous section looked at some measures mostly focused on the micro-scale level of the individual node or on macro-scale measures which describe general network features. If we move onto meso-scale analysis (concerned with subgraphs and certain portions of the observed network) and wish to look at emergent structures we have to come to terms with concepts such as:

- Motifs
- Graphlets
- Cliques
- Components
- Communities[15]

[14] Pertaining to the network's structure. We could also call them structural feautres.
[15] Also called modules or clusters.

2.2.1. Motifs and graphlets

Motifs are usually defined as subgraphs (smaller graphs present within a larger structure) which often repeat in a statistically significant manner. Milo (in [26]) called them the *building blocks* of networks - "*patterns of interconnections occurring in complex networks at numbers that are significantly higher than those in randomized networks*". The case is that certain types of networks, e.g. some closely related biological networks, will usually also share similar motifs. These similarities may also extend beyond the scope of networks of a certain family of networks (see Figure 16). By definition, significant motifs in complex networks must not be defined simply by counting their number of appearances within a network. Specifically, *network motifs are those patterns for which the probability P of appearing in a randomized network an equal or greater number of times than in the real network is lower than a cutoff value* [26].[16]

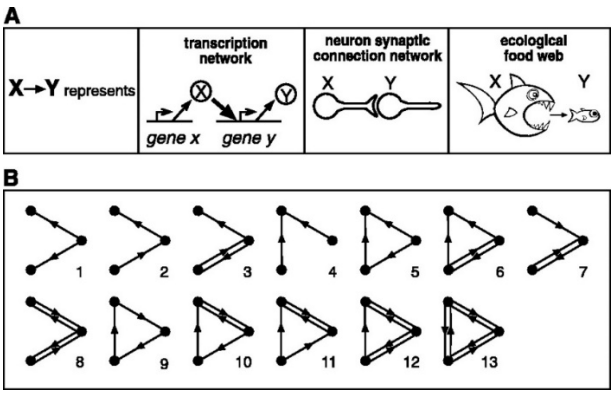


Figure 16 - Several networks with three-connected subgraph motifs [26]

The difference between motifs and graphlets is contained in the fact that, while motifs may be partial subgraphs, graphlets are always induced subgraphs[17] and do not rely on the comparison with randomized networks [27].

[16] The significance profile of a motif can be calculated upon this principle.
[17] Partial subgraphs (motifs) need not contain all the edges present in their network of origin and induced subgraphs (graphlets) must.

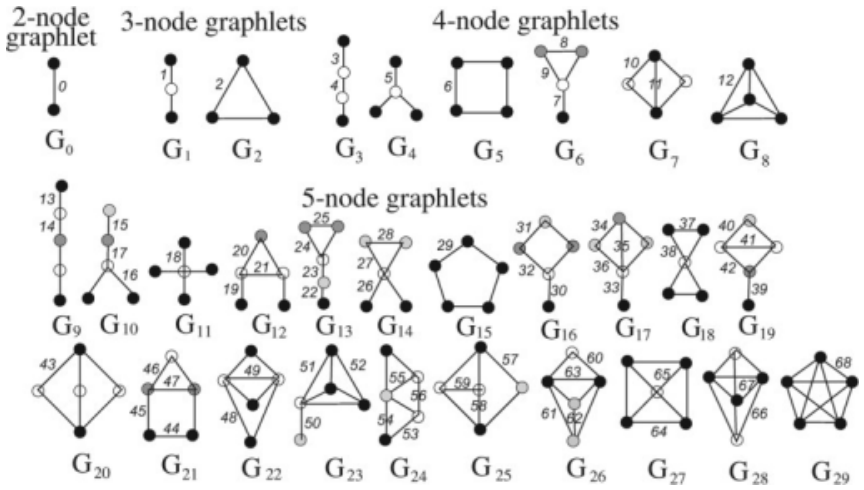


Figure 17 - Graphlets up to five nodes [28]

Some of the ways in which network structure analysis can be done using motifs and graphlets is via modeling and statistical analysis of motif occurrences or by looking at graphlet degree distributions, among others.

2.2.2. Cliques and communities

Per definition, *a clique C in an undirected graph G = (V, E) is a subset of the vertices, $C \subseteq V$, such that every two distinct vertices are adjacent*[18] [29]. In [4], Newman notes *that the occurrence of a clique in an otherwise sparse network is normally an indication of a highly cohesive subgroup*. So we must look at cliques as those portions of a network which are closely connected between each other but rather distanced to the rest of the network at large. In terms of graph theory, this usually entails a very small average path length between members and high density relative to the rest of the network.

An example are real-world cliques (a common example are terrorist groups, see [30]) wherein people organize into little communities based on some sort of internal familiarity.

There are several variations on the same concept, so we differentiate between cliques, *k*-plexes, *k*-cores and *k*-cliques. Since real-world cliques rarely satisfy the condition that any two nodes must be adjacent, *k*-plexes require that each node connects to *n-k* nodes, *n* specifying the number of nodes in the subgraph. In *k*-cores, each node must connect to at least *k* other nodes. In *k*-cliques, each node is less than *k* nodes away from all other nodes of the subgraph. There are also other ways to identify cliques that we will not be mentioning here.

[18] Thus satisfying the conditions for a complete (sub)-graph.

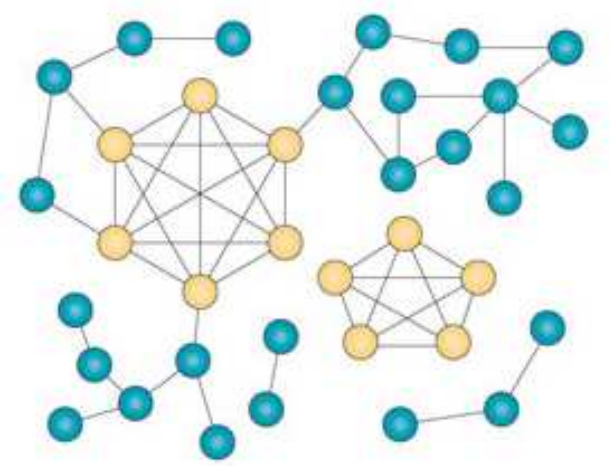


Figure 18 - Cliques within a network [31]

Another useful meso-scale structure are components, defined as the *maximal subset of vertices such that each is reachable by some path from each of the others* [4]. A variation on components are *k*-components, *k* representing the minimal number of vertex-independent[19] paths by which all vertices must be reachable.

The final concept related to meso-scale complex network structure we'll be looking at are communities. As with cliques or components, nodes within a community are more likely to connect to nodes within that same communities than to others[20]. Most commonly, the process of dividing a network into communities is done *so that the groups formed are tightly knit with many edges inside groups and only a few edges between groups* [4].

A network which clearly divides into several clusters of communities can be said to exhibit community structure. If the communities found within a network can be nested in whatever manner, then we can say that these also exhibit hierarchical or multi-scale structure. Motivation to analyze communities is commonly found in the fact that *identification of the community structure of complex networks provides insight into the relationships between network function and topology [32]*.

[19] Two paths sharing none of the same vertices other than the starting and the finishing one [4].
[20] This clustering tendency is also sometimes referred to as network transitivity.

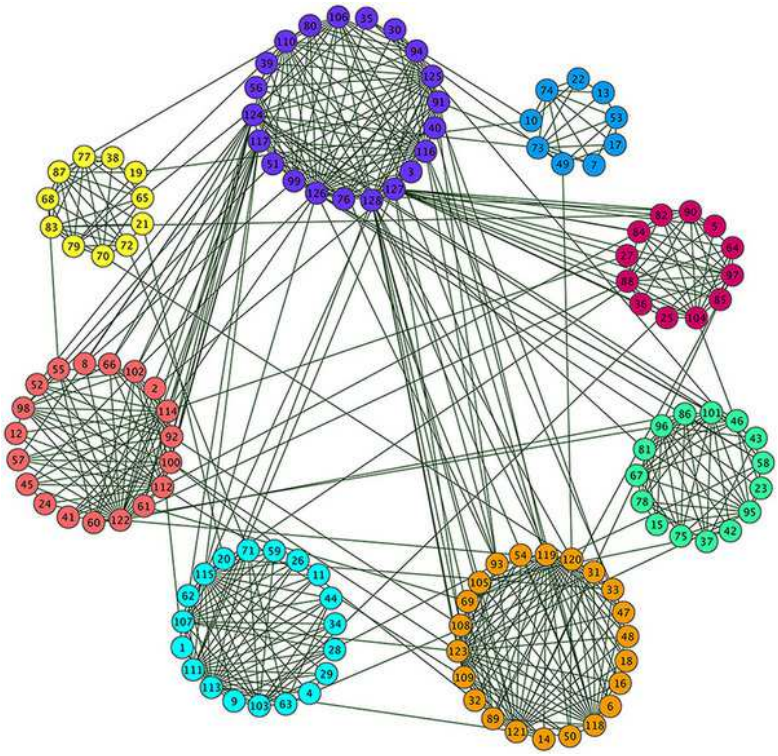


Figure 19 - Community structure showing non-overlapping communities [33]

When looking at communities we can differentiate between overlapping and non-overlapping communities within a network. In the first case, a node can be a member of more than one community, while in the latter the network is clearly divided and there are no overlaps.

Community detection is a non-trivial procedure which usually involves some amount of heuristics since they are not strictly delineated in real-world networks. Several algorithms are used for community detection and graph partitioning[21] such as hierarchical clustering, Girvan-Newman's algorithm, minimum-cut method, modularity maximization, clique percolation and others.

One of the most efficient is the Louvain method based on modularity maximization. It is a greedy optimization method that optimizes the modularity of a network's partitions. It is also successful in exposing the hierarchical structure of the network at hand.

The Louvain method follows a simple procedure [34] (Figure 20 visualizes the entire process):

[21] Different from community detection in that partitioning does not depend on the very topological organization of the network.

- Looking for small communities by doing a local modularity optimization
- Aggregating nodes belonging to the same community and building a new network whose nodes are the communities
- Repeating the previous two steps iteratively until a maximum of modularity is attained and a hierarchy of communities is produced.

