Ranklust

An extension of the Cytoscape clusterMaker2 plugin and its application to find network biomarkers in prostate cancer

Henning Lund-Hanssen

Thesis submitted for the degree of
Master in Programming and Networks
60 credits

Department of Informatics

Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Autumn 2016

Ranklust

An extension of the Cytoscape clusterMaker2 plugin and its application to find network biomarkers in prostate cancer

Henning Lund-Hanssen

Abstract

Next-generation technologies provide bioinformaticians with data on a scale worthy of adding a new label to the biomedicine industry as a whole, namely Big Data. With enormous amounts of data comes a great challenge as how to work with it. Algorithmic approaches benefit greatly from a bigger size in the amount of data to work with and has already shown to take great advantage of the data that comes from high-throuput sequencing technologies. One of the advantages is the potential to identify genes that promote diseases by analyzing networks biologically related networks. Another advantage is the newly gained understanding of the topological organization of large-scale molecular networks. Clustering groups of these networks into subnetworks depending on their function in the network is known as module detection in networks. The goal of this assignment was to combine these two newly adapted methods in bioinformatics to develop a tool in Cytoscape. This tool will rank clusters based on the score of the nodes (genes) they contain. To exercise an exact experiment with this tool, I performed an attempt at identifying prostate candidate cancer cluster biomarkers through the use of this tool. The tool itself in Cytoscape was developed in Java, and its task was to rank clusters based on the score of the nodes in a network. Comparing the results of these ranked clusters was then analyzed and plotted out through the use of Python scripts.

Contents

iv

List of Figures

List of Tables

Listings

Part I

Introduction

Chapter 1

Background

1.1 Oversight

This thesis is about implementing and using a tool named Ranklust. It is not a standalone tool, but rather a contribution to a Cytoscape plugin named clusterMaker2[1][2]. The goal of this tool is to rank clusters created from Protein-Protein Interaction (PPI) networks in Cytoscape. The ranks will be based on different node and edge attributes in the network. The resulting ranks will also indicate which clusters can be seen as cluster biomarkers, and which genes inside these clusters that could be considered to be single gene candidate biomarkers.

1.2 Biomarkers

A biomarker is a "biological measure of a biological state" [37]. Among other molecular entities, it can be represented by the levels of a specific protein in our blood, a specific gene, or a combination the two. Biomarkers can be used for different purposes. They can be used to measure the effect of cancer drug treatment. That the drug does what it is supposed to do. It can be used to predict disease development or the current stage of the disease. Here is a list of what biomarkers currently are being used for:

Usages for biomarkers: [68]

- Disease disposition

    - What is a patient's risk of developing cancer in the future?

