CONSTRUCTION AND ANALYSIS OF TISSUE/DISEASE SPECIFIC PROTEIN-PROTEIN INTERACTION NETWORKS BY INTEGRATING LARGE SCALE TRANSCRIPTOME DATA WITH GENOME SCALE PROTEIN-PROTEIN INTERACTION NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ARZU BURÇAK SÖNMEZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
MEDICAL INFORMATICS

JULY 2022

CONSTRUCTION AND ANALYSIS OF TISSUE/DISEASE SPECIFIC PROTEIN-PROTEIN INTERACTION NETWORKS BY INTEGRATING LARGE SCALE TRANSCRIPTOME DATA WITH GENOME SCALE PROTEIN-PROTEIN INTERACTION NETWORKS

submitted by **ARZU BURÇAK SÖNMEZ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Health Informatics Department, Middle East Technical University** by,

Prof. Dr. Deniz Zeyrek Bozşahin
Dean, **Graduate School of Informatics**

Assoc. Prof. Dr. Yeşim Aydın SON
Head of Department, **Health Informatics**

Prof. Dr. Tolga CAN
Supervisor, **Computer Engineering, METU**

Examining Committee Members:

Prof. Dr. Tansel ÖZYER
Computer Engineering, Ankara Medipol University

Prof. Dr. Tolga CAN
Computer Engineering, METU

Assoc. Prof. Dr. Hasan OĞUL
Computer Science, Østfold University College

Assist. Prof. Dr. Burçak OTLU SARITAŞ
Health Informatics, METU

Assoc. Prof. Dr. Yeşim Aydın SON
Health Informatics, METU

Date: 25.07.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname:	**ARZU BURÇAK SÖNMEZ**
Signature :	

ABSTRACT

CONSTRUCTION AND ANALYSIS OF TISSUE/DISEASE SPECIFIC PROTEIN-PROTEIN INTERACTION NETWORKS BY INTEGRATING LARGE SCALE TRANSCRIPTOME DATA WITH GENOME SCALE PROTEIN-PROTEIN INTERACTION NETWORKS

SÖNMEZ, ARZU BURÇAK
Ph.D., Department of Health Informatics
Supervisor: Prof. Dr. Tolga CAN

July 2022, 60 pages

Analysis of integrated genome-scale networks is a challenging problem due to heterogeneity of high-throughput data. There are several topological measures, such as graphlet counts and degree distributions, for characterization of biological networks. In this dissertation, we present methods for counting graphlet patterns in integrated genome-scale networks which are modeled as labeled multidigraphs. We have obtained physical, regulatory, and metabolic interactions between H. sapiens proteins from the Pathway Commons database. For the first part of the dissertation, the integrated network is filtered for tissue/disease specific proteins by using a large-scale human transcriptional profiling study, resulting in several tissue and disease specific sub-networks. We have applied and extended the idea of graphlet counting in undirected protein-protein interaction (PPI) networks to directed multi-labeled networks and represented each network as a vector of graphlet counts. Graphlet counts are assessed for statistical significance by comparison against a set of randomized networks. We present our results on analysis of differential graphlets between different conditions and on the utility of graphlet count vectors for clustering multiple condition specific networks. Our results show that there are numerous statistically significant graphlets in integrated biological networks and the graphlet signature vector can be used as an effective representation of a multilabeled network for clustering and systems level analysis of tissue/disease specific networks. The second part of the dissertation provides methods for comparing the network nodes based on the counts of 3- and 4-node multilabeled graphlets starting from these nodes. Then, we use different types of cancer pathways to retrieve the integrated interactions occurring between them from Pathway Commons dataset. These interactions are further investigated for identifying recurring 3- and 4-node graphlet patterns and Pathway Commons dataset and on different cancer pathways.

Keywords: Integrated Networks, Network Comparison, Directed Graphlets, Systems Biology

ÖZ

TRANSKRİPTOM VE GENOM ÖLÇEKLİ PROTEİN ETKİLEŞİM AĞLARININ BİRLEŞTİRİLMESİ İLE DURUM BAZLI SPESİFİK PROTEİN ETKİLEŞİM AĞLARININ OLUŞTURULMASI VE ANALİZİ

SÖNMEZ, ARZU BURÇAK
Doktora, Tıp Bilişimi Bölümü
Tez Yöneticisi: Prof. Dr. Tolga CAN

Temmuz 2022, 60 sayfa

Entegre genom ölçekli ağların analizi, yüksek hacimli verilerin heterojen yapısı nedeniyle zorlu bir problemdir. Biyolojik ağların nitelendirilmesi amacıyla kullanılan çizgecik sayıları ve derece dağılımları gibi çeşitli topolojik ölçütler bulunmaktadır. Bu tezde, etiketli, çoklu ve yönlü çizgeler şeklinde modellenmiş, entegre genom ölçekli ağlarda bulunan çizgecik kalıplarını saymak için yöntemler sunuyoruz. Bu çalışmanın ilk bölümünde, entegre ağları oluşturmak için, Pathway Commons veritabanından, proteinler arasında var olan metabolik, düzenleyici ve fiziksel etkileşim verileri alınmıştır. Daha sonra bu veriler, hastalık ve dokuya özgü proteinler kullanılarak filtrelenmiştir. Yönsüz protein-protein etkileşim ağları üzerinde uygulanan çizgecik sayma yaklaşımı yönlü ve çoklu etiketli etkileşim ağları için kullanılmak üzere genişletilmiştir. Belirlenen çizgecik sayıları karşılaştırmalı analizde kullanılmak amacıyla ağların göstericisi olarak belirlenmiştir. Ayrıca aynı derece dağılımlarına sahip gerçek ve rastgele oluşturulmuş ağların karşılaştırılması sonucunda çizgecik sayılarının istatistiksek önemi belirlenmiştir. Geliştirdiğimiz uygulama, durum bazında farklılık gösteren çizgeciklerin analizi ve çizgecik sayı vektörlerinin duruma özel oluşturulmuş ağların kümelenmesi yönlerinden değerlendirilerek sonuçlar paylaşılmıştır. Tezin ikinci kısmında ise, çizgecik sayıları ağ düğümü (protein) bazında değerlendirilmiştir. Belli kanser tiplerine özel proteinler kullanılarak oluşturulan entegre etkileşim ağları, 3 ve 4 ağ düğümlü çizgeciklerin sayımı için kullanılmış, bu etkileşim ağları ağ düğümü tabanlı çizgecik sayıları kullanılarak analiz edilmiştir.

Anahtar Kelimeler: Birleşik ağlar, Biyoloik Ağ Analizi, Çizgecik Analizi

dedicated to my son, ATEŞ

ACKNOWLEDGMENTS

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

PPI	Protein-Protein Interaction
HGNC	HUGO gene nomenclature comitee
SIF	Simple Interaction Format
HPRD	Human Protein Reference Database
MDT	Myometrium Disease Tissue
RINT	Retrocervical Infiltrate Normal Tissue
EODT	Endometrium Ovary Disease Tissue
SMSQNT	Skeletal Muscle Superior Quadracep Normal Tissue
PCA	Principal Component Analysis

CHAPTER 1

INTRODUCTION

1.1 Motivation

Systems biology is the study of relations between the components of a cell. It also investigates the effects of the interactions on the function and the behavior of the biological systems. The biology of an organism is associated with the interplay between some factors such as genes, proteins, and some other small biomolecules. Proteins are products of the protein-coding parts of the DNA material known as "genes" that are produced by a process called gene expression. The functioning of the biological system is largely accomplished by proteins. Proteins are not standalone biomolecules, in fact, they work together in biological processes and bio-pathways. They can interact with each other both directly and indirectly. So in order to reveal the underlying system of an organism, the proteins should be studied in the context of their interactions with other proteins, not in isolation.

There exist some challenges causing misinterpretation in biological networks. One major challenge is that high-throughput experiments measuring a single type of interaction can not display the full interaction landscape of a cell. There are different types of information including biological networks and pathways fragmented among many databases. So, in order to get the full picture of the biological systems, as much information as possible should be extracted and integrated (Mitra, Carvunis, Ramesh, & Ideker, 2013). Therefore, in our study, we consider an integrated set of protein interactions including physical, regulatory, and metabolic types.

Another problem in network biology is that the assembled interaction information is inconsistent with the dynamic nature of biological systems. Biological systems are highly dynamic that show continuous changes in order to respond to some factors such as environment, disease state, tissue type, development, etc. For this reason, there have been several studies investigating the dynamics of biological networks through experimental mapping of the networks by concentrating on the dynamic system response rather than the absolute properties of the system. As a result, these studies introduce the differential interaction maps that include the most clearly changing interactions among conditions compared with the interactions in static conditions (Cabusora, Sutton, Fulmer, & Forst, 2005),(Ideker & Krogan, 2012).

The interaction data between proteins are obtained by using some experimental techniques and deposited in many publicly available databases. These databases include various different types of interactions between protein pairs showing the same system from different perspectives. All interactions are usually represented mathematically as graphs (networks) and studied by graph-theoretic approaches. Graph theoretical approaches are the collection of methods trying to uncover the underlying interaction map between biological entities. A graph is a mathematical representation of relational data in which nodes correspond to the entities in the system and edges correspond to the relations between entities (Newman, 2009). There are basically many interesting and different biological problems in the computational biology field. Some of them can be solved exactly in the polynomial-time of the size of the input. But unfortunately, there are many questions that we ask about large biological networks that fall into the category of computationally intractable problems. The network comparison problem is one of them. In order to compare networks, we need to establish the list of topological differences which means we need to solve the subgraph isomorphism problem (Cook, 1971) This problem has shown to be NP-Complete (Garey & Johnson, 1979) and cannot be solved exactly in polynomial-time. So the only way we address these problems is to approximately solve them and these approximate ways are called heuristics. The "Graphlet" concept is one of the examples of the heuristics used to compare or align large biological networks. Graphlets are building blocks of large networks such as Lego pieces. Technically, graphlets are small, connected, induced, non-isomorphic subgraphs of large networks. This concept is first introduced by the study (Pržulj, 2007). Graphlet concept is used to extract some features of directed and undirected network types by several studies, and these works show that as the size of the graphlet increases, the computational complexity increases making the graphlet counting methods less applicable for larger networks. In order to overcome this challenge, some efficient counting strategies are sought. The work presented in (Ahmed, Neville, Rossi, & Duffield, 2016) proposed a parallel counting strategy for 3-4 node graphlets by also considering combinatorial arguments for fast counting. Their study is following the principle that 4-node graphlets can be decomposed into 3-node graphlets.

Eventually, several methods are proposed to count the directed and undirected graphlets of size 2-5 nodes efficiently. However, none of these methods consider both directed and undirected edges with multiple labels. To accomplish all the stated challenges, in this dissertation we propose methods for counting the graphlets of integrated interactions considering directed, undirected and reciprocal edges.

In the rest of this Chapter, we introduce some recent studies using the graphlets for the analysis of different type of networks. Then we introduce some basic concepts about networks and network properties. Finally, this section is concluded with the outline of the dissertation.

1.2 Introduction and Literature Review

As the biological databases are aggregating and curating the biological molecules such as proteins, chemical compounds and genes from different data sources, it gives an opportunity to make computational analysis on huge amounts of data and new insights into the research on human diseases and drug discovery. Some standard formats such as BioPAX (Demir et al., 2010) are used to represent and allow the exchange of different network types between applications and data sources. These sources include different types of data involving physical, regulatory and metabolic interactions, however an integrative analysis of all these types of interactions remains as a challenge. There exists some tools such as Paxtools (Demir et al., 2013), which is a software that allows us to query, convert and visualize the integrated networks in BioPAX format. However, the topological analysis of integrated networks by applying advanced algorithms remains a challenge.

Additionally, biological systems are altered in different ways depending on some factors such as tissue type, disease state, etc. Distinct studies show that the interaction networks are incurred several configurations during some response phases that can be either cellular or adaptive. That means the biological systems are so dynamic that constantly react to genetic and environmental changes (Ideker & Krogan, 2012). For this reason, we propose a method for differential network analysis dealing with the networks across multiple conditions. Several studies constructing condition specific functional interaction networks (Greene et al., 2015), protein-protein interaction networks (Will & Helms, 2016), (Kotlyar, Pastrello, Sheahan, & Jurisica, 2016), pathways (Park et al., 2015), and regulatory networks (Zhang, Tian, Tran, Choi, & Zhang, 2014) exist, but there is no study that integrates physical, regulatory and metabolic interactions to provide condition specific networks. In Chapter 2, we propose a method first detecting the differentially expressed human genes for some specific conditions. Then for each condition, the active genes are used to retrieve the condition specific interactions from the single integrated network involving physical, regulatory and metabolic interactions between human genes. For analysis of the condition specific networks, the topological similarities and differences should be identified. For this purpose, each condition specific network is represented by a vector of graphlet counts. Recall that the graphlets are first introduced by the study (Pržulj, 2007). This concept is first introduced in order to compare 14 eukaryotic PPI networks and 4 different model networks. Each real PPI network is compared with all four models and their similarity is measured by using graphlet-based statistics called graphlet degree distribution agreement. Graphlet degree is the generalization of the degree of a node. Node degree shows the number of edges touching a node while graphlet degree shows the number of graphlets touched by a node. The topologically similar nodes of the graphlets are also annotated and named as automorphism orbits (Pržulj, 2007), (Hočevar & Demšar, 2016). The work in (Pržulj, 2007) produced 2- to 5-nodes undirected graphlets with 73 automorphism orbits and conducted a comparative analysis of the real and model networks by observing the graphlet degree distribution of each. Figure 1.3 shows the 2-5 sized undirected graphlets produced by their work. Graphlets are small connected non-isomorphic induced subgraphs that are used to define the local topology around a protein/gene in a specific network and they are shown to be an effective way of the analysis and comparison of biological networks (Hayes, Sun, ulj, Editor, & Valencia, 2013), (Pržulj, 2007).

In another study, the undirected graphlets are extended to directed ones in order to compare directed networks. Their new methodology is used to interpret the topology of directed networks pertaining to different research areas such as economy, systems biology, etc. They produced 40 different directed graphlets with the size of 2-4 nodes. Their work also considers 128 topologically symmetric orbits. The directed graphlet-based similarity measure is tested if it is effectively clustering the networks corresponding to the six different model networks. First, with differing numbers of nodes and edge densities, 10 different networks for each model are generated. Totally 360 generated networks are clustered by using directed graphlet-based statistics and the clustering performance is evaluated. Despite the results showing the superiority of the method for comparing directed networks, the study has a limitation that the anti-parallel edges occurring between pairs of entities in real networks are not represented in the directed graphlets produced by their work. That means each anti-parallel directed edge separately triggers a corresponding directed graphlet (Sarajlić, Malod-Dognin, Ömer Nebil Yaveroĝlu, & Pržulj, 2016)

Some other studies also propose methods that counts the 2- to 5-node, directed and undirected graphlets effectively (Marcus & Shavitt, 2012), (Meira, Máximo, Álvaro L. Fazenda, & Conceião, 2014), (Hočevar & Demšar, 2014), but none of these deal with the multi labeled edges. Our proposed method in Chapter 2 considers the multiple edges occurring between nodes and represents all edges between node pairs by a single edge encoding vector. The counts of 2-3 node graphlets of each condition specific network are collected by a vector. The vectors of graphlet counts are then used to detect statistically significant graphlets by calculating the z-scores of each. Lastly, the success of using graphlets for clustering condition specific networks is revealed. Finally, the statistically significant graphlets appeared in different states of the same tissue networks are used for differential analysis. The details of the processes and the results are given in Chapter 2.

In Chapter 3, our proposed method constructs a node based similarity measure for the networks to be compared. Apart from the method in Chapter 2, for each node, how many times the node is touching a specific graphlet is calculated by accounting the topologically distinct points of the graphlet. Then the count values collectively comprise the signature vector for each node of a specific network. The implemented method is mainly based on the work of (Trpevski, Dimitrova, Boshkovski, Stikov, & Kocarev, 2016) which uses the directed graphlets revealing structural network similarities. They used an adjacency matrix to represent the networks to be compared. Unlike the study in (Sarajlić et al., 2016), here the anti-parallel edges are considered as reciprocal edges. Produced directed graphlets are used to reveal the graphlet dependencies occurring in each of the brain networks taken from 40 healthy subjects. This study uses the graphlet signature vector for the representation of the structure around each node of the network. Their strategy counts the graphlets touching a specific node of the network by an initiation point of the graphlets and assumes that the directed edges are considered w.r.t that particular node of graphlet, that means each graphlet is induced by a specific node and initiating node also corresponds to a different orbit of the graphlet. As a result, totally 1695 orbits are generated for up to 4 nodes. Since enumerating such a large number of orbits is not an easy task, they restrict their work to 2- to 3- node graphlets. The study produced 9 wedges and 27 triangles starting from a specific node. By grouping the isomorphic graphlets, 6 wedge and 7 triangle classes are obtained for this study. Figure 1.1 shows totally 27 triangles and 9 wedges that are grouped into 7 isomorphic triangle and 6 isomorphic wedge classes in order to reduce the number of features in each node's signature vector. The Figure 1.1 is from Figure 1 of (Trpevski et al., 2016)

Although the study considers parallel edges, they did not consider the types of the edges (both directed and undirected) in the analysis of the networks that also include these types of information. For the method in Chapter 3, the graphlet structures are determined by accounting the approach implemented by (Trpevski et al., 2016). The directed wedge and triangle patterns are combined with all possible undirected 3 node subgraphs in order to produce graphlets involving combined directed and undirected interactions. Figure 1.2 shows some samples of combined patterns. It is shown that the directed triangle and wedge patterns are combined with the disconnected and connected 3-node subgraphs respectively. The combination of the patterns emerges multi labeled 3-node graphlets comprising both directed and undirected multi labeled relations.

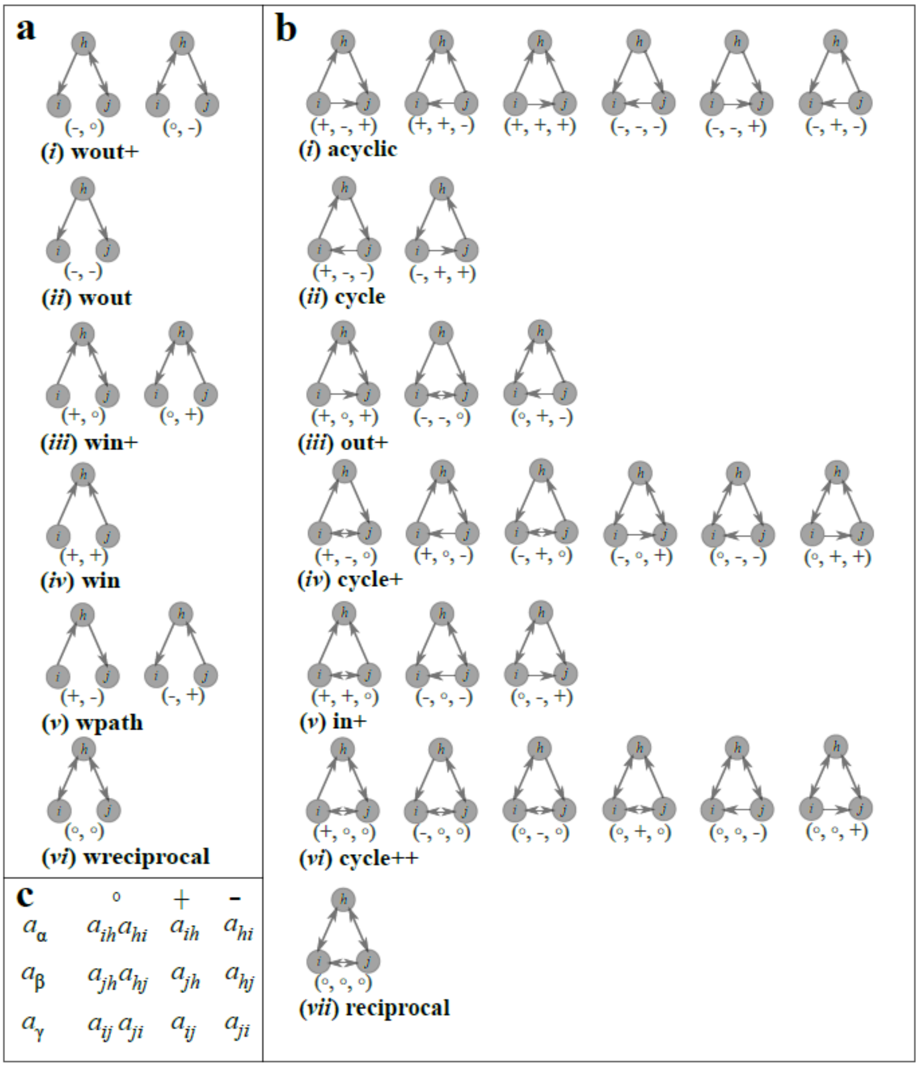


Figure 1.1: The isomorphic classes of directed wedges and triangles produced by Trpevski's study.