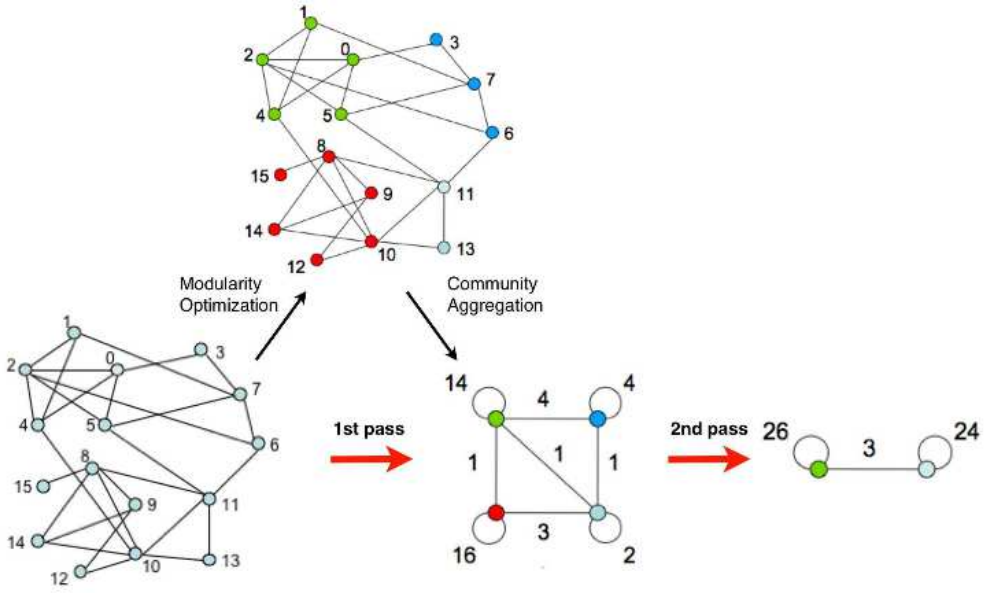


Figure 20 - Louvain method for community detection [35]

The results that can be achieved by using the Louvain algorithm will be presented in section 4.

3. Related work

Since the overarching topic of this work is modelling and analyzing Wikipedia as a complex network, it would be worthwhile to look at the efforts of others in the same field. This section will present the most important publications and breakthroughs on the mentioned topic.

Wikipedia can be modelled as a complex network in a way that Wikipedia entries are nodes, and links between two nodes are established if there is a hyperlink between these two entries. Early attempts to quantify Wikipedia using complex networks analysis were focused only on network structure of linked Wikipedia entries. In [39] Zlatić et al. present an analysis of Wikipedias in several languages as complex networks. They show that many network

characteristics (degree distributions, growth, topology, reciprocity, clustering, assortativity, path lengths and triad significance profiles) are common to Wikipedias in different languages and show the existence of a unique growth process. The same authors studied Wikipedia growth based on information exchange in [40]. In [41] the authors present an analysis of the statistical properties and growth of Wikipedia. Pembe and Bingol [42] have constructed two complex networks out of English and German Wikipedia corpora and analyzed conceptual networks in different languages.

The other research direction is focused on content found on Wikipedia and analyses Wikipedia as a (domain) knowledge network. In Fang [43] they first extract a specific domain knowledge network from Wikipedia (specifically, four domain networks on mathematics, physics, biology, and chemistry) and then carry out statistical analysis on these four knowledge networks. Also, they show that MathWorld and Wikipedia Math share a similar internal structure. In [44] Masucci et al. extract the topology of the semantic space and measure the semantic flow between different Wikipedia entries. They further analyze a directed complex network of semantic flow. In [45] the results of semantic language networks analysis are presented in general.

Motivated by the second approach that studies Wikipedia as a knowledge network, the first experiment wanted to study how the network structure is related to domain knowledge. The goal of the experiment was to extract centrally positioned entries in the network and analyze how these entries are related to domain knowledge and are some more important than other. In the second part of the experiment the task was to extract entries that belong to the same community and check whether they are semantically related.

The second experiment I will present rests firmly on employing network centrality measures for keyword extraction. Šišović in [46] and Beliga in [47] looked at applying network measures for the purpose of keyword extraction. For that purpose they constructed language networks out of web documents and news texts. Similar research was previously conducted by Lahiri et al. [48] wherein 11 various network measures were used and compared as candidates for successful keyword extraction.

Knowledge extraction from Wikipedia was variously also conducted by [49]:

- Creating a Wikipedia thesaurus using the *pfibf* (Path Frequency – Inversed Backward link Frequency) method

- Disambiguation and synonym extraction from anchor texts (page/entry names)
- Creating a Wikipedia ontology and XML-based API
- Construction of bilingual dictionaries through mining processes

Also, in [50], [51], [52], [53] and [54] various authors look at the topic of semantic relatedness as a property between concepts which can be extracted through the analysis of Wikipedia by using some of the following:

- Hyperlink structure analysis
- Category hierarchy and taxonomy-based measures
- Textual content and text overlap analysis
- Machine learning methods
- Spreading activation methods

They often compare their results with those obtained through the WordNet lexical database[22].

Being a rich knowledge corpus, Wikipedia is seen as truly fruitful resource for a variety of research. The two experiments that will be presented in the following sections detail some ways in which Wikipedia can be analyzed using methods for complex network analysis and some of the aforementioned, previously conducted research.

[22] See WordNet at Princeton University: https://wordnet.princeton.edu/

4. Experiment 1 – Domain knowledge extraction

After an overview of graph theory basics, Wikipedia and complex network structure and analysis, we can proceed onto the two actual experiments which were conducted by combining the three together in order to arrive at some novel conclusions.

The first experiment[23] consisted of constructing networks out of Wikipedia entries, calculating and analyzing global network metrics, centrality measure analysis and community detection of related concepts. This section will go through all steps related to the experiment.

4.1. Network construction

For the purpose of our experiment I collect entries from Wikipedia and construct networks related to the domain. My intention was to construct two types of networks: level 2 networks and level 4 networks.

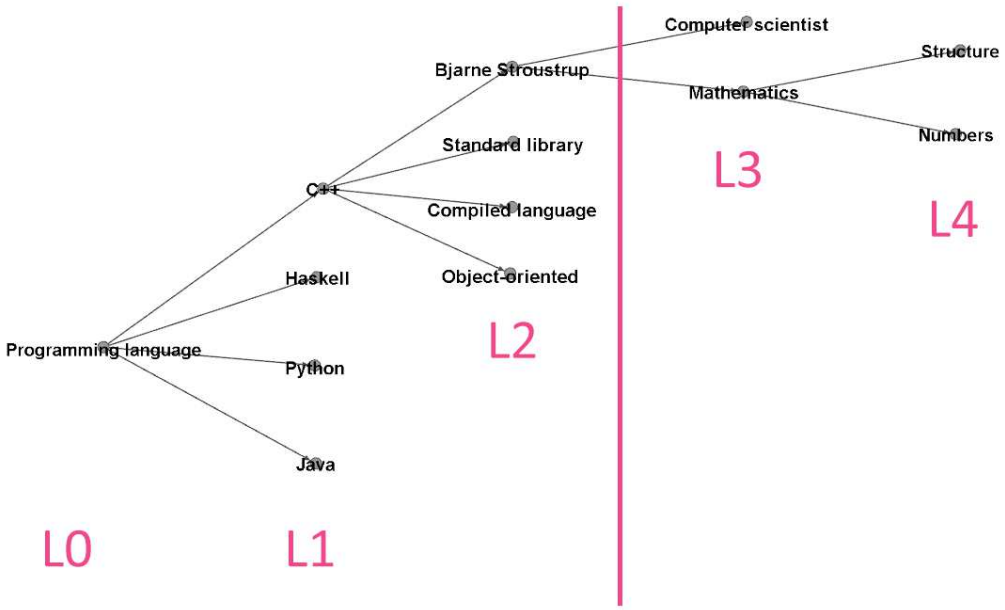


Figure 21 - Network construction from Wikipedia entries

Level 2 networks are constructed by starting with a chosen seed entry (e.g. "*Complex network*" or "*Data*"), storing all the hyperlinks to related entries from the seed entry's text (level 1) and proceeding to extract the hyperlinks from all the entry pages taken from the original entry (level

[23] Experiment and results were also summarized in *N. Matas; S. Martinčić-Ipšić; A. Meštrović. Extracting Domain Knowledge by Complex Networks Analysis of Wikipedia Entries. IEEE MIPRO 2015. pp. 1955-1960, 2015.*

2). Analogously, level 4 networks are constructed by taking the first 10 hyperlinks from a given entry page and proceeding to repeat the task three times, arriving at level 4 (see Figure 21 for a simplified visualization of the process).

The hyperlinks were limited to just the first 10 due to the computational complexity, at the same time having in mind that the most general hyperlinks are usually at the beginning of the entry's text.

Therefore, the first task is the construction of a web scraping program which would extract hyperlinks from a Wikipedia entry's text. The hyperlinks are extracted using a Python package for HTML parsing called Beautiful Soup which parses the HTML structure of a given HTML document into a parse tree. By navigating the tree one can locate the tag ID which corresponds to article content ("mw-content-text") and proceed to extract the hyperlinks which themselves are found within paragraph (<p>) tags and finally inside link (<a>) tags in that section of the page.

Finally, each network is stored in an edge list in the following format (\t representing a tab character in order to construct a TSV file[24]):

"entry title" \t "linked entry title".

Difficulties with processing non-ASCII script and hyperlinks that were not connected to other documents (citations, in-page references, etc.) were avoided by checking the data during the extraction process.

The code itself works in the following way - a function called `getURLS (articleURL)`, is constructed so that, for a given Wikipedia article URL (a string containing the link of a Wikipedia article such as http://en.wikipedia.org/wiki/Programming_language) gets all the other relevant hyperlinks and also stores the rows that will be written into the edge list to a previously declared array. The function is then called again for each element in the array of hyperlinks. This process can be repeated several times according to the number of levels we wish to extract.

The entire code with detailed comments can be examined below:

[24] Tab-separated values.

```
from urllib2 import quote
from urllib2 import unquote
from urllib2 import urlopen
from bs4 import BeautifulSoup

def getURLS (articleURL):

    ### articleLINKS stores the temporary incomplete URLs
    articleLinks = []

    ### articleURLS stores the complete list of URLs found in an article
    articleURLS = []

    ### articleEdgelist stores the rows that will go into the final edge    list
    articleEdgelist = []

    ### Getting the HTML file for a specified URL
    articleHTML = urlopen(articleURL).read()

    ### Creating a BeautifulSoup object
    articleBS = BeautifulSoup(articleHTML)

    ### Getting the title of the current article
    articleTitleMain = articleURL [29:].replace('_', ' ')

    ### Extracting and storing the relevant content
    articleContent = articleBS (id = "mw-content-text")

    articleParagraphs = articleContent[0].find_all('p')
    ### Extracting and storing all hyperlinks

    for ap in articleParagraphs:
        articleLinks.append (el.find_all('a'))

    ### Hyperlink checking, manipulation and creation
    for ap in articleLinks:
        for tag in ap:
            tagContent = tag['href']
            tagTitle = tag.string

            if (tagContent[0:6] == "/wiki/"):
                if (':' not in tagContent):
                    ### Appending a row to the articleURLS array
                    articleURLS.append("http://en.wikipedia.org" +   tagContent)
                    articleTitle = tagContent [6:].replace('_', ' ')

                    if ('#' in articleTitle):
                        articleTitle = articleTitle[0:articleTitle.index('#')]

                    if ('%' in articleTitle):
                        articleTitle = unquote(articleTitle).decode('utf-8')

                    ### Appending a row to the the articleEdgelist array
                    articleEdgelist.append ("\"" + articleTitleMain.lower() + "\"" +
                    "\t" + "\"" + articleTitle.lower() + "\"")

    return articleURLS

### Creating the array that will store the edge list's rows
articleEdgelist = []
### Specifying the seed entry URL
articleURL = "http://en.wikipedia.org/wiki/Programming_language"
```

```
### Level one edge list construction
articleURLS = getURLS (articleURL)

### Level two edge list construction
for url in articleURLS:
    articleURLS_L2 = getURLS (url)

articleEdgelist = sorted(set(articleEdgelist))

### Opening and writing to file all the contents from articleEdgelist array
f = open("PL2.edges", 'a')

for el in artEDLI:
    f.write(el.encode('utf-8'))
    f.write("\n")
    print el
```

In such a directed network, each entry's title represents a node and it is connected to other entries hyperlinked in its text, again represented as network nodes. I constructed a total of 10 domain networks for five chosen seed entries: "*Byte*", "*Complex network*", "*Computer science*", "*Data*" and "*Programming language*".