- Screening
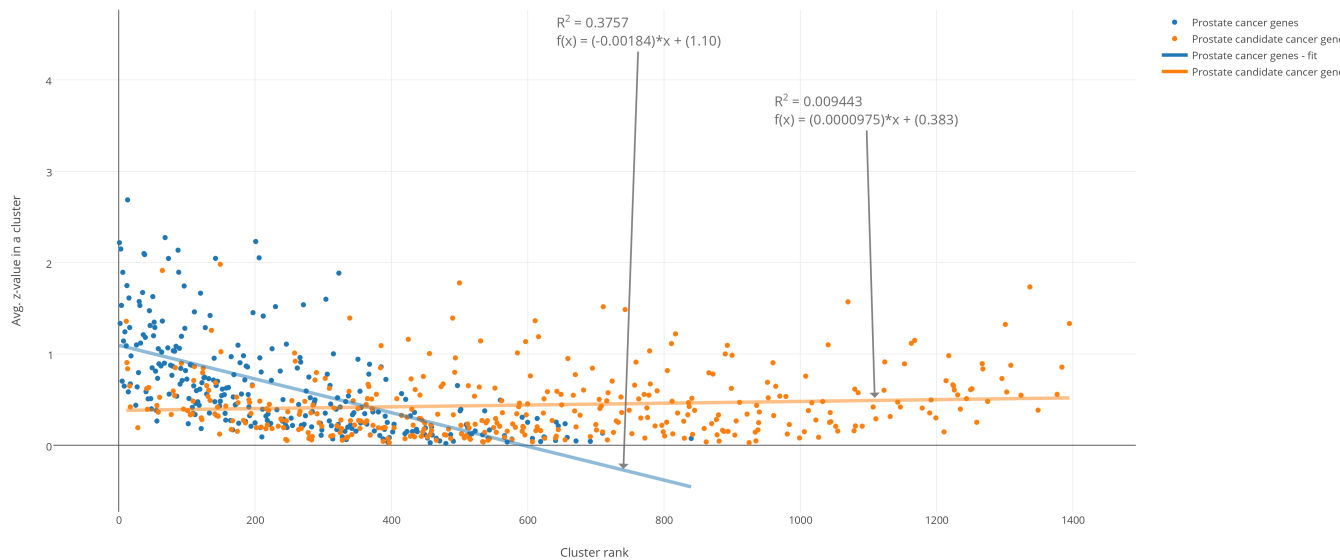
Figure 1.1: trololol

- Does earlier detection of patients with cancer decrease mortality?

- Diagnostic

  - Who has cancer? What is the grade of the cancer?

- Prognostic

  - What clinical outcome is most likely if therapy is not administered?

- Predictive

  - Which therapy is most appropriate?

- Monitoring

  - Was therapy effective? Did the patient's disease recur?

- Pharmacogenomic

  - What is the risk for adverse reaction to the prescribed therapeutic dose?

Characteristics for a good biomarker:

- Safe and easy to measure

- Cost efficient to follow up

- Modifiable with treatment

- Consistent across gender and ethnic groups

The National Institutes of Health Biomarkers Definitions Working Group has defined biomarkers as "a characteristic that is objectively measured and evaluated as an indicator of normal biological processes, pathogenic process, or pharmacologic responses to a therapeutic intervention."[41, 72]. The World Health Organization (WHO) has created a guideline for defining biomarkers: "almost any measurement reflecting an interaction between a biological system and a potential hazard, which may be chemical, physical or biological. The measured response may be functional and physiological, biochemical at the cellular level, or a molecular interaction". So a biomarker can be anything from the pulse of the heart or blood pressure. One of the goals of this assignment is to assess if it is possible to achieve novel information about network cancer biomarkers, by ranking clusters in the network. The result could in the best case be a new biomarker in itself, and a step in the direction of hierarchical biomarkers (biomarkers creating new biomarkers) in the form of cancer gene clusters.

1.2.1   Biomarkers and Clinical Endpoints

Some of the important characteristics of biomarkers is that they are objective and quantifiable as biological processes[72]. They are used as a state indicator of the biological object that is under scrutiny. When the biological object is a human being screened for cancer, biomarkers may help.

Biomarkers can be an indicator of how far cancer has progressed in the human body, even though the human subject may feel no difference at all. It can also be the opposite, that the human subject feels huge differences during several weeks that the cancer might have developed at a grand scale, but the biomarkers show no objective change. This proves that the human subject's experience and sense of state that it is in does not necessarily have to correlate.

Clinical endpoints is the opposite[72]. They describe how the subjects feel or describe how they function. It is not as objective as biomarkers and it demands more resources to gather the information, as the subjects has to be interviewed in some form, rather than interpreting pure data. But one thing has to be noted; patients does not seek treatment for cancer due to their biomarkers being "off the charts". They seek treatment because they feel that they do not feel ok or do not function sufficiently. So biomarkers

are not by any means far superior to clinical endpoints in all aspects.

Biomarkers can also be ruled out as a reliable predictor when population differences are too big[72]. As with clinical endpoints, people describe their feelings differently and might hold information back. As pointed out before, people's feelings are subjective, and different ways of interpreting their own body's state might skew statistical results with erronous feedback. Erronous feedback in this case can be examplified as two persons who are at the exact same state of cancer, with the same prerequisites, but they report how they feel differently. One person might be ignoring certain pains or lose hair after radiation treatment, but not the other. This can be mitigated to some degree with careful and accurate screening of patients admitted to treatment, but a totally unified group in terms of how they describe pain and other attributes that are interesting might be seen as borderline impossible. We are after all humans and very much fallible.

1.2.2 Prostate cancer biomarkers

An example of a single molecule biomarker is the prostate specific antigen (PSA). This is a protein produced by the prostate gland in male humans. The identification of cancer with PSA is simple, the higher the level of PSA, measured in ng/mL (nanograms per milliliter), the higher is the chance of the patient having prostate cancer [3].