Apart from the method in Chapter 2, the generic set of 3-node graphlets is produced for this part of the study. Here, the network analysis is conducted by considering each element of the generic set instead of considering all the graphlets encountered in the network. All possible 3-node multi labeled graphlets are investigated in the network since the graphlet patterns appeared at any frequency are assumed equally important for the graphlet analysis of the networks for this part of the study.

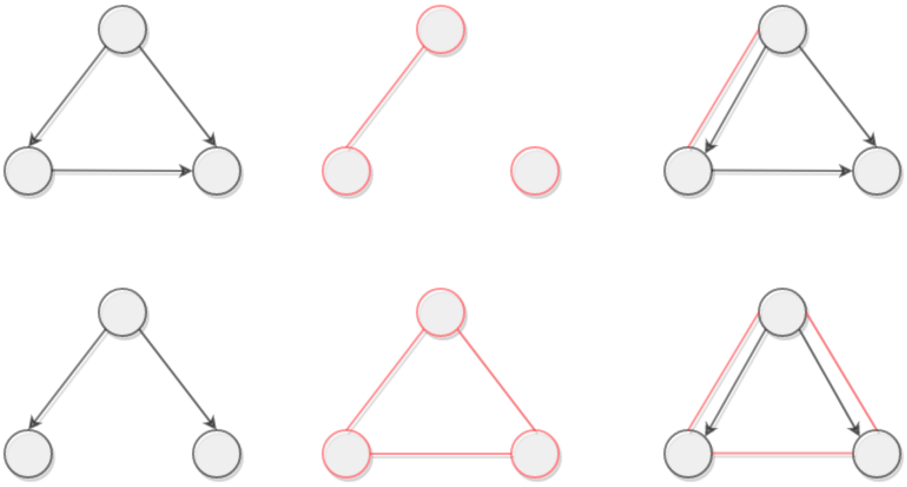


Figure 1.2: Samples of generated graphlets by combining undirected subgraphs and directed wedge and triangle patterns

Additionally, the automorphism orbits are important for revealing node based statistics. However, considering different types of edges results in the combinatorial expansion of different types of graphlets. As the size of the generic graphlet set increases, it becomes more challenging to overcome the computational efficiency issues. For this reason, the current approach is restricted for regarding two types of edges (two labels), one directed and the other is undirected.

Combination of two edge types emerges eight possible interactions between protein pairs. All possible combination of the pairwise relations to generate 3-node subgraphs result in 512 different graphlets including both connected and disconnected types. The disconnected ones are eliminated since the graphlet structures are connected by nature. An additional constraint is applied for the pairwise interactions involving in the graphlets such that defining a pattern (i,k,j) as a wedge or a triangle could only be possible by inspecting the existence of connection between staring node (node i) and the last node (node j). So, only the connection between i and j can be nonexistent, otherwise the graphlet is eliminated from the set of generic graphlet set.

One last elimination process is accomplished for the patterns including single undirected relation in multiple positions of the graphlet. This constraint occurs since the combined directed graphlets are connected.

After elimination processes, 360 multi labeled wedges and triangles are generated. Then the isomorphic graphlets are considered and grouped by collecting the isomorphic ones in the same class. The classification process is further reduced the size of the feature vector representing the network nodes. As a result, a feature vector of each network node considering pairwise relations, wedges and triangles associated with that node is defined to represent the local connectivity of each node of the networks.

The study of (Pržulj, 2007) determines the number of orbits in undirected 2- to 5-node graphlets as 73, while (Sarajlić et al., 2016) determines the number of orbits in 2-4 node directed graphlets with restricted contributions for both anti-parallel and multi labeled edges as 128. Additionally, the number of orbits for up to 4-node single labeled directed graphlets is reported as 1695 by the study considering the anti-parallel edges (Trpevski et al., 2016). The reference studies show that as the size of the graphlet increases, number of graphlets and orbits increases and makes the enumeration of the orbits unappealing. In addition to these, (Sarajlić, 2015) and (Yaveroglu, 2013) reported the orbit dependencies for all 2- to 5- node graphlets. Because of all the specified results, the proposed methods in this dissertation is restricted to deal with the graphlets size of up to 3-nodes. However, the number of 4-node graphlets existing around a specific protein is reported by the method in Chapter 3. Instead of producing a generic set for 4-node graphlets and orbits, combination of identified 3-node graphlets are used for producing 4-node directed multi labeled graphlets based on the work presented in (Ahmed et al., 2016). Their study is following the principle that 4-node graphlets can be decomposed into 3-node graphlets. So, their method merges the knowledge from the 3-node graphlet counts and some combinatorial approaches to determine the number of 4-node graphlets occurring in the networks faster. Despite implementing a fast and efficient method for graphlet counting, they do not consider the direction and types of the edges. And parallel edges are not considered in their work either. The details of the proposed study is given in Chapter 3.

1.3 Networks and Network Properties

Network is a set of entities and relations between the entities. The mathematical representation of the network is achieved by graphs where the node of the graph corresponds to the network entity and the edges correspond to the relationship between the entities. A graph is denoted by G = {V, E} where V represents the set of nodes and E represents the edges connecting each node pairs. The below list shows some common graph concepts:

- *Undirected graph* is a set of unordered pair of nodes.
- *Directed graph* is a set of ordered pair of nodes. The connection between node pairs has an orientation from source node to target node.
- *Reciprocal edge* also called two-way edge (anti-parallel edge) is a pair of directed edge having the same interactors in reverse order.
- A *loop* is an edge where source and target nodes are equal
- A *simple graph* is a type o graph that contains no loops and no multiple edges.
- A *neighborhood* of a node is the collection of nodes that are connected to this node up to a certain distance.
- A *path* is a simple graph whose nodes are the elements of an ordered list. The consecutive elements (nodes) in the list are also adjacent nodes in the path. The directed path is also an ordered list of nodes but this time there exists a starting and an end point of the path.
- A *connected graph* refers to the set of interacting nodes in which each pair of nodes belong to a path.

- A *cycle* is a type of path whose starting and end points are equal

- A *subgraph* indicated by $G' = \{V', E'\}$ is a subgraph of $G = \{V, E\}$ where $V' \subseteq V$and$E' \subseteq E$

- An *induced subgraph* is a subgraph of $G = \{V, E\}$ involving all elements of E between the nodes in $V' \subseteq V$

- A *partial subgraph* is a subgraph of $G = \{V, E\}$ which does not involve all elements of E between the nodes in V'

- An *isomorphism* between graphs $G = \{V_1, E_1\}$ and $H = \{V_2, E_2\}$ indicates the bijective function f between node sets V_1 and V_2. So, the nodes $i \in V_1$ and $j \in V_1$ are adjacent, if and only if $f(i) \in V_2$ and $f(j) \in V_2$ are also adjacent in H.

- A *motif* is a partial sub-graph occurring in a network with a frequency that is statistically significant. The network chosen as null model is important for the motif significance (Artzy-Randrup, Fleishman, Ben-Tal, & Stone, 2004).

- A *graphlet* is a small connected, induced, non-isomorphic subgraph. They are not partial subgraphs since they involve all the edges occurring between the elements of node subset. They are not necessarily dependent on a null model since they can be recur at any frequency. Different types of graphlets are reported by the several studies. The first time the graphlet notion is introduced, it is used for the analysis of undirected networks. Figure 1.3 shows the 2- to 5-node undirected graphlets. This Figure is from Figure 1 of (Pržulj, 2007)

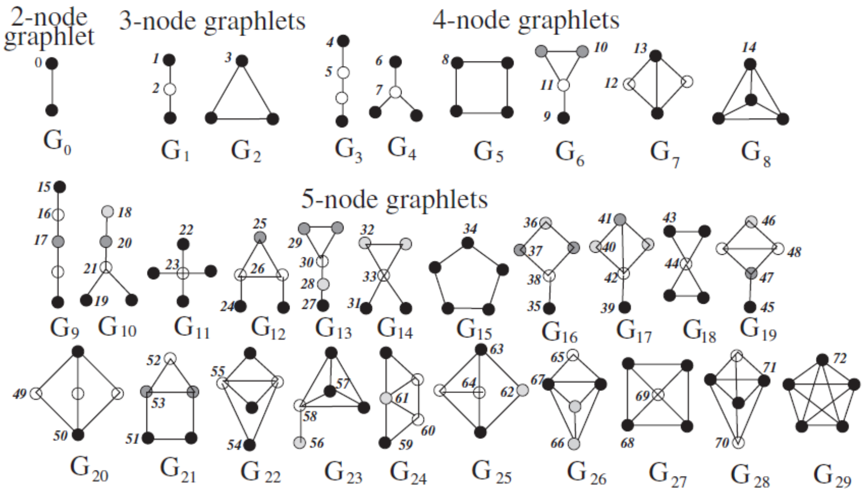


Figure 1.3: 2- to 5-node undirected graphlets produced by Pržulj's study

The set of graphlet involves 30 different patterns and 73 different points called "automorphism orbits" that can be touched by a specific node involving in the graphlet. By counting both graphlets and nodes touching a specific node of the graphlet, the measure of similarity is created in order to compare undirected networks. Graphlets can also be utilized for comparing directed networks. Some several studies produced a set of directed graphlets with the size up to 2- to 4-nodes. Figure 1.4 shows the directed graphlets and 128 different automorphism orbits produced by the study (Yaveroglu, 2013),(Sarajlić et al., 2016). This study adopts the extended version of the method analysing the undirected PPI networks in (Pržulj, 2007). The main drawback of the this study is restricted representation of multiple edges since the anti-parallel edges are not allowed while producing the graphlets. If multiple edge is encountered between a pair of nodes, each edge involving in the multi edge interaction is accounted for a separate graphlet. In Figure 1.4, the 3-node pattern including purple directed edge is counted for graphlet #21 and the pattern induced by the directed edge in blue color is counted for graphlet #25. The Figure is taken from Figure 1 of (Sarajlić et al., 2016)

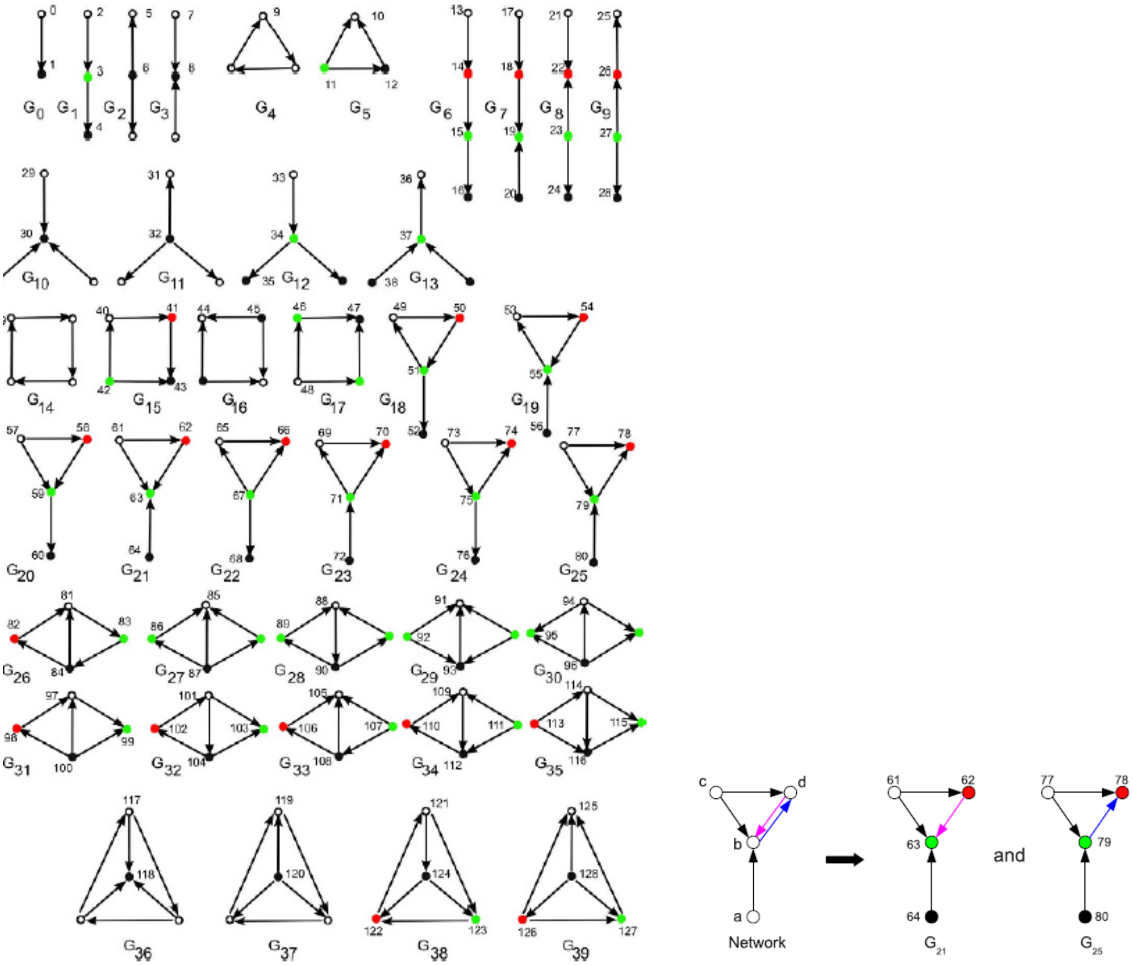


Figure 1.4: 2- to 4-node directed graphlets and the depiction of the approach for encountered anti-parallel edges.

1.4 Contributions of the Thesis

The methods provided in this dissertation suggest solutions to overcome some challenges in Systems Biology. Here we summarise the contributions made by this dissertation.

- The idea of directed and undirected graphlet analysis for network comparison is extended to provide graphlet analysis concerning both types of interactions (both undirected and directed edge types)
- An integrated interaction dataset is used to consider multi labeled interactions including physical, regulatory, and metabolic interaction types for the analysis of the networks instead of using single type of relation for a more complete understanding of the function and the behaviour of the biological systems.
- The tissue/disease network and cancer pathway specific protein interactions are considered for the graphlet analysis in order to reveal the differential interactions appeared by the the dynamic response of the cell against these conditions.

1.5 Organization of the Thesis

The study provided by the thesis is mainly based on counting the graphlet patterns for analysis of the integrated networks. Chapter 2 presents a method counting the directed multi labeled graphlets in integrated condition specific networks. The graphlet counts are used to make comparative analysis between distinct conditions. The differentially recurring graphlets are investigated and interpreting to reveal underlying processes related with specific conditions. Chapter 2 proposes a method for analysing the integrated networks by using a generic set of 3-node graphlets. Each 3-node graphlet in the generic set is produced by considering topologically similar node positions called orbits. Apart from Chapter 2 counting number of graphlets in the network, the method presented in Chapter 3 identifies the 3-node graphlets touched by the network nodes.. Finally, in Chapter 4, a brief summary of the study conducted by the thesis is provided. The results of the methods are discussed and future directions are given also in Chapter 4.

CHAPTER 2

COMPARISON OF TISSUE/DISEASE SPECIFIC NETWORKS BY USING DIRECTED GRAPHLETS

Our published study showing how network topology statistics can successfully be used for comparative analysis of integrated biological networks is presented in this chapter. For this part of the dissertation, the statistically significant graphlets are used to represent the network wirings involved in different types of diseases and tissues. Apart from other graphlet based methods, different types of directed and undirected interactions are considered by our method. We propose a straightforward method, however it is the first attempt dealing with the directed multi labelled graphlets for the analysis of integrated networks.

The presented method is composed of two steps. First step is to construct condition specific (tissue or different states of the tissues) subnetworks of the integrated Pathway Commons network w.r.t a human transcriptional profiling study. Then a hash based edge encoding approach is used to find the graphlet counts recurring in each network.

Proposed method is finally evaluated by determining the statistically significant garphlets. Graphlet z-scores are used to represent each condition specific network. Some clustering methods are applied to the same tissue networks in order to show the effectiveness of the graphlets when compared with the other parameters used for tissue comparison. Finally, a set of proteins inducing the differentially recurring significant graphlets between pairs of conditions (healthy/disease states of the same tissue) are identified and analyzed by using DAVID functional annotation tool. The functions associated with the protein set give insight about the possible abnormal behavior of disease state when compared with the healthy state of the same tissue network.

2.1 Datasets

We use two main sources of data for our study. First one comprises the pairwise interactions of human genes (proteins) with HGNC identifiers (Gray, Yates, Seal, Wright, & Bruford, 2014) and interaction types. The second one is the gene expression data pertaining to various human tissues and diseases. Gene expression data is used to filter the pairwise interaction dataset and constructing condition specific subnetworks. The following sections explain the processes in detail.

2.1.1 Pathway Commons Integrated Network

Pathway Commons database is created and made publicly available in 2009 (Cerami et al., 2011). It is basically developed in order to provide an efficient way for scientists to reach the biological data. Pathway Commons database is accomplishing the processes of collecting, normalizing and integrating distributed biological pathway and molecular interaction data. When it is first created, it integrated data from 9 different public databases and contained over 1400 pathways and 687000 interactions. As reported in the last update in 2019 (Rodchenkov et al., 2019), Pathway Commons currently aggregates 22 public databases (Stark et al., 2006),(Prasad et al., 2009), (Fabregat et al., 2017), (Schaefer et al., 2009), (Orchard et al., 2013), (Yamamoto et al., 2011), (Wishart et al., 2018), etc. Also the amount of biological pathways and molecular interaction data is incremented three times more than the amounts reported in 2010. Pathway Commons database is freely available and the researchers can utilize some services such as, search tools, bulk download option, reusable software modules adapted with several programming languages such as JAVA, R, Python, etc. and a web service for querying the entire dataset programmatically. The data collected from multiple database involves biochemical reactions, biomolecular complex assembly, transport and catalysis events, and physical interactions between DNA, RNA, proteins and small molecules. These information are retrieved from original databases in different formats. BioPAX format is more complex since it captures data with additional details represented by descriptions. These descriptions are annotated with links associated with the citations, experimental evidence and the information of external databases(Cerami et al., 2011). Since it includes so much details and our study focuses on pairwise interactions between human protein pairs, SIF formatted information is used for our study. SIF format include pairwise interaction network which is a suitable input for network analysis.

Pathway Commons Version 8 (*The Pathway Commons Database Version 8; [cited April 21, 2016].*, 2016) for human genes is downloaded to be used by this part of the study. The proteins are in the dataset is represented by HGNC (Gray et al., 2014) names. The dataset is the assembly of various databases and it also contains metabolic interactions with compounds; however, in this study, we ignored compound interactions and focused only on the protein-protein relations. After filtering compounds, 883,211 interactions between 19,537 human proteins remains in the resulting network. Table 1 shows the amount and type of interactions in the input network. Interacts-with and in-complex-with type interactions describe undirected associations; whereas, the other five types of interactions in the network represent the directed types

Table 1: Types and amounts of interactions between proteins in Pathway Commons Version 8

Interaction type	# of pairwise interactions
interacts-with	369,895
in-complex-with	153,603
controls-phosphorylation-of	15,636
Catalysis-precedes	120,948
Controls-expression-of	110,013
Controls-transport-of	6960
Controls-state-change-of	106,156

Since phosphorylation and transport control relations are also encapsulated by state-change interaction, we ignored these types which are existing in small amounts in the dataset w.r.t other interaction types. As a result, integrated human network of 19,537 proteins with 523,498 undirected and 337,117 directed edges is constructed and used as an input for this part of the study. The input network contains physical, regulatory, and metabolic interactions. As multiple edges between protein pairs are merged into a single multi labeled directed edge, the final network contains 1,521,508 directed edges between 760,754 protein pairs.

2.1.2 Condition Specific Networks

For this study, we use condition specific term in order to refer to both different tissues and different states of the same tissue such as healthy, disease, treated, etc. Condition specific network is defined as a subnetwork of the integrated network obtained from Pathway Commons database. A condition specific network includes all proteins/genes specific to that condition and the associated directed multi labeled edges. In other words, since a partial set of genes/proteins is active under any condition, a condition specific subnetwork considers the active protein/gene set and the set of interactions between them. Eventually, the expressions values of all genes/proteins determine which genes are active under a specific condition. Microarray experiment is a widely used technique revealing the differences in expression levels and helps the researchers to detect the active gene sets associated with the specific condition.