The naming scheme includes the level of a specific network in its name (e.g. the level 2 network for "Byte" is BT2, the others being BT4, CN2, CN4, CS2, CS4, DT2, DT4, PL2 and PL4). Since I have considered exclusively unweighted networks, double links were dismissed. This, along with the fact that some entries did not contain 10 hyperlinks resulted in the level 4 networks having less than 10^4 expected edges.

Figure 22 shows a visualization of the CN2 network (seed entry being "*Complex network*"). It was created using the open-source Gephi software suite with the the Yifan-Hu proportional layout algorithm and node sizing according to degree.

Figure 6 as shown on page 7 is an example of a building block out of which these networks are constructed.

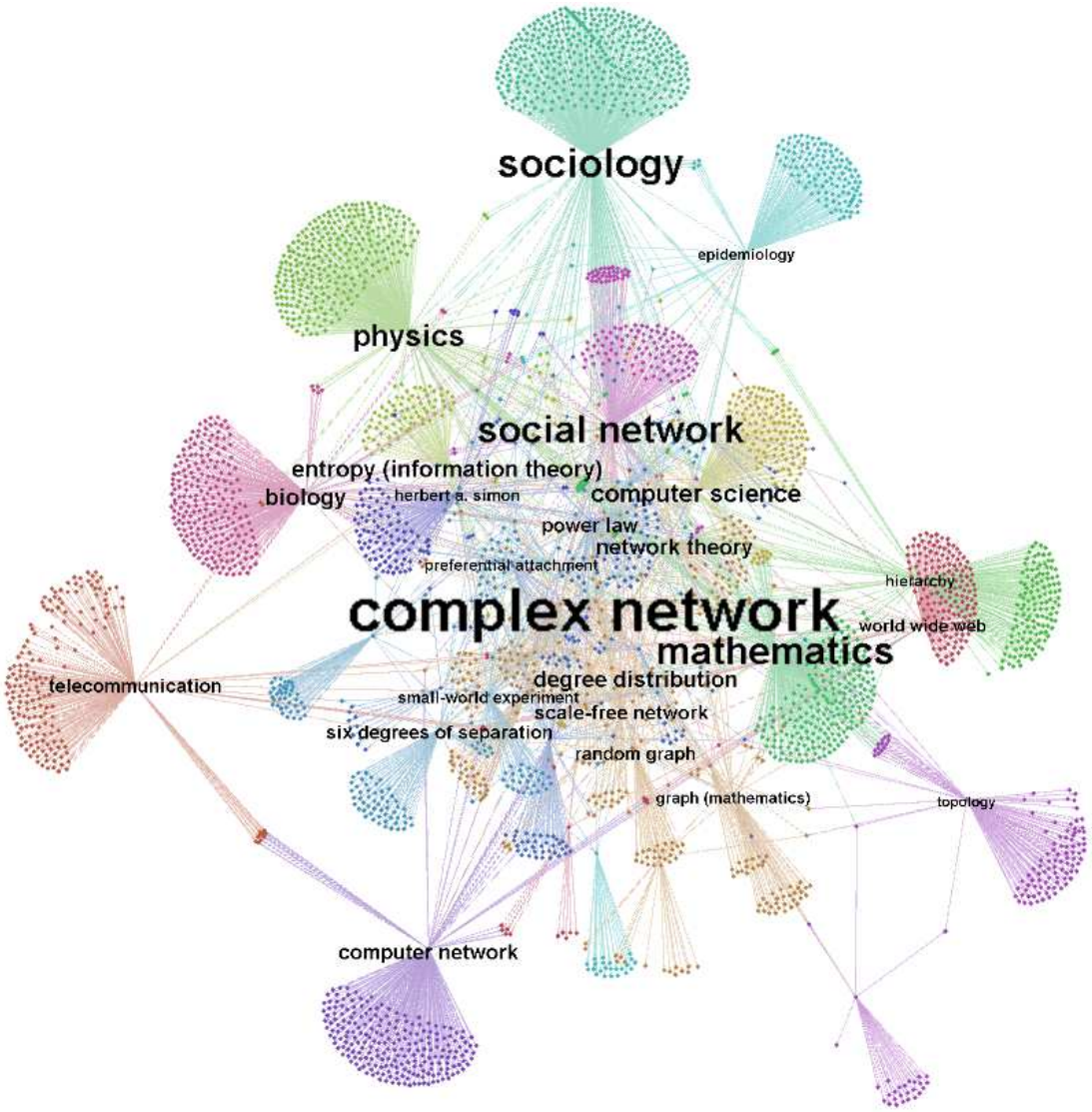


Figure 22 - CN2 network visualization

4.2. Global network measures

In this section I'll present the results of measurements done using some of the metrics described in section 2 such as average degree <*k*>, average path distance *L*, diameter *D*, average clustering coefficient *C,* density *d*, modularity *Q,* number of communities (N_c) and degree assortativity coefficient *r*. Table 2 holds all of the relevant data.

There are certain differences between measures for level 2 and 4 which are evident upon closer inspection. For instance, level 4 networks have significantly larger average path lengths, diameters, assortativity coefficients, often a significantly larger number of detected

communities and slightly larger average degrees. The modularity measure and density are comparable between the two, whilst level 2 networks show larger clustering coefficients.

Table 2 - Global network measures

Measure	**"Byte"**		**"Complex network"**		**"Computer science"**		**"Data"**		**"Program. Language"**	
Network	**BT2**	**BT4**	**CN2**	**CN4**	**CS2**	**CS4**	**DT2**	**DT4**	**PL2**	**PL4**
Number of nodes (N)	3945	3632	3405	3070	12881	3630	2297	3658	7467	3965
Number of edges (K)	5112	5611	4132	5008	18852	5851	2630	5531	13933	6215
Average degree ($<k>$)	1.296	1.545	1.214	1.631	1.464	1.612	1.145	1.512	1.145	1.612
Avg. shortest path (L)	3.693	6.834	3.198	9.218	3.417	6.277	3.086	6.369	3.127	6.277
Avg. shortest path (L_{ER})	8.6938	7.26622	9.16841	6.791134	8.80884	7.0022451	9.34083	7.414438	10.76366	7.0776521
Diameter (D)	9	15	6	22	7	14	7	14	6	22
Average clustering coefficient (C)	0.06	0.021	0.043	0.024	0.074	0.019	0.043	0.019	0.082	0.021
Average clustering coefficient (C_{ER})	0.0006	0.00085	0.00071	0.0010625	0.00023	0.0008882	0.001	0.000827	0.000307	0.0008131
Density (d)	0.0003	0.00042	0.00035	0.00053	0.00011	0.00044	0.00049	0.00041	0.00025	0.0004
Modularity (Q)	0.778	0.776	0.794	0.763	0.725	0.771	0.828	0.779	0.594	0.78
Number of communities (N_c)	17	32	17	21	23	27	18	31	19	30
Degree assortativity coefficient (r)	-0.592	-0.048	-0.521	0.021	-0.491	-0.028	-0.561	-0.048	-0.468	-0.059

For comparison with random networks, the table also includes two measures for equivalent random networks (Erdös-Rényi random graphs) – the average shortest path length ($L_{ER} = \frac{\ln N}{\ln <k>}$) and the average clustering coefficient ($C_{ER} = \frac{<k>}{N}$). The results show that the complex networks I have constructed have a significantly higher average clustering coefficient than their Erdös-Rényi random graph counterparts. This, in addition with a relatively small average shortest path length L can lead us to conclude that we are dealing with small-world networks as described by Watts and Strogatz [6]. For the purposes of this comparison I treated the networks as undirected.

Moreover, a distinctly high modularity coefficient Q (higher than 0.7 in all but one network, as obvious from Table 2) shows a clear tendency towards community clustering of nodes present in the networks. I did not observe any strict rule governing community size across networks, although level 2 networks have an understandably smaller N_c which can be contributed to the very construction principle as described previously.

The degree distributions were found to approximately fit a power-law, but the results were somewhat inconclusive so I cannot definitively state these networks are truly scale-free. Figure

23 and 24 present the results for 4 out of the 10 constructed network (CN2, CN4, CS2, CS4). The observed difference between the level 2 and 4 networks is believed to be due to the method of network construction.

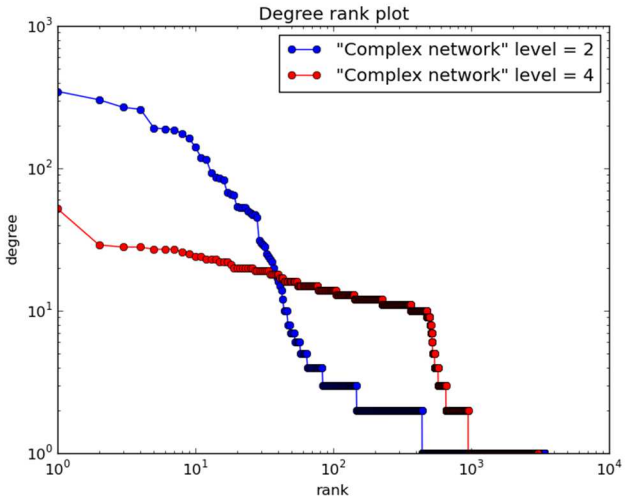


Figure 23 - CN2 and CN4 degree distributions

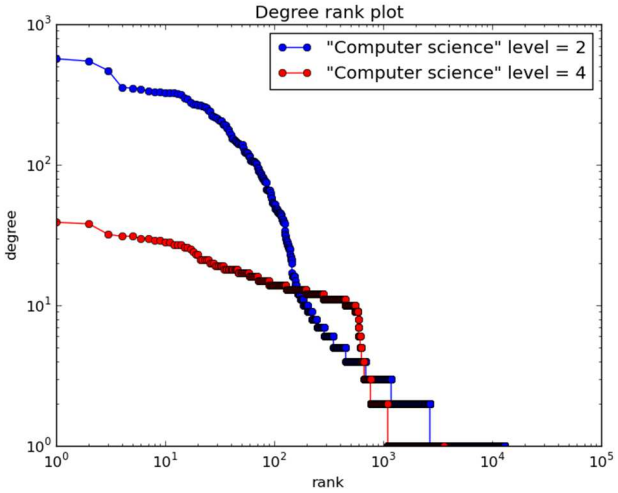


Figure 24 - CS2 and CS4 degree distributions

4.3. Micro-scale analysis using centrality measures

After the analysis on the global level, I have analyzed the networks on the local level in terms of centrality measures. Tables 3, 4, 5, 6 and 7 present lists of top ten entries according to three centrality measures (degree centrality, betweenness centrality and closeness centrality)

for all of the seed entries: "*Computer science*", "*Programming language*", "*Byte*", "*Complex network*" and "*Data*".