Today PSA is used for both identifying and evaluating the current stage of prostate cancer. This biomarker can be found by analyzing blood examples from patients, thus fulfilling some of the demands for a good biomarker, but not all of them. It is easy to measure and easy to acquire, but not reliable enough to be used as the only marker to identify and determine the stage or remission of prostate cancer. The low grade of reliability comes from the fact that even though higher levels of PSA shows higher chance of having prostate cancer, prostate cancer is not the only reason to have elevated levels of PSA [3]. Namely inflammation and enlargement of the prostate. Though, a man with both of these cases may or may not develop prostate cancer. Using PSA as the only factor for determinating the stage of cancer in screening has caused a great number of cases where overtreatment has occured[64].

So the conclusion for the PSA biomarker is that it is not reliable enough and are causing faulty treatment of prostate cancer that may not even exist. Because even if a patient has prostate cancer, it may not be harmful, promoting the case of not taking any action against it at all. So there is need for a new biomarker, or at least a better way to diagnose and predict the right treatment.

Chapter 2

Networks as a tool for biomarker discovery in cancer research

2.1 Existing biomarkers

The PSA biomarker is over 20 years old [39]. Through those years, it has been discovered other and better biomarkers for prostate cancer than PSA. Among those, PCA3, which is detectable through urine samples from patients. It also has the benefit of not being affected by the size of the prostate gland [42]. But still the results could be better. Therefore, it has been tried to combine these two biomarkers in order to see if it is beneficial to see the results from each biomarker in light of each other [68]. The results from these tests is that they complement each other to a level of significance that makes it compelling to analyze them both to diagnose prostate cancer. It is important to point out that even if these biomarkers are not the best at indicating if a patient has cancer or not, these biomarkers are good at indicating progression and recurrence of prostate cancer.

In the cases of pancreatic, lung, breast, brain and overian cancer, the somatic distribution of single-nucleotide polymorphisms (SNP) has a few altered genes that occur with a frequency higher than 10%, and many other genes that are mutated occur with a frequency of 5% or lower[34]. On the other hand, prostate cancer has relatively few SNPs and copy-number alterations (CNA), so that kind of cancer is more likely to be driven by another somatic variation, namely DNA methylation. Cancer driver genes can be detected by positive-selection signals in the mutation pattern of genes in tumors, but there is a drawback with this method. Genes that are less frequently mutated within tumor samples might not be identified by a statistical analysis, even if they might be functionally important. An

alternative method is to use prior knowledge of cellular mechanisms, and add this prior knowledge as attributes to genes represented as nodes in a network. Then we can use graph algorithms to gain novel information about cancer driver genes. Genes that is not listed as cancer driver genes can then be assigned a status "guilt-by-association" and tagged as a possible candidate cancer biomarker if they have a network-relationship with proven cancer driver genes. This way of detecting biomarkers is how Ranklust will do it, create clusters, rank the network, and through the score in the nodes in the network, give each cluster a score and rank them accordingly.

2.2   High-throughput sequencing

Today, rapid analysis of genes and proteins are made available through Next-Generation Sequencing (NGS)[4]. Networks offers us an informatic, algorithmic, visual and mathematical tool to study this bigger picture. This project will integrate the opportunity networks offer to discover new biomarkers in cancer. Through acquiring more data, faster than before, there now exists databases with large amounts of information that is easy to access. This makes room for building huge networks of proteins and genes, allowing for more extensively and thorough assays to be done. For example, what if something that is classified as a prostate cancer biomarker only is viable when proteins that has not been classified as a biomarker, also is present? Together they could represent a more appropriate *network cluster biomarker*. The amount of data that can be analyzed also opens up for another more personalized approach to each cancer patient. Finding patient-specific biomarkers could make a huge impact on the quality of treatment [38, 54].

2.3   Networks as a tool in cancer research

Viewing the cell as a network of proteins and genes presents us with several assumptions to make. A way of defining edges between nodes has to be established. It exists several ways of doing this, but it depends on the context.

Bayesian probability networks are one way of defining edges [43]. It is based on the probability that if a node A exists, then node B exists. That way it is possible to create networks based on the assumption that if node A exists, then node B is 80% likely to exist. Maybe node C and D have a 100% chance of existing if node B exists. These percentages represents how strong the edges between the nodes are, and makes it able to construct a directed acyclic graph (DAG). It is important to note that a true bayesian

network can not be achieved through random observations. Rather should some constant value(s) be introduced to be able to really measure the effect of the other variables.

The way of defining edges in a network also determines what kind of information that is possible to get from it. The different bio-databases has different ways of calculating edges. Should the database alone define the edges? If the database supply weights in the edges and/or nodes, should it be used, changed or ignored? The final decision of how and in what way the edges should be defined in this thesis has yet to come.