A large scale microarray study with accession number GSE7307 (Roth, 2007) is used to identify the active gene sets. This is a human transcriptional profiling study that profiles different tissues to figure out gene expression levels by using the Affymetrix U133 plus 2.0 array. Totally 677 samples related to 90 distinct tissue types are used for the experiment that means multiple samples may represent the same tissue types Additionally, different states of the same tissue are profiled yielding a total of 141 condition specific gene expression data.

2.2 Methods

In this section, the details of condition specific network construction process employing the integrated network and microarray data is described in detail. The directed and multi labeled graphlets and the hash-based counting method adopted for quantifying the graphlets are also introduced in this section. And we conclude the section by describing the details of statistical significance assessment of graphlet counts.

2.2.1 Creating Condition Specific Networks

Human transcriptional profiling study is used in order to identify the set of differentially expressed genes for each condition. The differentially expressed genes are also defined as active gene set under a specific condition. Then the set of active genes is used to extract a condition specific subnetwork of the integrated Pathway Commons network. The process of identifying the active genes among conditions first involves the investigation of the microarray experiment enclosing the total of 90 different human tissue types with 677 samples in total. Besides, some tissues are profiled considering the diseased, healthy or treated states of same sample. Each one of the 677 samples, includes a probe set and RMA (robust multi-array) normalized expression values. In order to identify the differentially expressed genes, these expression values need to be discretized first. There are different methods for discretizing the expression values (Li et al., 2010). However we follow a simple approach that determines a threshold level assuming the gene with the expression value over that threshold as active. For our method, the threshold level r=10.0 is used and all genes with the expression value less than 10.0 are eliminated from the sample.

This process is repeated for all tissue types and different states of the tissues producing 141 different types of condition specific gene sets with the amount of probes from 11,931 (retrocervical infiltrate normal sample) up to 15,853 (MDA-MB231 control sample) produced by our method. Table 2 shows the topmost five conditions with the highest number of active genes.

Table 2: Top 5 conditions with the highest active gene set sizes

Conditions	# of nodes
MDA-MB231 cells control	15853
myometrium disease	15785
MDA-MB231 cells treated	15781
myometrium normal	15777
paravertebral muscle disease	15683

Totally, 141 distinct files including the identifiers of active genes associated with each condition are produced by the process. Each line of a condition specific file is processed one by one to implement the underlying condition specific network. Algorithm 1 shows the details of the implementation by using a list of labeled interactions between pair of active genes for a single condition.

Algorithm 1 condition specific network construction

```
Input: $integratedNetwork, activeGeneList$
Output: condition specific network file
$H \leftarrow Hashtable$
for active gene G in $activeGeneList$ do
    if $H$ does not contain G then
        insert gene into hashtable $H$
    end if
end for
for each interaction $i$ in $integratedNetwork$ do
    if $H$ contains i.$gene1$ && $H$ contains i.$gene2$ then
        Add $i$ to to the condition specific network
    end if
end for=0
```

A global hash table holding the active genes as key values is defined. The default hash function for Java strings is used by the method. As an active gene is used as a key of the hash table, a hash code is returned. Equality of hash codes means the keys are equal. As a new gene is to be held by the hash table, it is controlled if it is previously used as a key. That means no genes are assigned as a key twice. The method provides a hash table with the unique keys of active genes. Then the hash table is used to filter out the irrelevant interactions from integrated Pathway Commons network involving non existing genes in the hash table. As a result each interaction line in integrated dataset associated with an active gene is stored in a condition specific network file. Condition specific network file is a text file which includes the identifiers of the interacting gene pairs and the associated edge type. Figure 2.1 shows an example interaction line included in the file. All the steps are repeated for 141 different conditions and 141 condition specific network files are produced as a result of the method given in Algorithm 1

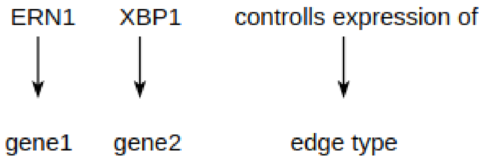


Figure 2.1: Sample line of interaction taken from condition specific network file

2.2.2 Representing Networks

In order to count the graphlets, graph representation of each condition specific network is concerned. Our implementation is based on the adjacency-list approach. Each multi labelled directed edge between protein pairs is also called 2-node graphlets. The 2-node graphlets are expressed by an 8-bit edge encoding vector. In this part of the study, we consider 5 different types of edges including 2 undirected and 3 directed types. Figure 2.2 shows the assignment of edge labels to specific bits in the 8-bit edge encoding vector. The first 2 bits of the edge encoding vector represent the undirected edge types. In order to consider the orientation of directed edges, the first 3 bits of the remaining part of the edge vector represent the outgoing edges, and the last 3 bits represent the incoming edges of the source node.

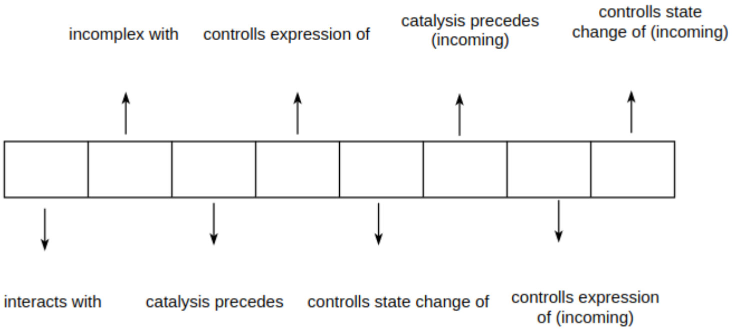

incomplex with
controlls expression of
catalysis precedes (incoming)
controlls state change of (incoming)
interacts with
catalysis precedes
controlls state change of
controlls expression of (incoming)

Figure 2.2: The assignment of edge labels

For the set of interactions between the genes A and B as given in Figure 2.3, in parallel with our adjacency-list based implementation, the encoded edge from protein A to protein B is 01010100 which appears in the adjacency list of protein A. The same edge is represented in the adjacency list of protein B by the edge vector of 01100010. These two edges are called symmetric edges and both edge vectors are kept by the adjacency list of both genes separately.

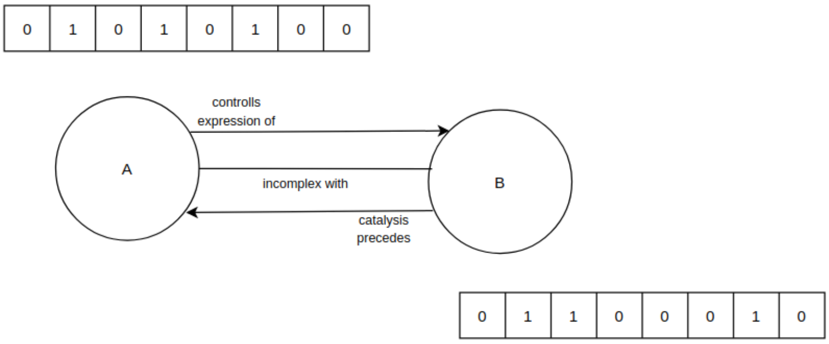


Figure 2.3: The 2-node directed multilabeled graphlet between A and B. The symmetric edge vectors associated with each interacting gene is stored separately.

Algorithm 2 shows the steps of the method which includes the process of constructing edge vectors by considering the edges between gene/protein pairs and storing the vertices of the graph and their neighborhoods with associated edge vectors. The network is stored in a data structure called hash table. Each time a line of condition specific interaction is processed, each individual gene is used as a key to map a value. Here gene1 refers to the source node (first node in interaction line in 2.1) and gene 2 refers to the target node (second node in in interaction line in 2.1) The value mapped by a gene is another hashtable including the target gene and the associated edge encoding vector as (key,value) pair. It refers to the variable "Map" in Algorithm 2.

Line 3 indicates the condition that if it is the first time a gene is mapping a value. An 8-bit edge vector is initially assigned to 00000000, assuming that there is no interaction associated with the considered gene yet. The type of the edge occurring at the line of interaction as shown in Figure 2.2 is used to specify which bit of the edge vector is going to be set to 1. Line 6 shows that, after modifying the edge vector, a new hash table (Map) is defined to store (target node, edge vector) pair as an entry. Then a globally defined hash table (H) representing the condition specific network is modified by inserting the new entry as shown in Line 7.

Line 8 indicates the condition in which the considered gene is stored as a key previously and mapped a specific value. That means there is an entry in the hash table with the considered gene and the existing edge vector should be modified because of the the recent line of interaction. So, the value is retrieved as in Line 9 first, and if the retrieved value does not include the target gene as a key, a new record with the key of target gene (gene2) and the edge vector corresponds to the the wirings between them is created. Besides, if there exists an entry associated with the target gene as a key, the corresponding edge vector is retrieved and modified according to the new edge expressed in the input line. Then the recent mapping is removed and the current mapping is inserted.

Same steps are repeated for the target gene. But this time, while constructing the edge vector, if the encountered edge is an incoming edge for the target node, last three bits of the edge vector with associated edge label should be set to 1. Figure 2.4 depicts the process for three sample lines of interaction pertaining to a condition specific network file.

Algorithm 2 Graph Representation

```
Input: condition specific network
Output: network hashtable
$H$ ←hashtable
for (gene1,gene2,edge) $\in network$ do
    if $gene1$ is not a key stored in $H$ then
        $edgeVector$ ←00000000
        $edgeVector[edgeIndex]$ ←1
        $Map$.put($gene2$,$edgeVector$)
        $H$.put($gene1$,$Map$)
    else
        $Map$ ←H.$get$(gene1)
        if $gene2$ is not a key stored in $Map$ then
            $edgeVector$ ←00000000
            $edgeVector[edgeIndex]$ ←1
            $Map$.put($gene2$,$edgeVector$)
        else
            $edgeVector$ ←Map.$get$(gene2)
            $edgeVector[edgeIndex]$ ←1
            $Map$.remove($gene2$)
            $Map$.put($gene2$,$edgeVector$)
        end if
    end for=0
```

Each iteration deals with a single line of condition specific interaction including interacting nodes and relation between them. First iteration processes the first line representing protein A controls a conversion or a template reaction that changes expression of protein B. As shown in Figure 2.4, protein A is assigned as a key first, and an entry with the target node B/edge vector pair is created and stored as a value. Since the outgoing "control expression of" interaction is represented by the fourth bit of the edge vector, this bit is set to 1. Despite it is not indicated in Algorithm 2, same steps in the loop should also be repeated for target node (gene2) to create a separate hash entry. To accomplish this, the network hashtable is investigated whether there exists a hash entry associated with the target node. Since it is not the case in the current iteration of 2.4, protein B is assigned as a key and mapped to the value of target node A/edge vector pair. As mentioned earlier, the last three bits represent the incoming edges of the source node. Since the "controls expression of" relation is the incoming edge for protein B, and the protein B is the source node in the current condition, the seventh bit of the edge encoding vector is set to 1. As a result of the first iteration, two separate hash entries are created for each interacting node. Each entry includes the value representing target node and collection of encountered edges between the source node (key) and the target node.

A	B	control expression of
A	B	in complex with
B	A	catalysis precedes

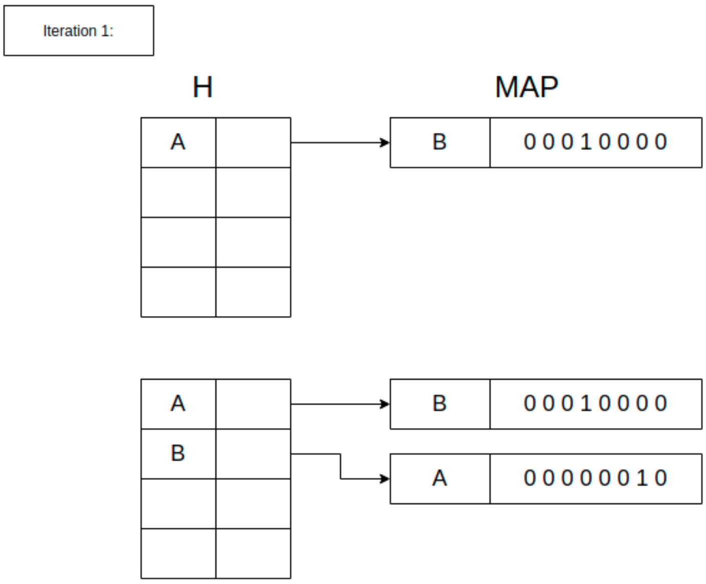


Figure 2.4: Result of first iteration creating separate hash entries for both proteins/genes.

Figure 2.5 shows the second iteration dealing with the second line of interaction. Second iteration processes the second line representing that the proteins are members of the same complex. This time, since A is a pre-stored key in the hash table, the value mapped by it is retrieved. Retrieved entry has also another entry with target node B as a key. So, the edge vector expressing the interactions between nodes A and B is retrieved and modified by setting the bit related to the undirected edge type “incomplex with”. Since it is an undirected edge type, the first two bits of the edge vector should be considered and the second bit corresponding to "interacts-with" type of relation is set to 1. After modifying existing edge vector by setting the second bit to 1, the existing hash entry is removed and a new entry with modified edge vector is placed instead. Same processes are repeated for protein B in the same iteration.

Finally, the third iteration processes the third line that represents protein B controls a reaction whose output molecule is input to another reaction controlled by the protein A. Since entries mapped by both proteins are created at the first iteration, no more entries are created and just the edge vectors are modified as shown in Figure 2.6. The predefined separate hash entries are modified by retrieving the edge vectors and setting the associated bits respectively. Since it is an outgoing relation from protein B, the third bit is modified in the edge vector whose source node is B and the target node is A. Similarly, the sixth bit is modified in the edge vector whose source node is A and the target node is B.

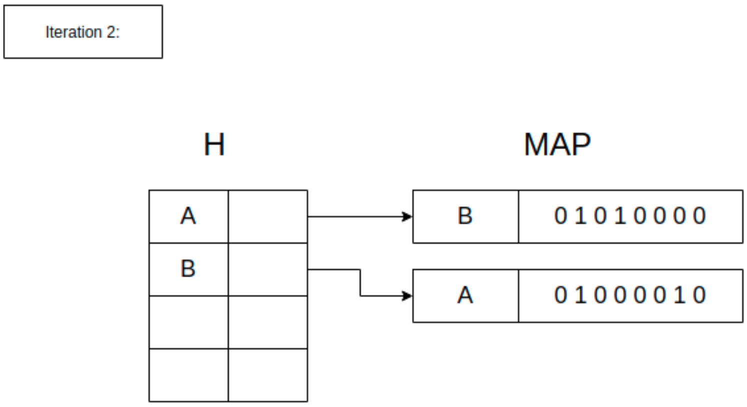


Figure 2.5: Result of second iteration showing how the existing entries are modified.

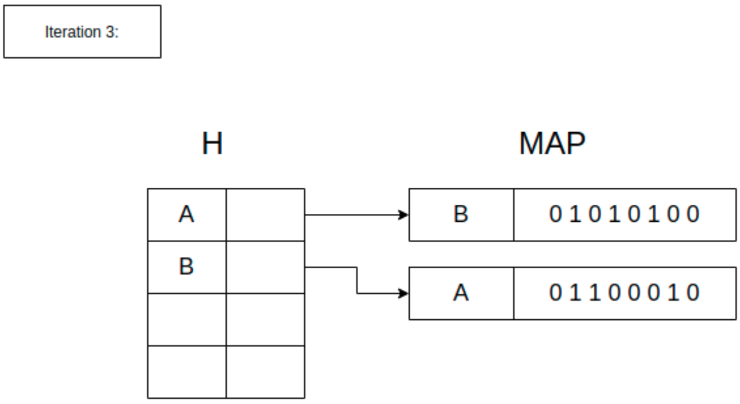


Figure 2.6: Result of third iteration showing how the existing entries are modified.

2.2.3 Counting Directed Graphlets

The method described in section 2.2.1 produces 141 different condition specific network files. Graphlet counts are the basic statistics for the comparative analysis of networks adopted by our approach. In this part of the study, our method counts up to 3-node graphlets in each condition specific network. We implement a hash-based strategy for counting the graphlets. The counting strategy is a naïve brute-force method with the time complexity of $\mathcal{O}(|V|.d^{k-1})$, where V is the set of genes in the network, k is the maximum size of the graphlet and d is the average degree in the network.

Algorithm 3 Graphlet Counting

```
Input: Hashtable: $H$
Output: Hashtable of 2-3 node graphlets
initialise $allEdges, count2node, count3node$ as Hashtables
for each gene G1 ∈H.$keys()$ do
    $Map1$ ←H.$get$(G1)
    for each gene G2 ∈Map1.$keys()$ do
        $edgeVec1$ ←Map1.$get$(G2)
        if $edgeVec1$ is not a key stored in $allEdges$ then
            insert $edgeVec1$ to $allEdges$
        end if
        if $edgeVec1$ is not a key stored in $count2node$ then
            $count2node$.put($edgeVec1$,1)
        else
            $count$ ←count2node.$get$(edgeVec1)
            ++$count$
            $count2node$.remove($edgeVec1$)
            $count2node$.put($edgeVec1$,$count$)
        end if
        $Map2$ ←H.$get$(G2)
        for each gene $G3$ ∈Map2.$keys()$ do
            if $G3$ = $G1$ then
                continue
            end if
            $edgeVec2$ ←Map2.$get$(G3)
            $Map3$ ←H.$get$(G3)
            $edgeVec3$ ←00000000
            if $G1$ is a key stored in $Map3$ then
                $edgeVec3$ ←Map3.$get$(G1)
            end if
            if $edgeVec1 + edgeVec2 + edgeVec3$ is not a key stored in $allEdges$ then
                insert $edgeVec1 + edgeVec2 + edgeVec3$ to $allEdges$
            end if
            if $edgeVec1 + edgeVec2 + edgeVec3$ is not a key stored in $count3node$ then
                $count3node$.put($edgeVec1 + edgeVec2 + edgeVec3$,1)
            else
                $count$←count3node.$get$(edgeVec1+edgeVec2+edgeVec3)
                ++$count$
                $count3node$.remove($edgeVec1 + edgeVec2 + edgeVec3$)
                $count3node$.put($edgeVec1 + edgeVec2 + edgeVec3$,$count$)
            end for=0
```

A 2-node graphlet is an 8-bit encoded edge vector between node pairs. The hash-based method for counting graphlets assign the edge vector as a key in a standard implementation of hash tables in the Java standard library by using the default hash function for strings. Accordingly, 3-node graphlets between nodes are represented by a 24 bit edge vector. Each 8 bit of the whole vector corresponds to the 2-node graphlets between each node pair. The last 8-bits representing the edge between first and third node is assigned to 00000000 if the adjacency list of each does not include the other. That means if there is no connection between the first and third node, the part of the edge vector associated with these nodes becomes 00000000. Algorithm 3 shows the steps of graphlet counting method.

The algorithm uses the network hashtable produced by Algorithm 2 as an input. It iterates through the keys of the network hashtable (Line 2). For counting 2-node graphlets, the two-level-deep neighbors of the visited node (key of the network hash table) are also visited by another loop (Line 4). The line 5 denotes the process of retrieving the edge vector including the interactions between the source node (G1) and one of its neighbor nodes (G2). The edge vectors representing the graphlets are assigned as a key and mapped the count of the edge vector as a value. Similarly, for counting 3-node graphlets, the algorithm is required to investigate up to three-level-deep neighborhood of the protein (gene) G1. In the same iteration, the counter of the edge vector is incremented if it is encountered before (Line 12-15). If it is the first time the edge vector is detected (Line 9), it is assigned as a key of the hash table storing the 2-node or 3-node graphlets encountered in the condition specific network.

Line 17 retrieves the values mapped by one of the two-level-deep neighbors of G1 in network hashtable. The retrieved value is also a hash table and the keys of the hash table refer to the three-level-deep neighborhood of the first node (G1). Since the first node might be in the neighborhood of its own neighbors (G2 or G3), line 19 checks if the neighbor node is similar to itself. Line 22 retrieves the edge vector between the neighbor of the first node (G1-G2) and the neighbor of the second node (G2-G3). Line 23-25 retrieves the neighborhood of the third node (G3) and checks if it includes first node (G1) as a key. If it exists in the neighborhood, the edge vector between them is retrieved. If it does not exists in the neighborhood, the edge vector is assigned to 00000000. Line 31 shows how three 8-bit vectors are combined to form a 24-bit edge vector and assigned as a key. The 24-bit edge vector is mapped to the count of the vector and stored in a hash table of 3-node graphlets (line 32).

