On a new metric to compare internal structures in biological networks

Razvan Valentin Marinescu
rvm10@imperial.ac.uk

January 2014

Contents

Chapter 1

Introduction

1.1 Abstract

Large biological data that has been made available in the past 20 years is infeasible for the humans to manually analyse. Several algorithms and analysis techniques have been introduced in order to process such data and extract critical information from biological networks. We have developed a novel metric that compares the local topological structure around a node belonging to such a network. When applied to biological networks such as protein-protein interaction networks or metabolic networks, this could tell us whether two proteins have a similar function or even identify key proteins in the metabolic chain. Moreover, it can also help us compare whole networks together and find out how similar or different they are with respect to certain properties. We hope that our technique would help scientists better understand biological and economic networks.

1.2 Motivation

Over the past few decades, major advancements in Genomics and Molecular Research technologies have made available large amounts of biological data that can help us better understand molecular processes. A large set of the data is in the form of biological networks, such as protein-protein interaction networks or metabolic networks. Analysing these networks can help scientists better understand the molecular interactions present in our body and can help aid drug discovery or treatment of various disorders. However, as these networks are too large to be analysed, computers are being extensively used to find patterns in the data. Bioinformatics and Computational Biology are therefore the two emerging fields that use computers to analyse biological information.

The past two or three decades have been an exciting time for Bioinformatics. From the computational side, several algorithms that align DNA sequences have been developed, such as BLAST[1] or Smith-Waterman[2]. These algorithms can not only quickly analyse large amounts of genetic data, but they can also be parallelised on a cluster of computers.

From a biological point of view, sequencing the human genome became reality for the first time through the Human Genome Project[3], a large collaboration of research institutes around the world. A project such as the Human Genome Project, as well as others have produced large amounts of network data that is waiting to be analised. Nowadays, there is a clear need of computer scientists and bioinformaticians to analyse these data and collaborate with biologists and chemists.

Our project presents a novel method of comparing biological networks and individual nodes, which can also be used to align networks or asses which random model fits the real data best. We believe that exploring new ways to analyse biological networks will help us shed a new light on

complex biological processes. This could further lead to an increased understanding of diseases such as cancer or cardiovascular disorders that are leading causes of death worldwide[4][5].

It is the case that when analysing networks of any kind, scientists generally look at global properties of networks, such as the degree distribution, average diameter or node centralities. These have so far been successfully used to identify enzymes or reactions that are crucial for the survival of organisms[6], model drug design trends[7] or modeling the world-wide airport network[8] However, not that much research has been done in exploring local properties of the network, that describe how a node interacts with its neighbourhood. Our technique of comparing nodes is based on local information, and it can thus be used to complement other established techniques.

Our technique builds upon work done by the Przulj group, which has developed a metric that is based on graphlets (i.e. small graphs) [9]. Their research has shown that the this metric has been successfully used to fit random network models to real world networks[10], uncover biological network function[11] and topologically align networks[12]. Our aim is to generalise this metric and use it help discover new biological functions. Given these previous results, we believe our technique will also be successfully used to fit random network models to real networks and topologically align networks.

1.3 Objectives

We would like to explore a new way of analysing biological networks by looking at local topology and structures around individual nodes. This would be performed by the means of a new metric called **Graphlet Cluster Vector** (GCV) that we will develop, which builds on previous work done by Natasa Przulj[9]. This metric measures the number of graphlets (i.e. small graphs with up to 5 nodes) that are found in the neighbouring subgraph of a node. For such a node, this metric quantifies the type of interaction its neighbours have with each other. As it will be explained later on, one can see it as a generalisation of the clustering coefficient.

Our first objective is to implement a fast algorithm that calculates the new metric for every single node in a given network. As some of the networks are large and the computation is quite intensive, we are also planning to paralellise the algorithm in order to enable it to run on multiple cores or even on a cluster of servers.

Our main objectives are to explore the properties of this novel metric and used it to compare not only individual nodes, but also whole networks together. First of all, we will be trying to find out what are the differences between the GCV and an older GDV metric that was developed by Tijana and Przulj[11]. This could help us find certain areas where it could be more efficient to use the new metric for biological analysis.

Afterwards, the next objective would be to compute an average GCV signature for the entire network, by averaging the individual GCV of every single node in it. This would enable us to perform early comparisons of the GCV between different network types:

- Real networks
    1. Protein-protein interaction networks
    2. Metabolic networks
    3. Trade networks
- Random networks
    1. Erdos-Renyi[13]
    2. Erdos-Renyi (arbitrary degree distribution)
    3. Geometric networks[14]

4. Barabasi-Albert (preferential attachment)[15]
5. Stickiness index-based[16]

Moreover, previous research has already shown that the old GDV metric has been successfully used to fit random network models to real world networks[10], uncover biological network function[11] and topologically align networks[12]. Therefore, our objective is to asses the performance of the new GCV metric on such problems and compare them with the results of the GDV metric. We are thus mainly interested to find out how to gather biological insights from the data we have. Apart from biological data, we would like to see how the metric performs on economic networks, which have a different structure. We will also be interested to evaluate a few other properties, such as robustness to noise.

Other extensions that could possibly be done are as follows:

- use the new GCV metric to get new insights from biological data
- explore the injectivity of the metric, by answering the following questions: Are there 2 nodes belonging to the same biological network that can have the same GCV result, even if the local structure around them is different? What is the probability of such a clash occurring?

Chapter 2

Background Research

2.1 Global Network properties

Global network properties give an overall view of the network, but are usually not detailed enough to capture complex topological characteristics. In the following sections we will present a few key global properties such as degree distribution, clustering coefficient and the average path length.

2.1.1 Degree Distribution

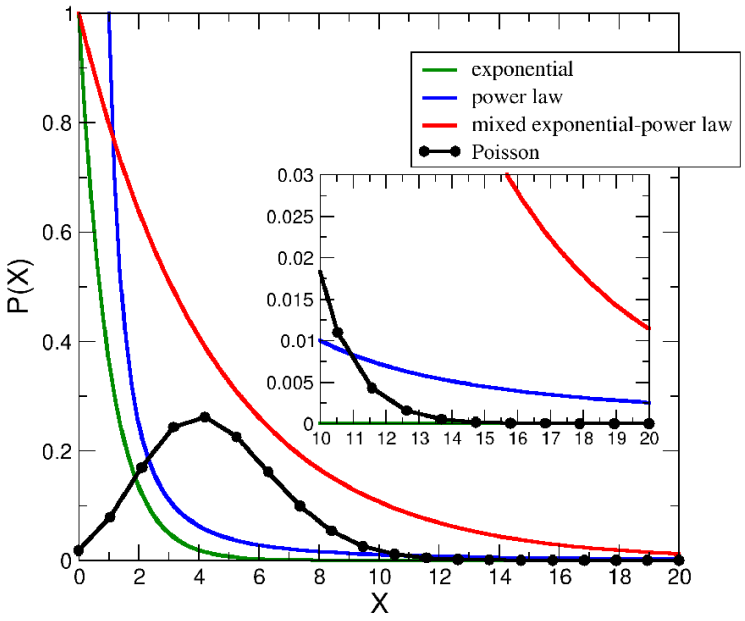


Figure 2.1: Graph depicting the most common classes of degree distributions: Poisson, Power-law and Exponential.

The degree distribution P(k) of a graph is the fraction of nodes in the network having degree k. Several current classes exist, with the most commonly-used ones being: Poison/Binomial, power-law and exponential (see fig. 2.1). A power-law degree distribution has a high number of nodes with low degree and a very small number of nodes with high degree, also called hub nodes.

Although many random graphs have a Poison degree distribution, it has been shown that many real networks have a power-law degree distribution instead. Such networks include the

Internet, transportation networks, anatomical circulatory networks, social networks and food webs.