2.3.1 Pathways and networks

Pathways are small networks of well studied processes where many areas are well documented. Networks on the other hand are bigger and less explored, but when properly analyzed, it might provide new information unknown to pathway systems.

Analyzing pathways and networks have an advantage over individual genes. They may reveal information that comes from molecular events that covers multiple genes or pathways. Aggregating this data can increase the chance of detecting driver genes through statistical analysis. Also, information gathered through pathways and networks may be enriched through genomic, transcriptomic and proteomic data and create a more unified view of the tumor biology.

Chapter 3

Network clustering: Toward network based biomarker discovery

Network clustering is the pre-processing of single gene prioritization that is necessary in order to come to the next step, clustering of the network. Clustering is about making a hierarchical view of a network to be able to look at the bigger picture. The reason to group a network into clusters and rank them is that the edges we create represents different connections. They can represent function, probability of existence, interactions or contents of the cell [31]. Also, abnormalities in a single gene is very rarely the cause of disease. More often, there is the interactions between several genes in a complex network[32], and their connections that can show us how diseases occur and evolve.

There is several algorithms to create clusters, and the most suited one seems to be Markov Cluster[30, 36]. The major differences on how these algorithms work is centered around how they handle nodes and edges. For example how the cluster is expanded from a single gene to several, what density level it is aiming for, how robust the algorithm is towards incomplete networks. It is feasible if the cluster-making algorithm is easily, or even already implemented as an app in Cytoscape. ClusterMaker2 is such an app, and it already has MCL implemented[1]. But there is faster cluster-creating algorithms than those implemented in ClusterMaker2, and an example is the SPICi [52] algorithm. So a possible solution for Ranklust could be to implement the SPICi algorithm into the ClusterMaker2 app, in order to reuse most of the code that represents the GUI, single gene prioritization and network creation. In addition to the cluster-creating algorithms, there is a need for cluster-ranking algorithms. They should be ranked in the order of being a cluster biomarker, more specificly described, a cluster biomarker for prostate cancer. The idea of clustering is to identify

relationships between cancer driver genes inside protein complexes and genes not related to cancer that reside within the same cluster. Since a cancer driver gene and a gene that has not been proved to be a cancer driver gene has a relation, the latter might play a role in cancer. These genes will be identified as candidate biomarkeres for prostate cancer. This was mentioned earlier as the "guilt-by-association"-principle.

Chapter 4

Ranklust: An tool to rank clusters

4.1 Cytoscape

Cytoscape is the open-source software platform the Ranklust app will be developed on. Its main purpose is to visualize molecular interaction networks and biological pathways. It is easy to integrate ***Apps*** and even combine multiple apps to solve new problems, given that the source code of the apps is available or that it exists an easy way of piping results from one app to another.

The goal of Ranklust is to rank the clusters in the network. Apps taking care of making the networks and creating clusters already exists, so Ranklust will only concentrate on the part that ranks the clusters and visualizes the results to the user of Cytoscape.

4.2 Programming language

The reason to choose Java above Python, Perl or other popular programming languages in bioinformatics is simply because of development experience. Java 8 is known for being this big bloated enterprise programming language, and Python as the fast and easy mockup tool to develop good programs fast. Python also has big biological computation libraries like *Biopython* [5] and sklearn[6], that makes it easy to build your own standalone apps. Though when used in a bigger environment as Cytoscape, Java shines, having sturdy packaging and modelling standards. The Cytoscape community is big, alive and has standards for how the architecture of apps should look like. The community promotes this through the use of the

OSGi standard [7].

Formatting and cleaning the data that Ranklust will provide and export out of Cytoscape is done in Python 2.7[69]. As mentioned over, it has many small libraries that can help with the bioinformatics part, and the scripts needed to format the data will rarely exceed over 100 lines of code. Third-party libraries like pandas[58] and numpy[35] will support the formatting and cleaning of data.

4.3   OSGi and design patterns

Developing OSGi software promotes modularization [8] of the code and increase the probability of the app being launched as an official Cytoscape app; in addition to provide other developers with the possibility of reusing my modules in their own apps. Also, it seems like Java 9 is aimed at making it easier to modularize apps along the lines of the OSGi standard[59], so it might be easier to refactor an application in the future from Java 8 to Java 9 when the architecture already is in place. There exists several design patterns that could prove to be useful in the development of Ranklust. Another strategy to follow may not be a direct design pattern [9], but more of a collection of them, and it is the clean code principles by Robert C. Martin [10]. Going against the coding principles promoted by the Cytoscape app community and exclude the use of OSGi modules will not be done, but third party Java libraries that is not OSGi-ready will be used, namely the JUNG library[11].