2.2.4 Identifying Isomorphic Graphlets

Each edge between protein pairs is represented by a specific bit in an edge vector. Additionally, both of the interacting proteins are associated with a hash entry in the hashtable representing the network. While the encoded edges from source node to target node are stored in the adjacency list of source protein, the same set of edges are also stored in the adjacency list of target protein by a separate edge vector. Despite these two edge vectors having the same set of edges, their representations are different from each other. The two edge vectors between the same protein pairs are called symmetric edges. Since the various orderings of the same nodes cause the different edge orientations, the edge vector may comprise the different combinations of symmetric edges which constitute isomorphic graphlets.

In order to count the graphlets accurately, the isomorphic graphlets should be considered. Otherwise, the same graphlet can be counted more than once since the same graphlet between same pair of nodes is represented by different edge vectors w.r.t the endpoints. Our method takes care of the isomorphic graphlets by dealing with the symmetric edges at the post-processing stage.

Algorithm 4 shows how to form the symmetric edge of each input vector (2-node graphlet) and use them to find out whether 2-node graphlets are isomorphic. Line 1 shows the condition that the symmetric edge vector of the first input eqauls the second input. getSymmetric() function in Algorithm 4 returns the symmetric edge vector of an 8-bit edge vector. That helper function considers each bit of the vector successively. Since the first two bits of the vector represent the undirected edges and the does not change w.r.t the source or target nodes, the content of the first two bits are the same as the ones of the input vector. On the other hand, each successive three bits represent the directed interactions from source to target node and from target to source node respectively. For producing the symmetric edges, we need to swap the endpoints of the edges, so the getSymmetric() method swaps the 3-bit blocks as shown in Figure 2.7.

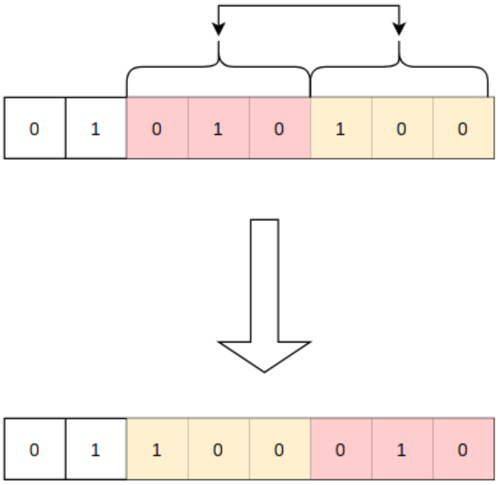


Figure 2.7: Generating symmetric edge

Algorithm 4 Identifying 2-node Isomorphic Graphlets

```
Input: Graphlets to be compared: $edgeVector1$,$edgeVector2$
Output: returns True if the graphlets are isomorphic
if $edgeVector1$ == getSymmetric($edgeVector2$) then
    Graphlets are isomorphic
else
    Graphlets are not isomorphic
end if=0
```

Similarly isomorphic 3-node graphlets are handled in order not to be counted multiple times. The Algorithm 5 shows the steps of the approach. The two 24-bit edge vectors are used as an input to be compared for the method. Since a 3-node graphlet is basically a union of three 8-bit edge vectors, each 8-bit part of one of the 3-node input graphlet is splitted into three 8-bit parts to get the symmetric edge vector of each (Line1-3).

Algorithm 5 Identifying 3-node Isomorphic Graphlets

```
Input: Graphlets to be tested:$edgeVector1$,$edgeVector2$
Output: returns True if the graphlets are isomorphic
1: $X \leftarrow edgeVector1[0:7]$
2: $Y \leftarrow edgeVector1[8:15]$
3: $Z \leftarrow edgeVector1[16:23]$
4: if $edgeVector2 == X + Y + Z$ ||
   $edgeVector2 == getSymmetric(Z) + getSymmetric(Y) + getSymmetric(Z)$ ||
   $edgeVector2 == getSymmetric(X) + getSymmetric(Z) + getSymmetric(Y)$ ||
   $edgeVector2 == Y + Z + X$ ||
   $edgeVector2 == Z + X + Y$ ||
   $edgeVector2 == getSymmetric(Y) + getSymmetric(X) + getSymmetric(ZY)$ then
5:     return true
6: else
       False
7: end if=0
```

As the symmetric of each 8-bit subpart of the 24-bit 3-node graphlet is produced, 8-bit subparts are joined according to the different combinations of constituting nodes (Line 4) as shown in Figure 2.8. As a result of the node orderings, six different 3-node graphlets occur. If there is a match between one of the edge combinations and the other input (edgeVector2), that means the graphlets are isomorphic and one of the input vector is not considered and does not change the count of the graphlet.

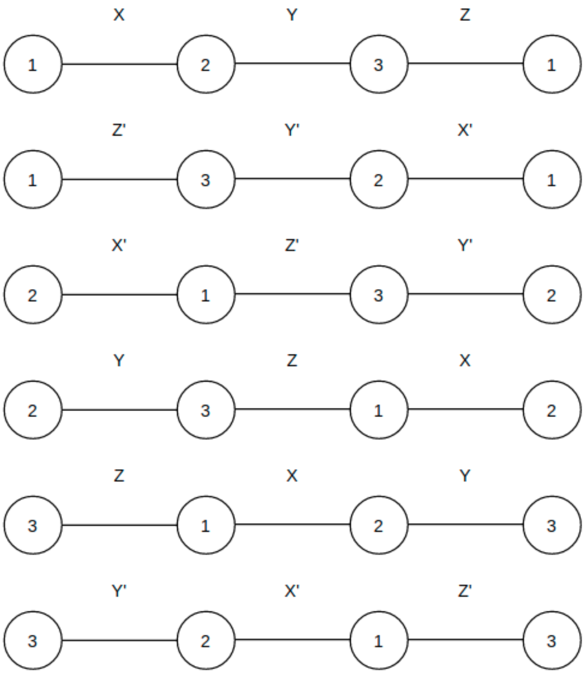


Figure 2.8: All possible 3-node graphlets produced by different node orderings.

2.2.5 Computing Statistically Significant Graphlets

After identifying the existing graphlets and eliminating the isomorphic ones in each condition specific network, the collection of encountered graphlet counts are used to represent each condition specific network. Additionally, the graphlet signature vector is obtained by computing the z-score of each graphlet existing in a condition specific network. In order to compute the z-score of each graphlet. First a random network corresponding to each condition specific network is constructed without modifying the degree distribution of it. That means, the degree distribution of the multi labelled directed edges of a node does not change. The randomization process of 141 condition specific networks is handled by edge shuffling method. This method shuffles randomly chosen pairs of edges with the same set of labels in order to create a same set on interactions with different interacting partners.

Algorithm 6 Create Random Network

```
Input: condition specific network
Output: random network file
$RND \leftarrow Hashtable$
$pairs \leftarrow Vector$
for (gene1,gene2,edge) $\in network$ do
    if $edge$ is not a key stored in $RND$ then
        $pairs$.addElement($gene1, gene2$)
        $RND$.put($edge$,$pairs$)
    else
        $pairs$ ←RND.$get$(edge)
        $pairs$.addElement($gene1, gene2$)
    end for
    for each edge ∈RND.$keys()$ do
        $pairs$ ←RND.$get(edge)$
        while $pairs$.size()>0 do
            $index1$=rand($pairs$.size())
            $index2$=rand($pairs$.size())
            $pair1$=$pairs$.elementAt($index1$))
            $pair2$=$pairs$.elementAt($index2$))
            Be sure that $pair1$ and $pair2$ have four distinct elements
            Generate a new interaction by shuffling the elements of $pair1$ and $pair2$
            new interaction line← $(pair1[0], pair2[1], edge)$
            new interaction line← $(pair2[0], pair1[1], edge)$
            remove the original pairs
        end for=0
```

Algorithm 6 shows the steps of creating a random network for a single condition specific network. Two types of data structure are used in the method. The hash table is holding the node pairs and the associated edges of the network while the vector is holding only the interacting pairs. Each element of the vector is a two element tuple including the endpoints of a specific edge. The vector is assigned as a value of the network hashtable and mapped by the edge associated with them (key).

Generating a random network is actually a process of creating a network file of modified interactions. The difference of the content of the network file from the original condition specific file is the shuffled interacting pairs. Steps 14-17 shows the process of retrieving the random node pairs mapped by the similar directed multi labeled edge. In order to avoid from self loops, the method makes sure that the interacting pairs to be shuffled are distinct from each other. After the random networks are produced, each of them is processed by the same methods described in previous section for counting the 2- and 3- node graphlets within the random networks. In order to compute the z-score of a specific graphlet, the count of the graphlet in a real condition specific network is compared against the collection of the counts in all 141 random networks. This process actually finds out how much the count of the graphlet in real network differs from its counts encountered in random networks. The graphlets with high z-scores are considered as statistically significant graphlets. The vector of graphlet z-scores representing each condition specific network is called graphlet signatures. Next section shows the results of the comparative analysis of condition specific networks by using graphlet signatures.

2.3 Results

The proposed counting method is a brute force method yielding the inefficient time complexity with increasing amount of graphlet sizes. Despite its expense for larger sized graphlets, the method is the first attempt making the comparative analysis of condition specific networks including integrated types of interactions. Output of the graphlet counting process shows that, there exists 56 different 2-node graphlets within the analyzed networks. 12 of the 56 different 2-node graphlets are recurring more than 1000 times while 28 of them are recurring more than 100 times on average within the 141 different condition specific real networks. Additionally, there exist 7346 different types of 3-node graphlets in the entire networks. It is also important to indicate that the subset of the labels constituting a multi label edge do not considered while counting the most generic graphlets. That means the components of the directed multi label edges are not accounted for the most specific occurrence of the graphlet.

The statistical significance of each graphlet is also calculated by comparing the count of each in real networks with count of it which occurs in the collection of randomized networks. This process gives the z-score of each graphlet and the 1-D vector of z-score values is used to represent each condition specific network. The vector is called graphlet signature of the represented network. The results show that there exists graphlets with zero counts in randomized networks and they are eliminated since we cannot assign a numerical value other than infinity. There are 2766 such graphlets with z-score of infinity. Eventually, 275 of 4636 remaining graphlets have z-score over 100.

In Figure 2.9, the most significant graphlet with the average z-sore of 6913.36 across 141 different conditions is given. While the frequency of this graphlet in real networks is more than 50000 times, it occurs only 10.35 times in average across the randomized networks. It is a three node graphlet which reveals a triangle pattern. A pair of nodes in the graphlet are the member of a same protein complex. Additionally both members are having mutual catalysis-precedes relation with a common neighbour that means each member of the protein complex reciprocally controls consecutive reactions with their common neighbour. The relations appeared in the graphlet is also an interesting biological finding showing such a high amount of protein pairs both precede and succeed each other in catalyzing a cascade of metabolic reactions.

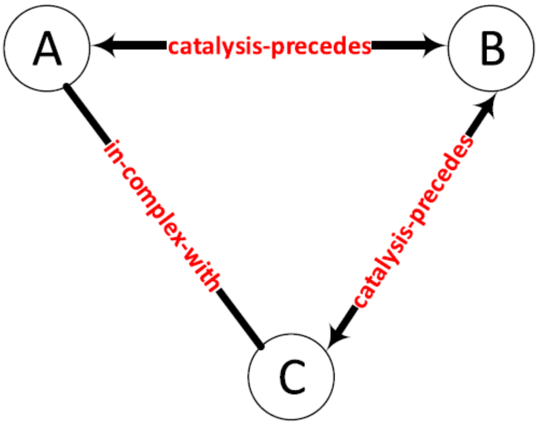


Figure 2.9: The most significant graphlet with the average z-score of 6913.36. It occurs more than fifty thousand times in real networks.

We also assessed the statistically significant graphlets for finding the graphlets that show high z-score variation. Figure 2.10 shows the graphlet that shows the highest variations across 141 conditions. The average z-score of the graphlet is 4656.97 with a standard deviation of 1570.49. All nodes of the graphlets are the members of the same protein complex and the components are also forming a feed-forward motif of change state interaction. The graphlet occurs most frequently in a diseased state of myometrium tissue (MDT) with a z-score of 7999.00. The least frequent occurrence of the same graphlet is in normal state of retrocervical infiltrate tissue (RINT) with the z-score of 1702.81. There exists a four-fold decrease in the frequency of the same graphlet between these two conditions. The distinction between the number of active genes in the conditions can be considered as a source of the frequency differences. However, while MDT network has less number of active genes than the control sample of MDA-MB231 cells, it also has 688 more occurrences of this graphlet compared to the MDA-MB231 tissue.

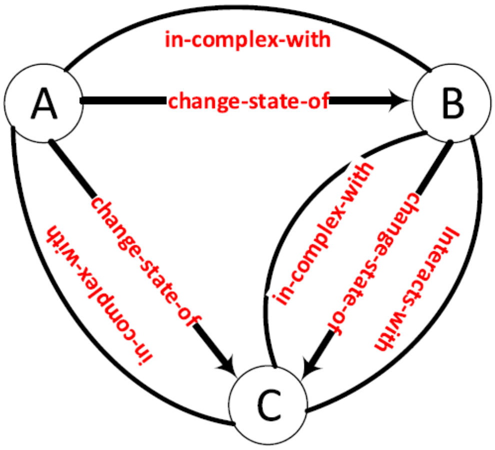


Figure 2.10: The 3-node graphlet with the highest z-score variation. The average z-score and the standard deviation are 4656.97 and 1570.49 respectively

Another high variant graphlet occurring among the condition specific networks is shown in 2.11. The two of the proteins are the components of the same protein complex while all three proteins mutually has an effect on an event changing the state of each other. The average occurrence of the graphlet in the randomized network is 2.0. However, it occurs in the endometrium ovary disease tissue (EODT) network 7564 times, while, in the skeletal muscle superior quadracep normal tissue (SMSQNT) it recurrs 1378 times.

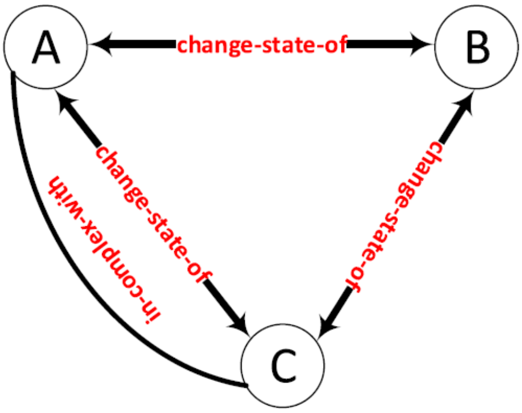


Figure 2.11: Statistically significant 3-node graphlet with high variance of z-score. The average occurrence in random networks is 2.0

All 141 condition-specific networks are represented by a one-dimensional vector of z-score of each graphlet in the networks. In order to evaluate the efficiency of the graphlet signatures for clustering conditions, the z-scores of the graphlets are used as input for TM4 MeV: MultiExperiment Viewer tool (Saeed et al., 2003) which is originally implemented for the analysis of microarray data, however any dataset with a set of objects represented by vectors can be analyzed by applying standard methods such as PCA and clustering. We performed PCA-based reduction on the graphlet signatures of all 141 condition specific networks. As a result, the first three principal components were able to capture 96.126% of the variance in the dataset that shows despite the thousands of different types of graphlets occur in the networks, the graphlet frequencies are highly correlated and much lower dimensions of the features can be used to represent the networks.

Another analysis is conducted on a part of the 141 condition-specific networks in order to show whether the clustering methods effectively partition the dataset by using graphlet signatures . The subset of 21 brain tissue is selected and hierarchical clustering is applied to the subset. Figure 2.12 shows the result of the analysis. The results show that the brain parts with similar functions are clustered together. Four of the five thalamus tissues are in the same subcluster if the tree is cut at a level to produce four main clusters. Another analysis is conducted by using the values in microarray dataset GSE7303 to cluster the brain tissues and the thalamus tissues fall in different clusters. The outcome of the analysis shows the discriminative power of statistically significant directed multi labeled graphlets on the comparative analysis of distinct conditions.

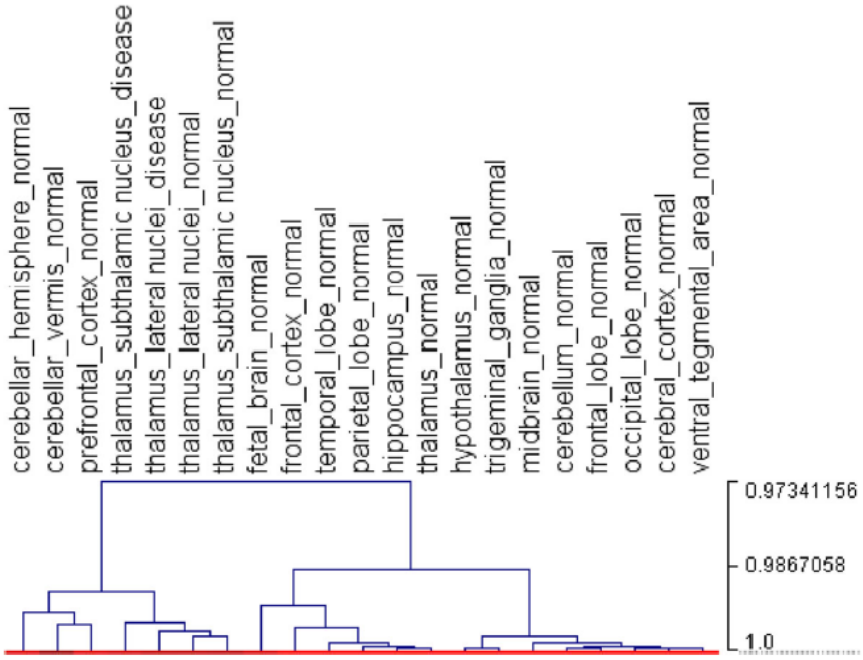


Figure 2.12: The result of hierarchical clustering of 21 brain tissues by using graphlet signatures.

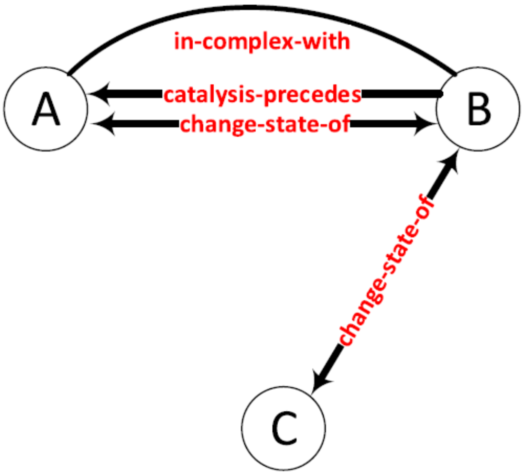


Figure 2.13: The differentially recurring graphlet in the normal and disease states of thalamus lateral nuclei tissue network.

Eventually, the graphlet signatures are used to compare the different states of the same tissue. The diseased and normal states of the prostate gland, thalamus lateral nuclei, and skin tissues are used for the analysis. The target of the analysis is to identify some differentially recurring graphlets across different states of the same tissue. Additionally, the proteins inducing the differentially recurring graphlets are analyzed for functional enrichment using DAVID functional annotation tool (Huang, Sherman, & Lempicki, 2009).

Figure 2.13 shows the graphlet that has the highest variation between the disease and the normal state of thalamus lateral nuclei tissue. All interacting proteins inducing the graphlet control a reaction that changes the state of each other. Additionally, 2 out of the 3 nodes are members of the same protein complex. And also, one of the components of the protein complex is controlling a reaction whose output molecule is used as an input to another reaction which is controlled by the other component of the same protein complex. The graphlet's z-score in the diseased network is -0.08, while it is represented by the z-score of 43.40 in the normal state of the same tissue. And also the four-fold difference is observed in the frequency of this graphlet in the network of each state.

This graphlet is induced by only six different proteins in the disease state of the thalamus lateral nuclei tissue. On the other hand, there exists 43 different proteins inducing the same graphlet in the normal state of the same tissue. There are four common proteins between the inducing protein sets. After excluding these four proteins from the set of 43 graphlet activating proteins in normal tissue, the identifiers of the remaining proteins used as an input to the DAVID tool for functional enrichment. The Table 3 shows the top three most significantly enriched functions. The count values show how many proteins out of the entire input set are involved in that specific function. The table shows that the activating proteins are related to the growth factor and positive regulation of cell division. The functional enrichment results indicated that possible defection of these functions may cause tissue be in the the disease state.