For the degree centrality calculations I have treated the networks as undirected. For each centrality measure and domain there are two lists of entries, one for level 2 networks and another for level 4 networks. I have noticed that the lists for level 2 networks consist of entries that are semantically related to the seed entries ("Computer science" or "Programming language") in a way that might be ascribed as belonging to a hierarchy.

This is especially evident for the closeness centrality measure. For example, the list of top ten entries according to the closeness centrality for the seed entry "Computer science" contains other scientific domains (*theoretical computer science*, *mathematics*, *artificial intelligence*, *physics*, *engineering*) and for the seed entry "Programming language", the list contains some prominent programming languages (*C*, *Java*, *Perl*, *Python*, *C++*).

Table 3 - Centrality analysis for CS2 and CS4 networks

	Degree centrality		**Betweenness centrality**		**Closeness centrality**	
	CS2	**CS4**	**CS2**	**CS4**	**CS2**	**CS4**
#1	human	mathematics	computer science	computer science	computer science	computer science
#2	university of cambridge	cell (biology)	computer	information	mathematics	information
#3	philosophy	computer science	mathematics	protein	theoretical computer science	science
#4	industrial revolution	computer	artificial intelligence	science	computer	cell (biology)
#5	gottfried wilhelm leibniz	information	philosophy	algorithm	artificial intellgience	mathematics
#6	physics	protein	human	logic	philosopy	ancient greek
#7	eletrical engineering	organism	gottfried wilhelm leibniz	organism	physics	latin
#8	artificial intelligence	dna	algorithm	cell (biology)	human	computing
#9	mathematics	computer program	theoretical computer science	computing	gottfried wilhelm leibniz	algorithm
#10	alan turing	philosophy	physics	mathematics	engineering	bit

Table 4 - Centrality analysis for PL2 and PL4 networks

	Degree centrality		Betweenness centrality		Closeness centrality	
	PL2	**PL4**	**PL2**	**PL4**	**PL2**	**PL4**
#1	history of computing hardware	mathematics	programming language	programming language	programming language	programming language
#2	internet	computer	computer	computer	c (programming language)	ancient greek
#3	english language	computer science	c (programming language)	software engineering	computer programming	computer
#4	computer	physics	compiler	computing	java (programming language)	mathematics
#5	c (programming language)	set (mathematics)	english language	computer science	perl	arithmetic
#6	python (programming language)	greek language	computer program	algorithm	compiler	science
#7	university of manchester	logic	internet	message	computer program	greek language
#8	perl	language	perl	communication	python (programming language)	physics
#9	programming language	central processing unit	python (programming language)	machine	control flow	latin
#10	php	electronics	java (programming language)	function (mathematics)	c++	computer science

Table 5 - Centrality analysis for BT2 and BT4 networks

	Degree centrality		Betweenness centrality		Closeness centrality	
	BT2	**BT4**	**BT2**	**BT4**	**BT2**	**BT4**
#1	internet	mathematics	byte	information	byte	information
#2	united states army	computer	C (programming language)	computer science	computing	computer science
#3	french language	information	internet	computing	internet	bit
#4	microprocesso r	number	computing	memory	ascii	computing
#5	alexander graham bell	integer	ascii	bit	bit	computer
#6	C (programming language)	computing	international system of units	computer	programming language	mathematics
#7	international system of units	data	microprocessor	mathematics	microprocessor	byte
#8	romanian language	computer science	bit	message	c (programming language)	communicatio n
#9	telecommunic ation	set (mathematics)	decibel	perception	binary-coded decimal	character (computing)
#10	ascii	language	metric prefix	number	alexander graham bell	message

Table 6 - Centrality analysis for CN2 and CN4 networks

	Degree centrality		Betweenness centrality		Closeness centrality	
	CN2	**CN4**	**CN2**	**CN4**	**CN2**	**CN4**
#1	sociology	mathematics	complex network	sociology	complex network	mathematics
#2	physics	set (mathematics)	sociology	social network	computer science	graph theory
#3	mathematics	greek language	mathematics	social structure	sociology	ancient greek
#4	computer network	statistics	social network	population	social network	computer science
#5	biology	computer science	physics	social network analysis	physics	mathematical model
#6	social network	graph theory	computer science	network theory	biology	graph (mathematics)
#7	telecommunication	function (mathematics)	entropy (information theory)	set (mathematics)	network theory	network theory
#8	world wide web	physics	biology	network science	mathematics	set (mathematics)
#9	computer science	real number	degree distribution	statistics	epidemiology	arithmetic
#10	hierarchy	organism	network theory	venn diagram	power law	complex network

Table 7 - Centrality analysis for DT2 and DT4 networks

	Degree centrality		Betweenness centrality		Closeness centrality	
	DT2	**DT4**	**DT2**	**DT4**	**DT2**	**DT4**
#1	computer	mathematics	data	data	data	information
#2	number	statistics	computer	information	computer	data
#3	mount everest	computer science	computer program	measurement	number	mathematics
#4	alphabet	information	number	mathematics	lisp (programming language)	computer science
#5	lisp (programming language)	set (mathematics)	level of measurement	observation	computer program	science
#6	computer program	science	alphabet	computer science	measurement	statistics
#7	measurement	data	information	set (mathematics)	character (computing)	observation
#8	information	logic	lisp (programming language)	level of measurement	alphabet	ancient greek
#9	analog computer	programming language	knowledge	knowledge	analog computer	knowledge
#10	knowledge	philosophy	in situ	mathematician	set (mathematics)	logic

4.4. Community detection

In the second part of the experiment I have analysed communities in all 10 networks in order to explore which entries are grouped together. Figures 25 and 26 show the most significant entries from the CS2 network grouped into communities.

Different communities are presented in different colors. For example, entries related to the mathematics domain (*mathematics*, *number*, *set*, *function*, *real number*, etc.) are in the red-

coloured community; entries related to the computer science domain (*computing*, *algorithm*, *compiler*, etc.) are in the orange-coloured community; entries that are related to the biology domain (*cell*, *organism*, *gene*, etc.) are in the light-orange coloured community and entries that are related to the philosophy domain (*reality*, *concept*, *knowledge*, etc.) are in the white-coloured community.

It can be observed that entries grouped into communities are more closely semantically related than entries from different communities. The results are similar for other networks; semantically related entries are grouped into communities much more than entries that are not semantically related.

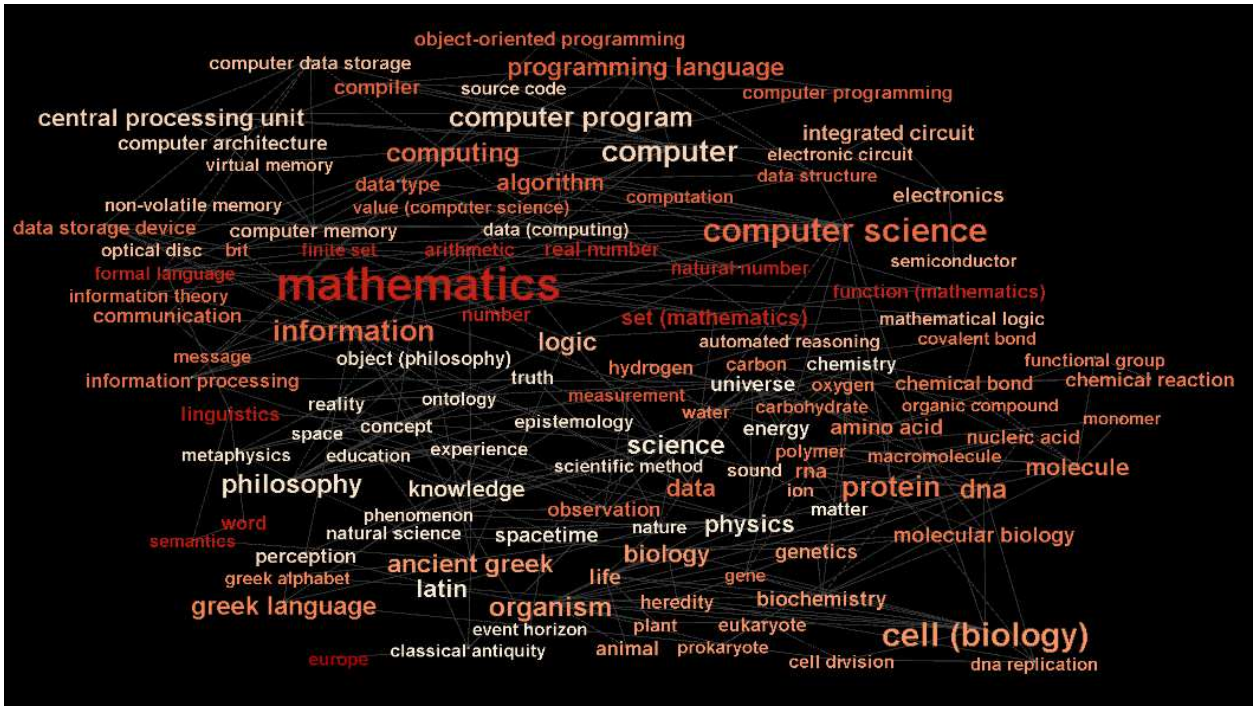


Figure 25 - Community structure of the CS2 network

Community detection in these networks was done by employing the Louvain method described before. A NetworkX implementation of the algorithm can be found on the following web page: http://perso.crans.org/aynaud/communities/.