2.1.2 Clustering Coefficient

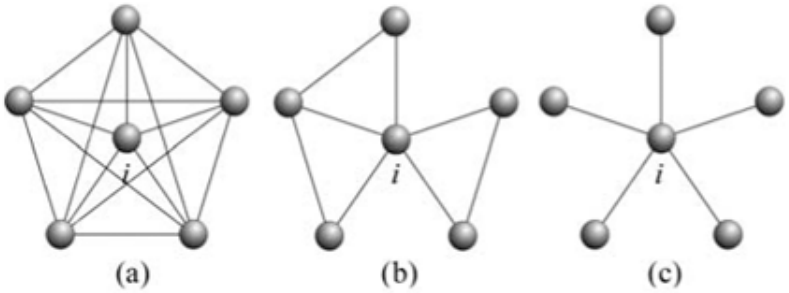


Figure 2.2: The clustering coefficient intuitively describes how connected are the neighbours of a particular node. In the above three scenarios, the custering coefficient decreases from (a) to (c). Image source[17]

The clustering coefficient is another important property of a graph that is used for data analysis and comparisons. It measures the tendency of nodes to cluster together.

Watts and Strogatz defined the clustering coefficient in the following manner: Consider a node n in a graph G that has k_n neighbours. The maximum number of edges between the neighbours of n is $\frac{k_n(k_n-1)}{2}$. Consider that only a fraction C_n of these edges are present in the set of neighbours of n. C_n can also be viewed as the probability of two neighbours of n being connected. This is then averaged against all the nodes in the graph and the final clustering coefficient C of graph G is obtained.

It has been shown that real networks have a high clustering coefficient. However, in the Erdos-Renyi graphs, the clustering coefficient is low when the probability p of connecting two nodes is also low. This makes these models unsuitable for modeling real data.

2.1.3 Average path length

The distance between two nodes is defined as the smalled number of links that have to be traversed to get from one node to another. The average path length of a network is therefore the average distance between any pair of two nodes.

Real networks have been shown to exhibit a small average path length. In the Erdos-Renyi graphs we have just mentioned, the average path length is large when the probability p of connecting two edges is small, and it gets smaller as p increases.

2.2 Local Network properties

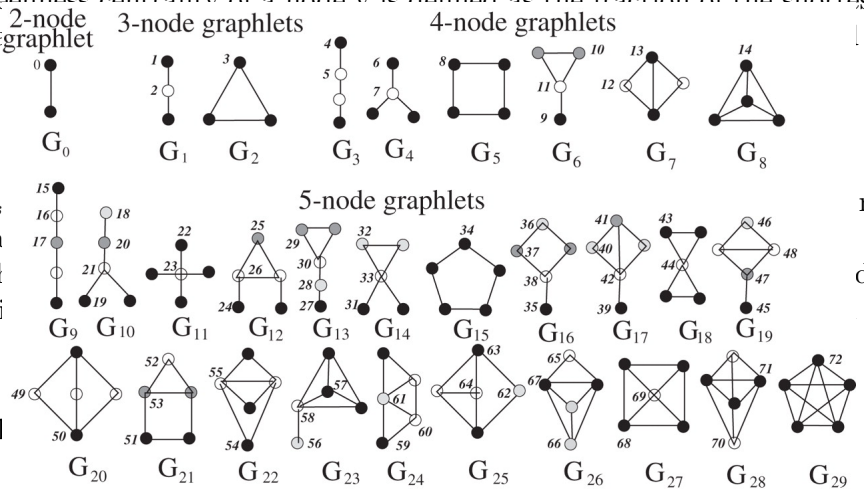
Local network properties capture detailed information about a local region in the network or even one single node. On the other hand, local properties cannot give an overall description of the network. In the next few sections we will present three main types of local metrics: Node centralities, *Graphlet Frequency Vector* and *Graphlet Degree Vector*

2.2.1 Node centralities

Another commonly used property for measuring the importance of a node in a network is the centrality. Several types of centralities include:

- Degree centrality: Measures the number of links that connect with the node. It is simply defined as the degree of the node.
- Betweenness centrality: Quantifies how many shortest paths pass through the node
- Closeness centrality: Measure how close the node is to the other nodes in the network.

The Betweenness centrality of a node v is defined as the fraction of the shortest paths from between two [illegible] as:



, where σ_s [illegible] node v, while σ_{st} is the tota[illegible]

On the otl[illegible] d the **farness** of a node whi[illegible] the network. Therefore the [illegible]

2.2.2 Gra[illegible]

Figure 2.3: Graphlets for sizes of 2, 3, 4 and 5 nodes. They are ordered in groups according to the number of nodes and numbered from the least-dense to the most dense. These are the graphlets that are generally counted when computing the GDV and GCV metrics. Source:[11]

Graphlets are small connected non-isomorphic induced subgraphs of a graph. They have been previously used by Natasa Przulj for developing some generalised measures of computing local topological metrics such as the Graphlet Distribution Vector (GDV).

For a given graph G, the *Graphlet Frequency Vector* can be calculated by counting the number of distinct graphlets of each type found in the network. From here, we can normalise these against the total number of graphlets in G to calculate the *Relative Graphlets Frequency Vector*. This is thus defined as:

$$GFV(G) = (F_1(G), F_2(G), ...F_{29}(G)) \tag{2.1}$$

where:

$$F_i(G) = -log\left(\frac{G_i}{\sum_{i=1}^{n} G_i}\right)$$

2.2.3 Relative Graphlet Frequency Distance

Using the Graphlet frequencies that have been previously defined, we can now compute a measure of disparity between two graphs by taking pairs of graphlet frequencies for each type and then summing their absolute difference. This is called the *Relative Graphlet Frequency Distance* (RGFD). More formally, if we consider two graphs G and H and we denote $F_i(G)$ and $F_i(H)$ to be the frequency of the i-th graphlet in G and H, then we compute the RGFD as:

$$D = \sum_{i=1}^{n} |G_i - H_i| \tag{2.2}$$

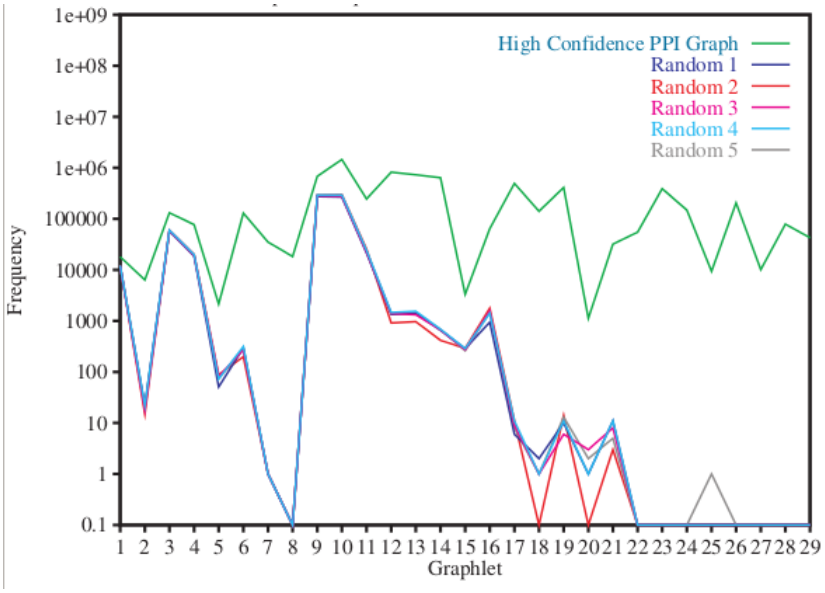


Figure 2.4: Example of a Graphlet degree vector for the PPI network of yeast *Saccharomyces cerevisiae*. As one can see from the graph, the GDV signature of the real network is considerably different from the ones of the random networks. The random networks have been generated using the Erdos-Renyi method. Source: [10]

2.2.4 Graphlet Degree Vectors

The Degree Distribution of a network calculates the number of nodes touching k edges for each value of k. However, we can generalise this concept by looking at the 73 automorphism orbits (see fig 2.3) and counting the number of nodes that touch a particular graphlet at a particular orbit. Finally, we get a spectrum of 73 *Graphlet Degree Distributions (GDDs)* measuring local properties of a network.