Chapter 5

Databases

5.1 Databases for network information

Which databases to use has to be considered. The reason to use databases is because they have information about how protein and genes form a network based on how they interact with each other. The initial database candidates in Ranklust were iRefIndex [12], GeneMania [13] and STRING [14]. These databases all have in common that it exists Cytoscape apps made to use these databases. STRING however, does not have any repository available through the Cytoscape app store, so interacting with the database through a new app in Cytoscape without making new plugins may be difficult. On the other hand, both iRefIndex and GeneMania have their repositories easily available to the public together with decent documentation. However, the difference between them is what they contain information about. iRefIndex contains data about protein-protein interaction (PPI), while GeneMania contains data about genes. Since proteins come from genes, GeneMania can also give us some information about proteins. The open-source plugins in Cytoscape to communicate with the databases are iRefScape [15] for iRefIndex and GeneMANIA [16] for GeneMania. The drawback with using apps for retrieving data from databases is that they have to be updated. The Cytoscape app of iRefIndex[15] is not up to date with Cytoscape version 3. The most suitable database and how to retrieve query data from it ended up being to visit the webpage of iRefWeb, and query a large amount of genes through the webpage's interface[17]. iRefWeb is webpage that works with the iRefIndex database and it supplies a network of over 100,000 interactions between proteins.

5.2 Databases for genes

To identify genes, a conversion from proteins to genes had to be performed. Since the data from iRefWeb was mined from several databases with a very low criteria of quality to get a large network, converting the nodes in the network from proteins to genes in order to be able to end up with prostate candidate cancer gene cluster biomarkers, HGNC[18] was used as the source for gene names. HGNC is able to produce large amounts of different aliases matched against approved gene symbols and names by cross referencing several databases. In this assignment, RefSeq IDs and UniProt IDs was used. This was also partially because the protein identifiers from iRefWeb was listed in this exact format.

5.3 Neo4J

Neo4J was considered as a source of data, and also as a way of maybe ranking the clusters in a network. Neo4J is a good database to work with when connections between nodes in a network is what will time to calculate. Before the final decision of opting for only node scores to score the network, making the choice of using Neo4J undesirable, Neo4J was therefore probed for its usages as a database together with Ranklust.

A problem we had with the Neo4J[49] database is the dump it creates. The dump creates a single Cypher[48] commit for the whole database. This way, if anything goes wrong while importing the data, it will get rolled back. It hinders faulty data and relationship between several parts of the data to be imported into the database. The problem with a single commit to import the data is the required memory. The personal computer used in this thesis and the computers at University of Oslo that we have access to did not have enough memory to import the database as a single commit in Cypher. Splitting up the dump to several commits does not work without quirks. It conquers the memory problem, but the way the relationships are created from the Neo4J dump requires the import process to be done in a single commit. A way to fix this is either not to use Cypher queries as a way of importing the data, but instead use GraphML[19]. Another way is to refactor the Cypher dump so that the creation of relationships does not need to be done in a single commit. We will use both ways, because not being able to use the Cypher queries because of too little memory will most likely be a problem for several people and not just me. Using GraphML is good for exporting data from Cytoscape and from the database, but interaction between the two of them while performing the algorithms is not feasible at the time. The last drawback is where Cypher queries are good, because it exists plugins that aid in using data from the database directly in Cytoscape. So initializing a database for testing could easily be done with GraphML, and then use Cypher queries to instruct a database to perform

computations on the data.

Creating a script to refactor the Cypher queries does not only help in regards of using the data with Cytoscape, but also for everyone using a Neo4J Cypher dump to create a database and does not have much memory. A problem that might come up is performance. Because the current way of creating relationships use node labels in Neo4J and it takes only a single line to create a relationship between two nodes. If we refactor the relationship-part of the Neo4J dump, it will increase to a two instruction query. Firstly, two nodes has to be found based on their indexed node name, which will relate to the protein/gene name. Secondly, we do the same thing that was done before the refactor, we create the relationship between the two nodes. The difference between the old and the new way of creating the relationship is how the nodes are found. The Neo4J dump creates temporary node variables prepended with an underscore and an integer ID. These ID's only exist within the commit the nodes are created. So splitting creation and relationship-making leaves us with worthless IDs that does not refer to any known node. That is when Neo4J tries to be smart and then creates new nodes based on the underscore ID and then creates a relationship between them. The end result, two new nodes that only contains the undescore IDs. So the refactored version aims at matching the underscore IDs in the relationship creation to the node creation, and to replace them with the name of the nodes instead of the temporary underscore ID. But as mentioned, matching the nodes with the name takes up more time and performance might be an issue. For comparison, with a Cypher dump of 4,500 lines, it takes about 20 seconds before the query engine gets a stack overflow error from too little memory (4GB memory). With GraphML, an XML file with 250,000 lines takes under 1 second to import and it gets relationships right without any form of refactoring.