Table 3: Top three clusters of functional annotation terms related to the graphlet in Figure 2.13

Annotation Cluster 1	Enrichment Score:20.21		
Catgory	Term	Count	P-value
GOTERM_BP_FAT	fibroblast growth factor receptor signaling pathway	13	2.3E-25
KEGG_PATHWAY	hsa04010:MAPK signaling pathway	13	4.8E-12
Annotation Cluster 2	Enrichment Score:11.04		
Catgory	Term	Count	P-value
KEGG_PATHWAY	hsa05218:Melanoma	10	5.3E-13
SP_PIR_KEYWORDS	growth factor	10	1.9E-12
GOTERM_MF_FAT	growth factor activity	10	1.2E-10
Annotation Cluster 3	Enrichment Score: 9.97		
Category	Term	Count	P-value
PIR_SUPERFAMILY	PIRSF001783:fibroblast growth factor	7	6.9E-14
SP_PIR_KEYWORDS	mitogen	7	8.7E-11
GOTERM_BP_FAT	positive regulation of cell division	7	4.0E-10
GOTERM_BP_FAT	regulation of cell division	7	1.3E-9

The disease and normal state of prostate gland tissue are also compared for detecting differentially recurring graphlets. The graphlet shown in Figure 2.14 is counted in a diseased state nine times more than in the normal state of prostate gland tissue. The z-score of the graphlet in the diseased network is 68.68, while it is 7.63 in the normal network. The control state change of relation exhibits the feed-forward loop motif. There also exists a single pair of proteins which are the elements of the same protein complex. Additionally, there exists a single node(protein) which is controlling the reactions that not only changes the state of its neighbor but also catalyzes a reaction before it.

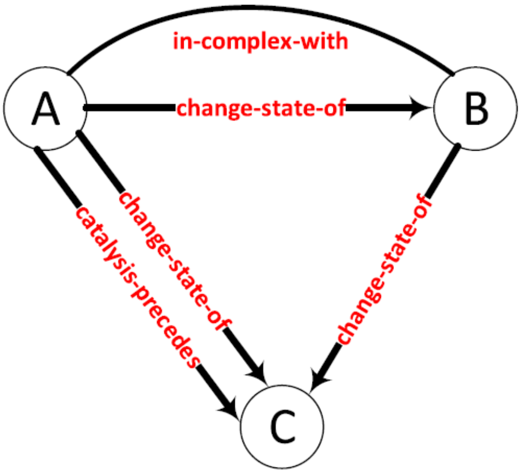


Figure 2.14: The differentially recurring graphlet in the normal and disease states of prostate gland tissue network.

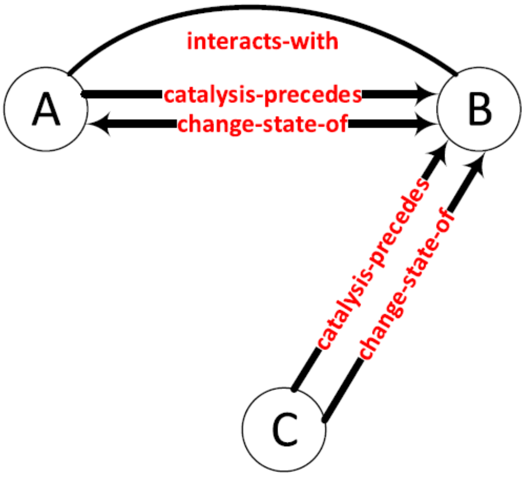


Figure 2.15: The differentially recurring graphlet in the normal and diseased states of skin tissue network.

There are 74 different proteins activating this graphlet in the diseased state of the prostate gland tissue. 28 of them also activate the same graphlet in the normal state of the same tissue. There remain 46 proteins specific to the diseased state. Table 4 shows the top three functional annotation clusters of the 46 disease state-specific proteins. The transmembrane receptor protein tyrosine kinase signaling pathway, growth factor activity, and positive regulation of cell division functions are highly associated with the 46 disease state-specific proteins.

Finally, both the normal and diseased states of skin tissue are compared to identify a differentially recurring statistically significant graphlet. Apart from the other analyzed tissues, the detected differential pattern does exist only in normal state. Figure 2.15 shows the graphlet that occurs only in the normal state of the skin tissue. A total of 69 proteins inducing the graphlet are also used for the functional enrichment analysis. Table 5 shows the result of the analysis. According to the results, most of the proteins take part in the transmembrane transport activities. The lack of these functions in diseased skin tissue may give an insight to understand the underlying factors of the disease.

Table 4: Top three clusters of functional annotation terms related to the graphlet in Figure 2.14

Annotation Cluster 1	Enrichment Score:17.90		
Catgory	Term	Count	P-value
GOTERM_BP_FAT	TM receptor protein tyrosine kinase signaling pathway	18	5.0E-20
GOTERM_BP_FAT	enzyme linked receptor protein signaling pathway	18	6.6E-17
Annotation Cluster 2	Enrichment Score:16.20		
Catgory	Term	Count	P-value
SP_PIR_KEYWORDS	growth factor	14	1.6E-18
GOTERM_MF_FAT	growth factor activity	14	6.0E-16
Annotation Cluster 3	Enrichment Score: 8.72		
Category	Term	Count	P-value
SP_PIR_KEYWORDS	mitogen	7	3.1E-10
GOTERM_BP_FAT	positive regulation of cell division	7	1.9E-9
GOTERM_BP_FAT	regulation of cell division	7	6.03E-9

2.4 Conclusion

In this chapter, the idea of graphlet counting in undirected protein-protein interaction networks is utilized and extended to directed multi labeled networks. The presented counting method is applied on 141 different condition specific networks filtered from an integrated network of physical, metabolic and regulatory interactions obtained from Pathway Commons database. Each condition specific network is represented as a vector of graphlet counts and the statistically significant graphlets are identified to be used for clustering and system level analysis of condition specific networks. The method presented is brute-force counting method which may not the computationally most efficient solution for counting the graphlets of larger sizes, however, the study is the first attempt dealing the multi labeled directed graphlets in genome scale integrated networks and results of the study shows that the biological significance of differential graphlets must be researched further before they can be used as useful biomarkers at the network level.

Table 5: Top three clusters of functional annotation terms related to the graphlet in Figure 2.15

Annotation Cluster 1	Enrichment Score:76.88		
Catgory	Term	Count	P-value
GOTERM_MF_FAT	ion channel activity	57	1.4E-77
GOTERM_MF_FAT	substrate specific channel activity	57	8.5E-77
GOTERM_MF_FAT	channel activity	57	6.8E-76
GOTERM_MF_FAT	passive transmembrane transporter activity	57	7.8E-76
Annotation Cluster 2	Enrichment Score:75.69		
Catgory	Term	Count	P-value
GOTERM_MF_FATS	metal ion transmembrane transporter activity	56	1.8E-79
GOTERM_MF_ALL	growth factor activity	56	3.6E-68
Annotation Cluster 3	Enrichment Score: 73.73		
Category	Term	Count	P-value
GOTERM_BP_FAT	calcium ion transport	52	4.5E-91
GOTERM_BP_FAT	di-, tri-valent inorganic cation transport	52	2.3E-85
GOTERM_BP_FAT	metal ion transport	52	9.7E-62
GOTERM_BP_FAT	rcation transport	52	9.6E-58

CHAPTER 3

IDENTIFICATION OF 3- AND 4-NODE MULTI-LABELED GRAPHLETS IN INTEGRATED HUMAN PROTEIN-PROTEIN INTERACTION NETWORK

In the previous chapter, we propose a method counting the 2- and 3-node graphlets and yielding a graphlet signature vector that represents distinct condition specific networks. The signature vector is employed to detect the statistically significant graphlets recurring in analyzed networks. Graphlet based network statistics maintain a detailed description of the network topology. The statistics may refer to the frequency of a graphlet either in a network or touched by a network node. In the previous chapter, we focus only on the network based graphlet frequencies. However, in this chapter, node based graphlet statistics are considered.

Since larger graphlets are formed by the combination of small graphlets, we can not assume that the different graphlets are independent. The method proposed by this chapter also identifies the six different types of 4-node graphlets occurring in the integrated protein interaction networks. The approach involves consecutive merge operations on the set of 3-node graphlets to identify the 4-node graphlets.

3.1 Methods

In this section, the details of the proposed approach is described. Below list shows each process conducted by the method.
1) Generation of the set of all possible 3-node multi labeled graphlets consisting both directed and undirected edges types.
2) Identification of isomorphic graphlets in the 3-node graphlet set and grouping them into isomorphic classes.
3) Generation of node neighbourhoods for all unique nodes in the integrated Pathway Commons data set.
4) Identification of 3-node wedge and triangle patterns appeared in the node neighborhoods
5) 4-node pattern detection by using the 3-node pattern combinations.

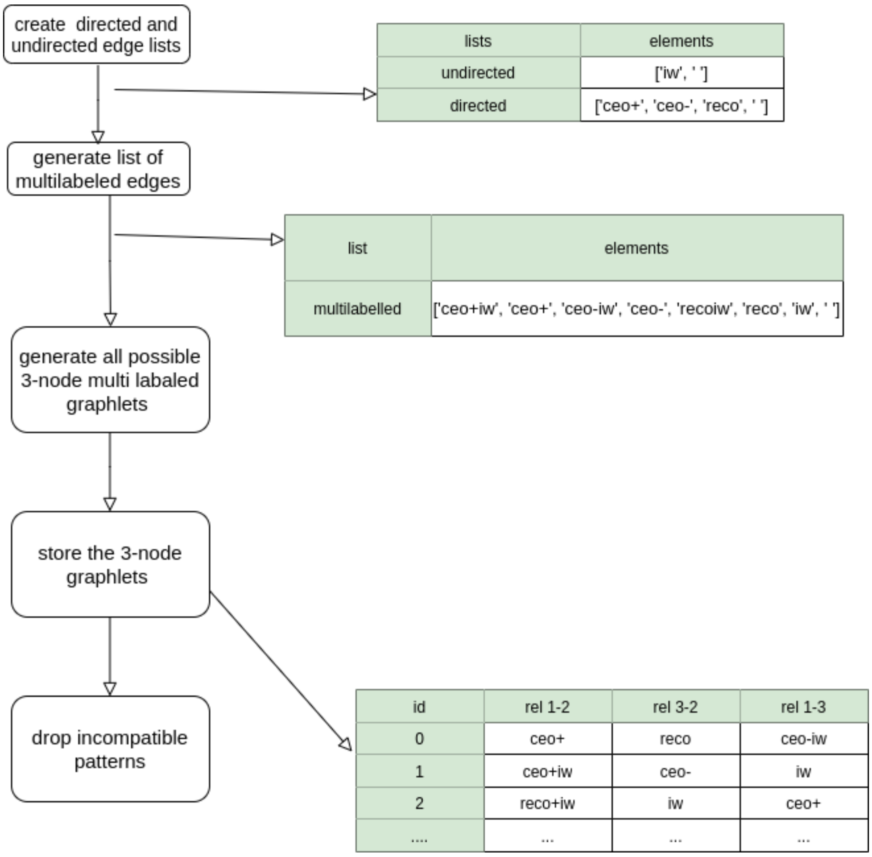
Python dictionary structure and Python pandas are frequently used for this part of the study. Python pandas library is a popular tool lately, especially for data scientists and analysts. As well as, ease of use, it is an open-source package offering flexible data structures to analyze labelled and relational data of different types. On top of that, it provides convenience for manipulating data by merge operations. Python dictionary is also an important data structure that speeds up the processes by assigning key-value pair for each item. So, it just take a constant time to look up a key and find the associated value. It also have some disadvantages since it takes up a lot more space as number of keys increase. But this drawback can be solved by another Python module called "shelve". This module includes dictionary-like shelf object which is persistently stored in a disk file. Additionally, using ’category’ type values in dataframes significantly reduces the memory usage.

3.1.1 Producing a Generic Set of 3-node Graphlets

In order to generate a generic set of multi labeled 3-node graphlets, directed triangles and wedges are combined with 3-node undirected subgraphs to expand the pattern set and show the actual behaviour of real-world biological networks since they naturally include both types. Devised method consists of three parts. First part is generating 3-node patterns composed of both undirected and directed relations. Pattern generation involves generating the string combinations of two types of interactions. This is achieved by first determining the set of all possible elements for each type of interaction. Undirected interaction set is a 2-element string set in which each element represents the presence and absence of the interaction between protein pairs. Besides, the directed interaction set is 4-element set, in which the incoming, outgoing, reciprocal edges and absence of any relation is represented respectively. By calculating Cartesian product of the sets, all possible string combinations involving elements of each sets are generated. List of directed and undirected edges with labels (control expression of and interact with) are given below. “controls expression of” interaction is abbreviated by “ceo” and “interacts with” type is abbreviated as “iw”.

```
dir_list = ["ceo+", "ceo-", "reco"," "];
undir_list = ["iw"," "] ;
```

3-node patterns are produced by simply using nested loops to assign the string combinations of edge labels (combined edge types) to all possible positions between node pairs. The generated 3-node graphlet set is then eliminated according to some rules and assumptions. Since our contribution is combining the directed graphlets with the undirected subgraphs to create an extended pattern set, the patterns that are either disconnected or not including directed relations in at least two edge positions are excluded. Additionally, since the implemented algorithm considers the patterns w.r.t an initiating node and determines whether the pattern is a 3-path (wedge) or a triangle by considering the presence of a connection in the third edge position (between starting node and the third node), the patterns with no connection at first (between node1 and node2) or second (between node2 and node3) edge positions are also eliminated as stated in the previous section.


create directed and undirected edge lists
generate list of multilabeled edges
generate all possible 3-node multi labaled graphlets
store the 3-node graphlets
drop incompatible patterns

lists	elements
undirected	['iw', ' ']
directed	['ceo+', 'ceo-', 'reco', ' ']

list	elements
multilabelled	['ceo+iw', 'ceo+', 'ceo-iw', 'ceo-', 'recoiw', 'reco', 'iw', ' ']

id	rel 1-2	rel 3-2	rel 1-3
0	ceo+	reco	ceo-iw
1	ceo+iw	ceo-	iw
2	reco+iw	iw	ceo+
....	...	...	...

Figure 3.1: The flowchart of generating 3-node multi labeled graphlets

While the generic 3-node graphlet set is produced, the direction of each edge between node pairs, if it is either outgoing or incoming, are determined according to a specific node of the graphlet. While the edge direction between starting node (first node) and the remaining nodes are determined w.r.t the starting node, the direction of the edge between second and third node is determined by the third node of the graphlet. As assignment and elimination processes are completed, all generated 3-node graphlets are stored in a dataframe structure. The dataframe is a 2-D tabular structure. Each row of the dataframe corresponds to a unique 3-node graphlet, while each column corresponds to a multi labeled edge occurring between the node pairs. Figure 3.1 shows the flow of the processes and some output samples as a result of the 3-node graphlet generation method.

3.1.2 Isomorphic Class Generation

There have been 360 different 3-node graphlets produced by the pattern generation method explained in the previous section. All produced patterns are stored in a dataframe structure with corresponding unique graphlet ids. However, some patterns out of the produced 360 graphlets are isomorphic.

Recall from Chapter2 that the graphlet counts are collectively used as a signature vector for network representation. Similarly they can be used to represent the nodes of the networks. The number of graphlets touched by a node might reveal the topology of a node's neighborhood and let us compare the nodes of different networks. This local topological measure is actually an extended version of node degree, however instead of counting the number of edges incident to a specific node, the counts of the graphlets touched by a node are accounted. Since an n-graphlet pattern is composed of n nodes, a network node can touch that graphlet from n different points. Some of these points are topologically similar and are the member of the same automorphism group. The automorphism is actually the permutation of the nodes without any changes in the set of the edges involving the permuted nodes. The topologically similar points are called automorphism orbits.

For the study presented in Chapter 3, our aim is to extract the node based profile of a network so instead of network based graphlet counts, node based orbit counts are considered. Since produced graphlet is assumed to be initiated from one out of three nodes involving in the considered graphlet, while the 3-node graphlet set is generated, same graphlet with different node orderings and same set of edges is also included in the generic set of 3-node graphlets. These types of graphlets are called isomorphic graphlets and grouped together under a specific isomorphic class id. This section explains the details of the isomorphic pattern detection and generating a map that engages a 3-node graphlet member to a specific class. Algorithm7 shows the details of the 3-node graphlet classification process.

Algorithm 7 Generating Isomorphic Classes

```
Input: Dataframe of 3-node graphlets
Output: Mapping between 3-node graphlets and corresponding isomorphic class
 1: INTERACTIONLIST =[], LIST=[]
 2: for each 3-node graphlet do
 3:     for each node of the graphlet do
 4:         Sort the node interactions alphanumerically
 5:         $INTERACTIONLIST \leftarrow sortedInteractions$
 6:         $LIST.append(INTERACTIONLIST)$
 7:     end for
 8:     Associate the $LIST$ with the graphlet id
 9: end for
10: for each unique 3-node graphlet id do
11:     Sort the LIST by first and second elements of the $INTERACTIONLIST$s respectively
12:     for each remaining 3-node graphlets do
13:         Repeat the same sorting process for the current graphlet
14:         if Compared $LIST$s are identical then
15:             The compared graphlet ids are assigned to be the member of the same isomorphic class
16:             Indicate the current graphlet id as "assigned" in order not to be reprocessed
17:         end if
18:     end for
19: end for=0
```

The couple of lists defined in Line 1 is used for holding the interactions associated with a single graphlet node and all the nodes of a single graphlet respectively. INTERACTIONLIST is a 2-element list holding the interactions incident to any node of a 3-node graphlet. It can be any edge type out of eight different edges produced by the combination of directed and undirected edge types. Lines 2-7 shows that while iterating over the set of 3-node graphlets, interactions involved by each node are represented by an element of INTERACTIONLIST. The interactions are sorted and reversed if it is necessary. The reverting process is needed for the interactions concerning both the second and third nodes of the 3-node graphlet, because of the distinct graphlet nodes accounted for while determining the edge directions. The interactions concerning all nodes of a specific 3-node graphlet are collected in the LIST and stored with the associated graphlet id as shown in Line 8. As a result, each unique 3-node graphlet is represented by a single LIST variable with three appended lists (INTERACTIONLIST) as elements each of which corresponds to the respective nodes of the graphlets with associated incident edge lists. (Line 2-8) Then the produced LIST variables are iterated in order find a match. In order to understand whether two patterns are isomorphic, the interaction lists (INTERACTIONLIST variables) corresponding each node of the graphlet should be sorted in the LIST variable in a way that all nodes in the LIST are in the same order. Lines 11-13 show how this process is achieved. If the LISTs are matched, the two 3-node graphlet is assigned under the same isomorphic class and the graphlet id in the inner loop is flagged not to be used in the outer loop again.

3.1.3 Generation of Node Neighborhoods in the Integrated Network

In this step of the proposed method, Pathway Commons Human PPI dataset version 11 (*The Pathway Commons Database Version 11; [cited January 28, 2019].*, 2019) which includes 1851007 interactions between protein pairs is processed to determine the neighborhoods of all unique nodes in the dataset. The collection of binary relations between protein pairs are residing in a SIF formatted file. In each line of interaction file, source node, target node and the edge type between the nodes are stated respectively. The either of outgoing or undirected type of interactions are expressed in the data file. Our method mainly focuses on two types of interactions between human genes. First one is a directed interaction called “controls expression of”. The amount of the interactions involving this edge type is 126396. Second type of interaction of interest is “interacts with”, and there exists 485608 interacting pairs with this type of interaction. After filtering the original dataset to acquire interested edge types, there remains 612004 entries including the single labeled undirected and directed edges. The aim of this step is to construct list of neighbors for each unique protein(starting node, node1) in the dataset. While constructing node neighborhoods and further detect the 3- and 4-node graphlets, we assume that they are all initiated by a node called starting (initiating) node. If a node interacts its neighbours with multiple interaction types, construction process also involves aggregation of these multiple edges and formation of multi-labeled single edges. Algorithm 8 briefly shows the steps of the process.

Line 1-2 shows a duplication and inversion processes which are necessary for creating a node neighbourhood starting from a specific node. As the original data is reversed, the target nodes in the original dataset is also appeared as a source node (starting node) in the duplicated dataset with associated incoming edges and also the undirected type of edges related with these nodes are concerned. In Line 3, the python dataframe structure is used to be loaded the extended dataset. The DATAFRAME variable is a 3-column tabular data structure that each column holds the string representation of Source node id, Target node id and the encountered edge between these nodes respectively. Edge types are represented by string abbreviations such that, "ceo" refers to "controls expression of", "reco" refers to "reciprocal controls expression of and "iw" refers to "interacts with" interactions.