2.2.5 GDD Agreement

We are now trying to compare the spectra of 73 Graphlet Degree Distributions belonging to a graph G to the ones corresponding to another graph H. There might be several ways to perform this, but we will present the method used by N. Przulj in 2006[9].

Let G be a graph and let $d_G^j(k)$ be a sample distribution of the number of nodes in G touching orbit j (j = 1-73) k times. Therefore, d_G^j represents the j-th graphlet degree distribution (GDD). We scale d_G^j as:

$$S_G^j(k) = \frac{d_G^j(k)}{k}$$

in order to increase the contribution of the higher degrees. The reason for doing this is because most of the information is retained in the lower degrees, whereas the high degrees mostly contain noise[9]. Afterwards, the distribution is normalised against its total area:

$$T_G^j = \sum_{k=1}^{\infty} S_G^j(k)$$

giving the normalised distribution:

$$N_G^j(k) = \frac{S_G^j(k)}{T_G^j}.$$

The reason why we are normalising the distribution is because a large network would have more nodes that potentially touch orbit j and therefore a large area under the curve. Normalising it would make large and small biological networks comparable in terms of their GDD. Finally, for two graphs G and H and an orbit j, we define the distance $D^j(G,H)$ between their normalized j-th distributions as:

$$D^j(G,H) = \sqrt{\sum_{k=1}^{\infty}[N_G^j(k) - N_H^j(k)]^2}$$

The distance $D^j(G,H)$ is between 0 and 1, where 0 means that G and H have the same GDD for automorphism orbit j. Now, we invert the this distance in order to get the *j-th GDD agreement*:

$$A^j(G,H) = 1 - D^j(G,H)$$

where $j \in 0, 1, ..., 72$. Finally, the GDD agreement between the two networks G and H is the arithmetic mean of $A^j(G,H)$ over all j:

$$A(G,H) = \frac{1}{73}\sum_{j=0}^{72} A^j(G,H)$$

2.3 Random graphs

Random graphs are graphs that are usually generated using a random process. They are used in data analysis for comparing or aligning them against real networks. They can clarify the behaviour of real-world networks, such as the world-wide web or protein-protein interaction networks. Random graph models have been successfully used in various biological settings, such as: Network motifs[18], De-noising of protein-protein interaction network data[19] or Guiding biological experiments[20].

2.3.1 Erdos-Renyi graphs

The work on random graphs started from the influential publications of Erdos and Renyi in the 1950s and 1960s, while Edgar Gilbert published a similar model later on. Erdos and Renyi described the $\mathbf{G_{n,m}}$ model[13], while Gilbert described the $\mathbf{G_{n,p}}$ model[21]. The two methods they described were as follows:

- $\mathbf{G_{n,p}}$:We start with n disconnected nodes and given probability p. We then go through every pair of nodes and connect them with probability p.
- $\mathbf{G_{n,m}}$:We start with n disconnected nodes and a target of m edges. Afterwards, we randomly select m pairs of nodes and connect them.

Although these networks are very easy to generate, it was later found that real networks have a different structure that is different from the Erdos-Renyi graphs. More precisely, they have a different degree distribution and a low clustering coefficient. For the Erdos-Renyi $G_{n,p}$ graph, the degree distribution is binomial:

$$P(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \tag{2.3}$$

which can be approximated with a Poisson distribution for a large n:

$$P(k) = \frac{z^k * e^{-z}}{k!} \tag{2.4}$$

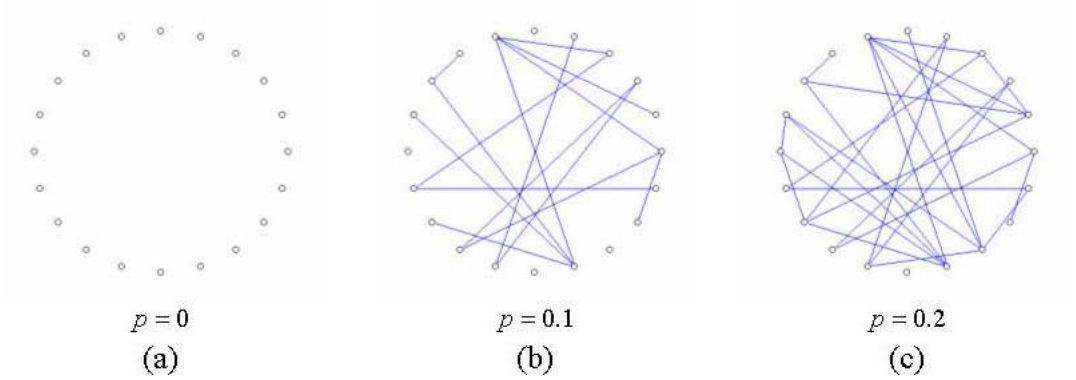


Figure 2.5: Example of three an Erdos-Renyi random graphs generated using the $G_{n,p}$ method. The three different graphs differ according to the probability p of connecting one pair of nodes: (a) The graph is completely disconnected. (b) The graph is sparsely connected as the probability p is low (c) The graph becomes more dense as p increases to 0.2. Source:[22]

2.3.2 Erdos-Renyi with the degree distribution of a real network

As we have previously seen, the degree distribution of an Erdos-Renyi graph does not match real data. We will now present a method for constructing an Erdos-Renyi network that preserves the degree distribution of a real network.

We start with n disconnected nodes. Each node is assigned a number of stubs according to the degree distribution of the real network that is being modelled. A stub is simply a slot belonging to a particular node from where an edge can be connected. Afterwards, edges are created only between random pairs of nodes with available stubs. After an edge is created, the number of stubs left available at the nodes that were just connected is decreased by one. Moreover, edges between one node and itself are not allowed.

This "stubs method" allows us to create ER (Erdos-Renyi) networks that have a power-law degree distribution and a small average path length. Unfortunately, they still have a low clustering coefficient.

2.3.3 Scale-free networks

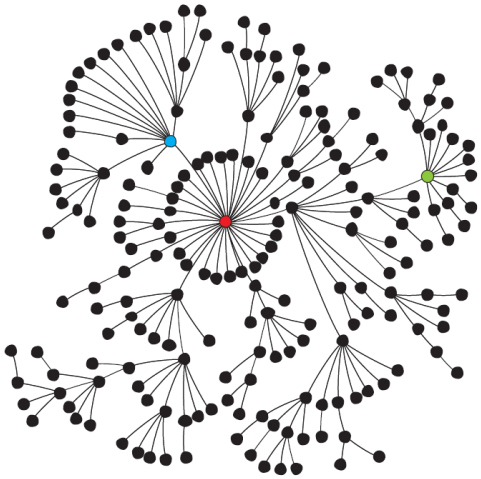
Scale free networks are networks that normally exhibit a power-law degree distribution (see fig 2.6). That is, $P(k) = k^{-\gamma}$, where $P(k)$ is the fraction of nodes having degree k. It is currently believed that many networks such as the world-wide web, social networks or biological networks exhibit a power-law degree distribution.