Part II

Methods and implementation

Chapter 6

Cytoscape

6.1 Intro

The *clusterMaker2* documentation for implementing new parts that is not directly connected with clustering algorithms is non-existing. This part explains the details about how it was done.

Cytoscape has a cookbook[75] listing a combination of best practices and tips on how to start developing an app, add menues, panels, algorithms, color schemes etc.. Instead of following all of these examples, we have read through the existing code in clusterMaker2[2] and tried to structure the code of the contributions in Ranklust to match the structure already implementet in clusterMaker2. Meaning that for each algorithm implemented, they will all have a corresponding *Factory* class and a *Context class*. For the panel that was implemented, it has a pure panel class with Java Swing[62] settings, a panel task class, create panel task class and destroy panel task class.

6.2 Workflow and usage

6.2.1 Installation

At the moment, clusterMaker2 does not officially contain Ranklust. So in order to use Ranklust, the user are required to compile the Java source code or download the updated JAR-file[61](URL: https://github.com/henninglh/ranklust-app/ranklust-1.0.0.jar). To build this whole project I have used Maven as a build tool[27]. To compile the project into an executable JAR-file Cytoscape can use, Maven has to be instructed to skip all tests, because the current tests has compilation errors.

Skipping tests with maven:

```
mvn clean install -Dmaven.test.skip
\end{verbatim}

The JAR-file that is being produced after the line over is executed in
a terminal or through the build tool of an \gls{ide}. This file has to be put in
the configuration folder of Cytoscape. The location of the configuration folder
can vary across operative systems and Cytoscape versions, but the Cytoscape
version 3.4.0 follows this path:
\begin{Verbatim}[fontsize=\scriptsize]
<user home folder>/CytoscapeConfiguration/3/apps/installed/<put the jar here>
```

After this step is done, the user can start up Cytoscape and should be able
to Ranklust.

6.2.2   Researcher workflow

The way researchers can use Ranklust is pretty simple. Step 1 and 2 will
only be required the first time the researcher want to use this app to solve
his/her problem. The rest has to be done every time, unless a previous
session of Cytoscape is loaded with more developed results.

1. Install Cytoscape

2. Install clusterMaker2 plugin through Cytoscape App manager

3. Upload the network to be clustered and ranked into Cytoscape

4. Use clusterMaker2 to cluster the network

5. Use the Ranklust-part of clusterMaker2 to rank the clusters

    (a) Decide what to score the clusters on

6. Use the Ranklust-part of clusterMaker2 to visualize the rankings

Step 6 should show the researchers the ranking of clusters based on what
attributes they wanted to include in step 5a. This ranking represents cluster
biomarkers. The score of the clusters and the order they are ranked in will
decide the state of the patient. State of the patient can be many different
things and what the rankings will mean to the researcher is not yet final. It
will be a new indicator, a biomarker, and information about what it means
has to be gathered empirically through clinical research.

6.3 Maven and the POM file

The POM-file has to be updated because libraries connected to the pdf exporting functionality is outdated. Updating the libraries, imports and the usage of these libraries in the source code is enough to make the whole project compile. Also, in order to get the third-party libraries to JUNG[11] into the project, the three packages that is being used has to be added to the POM file and the classes they contain has to be exposed in the OSGi module[26]. Exposing the JUNG classes inside the clusterMaker2 OSGi bundle could have been avoided if JUNG had OSGi modules in maven repositories, but there is only 2 out of 3 OSGi ready modules that was needed in this project. These were the changes needed:

Changed which packages was exported through the OSGi module.