Algorithm 8 Generating Node Neighborhood

```
Input: Binary interactions between proteins
Output: Neighborhood of each unique protein of the input data
 1: Duplicate pairwise interactions
 2: Reverse the columns in duplicated version
 3: $DATAFRAME \leftarrow$originalData+duplicatedData
 4: for each unique node $N$ in the $DATAFRAME$ do
 5:     Retrieve the dataframe entries with outgoing and incoming edges where Source node = $N$
 6:     $AGGREGATE \leftarrow outgoing + incoming$
 7:     Group the $AGGREGATE$ by Target nodes
 8:     if both incoming and outgoing edges have same target then
 9:         Define the edge as "Reciprocal"
10:     end if
11:     Retrieve the entries covering the undirected edges
12:     $ALL \leftarrow AGGREGATE$+$undirectededges$
13:     Group the $ALL$ by Target nodes
14:     if Both directed and undirected edges have same target then
15:         Combine the edge types and create multilabel single edge
16:     end if
17: end for=0
```

The collection of entries in a dataframe can be easily accessed. In lines 4-5, the entries where the values of the source node column corresponding a specific protein identifier and the relation column corresponding the string representations of directed edges are retrieved. Retrieved entries are aggregated and grouped by the target nodes in lines 7-8. This process reveals the entries having both incoming and outgoing "ceo" relations between same source and target nodes. The encountered interactions are then consolidated and called "reciprocal" (reco). Similarly, in line 12-16, the processes are repeated for undirected and directed (including reciprocal) edges. The result dataframe with the node ids and combined edges are stored as a Python dictionary value which is mapped by the id of the starting (source) node of the result dataframe. So, the method is mapped each unique node of the integrated network to the neighbor nodes and the associated multilabeled edges.

3.1.4 Identifying 3-node Graphlets by Using Node Neighborhoods

The generic 3-node graphlet set is composed of 3-node patterns that is either wedge or triangle. The absence or presence of the connection between starting node and third node help us to determine the type of the 3-node graphlet. For this part of the approach, the dictionary structure representing network specific node-neighborhood pair, is used to identify the wedges and triangles existing in the generic 3-node graphlet set. The two different versions of the identification method is implemented. First version is the iterative version which iterates over the set of both network nodes and generic 3-node graphlets in order to find out if there is a match between the connection pattern of each 3-node graphlet and connection pattern in a neighborhood of the specific node of the network. The second version is non-iterative implementation of the same process. This version uses the data modification method called "merge" from the Python Pandas module. To accomplish the non-iterative version, multiple dataframes are merged to find out whether the connection patterns are similar. This approach saved a lot of time since it is working 10x faster than the iterative version. Algorithm 9 shows the details of the approach.

Algorithm 9 3-Node Graphlet Identification

```
Input: Pairwise interactions of network nodes, Generic pattern set, Isomorphic class map
Output: Triangle/Wedge patterns starting from network nodes
 1: for each unique node $N$ in the network do
 2:     $allDATAFRAME$ ←all pairwise interactions of all network nodes
 3:     $nodeDATAFRAME$ ←all pairwise interactions where Source node is N
 4:     $MERGED1$ ←Merge(nodeDATAFRAME,allDATAFRAME,targetNode,sourceNode)
 5:     Drop the entries where $N$ = $thirdNode$
 6:     $MERGED2$ ←Merge(MERGED1,allDATAFRAME,(node1,node3),(sourceNode,targetNode))
 7:     $MERGED3$ ←Merge(MERGED2,patternDATAFRAME,(edges between node pairs))
 8:     $MERGED4$ ←Merge(MERGED3,IsomorphicClassDATAFRAME,(pattern id))
 9:     $MERGED4$ entries with a connection between $N$ and $thirdNode$ assigned as triangle
10:     $MERGED4$ entries without a connection between $N$ and $thirdNode$ assigned as wedge
11:     Map the triangle and wedge entries of $MERGED4$ by $N$ seperately
12: end for=0
```

This process is accomplished by successive merge operations applied on dataframes representing node neighborhood, 3-node graphlet set and isomorphic class map respectively. First merge operation shown in Line 4 combines the pairwise interactions starting from a specific node "N" and starting from all network nodes respectively. This is an inner merge operation where only the common values of the dataframes are merged. The merge operation is accomplished over target nodes of the relations starting from "N" and the source nodes of the whole multilabeled interactions involving in a network. That means, for the entries of the dataframes where the target node of a pairwise connection starting from node N is matched with the source node of any pairwise relation, these entries are merged. The result of the operation is also a dataframe including all nodes and connections between nodes. This result is modified by line 6 in order to eliminate the entries with self edges. And also if the relation between second and third nodes is a directed edge, it is reversed (Line 5). The reverting process is necessary for upcoming pattern matching process. Recall that the generic 3-node graphlet set is generated by the assumption that the connection orientation between node pairs are determined by different nodes of the graphlet. While the direction of the edges between starting node and the remaining nodes are determined by the starting node itself, the orientation between second and the third node is determined by the third node. So in order to make a straight comparison, the source of the relation between second and third node should be changed from the second node to the third node.

The aim of the first merge operation is to find a indirect neighbor connected to node "N" which constitutes a 3-node pattern. The result of this step detects 3-node patterns including N as a starting node and the remaining nodes existing in the pattern. In order to determine the type of the pattern, after inverting and dropping operations, another merge operation is applied on the dataframe produced by first merge operation and the dataframe representing the neighborhoods of all network nodes to find a connection between the starting node and third node of the graphlet. The entries with matching (node1,node3) pairs are retrieved and used to extend the result dataframe by inclusion of an additional column of relation information between the starting node and the third node (wedge/triangle determination).

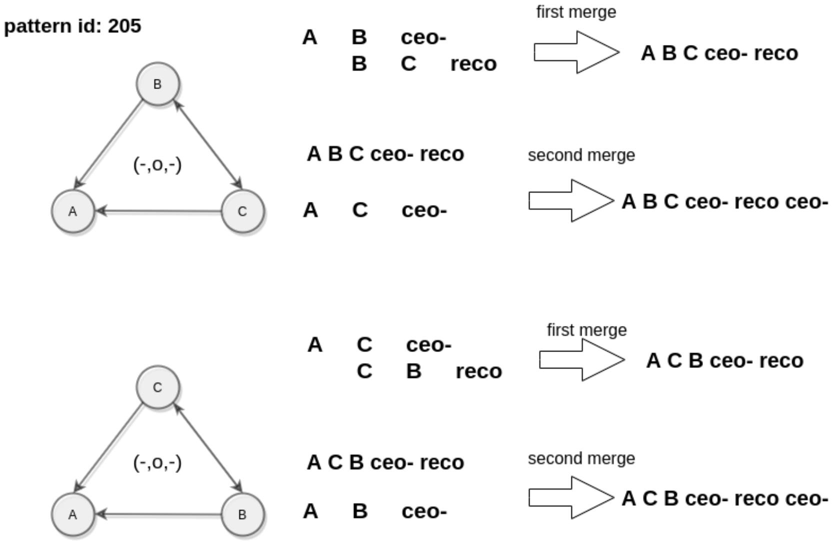


Figure 3.2: The generation of two isomorphic graphlets with pattern id 205

Eventually, in order to specify the connection patterns between the nodes of each 3-node graphlet, two successive merge operations are performed in line 8 and line 9 by pattern id and class id respectively. As a result, the information of the interacting nodes, edges between nodes, id of the 3-node graphlet constituted by the edges between nodes and the id of the class which the 3-node graphlet is the member of are all included in the result dataframe (MERGED4). Line 10 refers to the elimination of isomorphic graphlets with the same interactor ids (starting from a same node) and class membership. Identified patterns are seperately stored in order to be used by 4-node pattern identification processes explained in detail in the next section.

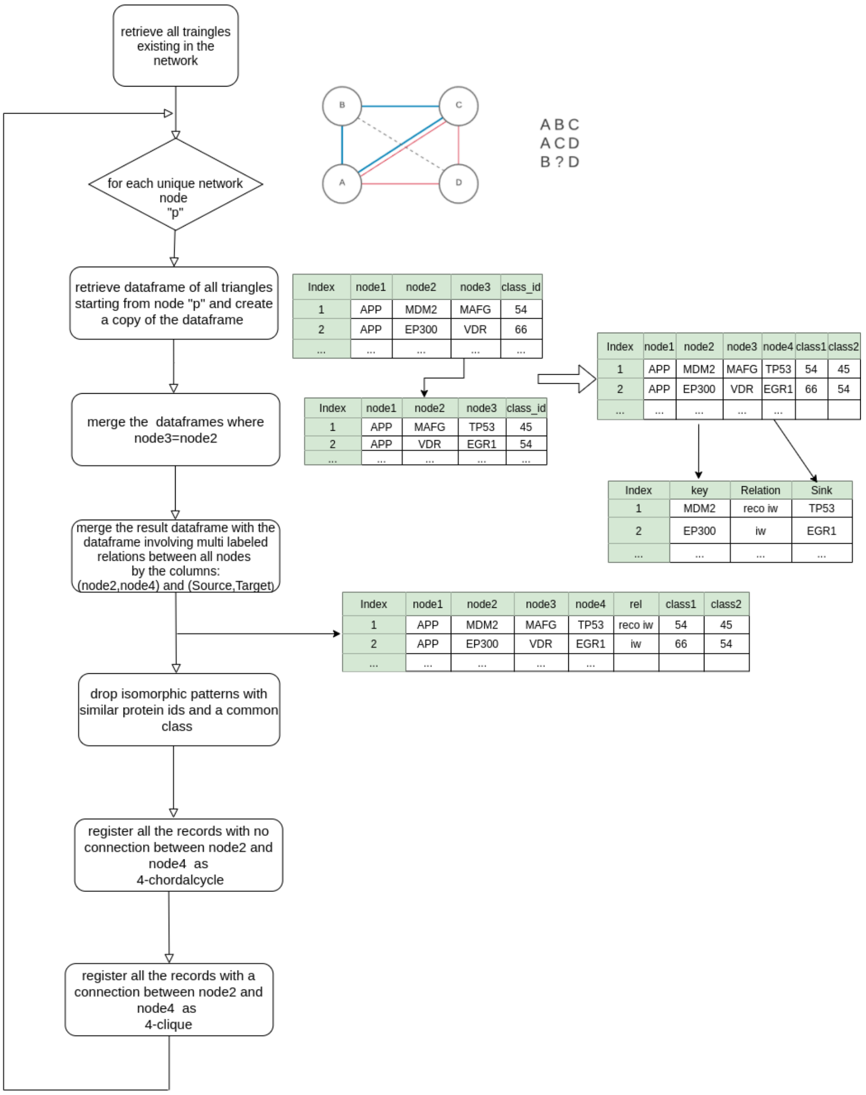
Figure 3.2 shows two isomorphic 3-node graphlets having the same interacting partners and connection patterns. Figure shows how protein A, B and C form a 3-node graphlet. This is the depiction of merge operations successively processed to identify the 3-node graphlets in the integrated interaction dataset. The first merge operation joins the same interacting pairs of each line of interaction (protein B for the first case which is the target node of the ceo- interaction and source node of the reciprocal directed edge between B and C). Then second merge operation is performed to find out whether there is a connection between first and third node that is the determiner of the type of the graphlet. As a result of the merge operations, a triangle pattern ABC with the id of "205" is identified. Since the original dataset is extended to include the interactions starting from each unique node of the network, the merge operation may result in the same patterns starting from the same protein but with different node orderings. Figure 3.2 shows that the same pattern with different node ordering is produced as a result of the merge operations performed on the extended interaction dataset. Our approach eliminates these isomorphic patterns. The isomorphic patterns are appeared as duplicate records in the result dataframe after taking the transpose of the original result and sorting the node ids thereafter.

3.1.5 Identification of 4-node Graphlets in the Integrated Network

Previous sections show how generic set of 3-node graphlets consisting both directed and undirected edges are generated and how the set of network nodes revealing the connection pattern associated by any of the graphlet in this generic set is identified. This part of the study additionally explains the detection of 4-node graphlets occuring in the integrated interaction network. The study (Ahmed et al., 2016) shows that there exists six different types of 4-node undirected graphlets: 4-cliques, 4-chordalCycles, 4-TailedTriangles, 3-Stars, 4-Cycles and 4-Paths. It is claimed in their study that 4-node graphlets can be decomposed into 3-node graphlets. Additionally, the counts of the 4-node patterns can be calculated by using both the count information of 3-node graphlets and the relationships between a 4-node pattern and its decomposed parts. Based on this study, a method to identify 4-node multi labeled graphlets is implemented by considering the combinations of the pre-identified 3-node graphlets.

3.1.5.1 Identification of 4-Cliques and 4-ChordalCycles

4-Clique and 4-ChordalCycle patterns can be imagined as the combination of two triangles. The 4-node pattern at the top of Figure 3.3 shows how combination of red and blue triangles originated from node A forms 4-node graphlet, possibly a 4-Clique or a 4-ChordalCycle depending on the existence of the connection between node B and node D. So, 4-Cliques/4-ChordalCycles detection is mainly a process of consolidating triangles over specific node positions in common and determining the type of the pattern according to the existence of an edge between the second nodes of the combined triangles. As demonstrated in the flowchart in Figure 3.3, multi labeled triangle patterns involving in a network are used as an input of the implemented method. The neighborhood of each unique network node which is an initiator of the triangle patterns is used by the method in order to identify 4-Cliques or 4-ChordalCycles. To accomplish the task, similar with the triangle/wedge detection process, successive merge operations are performed on the dataframes holding the information involved in the identified triangles. For a specific node, all triangles initiated by the node is retrieved from a global dataframe including all triangles occuring between network nodes. Retrieving operation only consider the portion of it associated by the initiator protein "p". Then the dataframe is merged with its exact copy. Actually, the merge operation is the process of combining two triangles initiated by same protein. If the protein id of third node of the first triangle is similar with the protein id of the second node of the second triangle, the triangles are combined to produce a 4-node pattern. As merging process is completed, a set of discovered pattern is emerged with related information involving the protein ids of the 4-node pattern and the class ids of each triangle component. Following the generation of the 4-node pattern set, records representing the triangles with the second (protein B in Figure 3.3) and fourth node (protein D in Figure 3.3) positions both contains the same protein identifier. And also there exist some isomorphic patterns in the discovered pattern set. All these records are eliminated. In order to eliminate the isomorphic patterns, the rows of the dataframe representing the patterns are selected according to the criteria of having at least one common class of triangle. The protein ids of the selected rows are sorted alphanumerically that means the positions of the proteins in the pattern may change. Then alphanumerically sorted rows are used as the input of *duplicated()* method which is a built-in method of Python Pandas module.


retrieve all traingles existing in the network
for each unique network node "p"
retrieve dataframe of all triangles starting from node "p" and create a copy of the dataframe
merge the dataframes where node3=node2
merge the result dataframe with the dataframe involving multi labeled relations between all nodes by the columns: (node2,node4) and (Source,Target)
drop isomorphic patterns with similar protein ids and a common class
register all the records with no connection between node2 and node4 as 4-chordalcycle
register all the records with a connection between node2 and node4 as 4-clique
B
C
A
D
A B C
A C D
B ? D

Index	node1	node2	node3	class_id
1	APP	MDM2	MAFG	54
2	APP	EP300	VDR	66
...	...	...	...	...

Index	node1	node2	node3	class_id
1	APP	MAFG	TP53	45
2	APP	VDR	EGR1	54
...	...	...	...	...

Index	node1	node2	node3	node4	class1	class2
1	APP	MDM2	MAFG	TP53	54	45
2	APP	EP300	VDR	EGR1	66	54
...	...	...	...	...		

Index	key	Relation	Sink
1	MDM2	reco iw	TP53
2	EP300	iw	EGR1
...	...	...	...

Index	node1	node2	node3	node4	rel	class1	class2
1	APP	MDM2	MAFG	TP53	reco iw	54	45
2	APP	EP300	VDR	EGR1	iw	66	54
...	...	...	...	...			

Figure 3.3: The flowchart of the implemented method for identifying multi labeled 4-Cliques and 4-ChordalCycles.

Each duplicated row is annotated by a boolean flag in O(1) time by this built-in method and duplicated rows are eliminated from the result. Eventually, to determine the type of 4-node graphlets as either 4-Clique or 4-ChordalCycle, one last merge operation is performed on the set of discovered 4-node graphlets and all pairwise multi label relations occurring between all nodes of the network. If the interacting pairs in the latter set is similar with the proteins appeared in both node2 and node4 positions of latter set, then the relation type in the pairwise interaction set indicates the edge type between node2 and node4 of the 4-node graphlet. If there is no connection, the pattern is determined to be a 4-ChordalCycle, otherwise it is identified as a 4-Clique.

3.1.5.2 Identification of 4-TailedTriangles and 3-Stars

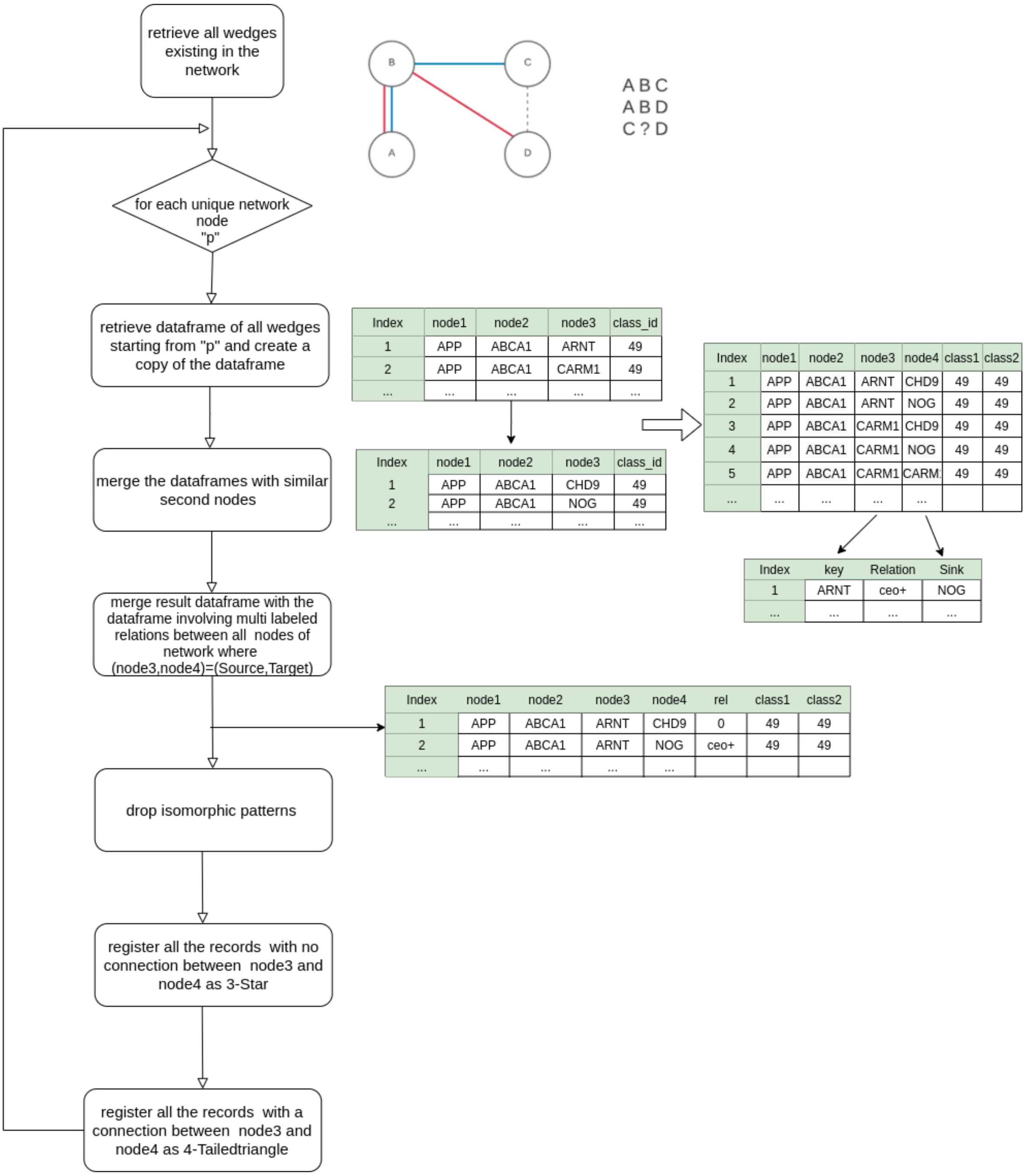
4-TailedTriangles and 3-Stars patterns are composed of two wedges with the originating node and the second neighbour in common. Existence of a connection between distinct nodes of each combined wedge helps to determine the type of the pattern. Generation of the pattern and the details of the process are given by the flowchart in Figure 3.4. The 4-node pattern at the top of Figure 3.4 shows how combination of red and blue wedges originated from node A forms 4-node graphlet, possibly a 4-TailedTriangle or 3-Star depending on the presence of the connection between distinct nodes of the combined wedges.