Barabasi-Albert model

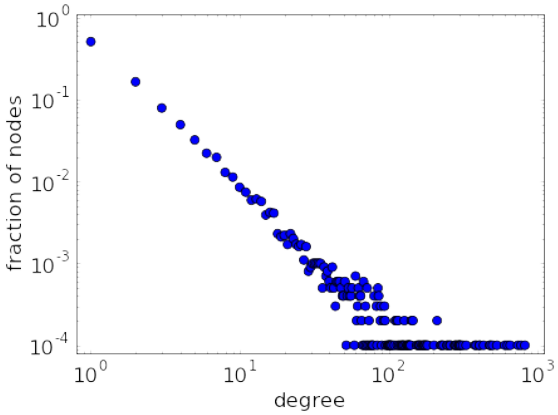
There are several proposed ways in which scale-free networks can be generated. The Barabasi-Albert model is one such technique that uses the *preferential attachment* mechanism, with which nodes of high degree have a high probability of receiving even more connections.

In order to construct a network using the Barabasi-Albert method, we start with an initial connected network of m_0 nodes. New nodes are consecutively added to the network one at a time. Each one of them is connected to $m <= m_0$ target nodes with a probability that is proportional to the degree of the target nodes. Formally, the probability p_i that the new node is connected to node i is:

$$p_i = \frac{k_i}{\sum_j k_j} \tag{2.5}$$



(a) Example of a scale-free network. Note the large number of nodes of small degree at the periphery of the network, while the number of hub nodes is very small.



(b) The degree distribution of a scale-free network. As the degree of the nodes gets larger, the fraction of nodes decreases exponentially. Notice the logarithmic scale on the Y axis. It has been observed that many real networks exhibit a scale free distribution.

Figure 2.6: Scale-free networks and degree distribution

where k_i is the degree of node i, while the sum is over all the nodes j that already existed in the network when the new node is added. Because of the preferential mechanism, heavily linked nodes (also called hub nodes) tend to quickly accumulate links, whereas nodes with a low degree are unlikely to be chosen. It has also been shown that the starting network heavily influences the properties of the resulting network[23].

2.3.4 Geometric graphs

Geometric graphs are generated by fixing a certain metric space and using metrics such as geometric distance or radius to connect edges together. A metric space is a space that has a distance norm associated to it, such as: the Euclidean distance, Chessboard distance or Manhattan distance.

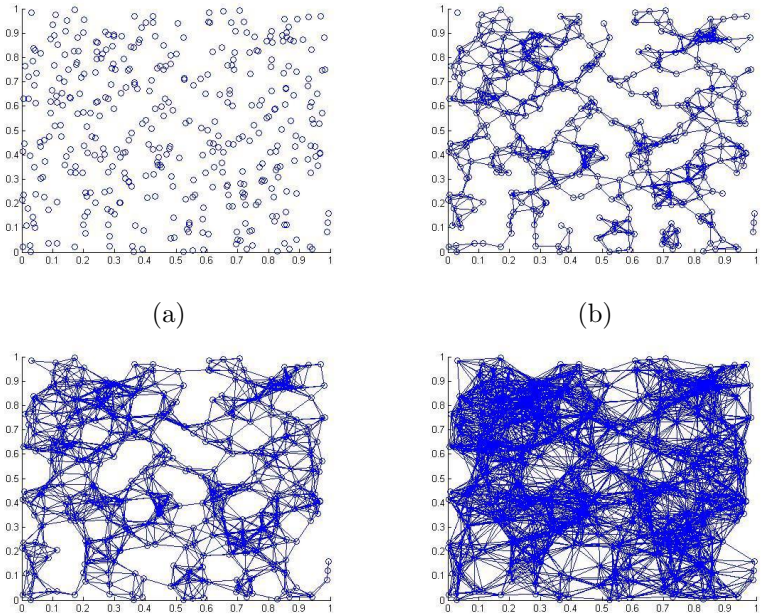
Such a network is generated in the following manner:

1. Choose a metric space and place nodes within the space using a uniform random distribution.
2. If any nodes are within distance d from each other, then connect them with an edge.
3. d needs to be chosen so that the end number of edges matches the network that is modeled.

2.3.5 Stickiness index-based graphs

Przulj et al. have proposed in 2006 a simple random graph model that inserts a connection according to the degree, or 'stickiness', of the nodes involved[16]. This model has been inspired from analysing protein-protein interactions and is based on two assumptions:

1. A node having a high degree, or stickiness, implies that the corresponding protein has many binding domains and/or its binding domains are commonly involved in interactions.
2. A pair of proteins is more likely to interact, or share complementary binding domains, if they both have a high degree/stickiness. On the other hand, if one or both of them have a low stickiness index, they are less likely to interact. Thus, the product of their stickiness values can be used as the probability of connecting the nodes.



(a) (b)

(c) (d)

Figure 2.7: Geometric networks for increasing distance d starting from 0. When $d = 0$ in graph (a), the nodes are all disconnected from each other. As d increses in graphs (b - d), the number of connections in the network also increases proportionally. When using a Geometric network to model a real network, one would normally use a value of d that would yield a similar number of edges as the real network.

Considering the above assumptions, a stickiness based random graph can be constructed as follows:

1. We start with a network of n nodes each having a degree deg_i sampled from a degree distribution of our choice
2. For each node i, we compute the stickiness index $\theta_i = deg_i / \sqrt{\sum_{j=1}^{N} deg_j}$. Note that $0 \leq \theta_i \leq 1$
3. For each pair of nodes (i, j), we connect them with probability $\theta_i \theta_j$

2.3.6 Random graph Comparisons

As can be clearly seen in table 2.1, real networks normally have a power-law degree distribution, high clustering coefficient and a small average path length. In terms of degree distribution, only Erdos-Renyi (with a custom degree distribution), Barabasi-Albert and Stickiness-based random networks have a power-law degree distribution, like that found in real networks. However, it must be noted that although most of the real networks have a power-law degree distribution, this is still a matter of research. For example, it has been shown that Interactome network can be better modeled with a Geometric network, instead of a Scale-free network having a power-law degree distribution[10].

On the other hand, only the Geometric and the Stickiness based models have a high clustering coefficient. This is again something which has been observed in most of the real networks such as social networks or biological networks. Finally, most of the networks have a small average path length. It can therefore be noted that the Stickiness based network is was the most successful at modeling real-world phenomena with respect to these three properties. However, there might be other network properties, such as various node centralities[24] or *Relative*

Comparison of real networks versus randomly generated networks

Model	Degree Distribution	Clustering coefficient	Average path length
Real networks	Power-law	High	Small
Erdos-Renyi	Poisson	Low	Large (for small p)
Erdos-Renyi - DD	Power-law	Low	Small
Barabasi-Albert	Power-law	Low	Small
Geometric (uniform)	Poisson	High	Small
Stickiness based	Power-law	High	Small

Table 2.1: As one can observe from the table above, the early models such as Erdos-Renyi are not suitable for modeling real networks according to these metrics. On the other hand, other random models such as the Stickiness based satisfy all the three criteria. Nonetheless, it might still be different from real networks with regards to other properties, such as node centralities[24].

Graphlet Frequency Agreement, with can also be employed to assess the suitability of the random networks. Identifying which of these properties can best compare various types of networks is still an open problem in Network Analysis.

2.4 Measuring Correlation

In previous sections we have presented two main methods for calculating how closely two GDV vectors match: *Relative Graphlet Frequency Distance* and *Graphlet Degree Distribution Agreement*. Although these are quite efficient when comparing two networks, we can devise a better metric by using some statistical techniques and a correlation coefficient such as *Pearson's product-movement correlation coefficient* or *Spearman's rank correlation coefficient*.

2.4.1 Pearson's product-movement correlation coefficient

Given two random variables X and Y from a population, *Pearson's correlation coefficient* or sometimes called *Pearson's population correlation coefficient* is defined as the ratio between the covariance of X and Y and the product of their standard deviation. It was introduced by Karl Pearson and it is based on a similar idea by Francis Galton in the 1880[25][26]. It is formally defined as:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[X - \mu_X]E[Y - \mu_Y]}{\sigma_X \sigma_Y}$$

,where $\rho_{X,Y}$ is Pearson's correlation coefficient between random variables X and Y, $cov(X,Y)$ is the covariance, while σ_X and μ_X are the standard deviation and the expectation of X.