Listing 6.1: POM-file OSGi changes

```
1  <Export-Package>!${bundle.namespace}.*,*;-split-
       package:=merge-first</Export-Package>
```

Added these dependencies

Listing 6.2: POM-file JUNG changes

```
1  <dependency>
2      <groupId>net.sf.jung</groupId>
3      <artifactId>jung-graph-impl</artifactId>
4      <version>2.1</version>
5  </dependency>
6  <dependency>
7      <groupId>net.sf.jung</groupId>
8      <artifactId>jung-algorithms</artifactId>
9      <version>2.1</version>
10  </dependency>
11  <dependency>
12      <groupId>net.sf.jung</groupId>
13      <artifactId>jung-api</artifactId>
14      <version>2.1</version>
15  </dependency>
```

As seen here, the 3 modules needed is jung-api, jung-graph-impl and jung-algorithms. Only the first 2 was OSGi ready. An alternative could have been to create a OSGi ready module of the jung-algorithms module, but taking on the responsibility for having a module updated at all times is too much for a single person. However, it could become the clusterMaker2's developers responsbility to create and update all 3 modules. The last alternative is to find other graph ranking algorithm libraries like Apache Spark[28], thas is OSGi ready. The reason for not choosing Apache Spark is

that it was not discovered until all of the Java implementation was finished, making it a future goal to reach, at best.

6.4   App registration and java class connections

First step of registration is to register a *service listener* to the already existing clusterMaker2 *CyActivator*. Here I describe the name of java methods in the *clusterManager* class with strings. These two methods for adding and removing ranking algorithms registers and unregisters the algorithms to the *App* menu in Cytoscape, making them accessible for the user through the GUI. Registering a new service listener is a way of keeping the Ranklust part out of clusterMaker2. Also, creating a standalone plugin at a later stage will be easier if Ranklust is properly compartmentalized.

The *RankFactory* class is a java public interface used for each of the ranking cluster algorithms, HITS, PR, PRWP, MAA and MAM. A class applying the task factory design pattern is meant to deliver an object of the class it is related to[76], and it can return different types of a class that has the same java superclass[60]. In this case, each class implementing the *RankFactory* will only have a single class to return. The RankFactory class also creates a *Context* class, which binds information the user can input to the GUI to variables, allowing the algorithm to gain access to parameter information about the algorithms before they run.

6.5   Algorithm

6.6   Ranking panel GUI

The code for the *RankingPanel* is a copy from the existing *ResultsPanel* in clusterMaker2. The results panel has some extra information that is removed in the ranking panel. The ranking panel only displays which clustering and ranking algorithm that was used and the network it is used on, together with the score of each cluster sorted descendingly, having the highest ranked cluster at the top. The ranking panel will display in the same place as the results panel, to the right of the network view.

The panel supports multiple selection of clusters through the *Control* key on the keyboard, much like its existing behaviour on Windows when selecting multiple directories or files in the *File Explorer*. Selecting the clusters in the panel will also select them in the network view, enabling the user to use other Cytoscape utilities on the nodes/clusters selected, for example create a new network based on only the selected nodes. The

color of the selected nodes will also change in the network view, when the user selects a cluster in the ranking panel, providing visual feedback to the user, in order to make it easier for the user to see where the selected nodes/clusters resides in the network.

When tasking clusterMaker2 with showing the ranking panel, the color of the nodes will also change. This is to visualize the rank of each cluster in the network view to the user. Coloring the highest scoring clusters with red or green and the lowest scoring with the opposite was the first and easiest solution. There was made a choice to color the whole cluster according to the rank it received, instead of individually color nodes according the their contribution to the clusters rank. The user might be interested in choosing what suits them best when it comes to this, so it might be implemented a menu for the ranking panel in the future to best meet the users needs.

Taking color blind people into consideration red and green is not the best combination. Colors resembling red and green in the form of hot-to-cold colors that all types of colorblind people can view was therefore a criteria to be met. A style satisfying these criteria will follow, having the hex value of te color[44], followed by the RGB value that was calculated using a site that converts from hex to RGB values[20]. *#ADD rgb*$170, 221, 221$ for a blueish and cold color representing low score. *#FEC rgb*$255, 248, 204$ represents off-brown color that is a step warmer than the blueish. The warmest color representing the highest scored clusters is *#F99 rgb*$255, 153, 153$. The drawback with this color combination is with people that has color blindness to the degree where they see only greyscales. The colors low to high will go from grey to light grey to dark gray, which does not seem logical. The color style might change if users experience trouble having these colors representing the cluster ranks.