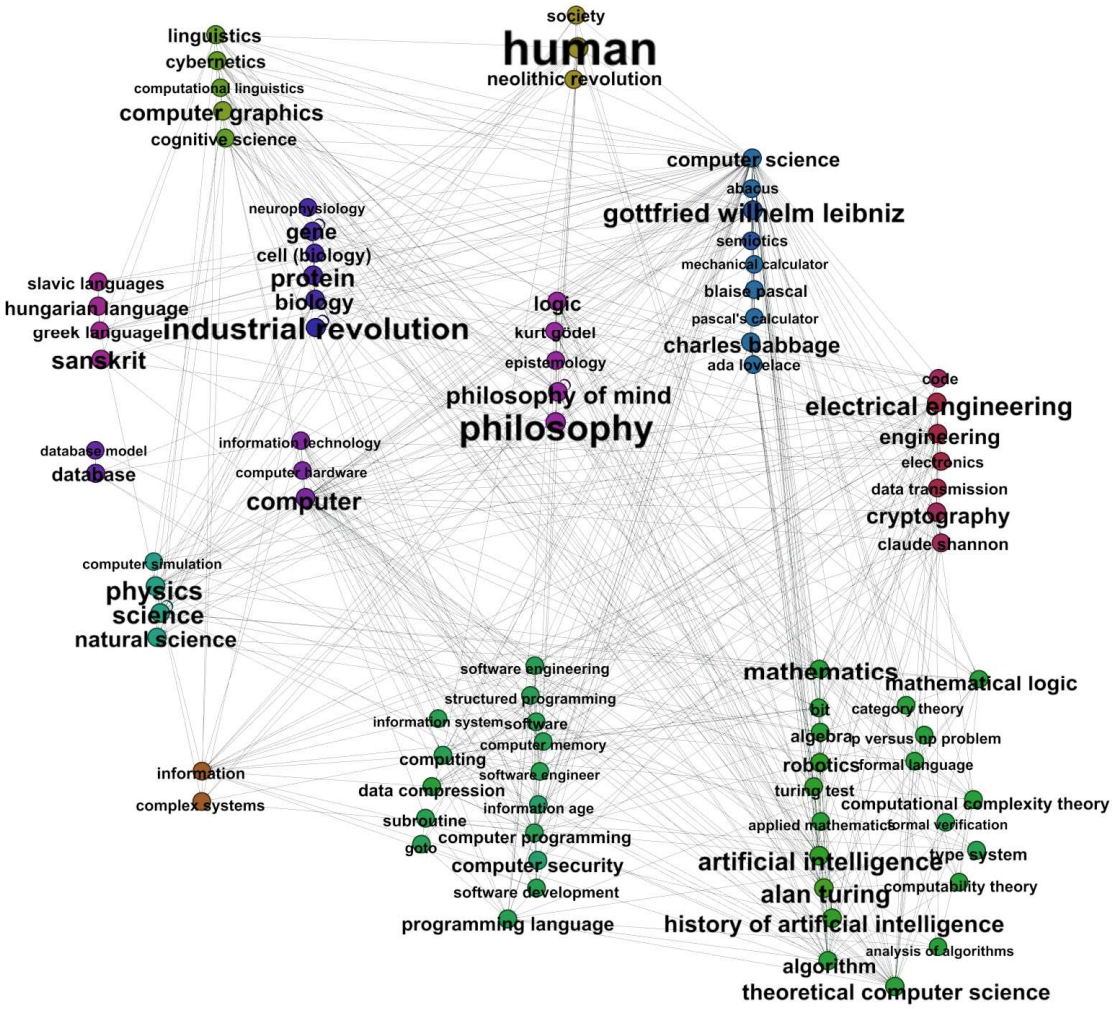


Figure 26 - Another visualization of the community structure of the CS2 network[25]

Table 8 is a list of key nodes in the communities present in the CS2 network. It is interesting to note how the communities depict the complex and interdisciplinary nature of computer science, seeing how communities revolve all sorts of relevant concepts including Charles Babbage ("*father of the computer*"), cryptography, electrical engineering, computer graphics, artificial intelligence, computer security, programming language, databases, computer simulation etc.

[25] Only a portion of the network is shown (0,66 % of nodes and 4,05% of edges). Node color corresponds to the community it belongs to. Node label size corresponds to node degree.

Table 8 - Communites and their representative nodes[26] in the CS2 network

Communities (representative nodes)
Charles Babbage
Cryptography
Protein
University of Cambridge
Cybernetics
Gottfried Wilhelm Leibniz
Bernoulli number
Electrical engineering
Computer graphics
IBM
Artificial intelligence
Industrial revolution
Philosophy
Computer security
Physics
Programming language
Purdue University
Human
Common Core State Standards Initiative
Data compression
Database
Sanskrit
Computer simulation

Table 9 shows the same for the CN2 network wherein we can observe the interdisciplinarity of complex networks and network science, with communities describe by nodes such as physics, mathematics, biology, sociology, etc.

Table 9 - Communites and their representative nodes in the CN2 network

Communities (representative nodes)
Entropy (information theory)
Six degrees of separation
Power law
Topology
Telecommunication
Sociology
Physics
Mathematics
Network theory
Hierarchy
Herbert A. Simon
Epidemiology
Computer network
Complex network
Biology
Social network
Computer science
World wide web
Entropy (information theory)

[26] Taken to be those nodes with the highest degree within a chosen community.

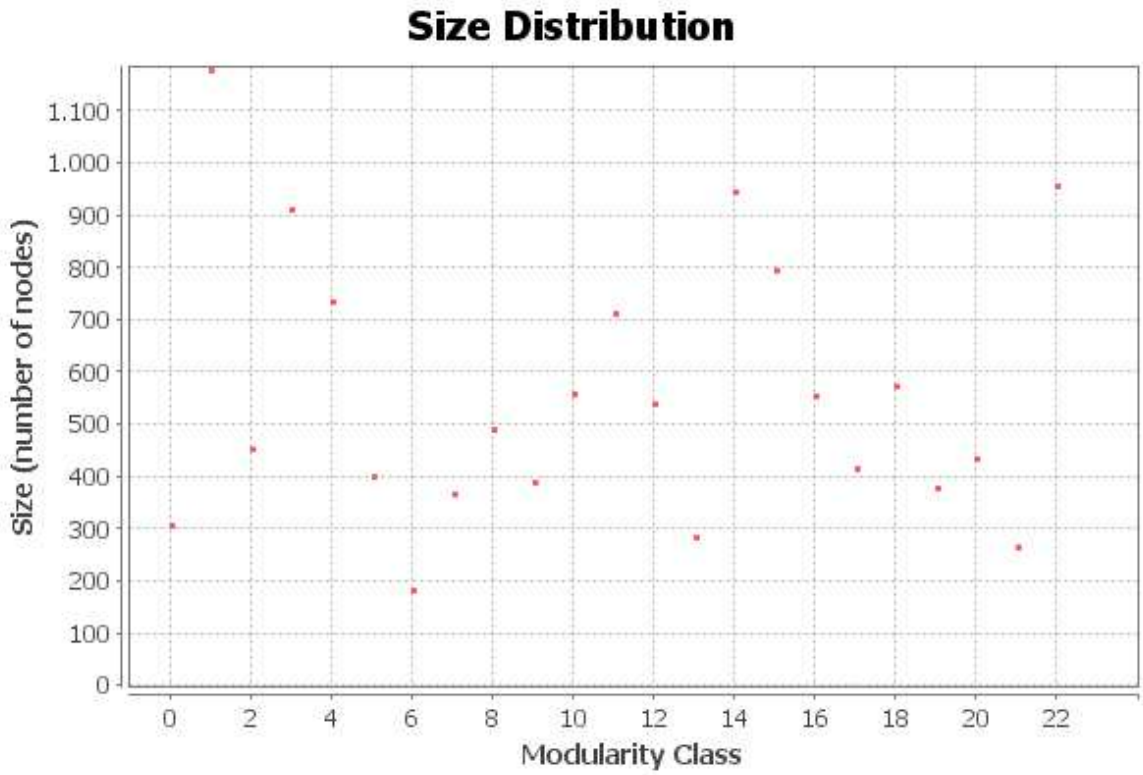


Figure 27 - Community distribution

Figure 27 presents the degree distribution of communities present in the "*Computer science*" level 2 network (CS2).

4.5. Discussion

In this section I present my initial attempt to study Wikipedia as a complex network. I extracted parts of Wikipedia related to 5 chosen seed entries and constructed 10 different networks using two different principles of construction.

Afterwards, I analyzed the global structure of all networks and showed that all networks have similar properties: a high average clustering coefficient in comparison to random networks, small distances, low density and community structure. From these global measures I may conclude that all 10 networks extracted form Wikipedia are small-world networks. These results are in line with previous studies of Wikipedia as a complex network.

Furthermore, I explored semantic relations in the constructed networks by using network centrality measures to extract entries in the networks that are significant according to the network structure. Three centrality measures are employed for this task: degree centrality, betweenness centrality and closeness centrality. It can be observed that for level 2 networks centrality measures obtain good results, especially in the case of closeness centrality. Among top ten entries according to the closeness centrality are entries that are semantically related to the domain. This can be useful for modelling taxonomy or domain ontology. Furthermore, semantically related entries are grouped into communities more often that entries that are not semantically related.

These findings can be partially explained as a consequence of network construction rules employed in this experiment. However, these results suggest that Wikipedia is well organized and its structure can be captured and explored by a complex networks approach. This led me to conduct another experiment which will be presented in the next section.

5. Experiment 2 – Keyword extraction using network centralities

5.1. Network construction

The second experiment follows a similar but deeper approach to complex network analysis of Wikipedia.

It consists of the following steps:

- Creation of a network (edge list) using a seed entry of choice
    - In this case the network of choice was a level 2 network using the first 20 links with the seed entry being "*Programming language*"
- Extraction of complete entry text for every node in the previously constructed network
- Text preprocessing by means of:
    - Transformation of each text into lower caps
    - Removal of punctuation from each text

    - Removal of stop words[27] from each text
    - Lemmatization[28] of each text
- Creation of a co-occurrence network from each text
- Extraction of 30 top keywords from each network (text) according to the following measures:
    - Closeness centrality
    - Betweenness centrality
    - Eigenvector centrality
    - Degree centrality
    - Current-flow betweenness centrality
    - Current-flow closeness centrality
    - Communicability centrality
- Creation of two new types of networks for each centrality (7 centralities × 4 thresholds × 2 network types = 56 networks):
    - A subset network multiplexed with the original network in which links are established if they exceed a certain threshold (the threshold representing how many keywords two distinct nodes / concepts share) and satisfy the condition of existence in the original network
    - A network in which a connection between all nodes is possible and is established only if they satisfy the threshold condition
- Comparison with the original network via the Jaccard overlap[29] in order to establish the differences between various measures in keyword extraction.
- Analysis of extracted keywords.

Figure 28 visualizes and further explains the entire process.

[27] Stop words are words, usually the most common closed class functional words in a language that are usually removed from the text in order not to influence results. This technique is commonly used in the field natural language processing.

[28] Lemmatization is the process of turning words in a text into their base, dictionary form (usually referred to as a *lemma*).

[29] Jaccard overlap, also known as the Jaccard index or the Jaccard similarity coefficient for comparing sets is defined by the following equation: $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$.