Pearson's correlation coefficient can also be applied to a sample from a given population, in which case it is called the *sample Pearson's correlation coefficient* and is commonly denoted by r. This can be calculated by using sample estimators for the covariance and standard deviation in the formula above. The formula for the *sample Pearson's correlation coefficient* is:

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{2.6}$$

The values of both the sample and the population variants of Pearson's correlation coefficients are between -1 and 1. Sample data points that have exactly 1 or -1 as their correlation coefficient will lie on a straight line. Moreover, Pearson's correlation coefficient is symmetric, because:

$$corr(X, Y) = corr(Y, X)$$

where corr(X,Y) is defined as the correlation between random variables X and Y.

Pearson's distance

Given a correlation coefficient $\rho_{X,Y}$, a distance metric called the Pearson's distance can be derived as follows[27]:

$$d_{X,Y} = 1 - \rho_{X,Y}$$

It should be noted that because Pearson's correlation coefficient lies between -1 and 1, then the Pearson's distance will have a value between 0 and 2.

Interpretation

Several researchers have provided guidelines into how to interpret the size of the correlation coefficient[28] (equation 2.6). However, interpretation is highly dependent on the context of the problem. For example, a correlation of 0.8 might be low if one verifies physical laws using measurements made with high-precision instruments, but it might be considered high when applied to the analysis of social networks, because of other hidden factors.

2.4.2 Spearman's rank correlation coefficient

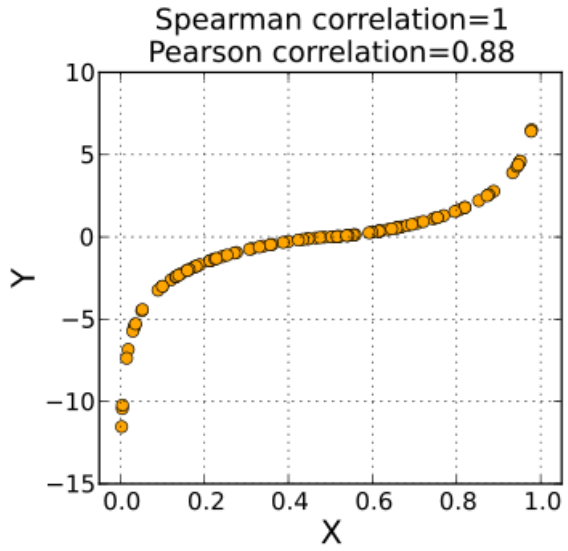


Figure 2.8: Spearman's correlation coefficient is measuring how well the dependence between two variables can be modeled using a monotonic function. A Spearman's correlation of 1 can result even when the data points are not linear, as long as they are monotonically related. Source:[29]

The Spearman's rank correlation coefficient or Spearman's rho, named after Charles Spearman, is a non-parametric estimator of the statistical dependence of two random variables[30]. It intuitively measures how well the dependence between two variables can be measured using a monotonic function. It is normally computed in two steps:

1. The ranks x_i, y_i of each of the data points in X and Y are calculated.

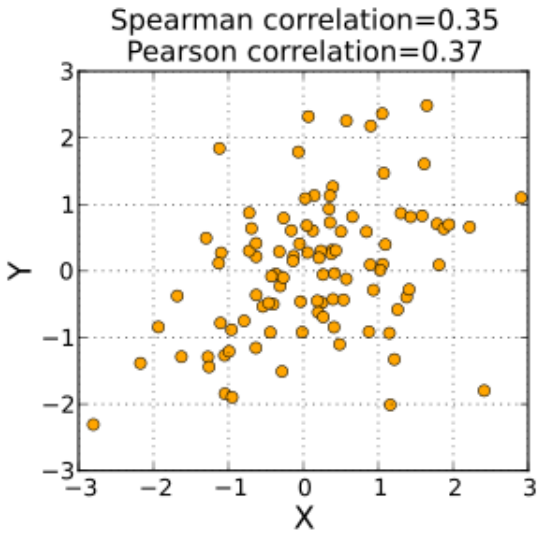


Figure 2.9: When the data points are evenly spread, both the Pearson's and the Spearman's correlation coefficients will be low (0.35 and 0.37 respectively). Source:[29]

Computing Spearman's ranks of the data points

Variable X_i	Position in ascending order	Rank
0.3	1	1
0.6	2	2
1.2	4	$\frac{4+5}{2}$
1.2	5	$\frac{4+5}{2}$
0.8	3	3
1.9	6	1

Table 2.2: The data is initially sorted in ascending order. If the data point is unique, then the rank is simply the position in the ordered list. Otherwise, the rank is computed as the average of the positions of all the data points with the same value.

2. The Spearman's rank correlation coefficient is computed by applying Pearson's correlation coefficient on the previously computed ranks of the data points[31].

The calculation of the ranks is best illustrated in table 2.2. After the ranks x_i, y_i of each data point is calculated are calculated, the Spearman's correlation coefficient is computed using the Pearson's correlation coefficient as follows:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.7}$$

Spearman's correlation coefficient is considered non-parametric in the sense that one does not need to know any prior on the X and Y random variables, as it does not require knowledge (i.e. the parameters) of the joint probability distribution of X and Y.

2.4.3 Computing the GDV correlation matrix of a network

We can use the Pearson's or Spearman's correlation coefficient previously described to compute a GDV correlation matrix for a given network in the following manner:

1. We compute the Graphlet Distribution Vector (GDV) for every node in the input network

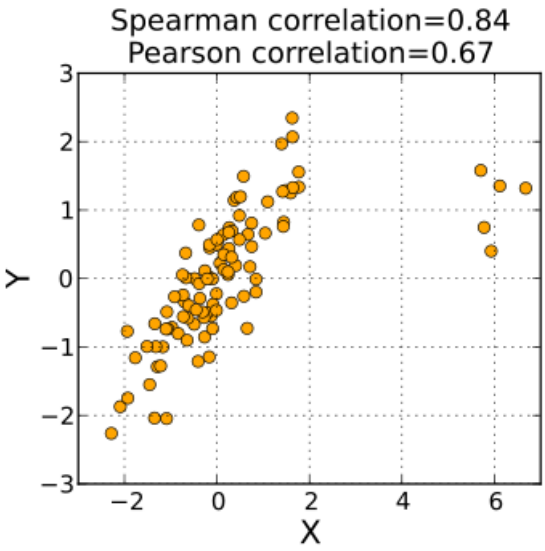


Figure 2.10: Spearman's correlation coefficient is more robust to outliers than Pearson's, because it projects the data points to their ranks. As a result, Spearman's correlation coefficient is much higher than Pearson's (0.84 vs 0.67) Source:[29]

2. We then construct samples $S_i, i \in 1, 2, 3, .., 29$ containing all the frequencies of the graphlet of type i found in the GDVs of the nodes.

3. We compute the Pearson's (or Spearman's) correlation coefficient of each pair of samples (S_i, S_j) and we put them in the 29x29 correlation matrix C_{ij}.

The newly obtained graphlet correlation matrix will be symmetric with respect to the main diagonal, as Pearson's correlation coefficient is also symmetric. In order to display such a matrix, we can use a heat map, with blue representing -1 and red representing 1.

Given two matrices from two different networks G and H, we can then calculate the distance between them by performing pairwise-subtractions of the elements:

$$D(G, H) = \sum_{i,j} (G(i, j) - H(i, j))^2$$

Note that the computed distance is always greater than 0. Using the distance, the *Graphlet Correlation Matrix Agreement* between G and H is defined as:

$$Agreement = 1 - D(G, H)$$

One advantage of using such an Agreement based on Pearson's correlation coefficient is that it has been shown to be more robust to noise in the network data[32].