The implemented method for generating 4-TailedTriangle/3-Star patterns is similar with the one identifying the 4-Clique/4-ChordalCycle patterns. Apart from this approach, 4-TailedTriangle/3-Star identification method combines the wedge patterns when the same protein ids are encountered in "node2" neighbour of both wedges. That mean, the wedges starting from the same protein and having a second neighbour in common are combined to produce a possible 4-TailedTriangle/3-Star pattern. The result set is investigated for the elimination of 4-node patterns composed of identical wedges (the third and fourth nodes are same). The process of both identifying the connection between distinct nodes and eliminating the isomorphic graphlets are similar with the processes performed to accomplish the same tasks in 4-Clique/4-ChordalCycle identification method.

3.1.5.3 Identification of 4-Cycles and 4-Paths

4-Cycles and 4-Paths are also constructed by combining wedge patterns. Apart from the construction of 4-node patterns in previous methods, one of the wedge component is originated from a different starting node. So, the merge operation is performed on two wedges initiated by different nodes. The 4-node pattern at the top of Figure 3.5 shows how combination of red and blue wedges originated from node A and node B respectively forms 4-node graphlet, possibly a 4-Cycles or 4-Paths depending on the presence of the connection between distinct nodes of the combined wedges.

As demonstrated by the flowchart in Figure 3.5, multi labeled wedge patterns involving in a network are used as the input of the implemented method. The merge process combines a wedge pattern which is initiated by a specific network node "p" with another wedge which is initiated by the middle nodes of the wedge patterns initiated by "p". To accomplish this task, all distinct nodes touching a middle node of wedge pattern (wedges initiated by p) are appended to a list. For each element of the list, all wedges initiated by this element are retrieved. The retrieved records are then merged with the wedges initiated by "p" if the third node one of the wedge pattern similar with the second node of the other wedge pattern. Similar with the other 4-node graphlet detection methods previously described, the edge between distinct nodes of the wedge components (first and fourth node of the generated 4-node graphlet)is determined by performing another merge operation. And lastly, the isomorphic patterns are eliminated. The results are registered as different types, either 4-Cycle or 4-Path, according to the existence of the edge between node1 and node4.


retrieve all wedges existing in the network
for each unique network node "p"
retrieve dataframe of all wedges starting from "p" and create a copy of the dataframe
merge the dataframes with similar second nodes
merge result dataframe with the dataframe involving multi labeled relations between all nodes of network where (node3,node4)=(Source,Target)
drop isomorphic patterns
register all the records with no connection between node3 and node4 as 3-Star
register all the records with a connection between node3 and node4 as 4-Tailedtriangle
B
C
A
D
A B C
A B D
C ? D

Index	node1	node2	node3	class_id
1	APP	ABCA1	ARNT	49
2	APP	ABCA1	CARM1	49
...	...	...	...	...

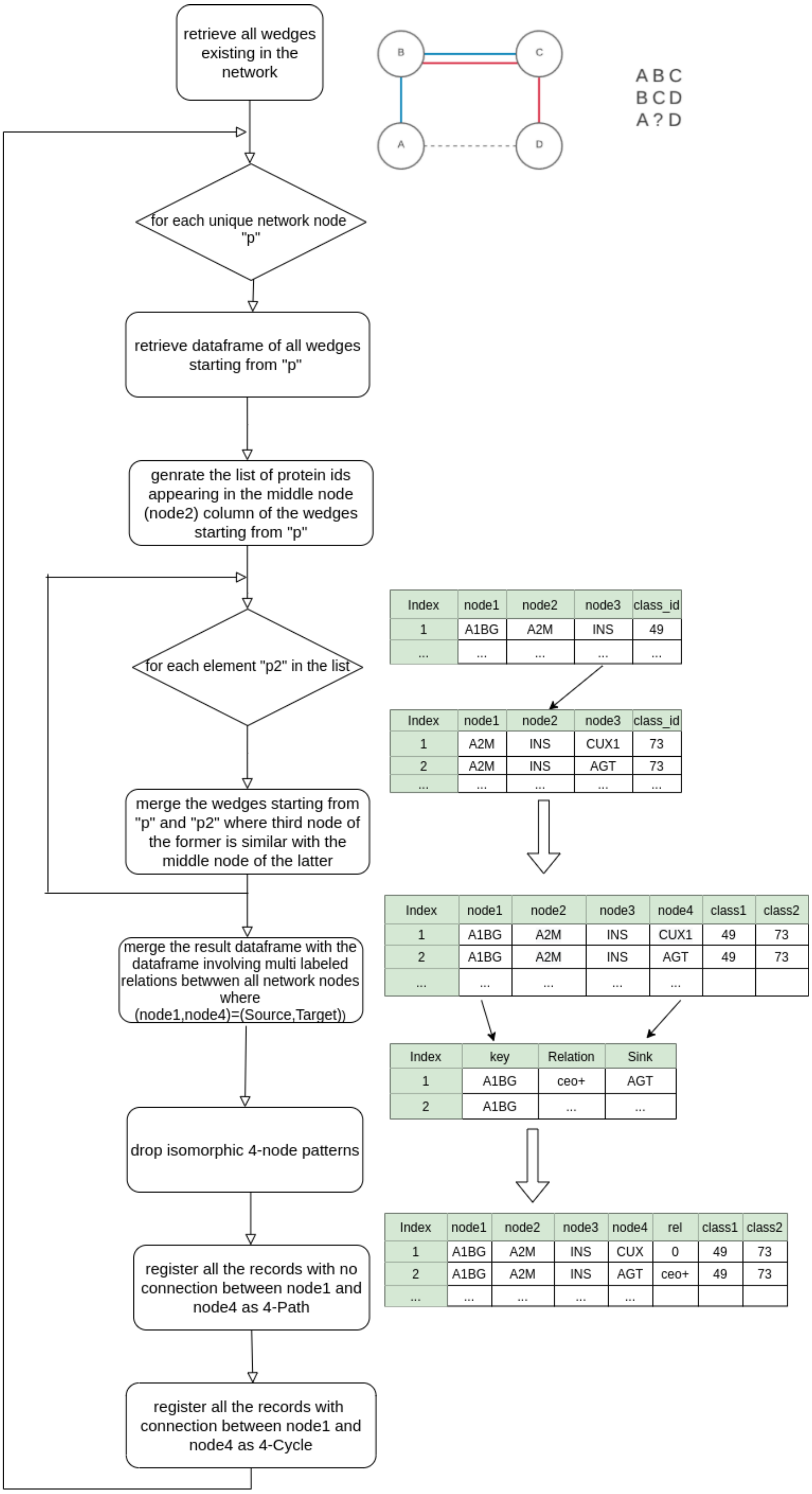
Index	node1	node2	node3	class_id
1	APP	ABCA1	CHD9	49
2	APP	ABCA1	NOG	49
...	...	...	...	...

Index	node1	node2	node3	node4	class1	class2
1	APP	ABCA1	ARNT	CHD9	49	49
2	APP	ABCA1	ARNT	NOG	49	49
3	APP	ABCA1	CARM1	CHD9	49	49
4	APP	ABCA1	CARM1	NOG	49	49
5	APP	ABCA1	CARM1	CARM:	49	49
...	...	...	...	...		

Index	key	Relation	Sink
1	ARNT	ceo+	NOG
...	...	...	...

Index	node1	node2	node3	node4	rel	class1	class2
1	APP	ABCA1	ARNT	CHD9	0	49	49
2	APP	ABCA1	ARNT	NOG	ceo+	49	49
...	...	...	...	...			

Figure 3.4: The flowchart of implemented method for identifying multi labeled 4-TailedTriangles and 3-Stars.


retrieve all wedges existing in the network
B
C
A
D
A B C
B C D
A ? D
for each unique network node "p"
retrieve dataframe of all wedges starting from "p"
genrate the list of protein ids appearing in the middle node (node2) column of the wedges starting from "p"
for each element "p2" in the list
merge the wedges starting from "p" and "p2" where third node of the former is similar with the middle node of the latter
merge the result dataframe with the dataframe involving multi labeled relations betwwen all network nodes where (node1,node4)=(Source,Target))
drop isomorphic 4-node patterns
register all the records with no connection between node1 and node4 as 4-Path
register all the records with connection between node1 and node4 as 4-Cycle

Index	node1	node2	node3	class_id
1	A1BG	A2M	INS	49
...	...	...	...	...

Index	node1	node2	node3	class_id
1	A2M	INS	CUX1	73
2	A2M	INS	AGT	73
...	...	...	...	...

Index	node1	node2	node3	node4	class1	class2
1	A1BG	A2M	INS	CUX1	49	73
2	A1BG	A2M	INS	AGT	49	73
...	...	...	...	...		

Index	key	Relation	Sink
1	A1BG	ceo+	AGT
2	A1BG	...	...

Index	node1	node2	node3	node4	rel	class1	class2
1	A1BG	A2M	INS	CUX	0	49	73
2	A1BG	A2M	INS	AGT	ceo+	49	73
...	...	...	...	...			

Figure 3.5: The flowchart of implemented method for identifying multi labeled 4-Cycles and 4-Paths.

3.1.6 Results

This part of the study shows how to identify 3-node graphlets concerning both directed and undirected edge types. And it also shows how to identify the 4-node patterns derived from combinations of the predetected 3-node graphlets. The identification processes are applied on an integrated network provided by Pathway Commons dataset (*The Pathway Commons Database Version 11; [cited January 28, 2019].*, 2019). The original dataset is filtered to exclude all relations other than “controls expression of” and “interacts with”. Table 6 shows the amount of records concerning these types in Pathway Commons dataset version 11.

Table 6: Amounts of pairwise interactions including interacts-with and controls- expression-of relations in Pathway Commons Version 11

Interaction type	# of interactions
interacts-with	485,608
controls-expression-of	126,396

The filtered dataset is duplicated in order to include both incoming and outgoing edges. Resulting dataset consists of 1.224.008 interactions. After combination of multiple edges to form multi labelled edge types, the size of the dataset decreases to 1.215.882. Table 7 shows the amount of interactions after combining the directed and undirected edges occuring between same pair of nodes in the dataset. For the types of the edges in table, "ceo+" and "ceo-" stands for outgoing and incoming "control expression of" interaction respectively, "reco" stands for "reciprocal" edge type, and "iw" stands for "interacts-with" relation.

The 3-node graphlet analysis of the Pathway Commons dataset shows that from the 18648 unique proteins occuring in the dataset, 14162 of them induce at least a triangle pattern and 15407 of them induce at least a wedge pattern. Table 8 and 9 show first five proteins initiating the highest amount of wedges and triangles respectively.

Table 7: Amount of multi labeled edges after combination process

Interaction type	# of interactions
ceo+	121,972
ceo+iw	3,430
ceo-	121,972
ceo-iw	3,430
iw	964,084
reco	722
recoiw	272

Additionally, when the node degrees and the number of triangles initiated by the nodes existing in the dataset is compared, it is observed that there is a relationship between the nodes that are initiating the highest number of triangles and having the highest node degrees. Table 10 shows the first five proteins of the dataset having the highest node degrees. %60 of the first five highest degree proteins are also appeared as an initiator of the triangles observed with highest amounts.

Table 8: First five proteins associated with the highest wedge amounts

Protein id	# of wedges
KTN1	95,798
LMO3	63,038
HOXC4	59,676
PITX2	58,373
DMD	57,100

Table 9: First five proteins associated with the highest triangle amounts

Protein id	# of triangles
NOG	79,176
SP1	42,524
INS	41,738
LEF1	37,261
MAZ	33,687

Table 10: First five proteins having the highest node degrees

Protein id	of node
NOG	5740
APP	3759
INS	3510
SP1	3341
HNF4A	3113

Besides the amount of triangle and wedge patterns initiated by a specific node, the class of the initiated pattern is also important. For this part of the study, the generic set of 360 graphlet patterns is produced and further reduced to 86 by grouping the isomorphic patterns into different classes. The node based graphlet counts of "NOG" protein which initiates the highest amount of triangles reveals that the most frequent triangle pattern starting from that protein is the one with the class id "66" with the frequency of 53004. The frequency of this pattern is three times more than the second highest frequent one starting from the same protein.

Additionally the graphlet counts of KTN1 protein which is initiating the highest amount of wedge patterns is investigated. There exists three different class of wedge patterns. The most frequent class of wedges is "73" with the frequency of 94972. The most frequent wedge pattern occurs nearly 1000 times more than the least frequent one. The Figure 3.6depicts the most frequent patterns starting from KTN1 and NOG proteins respectively.

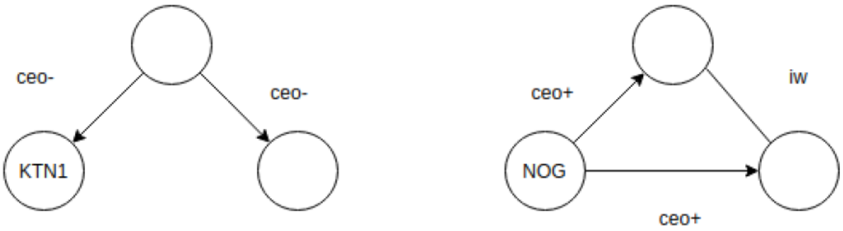


Figure 3.6: The most frequent wedge and triangle patterns starting from KTN1 and NOG proteins respectively.

Eventually, some cancer types are investigated in detail by using the node based 2- to 4-node graphlet counts. The pathway maps are taken from KEGG database. The pathway specific protein set in the cancer pathways are extracted first. Then for each investigated cancer type, the set of pathway specific proteins are used to retrieve the integrated interactions occurring between the proteins in the set. The 3- and 4-node pattern identification processes are accomplished by concerning the pathway specific protein sets pertaining to a specific cancer type. Out of 86 isomorphic pattern classes, 21 of them represent the distinct wedge patterns while the 65 of them represent the triangle patterns. And there are also 6 different types of 4-node graphlets produced by combining all possible 3-node patterns existing between considered nodes. For this part of the study, the analysis of the concerned cancer pathways is accomplished by considering the counts of the 86 distinct classes of graphlets starting from the proteins existing in the pathways.

Breast cancer pathway is the one of the cancer type investigated by our approach. There exists 147 different proteins appeared in the Breast Cancer pathway obtained from KEGG database. The most frequent 3-node pattern is the one starting from BRAF protein with the class id of "73". Its pattern id is "193" and it recurs 138 times between BRAF protein and its neighbours. BRAF protein initiates 6 different 3-node patterns. The half of this amount of patterns are triangles and the other half are the wedges. However, the most frequent pattern is a wedge pattern. The recurrence of the other patterns starting from BRAF are less than 10. On the other hand, the most frequent 4-node pattern starting from BRAF is a 3-Star pattern with the frequency of 2332. 2149 out of this amount is the combination of the wedges both with the class id of 73 which is not an unexpected observation since it is the most frequent wedge pattern starting from BRAF protein. Figure 3.7 shows the samples of most frequent 3- and 4-node patterns occurring between BRAF protein and its neighbors.

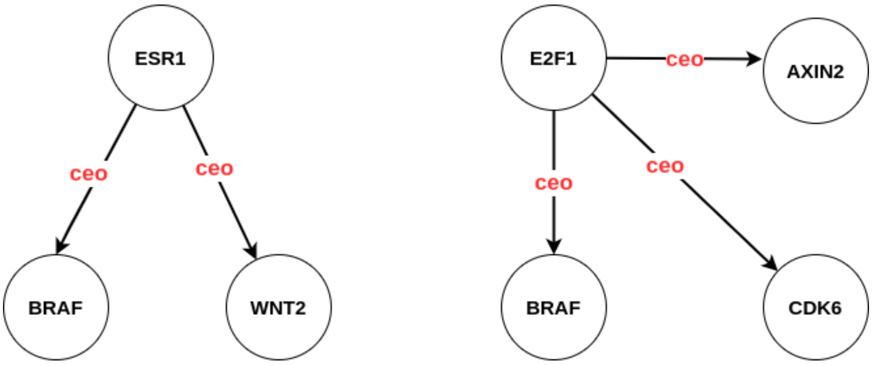


Figure 3.7: The most frequent wedge and 3-Star patterns starting from BRAF protein in Breast Cancer Pathway.

As 4-node patterns are inspected in detail among Breast Cancer specific proteins, it is observed that the 3-Star pattern recurs more than the other 4-node patterns. The maximum frequency of this pattern is the pattern starting from WNT5B. The count of the pattern is 4367. This protein initiates 3 classes of patterns which are all wedges. The other two classes recurs less than 3 times while the pattern with class id "73" recurs 95 times. The high frequency of this wedge patterns also explains the high amount of 3-Star pattern appeared between WNT5B protein and its neighbours since our approach identifies the 3-Star patterns by combining the wedges. The Figure 3.8 shows one of the most frequent 3-Star pattern starting from WNT5B protein.

The second cancer pathway which is analyzed is Gastric Cancer Pathway. There exists 149 pathway specific proteins in this pathway. The most significant difference between the highest and lowest count values of a 3-node pattern is observed for the class id "75" with the 46 fold difference. The pattern is a wedge and most frequently initiated by the TCF7 protein. The Figure 3.9 shows one of the most frequent pattern with class id "75" starting from TCF7.

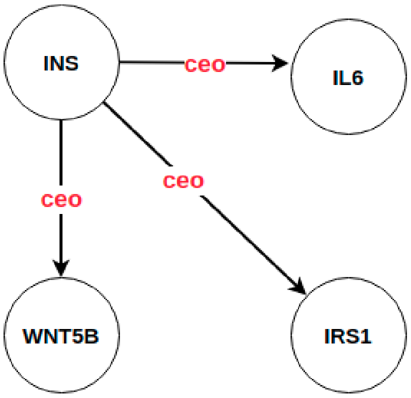


Figure 3.8: One of the most frequent 3-Star pattern starting from WNT5B protein with the highest count value among other 4-node graphlets in Breast Cancer pathway.

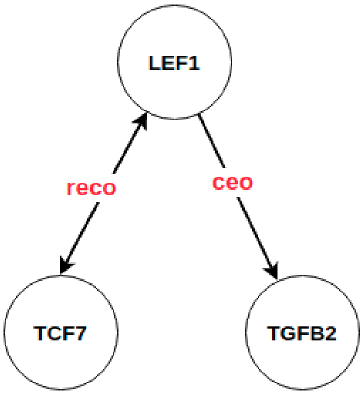


Figure 3.9: One of the most frequent pattern starting from TCF7 protein with class id "75" in Gastric Cancer Pathway.

As the interaction patterns between Gastric cancer specific proteins are investigated, it is revealed that the triangle pattern with class id "66" is the one initiated by the highest amount of proteins. That means, out of 149 pathway specific proteins, 61 distinct proteins initiates this type of triangle. This pattern is most frequently initiated by LEF1 protein with the count of 35.

As the protein set is examined to figure out the proteins initiating the highest number of 3-node graphlets, it is found that both LEF1 and MYC proteins initiates 13 distinct types of 3-node graphlets. Out of 13 different types of 3-node graphlets, LEF1 protein initiates 5 different wedge patterns while MYC protein initiates 4 different wedge patterns. The remaining patterns are triangles. The most significant difference between the counts of the triangle patterns is appeared for the pattern which is the member of class "48". While the count of this pattern starting from MYC protein is 2, the frequency of the pattern starting from LEF1 protein is 17. This is the highest amount of difference between the same class of patterns starting from LEF1 and MYC proteins. Figure 3.10 shows the sample triangles starting from MYC and LEF1 protein showing the highest difference among the counts of this common pattern.

Last analysis is conducted to compare the two types of lung cancer pathways, small cell lung cancer pathway and non-small cell lung cancer pathway. There exists 72 pathway specific proteins for non-small cell lung cancer pathway and 92 pathway specific proteins for small cell lung cancer pathway. The amount of common proteins for these two types of proteins is 31. The 3- and 4-node graphlet counts are analyzed for each cancer type by concerning common proteins existing in both types of lung cancers. Table 11 shows three common proteins for both types of the lung cancer (small cell vs. non-small cell). There is a significant difference between the count values of both 3-Star and 4-TailedTriangle patterns starting from TP53 protein between small and non-small cell lung cancer pathways. Another significant difference occurs between 4-Clique and 4-ChordalCycle patterns initiated by CDKN1A protein. BAX is another protein initiating the wedge pattern with the class id of 73 with high amount of difference among the considered pathways. BAX initiates the patterns in small cell lung cancer pathway 24 times more than in non-small cell lung cancer pathway.