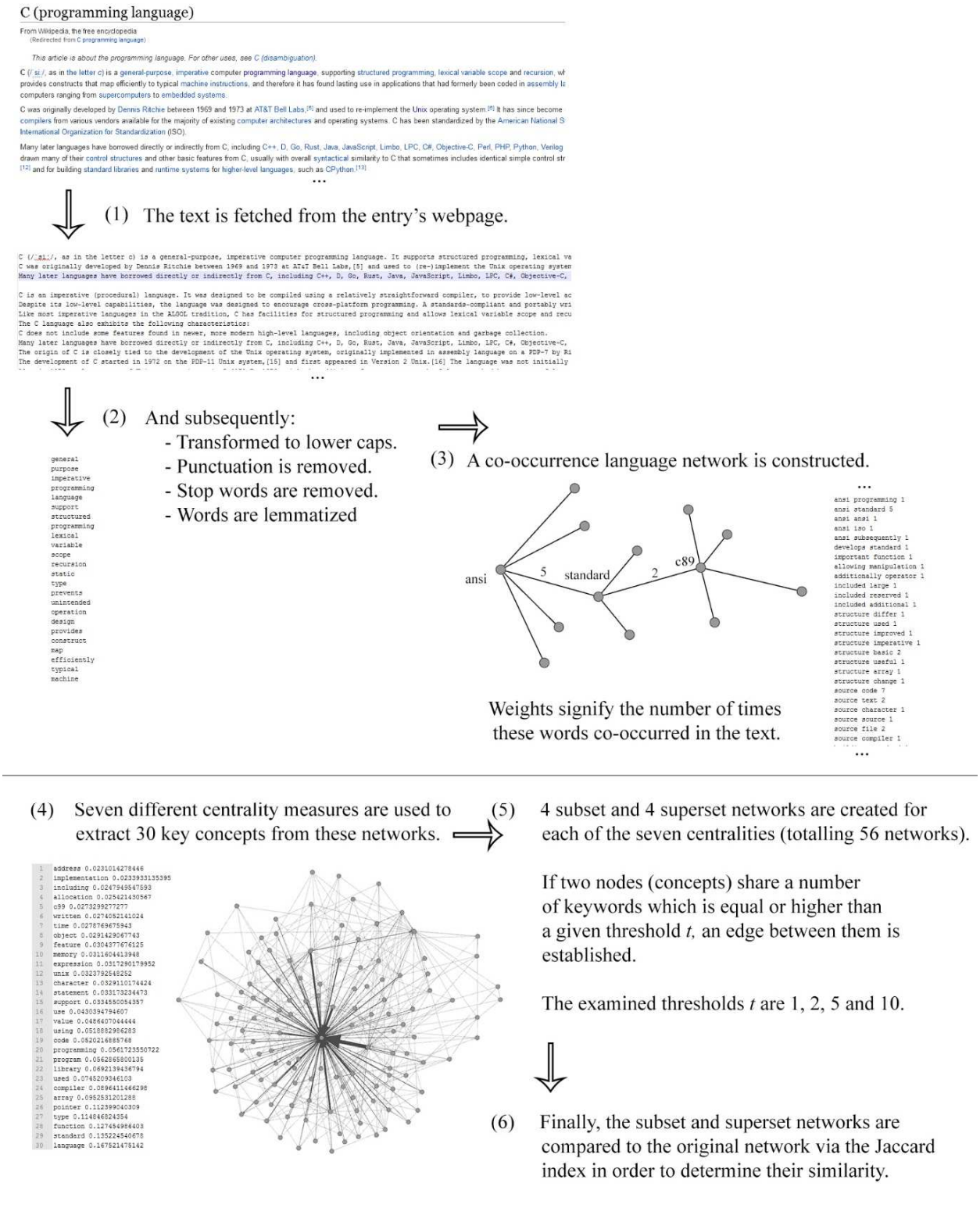


Figure 28 - Experiment 2 process

Figure 29 clearly shows how the networks are constructed and in which way subset and superset networks differ.

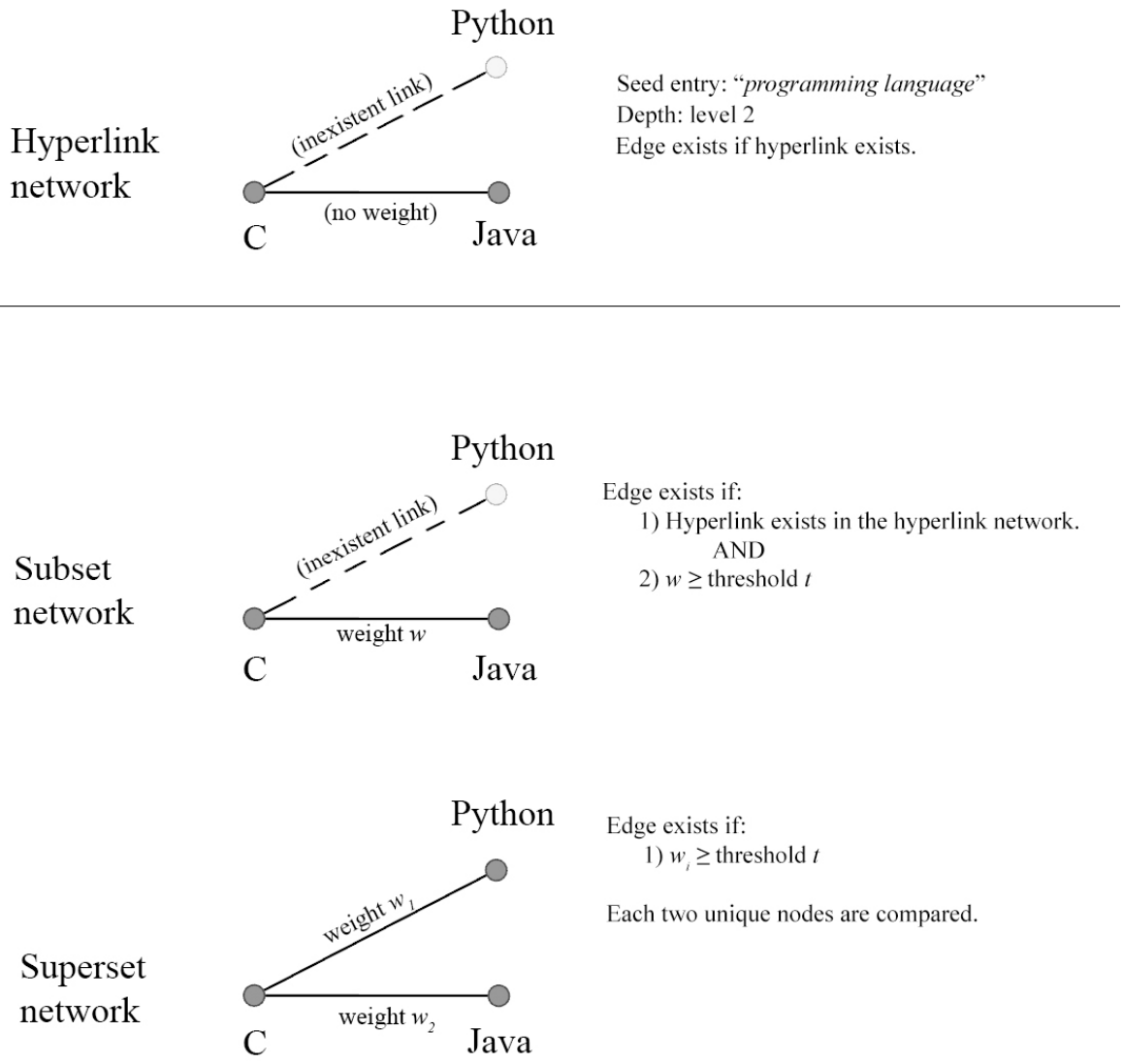


Figure 29 - Network construction in the second experiment

Most of the process (especially the calculation of centrality metrics) was again accomplished with the help of the NetworkX and BeautifulSoup (used for web scraping) software packages, but some additional software and resources were also used.

Lemmatization was done by using the NLTK Python toolkit (*Natural Language Toolkit*[30]) and the included Wordnet lemmatizer.

[30] See [36].

The list of stop words that I used in order to prepare the texts for the creation of co-occurrence networks was borrowed from Wikiminer (see [37]) and later expanded on my own with suitable stop words that were found missing from the original list.

Finally, the very removal of stop words and punctuation and the creation of co-occurrence networks was accomplished by using the LaNCoA (*Language Networks Construction and Analysis* [38]) Python package.

Due to the size and complexity of the Python scripts that were produced in order to arrive at the final results, they will not be included here.

5.2. Centrality measures - results and comparison

This section is used to divulge the results of the experiment and explain how they relate to one another.

The first thing I must mention is that the original hyperlink network had 302 nodes and 356 edges. As mentioned beforehand, its seed entry was "*Programming language*" and it was constructed on the basis of it being a level 2 network with the first 20 hyperlinks from each page being taken. The fact that not all links were used is due to computational complexity and bandwidth requirements of the task at hand.

The following seven tables look at how similar to the original hyperlink network are the networks ultimately produced by each of the employed centralities.

Each row in table represents one network. The first column specifies the type of network (subset or superset, see Figure 29 for detailed explanation). The second column specifies the threshold used in order to determine whether connections between nodes were established or not. The second two columns merely specify the basic metrics (number of nodes and edges).

The Jaccard overlap is a measure of similarity between the hyperlink network and the network at hand, whether it is a subset or superset network. It is calculated by the dividing the second to last and the last column. The second to last column represents the number of intersecting edges between the two networks. The last column represents the total number of unique edges in both networks.

As expected, the higher the threshold for link-establishment between edges (concepts), the lower the similarity between the networks.

Obviously the number of intersecting edges is the same in both subset and superset networks because the structure of the original network against which these are compared does not change.