2.5 Canonical Correlation Analysis

Canonical Correlation Analysis is a statistical method of analysing interdependence between two random variables X and Y. The method was first introduced by Harold Hotelling in 1936[33] and it has been used for analysing and interpreting data in various fields including Psychology[34], Marketing[35] and Operations Research[36].

Given two random variables X and Y, canonical correlation analysis (CCA) aims to find vectors a and b such that the correlation $\rho = corr(aX, bY)$ is maximised.

Chapter 3

Project Plan

3.1 Main Objectives

The aim of the project is to implement this *Graphlet Cluster Vector* (GCV) signature and explore its properties. We are interested in finding out to what degree it is different from the older *Graphlet Degree Vector* (GDV) developed by Przulj et al[9]. The main objectives that we have set are the following:

1. Implement the algorithm for calculating the GCV signature
2. Parallelise it in order to make it speed up analysis. This is particularly important when analysing large networks, such as the protein-protein interaction networks
3. Implement an algorithm to compute the average network GCV
4. Compute the average GCV of different networks and compare them with each other. Find out whether this metric can be used to distinguish between them.
5. Generate random networks from different models (Erdos-Renyi, Geometric, Barabasi-Albert, Stickiness based) using the real networks as models.
6. Calculate the GCV of the generated random networks and compare the results against the real networks. Compute the Graphlet frequency agreement and find out which random network fits the real data best according to the GCV metric.
7. Generate more instances of the random networks (around 10 for each model), compute their GCV and then plot the standard deviation of the GCV

3.1.1 Extensions

Accomplishing all the these objectives would be enough to get a clear understanding of the properties of the GCV signature. If time allows, a more comprehensive analysis can be peformed by doing several extras:

1. Use this metric to align two networks with each other using the GRAAL algorithm[37]
2. Explore the injectivity of the metric, by answering the following questions: Are there 2 nodes belonging to the same biological network that can have the same GCV result, even if the local structure around them is different? What is the probability of such a clash occuring?
3. Find out the evolution of world trade over time by analysing different trade networks from successive years. We can compute the GCV of the trade networks from successive years and find out what kind of change in patterns we get when:

- An economic crisis occurs: 2007 submortgage economic crisis, Eurozone crisis
- A major global political and economic shift occurs: Velvet Revolutions of Eastern Europe, Arab Spring
- Free trade areas are created

3.2 Work done so far

I have started working on this project in April last year, immediately after we started our industrial placements. However, I could not dedicate too much time on it, for I was working full time for my industrial placement. Nonetheless, I managed to do some background reading and to explore the mathematical model behind the graphlet counting procedure. More precisely, I spent a considerable ammount of time trying to devise a more precise normalisation procedure for the Graphlet Cluster Vector. More precisely, I tried to find out what is the maximum number of graphlets of each type one could have in a graph of n nodes, in order to use these numbers for normalisation. However, this turned out to be a very complex mathematical problem that I could not practically solve. The reason for this was because it is really hard to:

- find out what the ideal network architecture is in which one can get the maximum number of graphlets of a particular type
- mathematically find out the maximum number of graphlets in such an ideal architecture as a formula of N, the number of nodes in the network.

Therefore, we have decided to only perform a simple normalisation, by dividing the frequencies of each graphlet types against the sum of all frequencies.

So far, I have managed to do the following:

- Implement the algorithm which calculates the GCV for a particular node in the network
- Parallelise the computation over several cores
- Implement an algorithm to compute the average network GCV
- Compute the average GCV of different networks and compare them with each other.
- Generate random networks from different models (Erdos-Renyi, Geometric, Barabasi-Albert, Stickiness based) using the real networks as models.
- Calculate the GCV of the generated random networks.

3.3 Future milestones

The milestones for the next key tasks are described below:

- **30 February**: Calculate the Relative Cluster Frequency Distance (RCFD) between random networks and real networks
- **30 March**: Generate 10 network instances for each random model. Calculate the average GCV and plot it along with the standard deviation.
- **14 April**: Calculate the Pearson's correlation coefficient between pairs of column vectors of the GCV matrix of a network. Derive a 30x30 matrix of correlation pairs that is used as a global signature of the network. Given 2 such matrices, we can compare then and find out how similar two networks are.

- **15 May**: Use the GCV and the correlation matrix to derive biological or economic insights from the network data. For example, we can analyse gene networks and transfer function from one gene to another by performing alignment.
- **30 May**: Use the GCV to align two networks with each other using the GRAAL algorithm[37]. Compare the performance with that of the old GDV signature.
- **15 June**: Perform a time-dependent analysis of trade networks. Find out the RCFD between trade networks at different points in time.
- **27 June**: Write the final report

Chapter 4

Design and Implementation

4.1 Mathematical model

We shall now introduce the new metric that can be seen as a generalisation of the Graphlet Frequencies described above. This metric, which we shall call the Graphlet Cluster Vector (for reasons that will soon become obvious), is a key element of this project. As this is a novel metric, we would like to explore its properties and compare it to the Graphlet Frequency Vector and Graphlet Degree Distribution described above.

For a particular graph G and node n in G, we denote by S_n the set of neighbouring nodes of n and by S_n^i the number of graphlets of type i in S_n. Now, we consider the *Graphlet Cluster Vector* to be:

$$GCV(n) = \left(F_n^1, F_n^2, ...F_n^{29}\right)$$

where

$$F_n^i = \frac{S_n^i}{\sum_{i=1}^{n} S_n^i} \tag{4.1}$$

The new Graphlet Cluster Vector (GCV) metric is therefore generalising the GDV so that it includes information from the cluster of neighbouring nodes.

4.1.1 Relative Cluster Frequency Distance

Similarly to how the Relative Graphlet Frequency Distance is calculated, we will now define a similar measure for the distance measure for the new GCV signature. If we consider two nodes p and q in graph G and we denote F_p^i and F_q^i) to be the frequency of the i-th graphlet in nodes p and q, then we compute the *Relative Cluster Frequency Distance* (RCFD) as follows:

$$RCFD(p,q) = \sum_{i=1}^{n} |F_p^i - F_q^i| \tag{4.2}$$

4.2 Implementation of the Graphlet Cluster Vector

After the first stage of the project has been done, which was concerned with the mathematical model behind the Graphlet Cluster Vector, we started implementing the model in the C++ programming language. The two main reasons for implementing the algorithm in C++ were:

1. Dr. Przulj's research group already had a C++ function that was able to count the number of graphlets in a given graph. We were therefore able to leverage that code.

2. As, C++ is a compiled language that does not run in a virtual environment, it is computationally-speaking really fast. Since our algorithm was meant to execute very intensive calculations on large biological networks, it was therefore a good decision to use C++.

The C++ file that I was given from Przulj's group, called ncount.cpp, was used to count both the number of graphlets in a graph and also the number of automorphism orbits that nodes would touch. I also received three types of networks to test my algorithm on:

1. Protein-protein interaction network: Such networks describe the physical contacts established between two or more proteins as a result of biochemical processes or electrostatic forces. In this network, nodes would correspond to proteins and would get connected by an edge if there is a direct interaction between the underlying proteins.

2. Metabolic network: These networks describe the complete set of metabolic and physical processes that determine the physiological and biochemical properties of a cell. As such, these networks are composed of metabolic pathways as well as regulatory interactions that guide the chemical reactions taking place.

3. Trade network: This network describes the amount of economic trade that has been taking place in 2010 between major countries around the world. For each pair of countries, a number was given describing the trade volume that took place between them in 2010. This network would be ideally modeled using a weighted network. However, since our algorithm works only on unweighted networks, we had to trim down the edges between two countries that traded less than a certain volume of goods.

A third and last script that I was given was used to convert the given networks to a file format called LEDA, that is easy to be read and processed by the graphlet counting function.