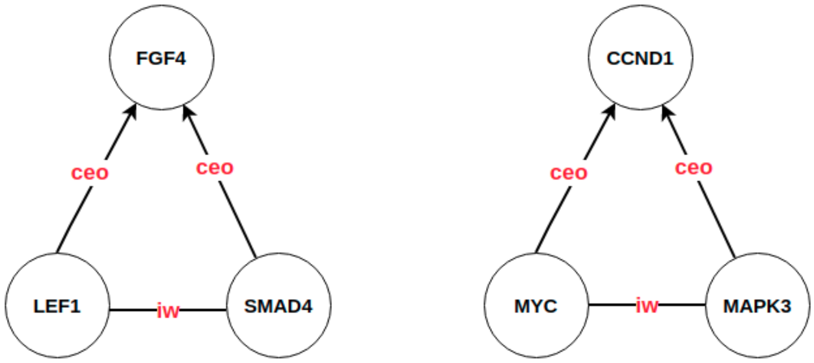


Figure 3.10: The triangle pattern starting from LEF1 and MYC proteins revealing the highest amount of difference between the count values.

Table 11: The differences between some patterns starting from common proteins specific to both types of lung cancers.

TP53		
Cancer Type	Graphlet Type	Count
small cell lung cancer	3-star	178
small cell lung cancer	4-TailedTriangle	12
non-small cell lung cancer	3-Star	1
non-small cell lung cancer	4-Tailed Triangle	0
CDKN1A		
Cancer Type	Graphlet Type	Count
small cell lung cancer	4-Clique	2
small cell lung cancer	4-ChordalCycle	5
non-small cell lung cancer	4-Clique	20
non-small cell lung cancer	4-ChordalCycle	19
BAX		
Cancer Type	Class id	Count
small cell lung cancer	73	24
non-small cell lung cancer	73	1

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

Network-based computational methods had gained so much attention for many years. The methods especially focusing on interaction data have been widely used for network analysis for detecting topological similarities of different networks. One of the most widely used heuristics revealing structural similarity is graphlets. A motif is another type of heuristic used for network analysis. However, apart from the graphlets, motifs are partial sub-graphs involving some of the edges between the nodes. However, graphlets are induced subgraphs so all the edges between a set of nodes are concerned for the network analysis. The counts of the graphlets are used to quantify the similarity of the networks. There have been several studies implementing graphlet-based methods counting the graphlets efficiently. But most of the methods mainly focus on the undirected networks while implementing the graphlet-based properties. However, there are different types of interaction networks such as metabolic, regulatory, etc. in which the direction of the edges between entities is also important. There also exist some methods considering the edge directions, but they have some limitations such as restricted anti-parallel edges, consideration of single label edges, etc. And eventually, there is no study focusing on the analysis of networks including both undirected and directed edge types.

Thus, in Chapter 2, we offer a method of counting the graphlets in condition specific integrated networks. To achieve the integrated network analysis, our method regards the edges with 5 different labels referring to both directed and undirected edge types. Each encountered edge is encoded in bitwise edge vectors by setting the corresponding bit. Each edge vector is a combined version of all edges occurring between node pairs. The method uses an adjacency list based method for representing the network and a hash-based strategy for counting the graphlets. As a set of graphlet detected among 141 different condition specific networks, the z-score of each graphlet is computed by comparing it with the collection of random networks constructed with the same degree distribution of each real condition specific network. The signature vector of each condition specific network is the collection of these z-scores. The method finds 7346 different types of 3-node graphlets and 56 different types of 2-node graphlets. Then the statistically significant graphlets are determined by inspecting the z-score of each. The graphlets with max z-score and max z-score variation are reported respectively. Another interesting result of the study is the performance of graphlet signatures used while performing clustering of the tissue specific networks. 4 out of 5 similar tissues (thalamus tissues) are fallen in the same cluster. The result shows that the graphlet signature is superior to gene expression data that cause the same tissue specific networks to fall in different clusters. Ultimately, pairs of different states of the same tissue network are compared to reveal the potential reasons causing the state changes. The statistically significant graphlets overrepresented in one of the states with respect to the other state are detected. The proteins distinctly inducing the differentially recurring graphlets among the states are also analyzed by using a functional enrichment tool to find the biological processes associated with the protein set. The proposed method is the first attempt to analyze the condition specific networks including different types of interactions, both directed and undirected, between proteins. Since the method is not a computationally most efficient method, some effort is needed to decrease the time of computation. And we can seek some solutions to increase the number of edge labels to be considered in order to add new levels of information by integrating more edges.

Despite the promising results, the discovered graphlets and the revealed functional relationships should be investigated in a biological context for a deeper understanding of using these graphlets as biological indicators at the network level.

The graphlet based strategy implemented in Chapter 2 is reflecting the network based structural properties. However, the graphlet counts can be used to represent the topologically similar nodes of distinct networks. Hence, the method in Chapter 3 offers a solution for comparing the network nodes based on the counts of the patterns starting from that nodes. Since the signature vector is representing the structural information around the nodes, for a specific k-node graphlet, it is important to indicate the topologically similar and different points of graphlets touched by a specific node. Therefore, apart from the method in Chapter 2 generating graphlets reflecting the topological properties of a network, a generic set of 3-node graphlets is produced first for the method provided in Chapter 3. Each provided graphlet is assumed to be initiated by a different node of that graphlet and each starting point corresponds to an orbit. As a result, a total of 360 3-node graphlets is generated. Each graphlet is composed of both directed and undirected types of edges. However, this part of the study is restricted to considering two edge labels (interacts-with, controls expression of) because of the increasing amount of the graphlets produced as a result of considering multi labeled edges. The generic set of 3-node graphlets includes isomorphic patterns since there exist the same patterns starting from the different nodes of the same graphlet. So, the isomorphic patterns are grouped and retained in the corresponding classes. This process reduces the size of the feature vector. Lastly, the identified 3-node graphlets in the integrated network are also used to detect 4-node graphlets which are composed of the identified 3-node graphlets. The method provided here is different from the one given by Chapter 2. The current method uses the modification methods offered by the Python pandas module. As assigning the detected patterns to a Python dataframe, merging operations are applied to the dataframe variables to identify both 3-node graphlets from occurring pairwise interactions and 4-node graphlets from previously identified 3-node graphlets. Despite some memory issues, this approach reduces the computational time. While identifying 3-node multi label directed graphlets in Pathway Commons dataset including 16646 unique proteins and 476051 binary relations by using the iterative approach takes nearly 21 hours, it takes nearly 2 hours by using the merge operations over the Pandas dataframe structures. The reported results of the process show that as the degree of each node and the number of triangles are compared, it is concluded that most of the proteins with the highest degree initiate the highest number of triangles. This is not the case for wedges, which means the proteins with the highest degree are not likely to initiate a wedge pattern. Apart from the analysis conducted on the integrated Pathway Commons dataset, the pathway data of some cancer types taken from KEGG database are also analyzed. Breast Cancer and Gastric Cancer specific nodes are used to identify the 3- and 4- node graphlet patterns starting from each pathway specific node and occurring between that nodes and their neighbors. Our approach accomplishes some node based and pattern based evaluations and detect some proteins revealing significant differences between different types patterns. Additionally different types of same cancer pathway (small vs. non-small cell lung cancer) are used to detect the common proteins revealing different profiles across these two types. The most important improvement achieved by the method in Chapter 3 is the decreased time of computation which facilitates large-scale network analysis. Although there are some memory issues caused by some redundant information involved in the result dataframe, both using filtering processes and the less memory demanding type of data could solve the problem. Additionally Python allows us to use some objects such as “shelf” which is a dictionary-like object but stored in a disk file and lets us use the memory efficiently.

In this dissertation, we have represented a graphlet counting method for the analysis of condition specific integrated networks. As a further study, the graphlet counts can be used to construct a scoring schema for cancer pathway specific genes. To be more specific, the graphlet counts can be used to define a measure to be used as a score of each cancer specific gene. Then these scores may be used to differentiate the up-regulated and down-regulated genes. Moreover, after detecting the differentially recurring 3- and 4-node graphlets between conditions, for instance the wedge pattern with class id 73 originating from BAX protein in small cell lung cancer occurs 24 times more than the same wedge pattern starting from the same protein in non-small cell lung cancer, the originating proteins causing this significant difference can be investigated by scanning the studies conducted on this specific condition in order to find a relevance with our results and extract the biological meanings of our findings. Another process leaved as a future work is to compare the results of the study provided in Chapter 2 with the results of the studies citing our work. Our study is published in 2017 and our aim is to find latter studies that validates the results produced by our study. Finally, as a further study, a research tool is going to be implemented by using the graphlet counting methods proposed in this dissertation. This tool is going to be accepted the differentially expressed genes pertaining to a specific condition. For instance, two separate microarray experiment data among different states of the same condition (healthy and disease state of the same tissue) can be used as an input by the developed tool and the node based or network based differential graphlet counts are reported as a vector of quantities to be used as a feature map of each state of the same condition for further analysis.

Bibliography

Ahmed, N. K., Neville, J., Rossi, R. A., & Duffield, N. (2016, 1). Efficient graphlet counting for large networks. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2016-January*, 1-10. doi: 10.1109/ICDM.2015.141

Artzy-Randrup, Y., Fleishman, S. J., Ben-Tal, N., & Stone, L. (2004). Comment on "network motifs: simple building blocks of complex networks" and "superfamilies of evolved and designed networks". *Science (New York, N.Y.)*, *305*. doi: 10.1126/SCIENCE.1099334

Cabusora, L., Sutton, E., Fulmer, A., & Forst, C. V. (2005). Differential network expression during drug and stress response. *Bioinformatics Original Paper*, *21*, 2898-2905. doi: 10.1093/bioinformatics/bti440

Cerami, E. G., Gross, B. E., Demir, E., Rodchenkov, I., Babur, O. Z. N., Anwar, N., ... Sander, C. (2011). Pathway commons, a web resource for biological pathway data. Retrieved from `http://www.sbcny.org/` doi: 10.1093/nar/gkq1039

Cook, S. A. (1971). The complexity of theorem-proving procedures. In (p. 151-158).

Demir, E., Cary, M. P., Paley, S., Fukuda, K., Lemer, C., Vastrik, I., ... Bader, G. D. (2010, 9). Biopax – a community standard for pathway data sharing. *Nature biotechnology*, *28*, 935. doi: 10.1038/NBT.1666

Demir, E., Özgün Babur, Rodchenkov, I., Aksoy, B. A., Fukuda, K. I., Gross, B., ... Sander, C. (2013, 9). Using biological pathway data with paxtools. *PLoS Computational Biology*, *9*. doi: 10.1371/JOURNAL.PCBI.1003194

Fabregat, A., Jupe, S., Matthews, L., Sidiropoulos, K., Gillespie, M., Garapati, P., ... D'eustachio, P. (2017). The reactome pathway knowledgebase. *Nucleic Acids Research*, *46*, 649-655. Retrieved from `https://reactome.` doi: 10.1093/nar/gkx1132

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of np-completeness (series of books in the mathematical sciences) 1st edition.*

Gray, K. A., Yates, B., Seal, R. L., Wright, M. W., & Bruford, E. A. (2014). Genenames.org: the hgnc resources in 2015. *Nucleic Acids Research*, *43*, 1079-1085. Retrieved from `http://www.genenames.org/` doi: 10.1093/nar/gku1071

Greene, C. S., Krishnan, A., Wong, A. K., Ricciotti, E., Zelaya, R. A., Himmelstein, D. S., ... Troyanskaya, O. G. (2015, 5). Understanding multicellular function and disease with human tissue-specific networks. *Nature genetics*, *47*, 569-576. Retrieved from `https://pubmed.ncbi.nlm.nih.gov/25915600/` doi: 10.1038/NG.3259

Hayes, W., Sun, K., ulj, N. P., Editor, A., & Valencia, A. (2013). Graphlet-based measures are suitable for biological network comparison. , *29*, 483-491. Retrieved from https://academic.oup.com/bioinformatics/article/29/4/483/199955 doi: 10.1093/bioinformatics/bts729

Hočevar, T., & Demšar, J. (2014). Data and text mining a combinatorial approach to graphlet counting. , *30*, 559-565. Retrieved from http://www.biolab.si/supp/orca/orca.html. doi: 10.1093/bioinformatics/btt717

Hočevar, T., & Demšar, J. (2016). Computation of graphlet orbits for nodes and edges in sparse graphs. *Journal of Statistical Software*, *71*. doi: 10.18637/jss.v071.i10

Huang, D. W., Sherman, B. T., & Lempicki, R. A. (2009). Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nature protocols*, *4*, 44-57. Retrieved from https://pubmed.ncbi.nlm.nih.gov/19131956/ doi: 10.1038/NPROT.2008.211

Ideker, T., & Krogan, N. J. (2012). Differential network biology. *Molecular Systems Biology*, *8*. doi: 10.1038/MSB.2011.99

Kotlyar, M., Pastrello, C., Sheahan, N., & Jurisica, I. (2016). Integrated interactions database: tissue-specific view of the human and model organism interactomes. *Nucleic Acids Research*, *44*. Retrieved from https://academic.oup.com/nar/article/44/D1/D536/2502619 doi: 10.1093/nar/gkv1115

Li, Y., Liu, L., Bai, X., Cai, H., Ji, W., Guo, D., & Zhu, Y. (2010, 10). Comparative study of discretization methods of microarray data for inferring transcriptional regulatory networks. *BMC Bioinformatics*, *11*, 1-6. Retrieved from https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-11-520 doi: 10.1186/1471-2105-11-520/FIGURES/3

Marcus, D., & Shavitt, Y. (2012, 2). Rage – a rapid graphlet enumerator for large networks. *Computer Networks*, *56*, 810-819. doi: 10.1016/J.COMNET.2011.08.019

Meira, L. A., Máximo, V. R., Álvaro L. Fazenda, & Conceião, A. F. D. (2014, 9). acc-motif: Accelerated network motif detection. *IEEE/ACM transactions on computational biology and bioinformatics*, *11*, 853-862. Retrieved from https://pubmed.ncbi.nlm.nih.gov/26356858/ doi: 10.1109/TCBB.2014.2321150

Mitra, K., Carvunis, A. R., Ramesh, S. K., & Ideker, T. (2013, 10). Integrative approaches for finding modular structure in biological networks. *Nature Reviews Genetics*, *14*, 719-732. doi: 10.1038/nrg3552

Newman, M. (2009). *Networks: An introduction.*

Orchard, S., Ammari, M., Aranda, B., Breuza, L., Briganti, L., Broackes-Carter, F., ... Hermjakob, H. (2013). The mintact project-intact as a common curation platform for 11 molecular interaction databases. Retrieved from http://www.imexconsortium.org doi: 10.1093/nar/gkt1115

Park, C. Y., Krishnan, A., Zhu, Q., Wong, A. K., Lee, Y. S., & Troyanskaya, O. G. (2015, 4). Tissue-aware data integration approach for the inference of pathway interactions in metazoan organisms. *Bioinformatics*, *31*, 1093-1101. doi: 10.1093/BIOINFORMATICS/BTU786

The pathway commons database version 11; [cited january 28, 2019]. (2019). Retrieved from http://www.pathwaycommons.org/archives/PC2/v11/

The pathway commons database version 8; [cited april 21, 2016]. (2016). Retrieved from http://www.pathwaycommons.org/archives/PC2/v8/

Prasad, T. S. K., Goel, R., Kandasamy, K., Keerthikumar, S., Kumar, S., Mathivanan, S., ... Pandey, A. (2009). Human protein reference database-2009 update. *Nucleic Acids Research*, *37*, 767-772. Retrieved from https://academic.oup.com/nar/article/37/suppl_1/D767/1019294 doi: 10.1093/nar/gkn892

Pržulj, N. (2007). Biological network comparison using graphlet degree distribution. In (Vol. 23). Oxford University Press. doi: 10.1093/bioinformatics/btl301

Rodchenkov, I., Babur, O., Luna, A., Aksoy, B. A., Wong, J. V., Fong, D., ... Sander, C. (2019). Pathway commons 2019 update: integration, analysis and exploration of pathway data. *Nucleic Acids Research*, *48*, 489-497. Retrieved from https://www.pathwaycommons. doi: 10.1093/nar/gkz946

Roth, R. (2007). *Gse7307: Geo accession viewer.* Retrieved from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7307

Saeed, A. I., Sharov, V., White, J., Li, J., Liang, W., Bhagabati, N., ... Quackenbush, J. (2003, 2). Tm4: A free, open-source system for microarray data management and analysis. *BioTechniques*, *34*, 374-378. doi: 10.2144/03342MT01

Sarajlić, A. (2015). *Analysing directed network data .*

Sarajlić, A., Malod-Dognin, N., Ömer Nebil Yaveroĝlu, & Pržulj, N. (2016, 10). Graphlet-based characterization of directed networks. *Scientific Reports*, *6*. doi: 10.1038/srep35098

Schaefer, C. F., Anthony, K., Krupa, S., Buchoff, J., Day, M., Hannay, T., & Buetow, K. H. (2009). Pid: the pathway interaction database. *Nucleic Acids Research*, *37*. Retrieved from https://academic.oup.com/nar/article/37/suppl_1/D674/1002223 doi: 10.1093/nar/gkn653

Stark, C., Breitkreutz, B.-J., Reguly, T., Boucher, L., Breitkreutz, A., & Tyers, M. (2006). Biogrid: a general repository for interaction datasets. Retrieved from http://biodata.mshri.on.ca/osprey doi: 10.1093/nar/gkj109

Trpevski, I., Dimitrova, T., Boshkovski, T., Stikov, N., & Kocarev, L. (2016, 11). Graphlet characteristics in directed networks. *Scientific Reports*, *6*. doi: 10.1038/srep37057

Will, T., & Helms, V. (2016, 2). Ppixpress: construction of condition-specific protein interaction networks based on transcript expression. *Bioinformatics*, *32*, 571-578. Retrieved from https://academic.oup.com/bioinformatics/article/32/4/571/1744136 doi: 10.1093/BIOINFORMATICS/BTV620

Wishart, D. S., Feunang, Y. D., Guo, A. C., Lo, E. J., Marcu, A., Grant, J. R., ... Wilson, M. (2018). Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleic Acids Research*, *46*. Retrieved from www.drugbank.ca doi: 10.1093/nar/gkx1037

Yamamoto, S., Sakai, N., Nakamura, H., Fukagawa, H., Fukuda, K., & Takagi, T. (2011). Inoh: ontology-based highly structured database of signal transduction pathways. Retrieved from https://academic.oup.com/database/article/doi/10.1093/database/bar052/469742 doi: 10.1093/database/bar052

Yaveroglu, O. N. (2013). *Graphlet correlations for network comparison and modelling: World trade network example* .

Zhang, S., Tian, D., Tran, N. H., Choi, K. P., & Zhang, L. (2014). Profiling the transcription factor regulatory networks of human cell types. *Nucleic Acids Research*, *42*, 12380-12387. doi: 10.1093/nar/gku923

CURRICULUM VITAE

ARZU BURÇAK SÖNMEZ

Education

July 2022 Doctor of Philosophy in Health Informatics, Middle East Technical University, Ankara, Turkey.

July 2008 Master of Science in Computer Engineering, Çankaya University, Ankara, Turkey.

July 2005 Bachelor of Science in Computer Engineering, Çankaya University, Ankara, Turkey.

July 2005 Bachelor of Science in Electronic and Communication Engineering, Çankaya University, Ankara, Turkey (Double Major).

Field of Study

Bioinformatics, Computational Biology, Biological Networks, Graph Theory, Network Analysis, Machine Learning Techniques, Data Mining.

Professional Experience

Research Assistant, Department of Computer Engineering, Çankaya University, Ankara, Turkey, 2005-2013

Publications

A. B. Sonmez and T. Can
Comparison of tissue/disease specific integrated networks using directed graphlet signatures.
BMC Bioinformatics, 18(4):135, Mar 2017.

H.Ogul, C.Beyan, O.Eren, K.Yildiz, T.Ercelebi, B.Sonmez
MicroRNA target recognition from compositional features of aligned microRNA-mRNA duplexes.
Proceedings of International Symposium on Innovations in Intelligent Systems and Applications, 2010, Kayseri, Turkey.