Table 10 - Experiment results for closeness centrality

CLOSENESS CENTRALITY						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	287	338	0,947	337	356
Subset	3	218	255	0,713	254	356
Subset	5	127	140	0,39	139	356
Subset	10	6	3	0,008	3	356
Superset	1	303	30818	0,011	337	30835
Superset	3	303	9117	0,027	254	9218
Superset	5	267	2468	0,051	139	2685
Superset	10	86	87	0,007	3	440

Table 11 - Experiment results for betweenness centrality

BETWEENNESS CENTRALITY						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	291	343	0,96	342	356
Subset	3	246	284	0,795	283	356
Subset	5	182	208	0,581	207	356
Subset	10	41	34	0,095	34	356
Superset	1	303	35342	0,01	342	35356
Superset	3	302	13929	0,02	283	14002
Superset	5	289	4867	0,041	207	5016
Superset	10	133	322	0,052	34	644

Table 12 - Experiment results for eigenvector centrality

EIGENVECTOR CENTRALITY						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	284	334	0,935	333	356
Subset	3	207	240	0,671	239	356
Subset	5	110	121	0,337	120	356
Subset	10	12	7	0,02	7	356
Superset	1	303	32344	0,01	333	32367
Superset	3	302	8963	0,026	239	9080
Superset	5	256	2014	0,053	120	2250
Superset	10	81	80	0,016	7	429

Table 13 - Experiment results for degree centrality

DEGREE CENTRALITY						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	296	349	0,977	348	356
Subset	3	257	301	0,842	300	356
Subset	5	193	226	0,632	225	356
Subset	10	61	56	0,154	55	356
Superset	1	303	36920	0,009	348	36927
Superset	3	303	16197	0,018	300	16252
Superset	5	294	5931	0,037	225	6062
Superset	10	183	493	0,069	55	794

Table 14 - Experiment results for current-flow betweenness centrality

CURRENT-FLOW BETWEENNESS C. (RANDOM-WALK BC)						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	290	343	0,96	342	356
Subset	3	253	295	0,8257	294	356
Subset	5	178	207	0,58	206	356
Subset	10	47	0,4	0,103	37	356
Superset	1	303	35707	0,01	342	35721
Superset	3	303	14471	0,02	294	14533
Superset	5	282	5196	0,038	206	5346
Superset	10	156	380	0,053	37	699

Table 15 - Experiment results for current-flow closeness centrality

CURRENT-FLOW CLOSENESS C. (INFORMATION CENTRALITY)						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	288	340	0,952	339	356
Subset	3	252	294	0,823	293	356
Subset	5	183	211	0,59	210	356
Subset	10	65	56	0,154	55	356
Superset	1	303	34922	0,01	339	34939
Superset	3	301	14277	0,02	293	14340
Superset	5	284	4969	0,041	210	5115
Superset	10	180	454	0,073	55	755

Table 16 - Experiment results for communicability centrality

COMMUNICABILITY CENTRALITY						
Network	***t***	**Nodes**	**Edges**	**Jaccard overlap**	**|A∩B|**	**|A∪B|**
Subset	1	291	345	0,966	344	356
Subset	3	237	278	0,778	277	356
Subset	5	172	196	0,547	195	356
Subset	10	31	25	0,07	25	356
Superset	1	303	34261	0,01	344	34273
Superset	3	300	12576	0,022	277	12655
Superset	5	283	3655	0,051	195	3816
Superset	10	131	220	0,045	25	551

In order to compare the results given by the centralities amongst themselves, they were ordered in Table 17 with the only criterion by their respective Jaccard overlaps. It puts degree centrality narrowly in front, with the only stand-out centrality being eigenvector centrality with somewhat lower results altogether.

Table 17 - A comparison of centrality measures

PLACEMENT	CENTRALITY MEASURE
#1	Degree centrality
#2	Current-flow closeness centrality
#3	Current-flow betweenness centrality
#4	Betweenness centrality
#5	Communicability centrality
#6	Closeness centrality
#7	Eigenvector centrality

5.3. Keyword extraction

And finally, the following tables look at what were the actual top 10 defining keywords for several three different network nodes. These were selected arbitrarily and they are: *algorithm*, *language* and *object-oriented programming* (OOP).

Table 18 - Keywords extracted via closeness centraltiy

Closeness centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***object***
machine	***word***	***language***
program	***meaning***	***programming***
example	*example*	***class***
term	***english***	***method***
given	*different*	***oop***
time	***sign***	***data***
euclid	***form***	***support***
type	*used*	***feature***

Table 19 - Keywords extracted via betweenness centraltiy

Betweenness centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***object***
machine	***word***	***language***
number	***meaning***	***class***
example	***human***	***oop***
program	*called*	***programming***
instruction	***linguistic***	***method***
use	*example*	***data***
work	***sound***	*known*
turing	*different*	***software***

Table 20 - Keywords extracted via eigenvector centraltiy

Eigenvector centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***programming***
machine	***spoken***	***language***
program	***family***	***oriented***
known	***use***	***object***
usually	***word***	***support***
complete	***change***	***class***
instruction	*called*	***feature***
used	*used*	***oop***
turing	*area*	*typically*

Table 21 - Keywords extracted via degree centraltiy

Degree centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***object***
number	***word***	***class***
machine	***meaning***	***language***
instruction	***human***	***oop***
program	*called*	***programming***
turing	***sound***	***method***
example	*different*	*used*
problem	***linguistic***	***data***
use	*example*	*example*

Table 22 - Keywords extracted via current-flow betweenness centrality

Current-flow betweenness centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***object***
machine	***word***	***language***
number	***sound***	***oop***
turing	***meaning***	***method***
step	***human***	***class***
use	***linguistic***	***programming***
problem	*called*	*net*
example	*different*	*employee*
time	*example*	***software***

Table 23 - Keywords extracted via current-flow closeness centraltiy

Current-flow closeness centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm	***language***	***object***
number	***word***	***language***
machine	***meaning***	***programming***
instruction	***sound***	***class***
turing	***human***	***oriented***
program	*different*	***oop***
example	***sign***	***method***
use	*called*	***data***
problem	*example*	*used*

Table 24 - Keywords extracted via communicability centraltiy

Communicability centrality		
ALGORITHM	**LANGUAGE**	**OOP**
algorithm.	***language***	***object***
machine	***word***	***class***
instruction	***meaning***	***language***
program	***sign***	***programming***
problem	***sound***	*method*
example	*different*	*used*
following	*called*	*known*
language	***human***	***data***
step	*example*	***oop***

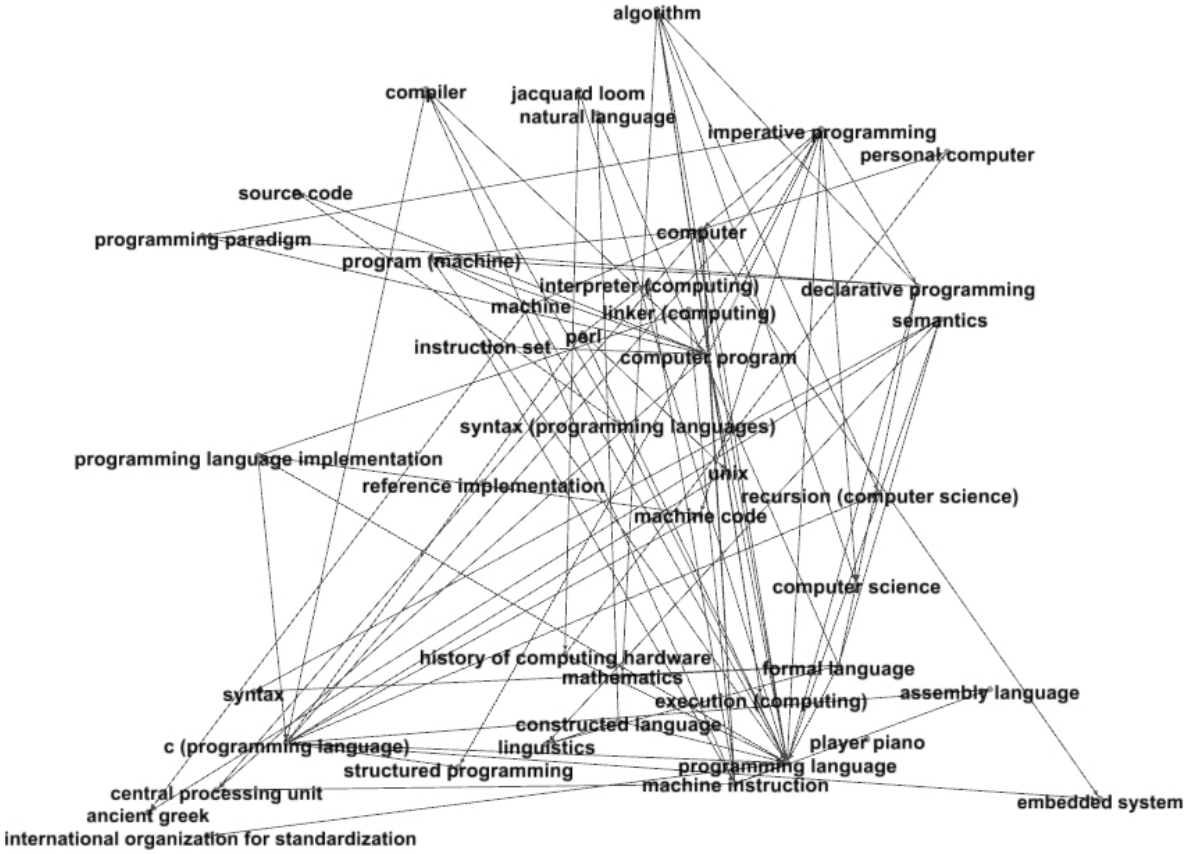


Figure 30 - Subset network (t=3) with keywords extracted via degree centrality[31]

[31] Only most central nodes are shown.

5.4. Discussion

In this experiment two things were accomplished.

Firstly, centrality measures were inspected and scored as tools for keyword extraction from Wikipedia texts. This has shown that many can be used as successful tools for this aim putting degree centrality narrowly in front as preferable, at least when the English language Wikipedia is concerned.

Secondly, the very keyword extraction process yielded significant results which enabled us to look at semantically related entities to various Wikipedia entries which are extracted both from the network structure and the texts themselves.

For instance, entries such as *number*, *machine*, *instruction*, *Alan Turing*, *program* and *problem* all obviously refer to an algorithm, a computer science concept often related with computer *programs*, *problem*-solving, *numerical* computation and calculability, *Turing machines* and so on.

In order to escape some functional words entering these results, a more comprehensive stop words list could have been employed or a filter for certain word types – such as verbs. The filtering could also have been done after the fact, but I chose to present the results as they were produced. We can see that the results given by most centralities are similar and rather satisfactory, since there are few words that could be marked as being semantically inappropriate, i.e. not belonging to the wider semantic context of a selected term.

Furthering this research could in turn enable a construction of a fully capable context extraction mechanism for any entity or term by looking at an increasingly wide semantic neighborhood. Another possible direction is missing link detection by contrasting the original link network with its subset and superset counterparts.

6. Conclusion

In this work I have looked at graph theory and its fundamental role in network science. Furthermore, I have explained and detailed some approaches and methods to the analysis and structure of complex networks and how these can be applied to the analysis of a large knowledge database and web encyclopedia – Wikipedia.

In the first experiment, *domain knowledge extraction*, centrality measures and community detection where used in order to look at whether similar concepts found on Wikipedia group into domain-based clusters. Also, general network metrics were employed to establish and confirm that, among other things, the structure of Wikipedia distinctly emulates the proposed small-world model.

In the second experiment, *keyword extraction using network centralities*, centrality measures were again used in order to extract keywords from the entries' texts and in order to look at how centrality measures fare amongst themselves when looking at the quality of Wikipedia's link structure contrasted with the semantic content found in the texts themselves.

The two experiments again confirm that using complex network analysis on Wikipedia can yield positive results. They show both the width and breadth of complex network science and its natural propensity to model existing structures as well as the wide-reaching and always changing landscape of Wikipedia as a dataset and knowledge database.