4.2.1 Node-based Graphlet Cluster Vector

I started writing the code by first modifying the graphlet counting function (from ncount.cpp) and removed the unnecessary code that was dealing with automorphism orbits. Afterwards, I realised that the function was still hard to work with, for it was way too long and seemed to resemble a 'God-function' that was responsible for everything: reading from the input file, parsing it, building an efficient data structure to store the input in, counting the graphlets and writing to the output file. I therefore decided to split it up into modules according to their responsibility, concepts I learned from the Software Engineering course in the second year.

Afterwards, I started writing the code that would iterate over each node in the input graph and extract the neighbouring subgraph (i.e. the graph containing all the neighbours of a particular node plus the edges between them.). It is worth mentioning that, for a given node n, the neighbouring subgraph of n does not contain n itself, nor any of the edges between n and its neighbours.

This part of the code took a while to implement mainly because I had to deeply understand how the data structure what was used to store the graph worked. The programmers who implemented the initial graphlet-counting function were using a specially-optimised data structure that was storing the network both in an adjacency matrix and in an adjacency list at the same time. This turned out to be a good idea, because using both the list and the matrix forms allowed for many operations to be executed in O(1) time:

- The adjacency matrix allowed one to check whether two arbitrary nodes are connected in O(1) time.

- The adjacency list allowed one to get the list of all the neighbours of a node in O(1) time. This was especially useful for my extended function, where I needed to extract the neighbourhood of a node in order to count the number of graphlets in it.

After I understood the underlying concepts behind the optimised data structure, I implemented the neighbourhood extraction part of the algorithm and unit tested it using some manually crafted network examples of up to 20 nodes. Once the neighbourhood extraction phase was complete, I passed the neighbourhood to the graphlet counting function (ncount) and I got a signature made of 29 frequencies, one for each graphlet type. From here, it was simple to repeat the operation over all the nodes in the input network.

4.2.2 Parallelisation

Soon after I started testing the newly-written algorithm on the large biological networks, I realised that the algorithm was taking hours to finish calculating the GCV for every node in the network. Therefore, I have decided to paralellise the code across multiple cores by forking different threads. I have decided to do this as an initial improvement, and if there was need for even more parallel processing, I would later on extend it to run on a cluster of computers.

The most effective way to parallelise was to split the node set into N chunks and then have each thread compute GCV signatures for all the nodes from the corresponding chunk. The advantae of forking different threads is that they have access to the same shared memory, so all of them could access the same input data easily. At the end, I made it work so that the desired number of threads could be passed as a command line argument, and changed the Makefile accordingly.

Each thread would write the output to its own file, which had the thread id at the end of its name. This way, the master thread would wait until all its children have finished processing and then it would compile the output files from each child thread into a single file.

4.2.3 Average Network GCV

Once the GCV signature of every single node in the input network was computed, I wrote a different C++ program that would take these signatures and compute an average GCV signature for the whole network. This allowed me to perform comparisons between various network types, both real networks as well as random networks.

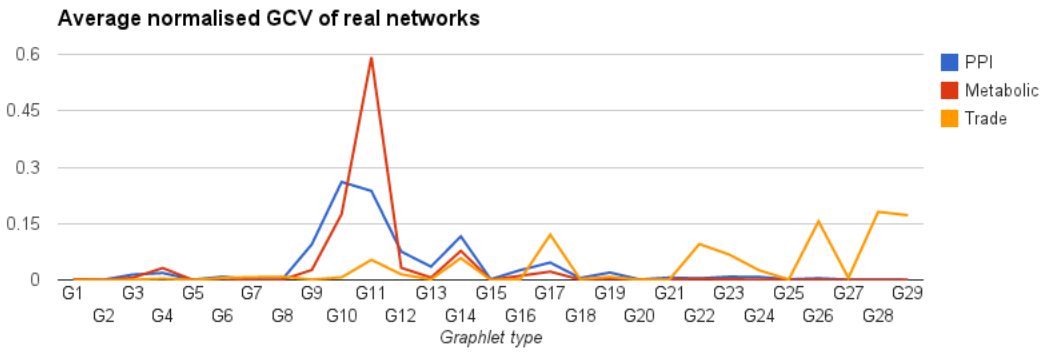


Figure 4.1: Comparison of the graphlet degree vectors for three different real networks: Protein-protein interaction network, Metabolic network and a 2010 Trade network between countries

What we observe from figure 4.1 is that there are slight differences between the GCV of the three networks analysed. More precisely, Graphlets G_{11} and G_{10} seem to discriminate well between them, with the Metabolic network having the highest number of type G_{11} graphlets while the trade network having the least. Moreover, the trade network also seems to have considerably more graphlets of type G_{22}, G_{26} and G_{28}.

4.2.4 Random Networks

After we performed a few comparisons of the average GCV of real networks, our next step was to experiment with the following random network models:

1. Erdos-Renyi[13]
2. Erdos-Renyi (arbitrary degree distribution)
3. Geometric networks[14]
4. Barabasi-Albert (preferential attachement)[15]
5. Stickiness index-based[16]

We have performed one initial experiment on the Human PPI network, using both the real network and the random models mentioned above(see fig. 4.2). We have not had enough time to calculate the GDV agreements between these networks, but once that is done we would be able to tell how well the random models fit the data and which random model fits the real network best.

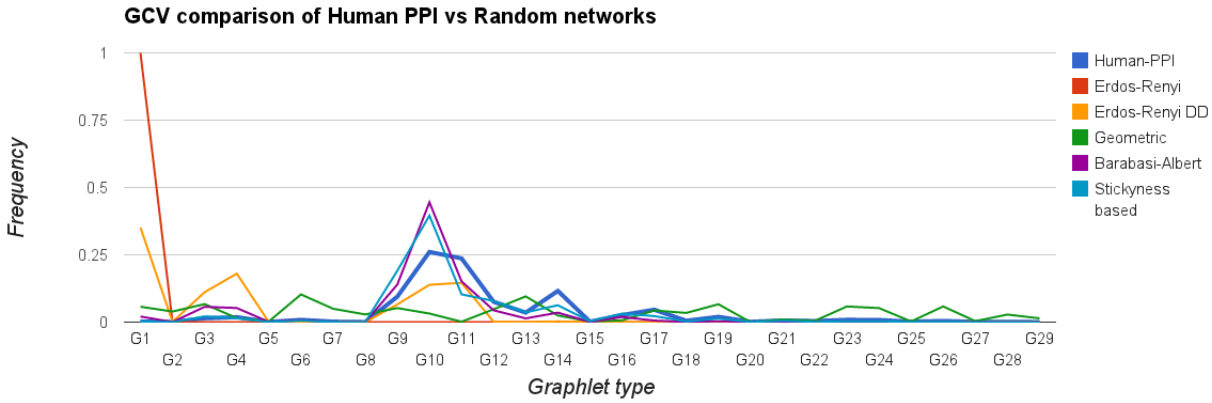


Figure 4.2: Comparison of the graphlet degree vectors between Human PPI network and random models: Erdos-Renyi, Erdos-Renyi with the Degree Distribution of the Human PPI network, Geometric, Barabasi-Albert Preferential Attachment, Stickiness-based

4.3 Upcoming tasks

The next task that I am looking into is to compute the Graphlet Frequency Agreement between random networks and real networks. This will allow us to find out which models best fit the data. Afterwards, I will generate more instances of each type of random networks in order to find out what is the standard deviation of the frequencies in the GCV. This would allow us to establish how robust the new GCV signature is to noise.

Chapter 5

Evaluation Plan

Evaluation of the project will be mainly done on the number of proposed goals that I manage to achieve. As a lot of the work has been already done, half of the goals have therefore already been achieved by now. I have managed to implement the core algorithms, parallelise it and perform a few initial experiments. Moreover, we will also be able to evaluate the project success on the biological and economic insights that we manage to uncover from the data available.

Another benchmark we could use to evaluate the success of our project is through the number of positive results we find from out experiments. For example, assume that we try to use the GCV metric to align two networks. If we feed the GCV signatures into the GRAAL algorithm and get a good alignment, then we have successfully used it in this particular experiment. A good alignment could easily be defined as:

- At least 70% of the nodes from both networks are completely aligned.
- At least 50% of the edges connecting two nodes are aligned.

Another experiment that could be used to measure success is using the average network GCV for discriminating between different network types. If we successfully manage to discriminate with an F1 score higher than 80% between the different network types(ex: Protein-protein interaction vs Trade, Metabolic vs Erdos-Renyi, etc ..), then we can say to have successfully used the GCV as a classifier. In order to do that, we indeed need to build a machine learning classifier that would initially be trained on a variety of networks and then used to classify new data. The F1 score previously mentioned is calculated as follows:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.1}$$

where:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

Since our project builds upon earlier work done by Przulj et al on the Graphlet Degree Vectors[9][11], we will be interested to find out in what respect our new GCV signature is different. Therefore, if we find out that the GCV is different in certain respects and can be used in different scenarios, then we can also use it as a benchmark score for success.

Last but not least, we are also planning to publish a paper with these results in one Bioinformatics journal. We can also say that the project has been successful if we managed to get good-enough results in order to publish the paper and have it peer-reviewed and accepted for publication.

Chapter 6

Conclusion

This interim report offers a clear perspective on the work done on this project so far. I have successfully developed the mathematical model for the GCV signature and I have implemented in an efficient manner the core algorithms necessary for the project. I have also performed some early experiments that showed some interesting properties of the metric.

Nonetheless, I also encountered several problems that I could not really foresee, which delayed the usual progress of the project. For instance, right at the very beginning I when I was working on the mathematical model, I could not mathematically find what the maximum possible number of graphlets of a particular type in a graph of N nodes. After I finished the mathematical model, I also had to spend some time reformatting the graphlet counting function that I was given. Although it took me a while to solve these tasks, I believe it was good that I tackled them in this manner, as it helped me better understand the nature of the project I was working on, both from a mathematical and a computational perspective.

6.1 Future work

As I have previously mentioned, our future work mainly consists in performing more experiments with the GCV signature in order to find out its properties. Our group is fairly confident that we will get a lot of interesting results. We are therefore planning to publish one paper with these results in a Bioinformatics journal.

One other idea we could explore is to try to derive several related metrics using different normalisation procedures or even combine the GCV with the older GDV metric. This could allow us to effectively use the power of both metrics at the same time.

Another idea that I had while discussing with Zoran, one of Dr. Przulj's collaborators, was to find out how important each of the elements from the *Graphlet Cluster Vector* was by assigning a weight to each of them. Using some machine learning techniques or linear regression, some optimal weights could be derived which would make the signatures more efficient when comparing them with each other or when calculating the *Relative Cluster Frequency Distance*. Unfortunately, these tasks are outside the scope of our project, as they can take a few months to complete and I wouldn't have enough time to finish them by July.

Finally, we hope that our work will help us better understand local properties of biological or economic networks and that it will make an impact in the Bioinformatics community. Ultimately, network analysis is a never-ending task: one can always find better ways to explain phenomena or behaviour, and even more, as this phenomena or behaviour changes over time, new models need to be developed that model then as closely as possible.

Chapter 7

Bibliography

[1] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[2] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[3] James D Watson. The human genome project: past, present, and future. *Science*, 248(4951):44–49, 1990.

[4] Ahmedin Jemal, Rebecca Siegel, Elizabeth Ward, Yongping Hao, Jiaquan Xu, Taylor Murray, and Michael J Thun. Cancer statistics, 2008. *CA: a cancer journal for clinicians*, 58(2):71–96, 2008.

[5] World Health Organization et al. Annex table 2: Deaths by cause, sex and mortality stratum in who regions, estimates for 2002. *The world health report*, 2004.

[6] Syed Asad Rahman and Dietmar Schomburg. Observing local and global properties of metabolic pathways:load points and choke points in the metabolic networks. *Bioinformatics*, 22(14):1767–1774, 2006.

[7] Muhammed A Yıldırım, Kwang-Il Goh, Michael E Cusick, Albert-László Barabási, and Marc Vidal. Drugtarget network. *Nature biotechnology*, 25(10):1119, 2007.

[8] Roger Guimera and Luis A Nunes Amaral. Modeling the world-wide airport network. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):381–385, 2004.

[9] Nataša Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.

[10] N Pržulj, Derek G Corneil, and Igor Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[11] Tijana Milenkoviæ and Nataša Pržulj. Uncovering biological network function via graphlet degree signatures. *Cancer informatics*, 6:257, 2008.

[12] Oleksii Kuchaiev, Tijana Milenković, Vesna Memišević, Wayne Hayes, and Nataša Pržulj. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface*, 7(50):1341–1354, 2010.

[13] Paul Erdős and Alfréd Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[14] Mathew Penrose. *Random geometric graphs*, volume 5. Oxford University Press Oxford, 2003.

[15] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[16] Nataša Pržulj and Desmond J Higham. Modelling protein–protein interaction networks via a stickiness index. *Journal of the Royal Society Interface*, 3(10):711–716, 2006.

[17] L da F Costa, Francisco A Rodrigues, and Alexandre S Cristino. Complex networks: the key to systems biology. *Genet Mol Biol*, 31:591–601, 2008.

[18] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[19] Oleksii Kuchaiev, Marija Rašajski, Desmond J Higham, and Nataša Pržulj. Geometric denoising of protein-protein interaction networks. *PLoS computational biology*, 5(8):e1000454, 2009.

[20] Michael Lappe and Liisa Holm. Unraveling protein interaction networks with near-optimal efficiency. *Nature biotechnology*, 22(1):98–103, 2003.

[21] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[22] Paul Erd6s and A Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5:17–61, 1960.

[23] Fereydoun Hormozdiari, Petra Berenbrink, Nataša Pržulj, and S Cenk Sahinalp. Not all scale-free networks are born equal: the role of the seed graph in ppi network evolution. *PLoS computational biology*, 3(7):e118, 2007.

[24] Mark Newman. *Networks: an introduction*. Oxford University Press, 2009.

[25] Stephen M Stigler. Francis galton's account of the invention of correlation. *Statistical Science*, 4(2):73–79, 1989.

[26] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[27] MH Fulekar. *Bioinformatics: Applications in life and environmental sciences*. Springer, 2009.

[28] Jack Cohen. *Statistical power analysis for the behavioral sciencies*. Routledge, 1988.

[29] Skbkekas. Wikipedia spearmans rank correlation coefficient, December 2009.

[30] Ann Lehman. *JMP for basic univariate and multivariate statistics: a step-by-step guide*. SAS Institute, 2005.

[31] Jerome L Myers, Arnold D Well, and Robert Frederick Lorch. *Research design and statistical analysis*. Routledge, 2010.

[32] Oleksii Kuchaiev and Natasa Przulj. Learning the structure of protein-protein interaction networks. In *Pacific Symposium on Biocomputing*, volume 14, pages 39–50, 2009.

[33] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 1936.

[34] William W Cooley and Paul R Lohnes. *Multivariate data analysis*. J. Wiley, 1971.

[35] Peter S Fader and Leonard M Lodish. A cross-category analysis of category structure and promotional activity for grocery products. *Journal of Marketing*, 54(4), 1990.

[36] R Mohan Pisharodi and C John Langley. Interset association between measures of customer service and market response. *International journal of physical distribution & logistics management*, 21(2):32–44, 1991.

[37] Vesna Memišević and Nataša Pržulj. C-graal: Common-neighbors-based global graph alignment of biological networks. *Integrative Biology*, 4(7):734–743, 2012.

[38] Yushi Jing and Shumeet Baluja. Pagerank for product image search. In *Proceedings of the 17th international conference on World Wide Web*, pages 307–316. ACM, 2008.