The section on related work shows some of the directions in which similar research may be headed. I believe that furthering such research can also greatly benefit the fields of machine learning, natural language processing and data mining, among others. It may allow us greater insight into the structures underlying human knowledge.

7. Literature and references

[1] R. Merris. *Graph Theory*, 1st edition. New York: Wiley-Interscience, 2000.

[2] E. W. Weisstein. "Graph." [Online]. Available: http://mathworld.wolfram.com/Graph.html [Accessed: 02-Jul-2015].

[3] "Graph (mathematics) - Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/Graph_%28mathematics%29#Mixed_graph [Accessed: 02-Jul-2015].

[4] M. Newman. *Networks: An Introduction*, 1st edition. Oxford; New York: Oxford University Press, 2010.

[5] A. Barabási. "Network Science." [Online]. Available: http://barabasi.com/networksciencebook/. [Accessed: 02-Jul-2015].

[6] N. Antulov-Fantulin, A. Lanlić, T. Šmuc, H. Štefančić and M. Šikić. "Identification of Patient Zero in Static and Temporal Networks: Robustness and Limitations." *Physical Review Letters,* vol. 114, June 2015, p. 248701.

[7] D. J. Watts and S. H. Strogatz. "Collective dynamics of `small-world' networks." *Nature*, vol. 393, no. 6684, Jun. 1998, pp. 440–442.

[8] Q. K. Telesford, K. E. Joyce, S. Hayasaka, J. H. Burdette, and P. J. Laurienti. "The Ubiquity of Small-World Networks." *Brain Connect*, vol. 1, no. 5, Dec. 2011, pp. 367–375.

[9] A.-L. Barabási and R. Albert. "Emergence of Scaling in Random Networks". *Science*, vol. 286, no. 5439, Oct. 1999, pp. 509–512.

[10] "Node degree distribution". [Online]. Available: http://www.network-science.org/powerlaw_scalefree_node_degree_distribution.html [Accessed: 11-Jul-2015].

[11] "Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia [Accessed: 12-Jul-2015].

[12] J. Giles. "Internet encyclopaedias go head to head." *Nature*, vol. 438, no. 7070, Dec. 2005, pp. 900–901.

[13] "Seven Bridges of Konigsberg." [Online]. Available: https://sites.google.com/site/sevenbridgesofkonigsberg/ [Accessed: 12-Jul-2015]

[14] W. W. Zachary. "An information flow model for conflict and fission in small groups." *Journal of Anthropological Research*, vol. 33, 1977, pp. 452–473.

[15] "The igraph library." [Online]. Available: http://cneurocvs.rmki.kfki.hu/igraph/screenshots2.html [Accessed: 13-Jul-2015]

[16] L. C. Santillán and A. C. Pérez. "Discovering epistemological axes for academic programs in computer science through network analysis." *Revista electrónica de computación, informatica, biomédica y electronica*, vol. 1, Nov. 2012.

[17] G. Rosenblatt. “What is Network Density – and How Do You Calculate It.” [Online]. Available: http://www.the-vital-edge.com/what-is-network-density/ [Accessed: 15-Jul-2015]

[18] M. E. J. Newman. “Modularity and community structure in networks.” *PNAS*, vol. 103, no. 23, Jun. 2006, pp. 8577–8582.

[19] S. Muff, F. Rao, and A. Caflisch. “Local modularity measure for network clusterizations.” *Phys. Rev. E*, vol. 72, no. 5, Nov. 2005, p. 056107.

[20] “Network centrality.” [Online]. Available: http://cs.brynmawr.edu/Courses/cs380/spring2013/section02/slides/05_Centrality.pdf [Accessed: 15-Jul-2015]

[21] “How Google Finds Your Needle in the Web's Haystack.” [Online]. Available: http://www.ams.org/samplings/feature-column/fcarc-pagerank [Accessed: 16-Jul-2015]

[22] “Centrality.” [Online]. Available: https://en.wikipedia.org/wiki/Centrality [Accessed: 16-Jul-2015].

[23] U. Brandes and D. Fleischer. “Centrality Measures Based on Current Flow.” *STACS 2005*, V. Diekert and B. Durand, Eds. Springer Berlin Heidelberg, 2005, pp. 533–544.

[24] M. Benzi. “Centrality and Communicability Measures in Complex Networks.” [Online]. Available: http://www.dsi.unive.it/isscn/wp-content/uploads/2014/07/Benzi1.pdf [Accessed: 18-Jul-2015].

[25] E. Estrada and N. Hatano. “Communicability in complex networks.” *Phys. Rev. E*, vol. 77, no. 3, Mar. 2008, p. 036111.

[26] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. “Network motifs: simple building blocks of complex networks.” *Science*, vol. 298, no. 5594, Oct. 2002, pp. 824–827.

[27] “Graphlets.” [Online]. Available: https://en.wikipedia.org/wiki/Graphlets [Accessed: 18-Jul-2015].

[28] O. Kuchaiev, T. Milenković, V. Memišević, W. Hayes, and N. Pržulj. “Topological network alignment uncovers biological function and phylogeny.” *Journal of The Royal Society Interface*, vol. 7, no. 50, Sep. 2010, pp. 1341–1354.

[29] “Clique.” [Online]. Available: https://en.wikipedia.org/wiki/Clique_%28graph_theory%29 [Accessed: 18-Jul-2015].

[30] “Cliques, Clusters and Components.” [Online]. Available: https://www.safaribooksonline.com/library/view/social-networkanalysis/9781449311377/ch04.html [Accessed: 19-Jul-2015].

[31] D. Quigley and A. Balmain. “Systems genetics analysis of cancer susceptibility: from mouse models to humans.” *Nature Reviews Genetics*, vol. 10, no. 9, Sep. 2009, pp. 651–657.

[32] B. Saha, A. Mandal, S. B. Tripathy, and D. Mukherjee. “Complex Networks, Communities and Clustering: A survey.” *arXiv*:1503.06277 [cs], Mar. 2015.

[33] “Detecting Communities Based on Network Topology : Scientific Reports.” [Online]. Available: http://www.nature.com/articles/srep05739 [Accessed: 19-Jul-2015].

[34] “The Louvain method for community detection in large networks.” [Online]. Available: https://perso.uclouvain.be/vincent.blondel/research/louvain.html [Accessed: 19-Jul-2015].

[35] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. “Fast unfolding of communities in large networks.” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, Oct. 2008, p. P10008.

[36] “Natural Language Toolkit.” [Online]. Available: http://www.nltk.org/ [Accessed: 21-Jul-2015].

[37] “Wikipedia Miner.” [Online]. Available: http://wikipedia-miner.cms.waikato.ac.nz/ [Accessed: 21-Jul-2015].

[38] D. Margan and A. Meštrović. "LaNCoA: A Python Toolkit for Language Networks Construction and Analysis". *IEEE 38th International Convention on Information and Communication Technology, Electronics and Microelectronics - (MIPRO 2015)*, Opatija, 2015, pp. 1961–1966.

[39] V. Zlatić, M. Božičević, H. Štefančić, and M. Domazet. “Wikipedias: Collaborative web-based encyclopedias as complex networks.” *Phys. Rev. E*, vol. 74, no. 1, Jul. 2006, p. 016115.

[40] V. Zlatić and H. Štefančić. “Model of Wikipedia growth based on information exchange via reciprocal arcs.” *EPL (Europhysics Letters)*, vol. 93, no. 5, Mar. 2011, p. 58005.

[41] A. Capocci, V. D. P. Servedio, F. Colaiori, L. S. Buriol, D. Donato, S. Leonardi, and G. Caldarelli. “Preferential attachment in the growth of social networks: The internet encyclopedia Wikipedia.” *Phys. Rev. E*, vol. 74, no. 3, Sep. 2006, p. 036116.

[42] F. C. Pembe and H. Bingol. “Complex Networks in Different Languages: A Study of an Emergent Multilingual Encyclopedia.” *Unifying Themes in Complex Systems*, Springer Berlin Heidelberg, 2008, pp. 612–617.

[43] Fang, Zheng, et al. “Wikipedia as Domain Knowledge Networks-Domain Extraction and Statistical Measurement.” *Proceedings of international conference on knowledge discovery and information retrieval (KDIR 2011)*, Paris, Oct 2011.

[44] A. P. Masucci, A. Kalampokis, V. M. Eguíluz, and E. Hernández-García. “Wikipedia information flow analysis reveals the scale-free architecture of the Semantic Space.” *PLoS ONE*, vol. 6, no. 2, Feb. 2011, p. e17333.

[45] J. Borge-Hothoefer and A. Arenas. “Semantic Networks: Structure and Dynamics.” *Entropy*, vol. 12, 2010, pp. 1264–1302.

[46] S. Šišović, S. Martinčić-Ipšić, and A. Meštrović. “Toward Network-based Keyword Extraction from Multitopic Web Documents.” *Proceedings of 6th International Conference on*

Information Technologies and Information Society (*ITIS2014*), Šmarješke toplice, Slovenia, 2014, pp. 18–27.

[47] S. Beliga, A. Meštrović and S. Martinčić-Ipšić. "Toward Selectivity Based Keyword Extraction for Croatian News." *CEUR Proceedings* (*SDSW 2014*), Vol. 1301, Riva del Garda, Trentino, Italy, 2014, pp. 1–14.

[48] S. Lahiri, S. R. Choudhury, and C. Caragea. "Keyword and Keyphrase Extraction Using Centrality Measures on Collocation Networks." *arXiv*:1401.6571 [cs], Jan. 2014.

[49] K. Nakayama, M. Pei, M. Erdmann, M. Ito, M. Shirakawa, T. Hara, and S. Nishio. "Wikipedia Mining Wikipedia as a Corpus for Knowledge Extraction." *WISE 2007, LNCS 4831*, 2007, pp. 322–334.

[50] M. Strube and S. P. Ponzetto. "WikiRelate! Computing Semantic Relatedness Using Wikipedia." in *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, Boston, Massachusetts, 2006, pp. 1419–1424.

[51] M. S. Han and E. E. Mon. "Computing Semantic Relatedness using Wikipedia Taxonomy by Spreading Activation." *International Conference on Advances in Engineering and Technology (ICAET'2014)*, 2014.

[52] K. Nakayama, T. Hara and S. Nishio. "Wikipedia Link Structure and Text Mining for Semantic Relation Extraction." *SemSearch 2008 (CEUR Workshop Proceedings)*, Jun. 2008, pp. 59–73.

[53] K. Nakayama. "Extracting Structured Knowledge for Semantic Web by Mining Wikipedia."

[54] S. P. Ponzetto and M. Strube. "Knowledge Derived From Wikipedia For Computing Semantic Relatedness." *Journal of Artificial Intelligence Research,* vol. 30, 2007, pp. 181–212.

8. Appendix

Index of figures

Index of tables