6.7   Current state of development

The pull request[47] made from the Ranklust contribution of this thesis has been accepted[70], adding 3750 lines of code and deleting 1051. The changes was distributed across 50 files. The type of files edited ranges from the Maven POM-settings file and .gitignore to plain Java source code files. Further changes will be made, as the work on the Ranklust contribution will not end when this thesis is delivered. One of the most recent changes to the contribution is that the *PageRankWithPriors* algorithm used to analyze the networks in this thesis assumes undirected edges because of the *UndirectedSparseMultigraph* graph used. For normal use, directed edges is believed to be the most sought after choice. Therefore, the pull request was hardcoded with directed edges. Choosing between directed and undirected edges should in the end be up to the users of clusterMaker2 and is on the TODO-list of futures to implement.

6.8   Known bugs

6.8.1   Menu bugs

Unintentional execution of clustering algorithm

1. Open the clusterMaker2 clustering algorithm menu

2. Select a clustering algorithm

3. Exit the menu for the algorithm without running it

4. Repeat step 1-2 for the same algorithm and it results in running the previous algorithm that was exited without displaying the parameter prompt

This was the behaviour before Ranklust was added to clusterMaker2, so it might be clusterMaker2 or it might be Cytoscape itself. Most likely, the way clusterMaker2 algorithms is started through its *TaskFactory*'s needs to be changed, but this is just a hunch and not based on any analysis.

6.8.2   Ranking panel bugs

It is a known bug with the style for node coloring that the ranking panel causes. The colors in the nodes might flicker when several ranking algorithms has been run and the colors of each node has changed several times. This is probably related to the previous color style for the nodes and it will be fixed in the future.

6.9   Clustering

PPI networks already have a great deal of edges, and can be seen as clusters that I should not alter. On the other hand, using clustering algorithms to make clusters out of PPI-networks gives us the more control over the clusters and has the potential to identify protein complexes[77]. How big they should be, how many of them I want, should I cluster on a certain attribute, or even several? I have chosen to go with the *Markov Cluster*(MCL)[36] clustering algorithm in *clusterMaker2*, without any limitations to the amount of clusters or weighting of the nodes. I prototyped Ranklust with *Affinity Propagation*(AP)[40] because of its ease of use and low execution time, but through a comparison between MCL and AP, it was concluded with MCL having better performance in many aspects when it came to cluster unweighted PPI networks[77] with binary interactions.

These aspects being noise tolerance and more robust behaviour, which will contribute greatly to cluster the iRefWeb network.

6.10 Ranking clusters

I have used five algorithms to rank the clusters, MAA, MAM, PR, PRWR and HITS. The first two are simple algorithms that through addition or multiplication calculates a cluster's average score. The three others utilize network ranking algorithms from the Java JUNG[11] library.

Every algorithm implemented in Ranklust for ranking clusters except HITS takes node and/or edge scores as input for calculating the scores for each cluster.

6.10.1 Multiple Attribute Additive Method (MAA)

Multiple attribute additive method is the first algorithm implemented. The user has the option of choosing an unlimited amount of attributes from nodes and edges.

It goes through all of the nodes in each cluster and sums up the number-attributes the user chose. Each cluster is then ranked based on the average sum in each cluster and ranked descending, with the highest ranking cluster as the most likely prostate cancer biomarker cluster.

There is a question as to how to rank the edges in the cluster. We chose to rank each edge as it is listed to the user in Cytoscape. So if it is listed only once time in the edge table, it will only be scored additively once. This decision was based on simplicity. To not represent the edge as something the user did not define it as, or is unable to understand. Some clustering algorithms might assign the same node or edge to several clusters, though this is not the case with the algorithms I use in this thesis. Support for this is only implemented by MAA and MAM, as they were implemented before a final decision on which clustering algorithms should be used. If MAA/MAM discovers this special case of several scores for a single node or edge, it will assign it the highest value. The reason for leaving this feature in Ranklust is to have an example on how it can be done, should it be a problem in the future.

6.10.2   Multiple Attribute Multiplication Method (MAM)

Multiple attribute multiplication method is to some degree redundant, considering what exists from before in Ranklust. The only difference from MAA is the scale the scores will be in. MAA adds the scores from each node and edge in the cluster through addition, MAM does it with multiplication.

A problem that occurs with multiplication is calculating scores for clusters that contain nodes with a score between 0 and 1, since the score would decrease to such a degree that it would be difficult to work with when normalizing the scores. The solution I have chosen for this problem is to make a new score from the score that is to be added to the cluster average, and add 1.0 to it. This way, when the existing average score in the cluster is multiplied with the new score, it will always increase, unless the old value was 0.0, or both the old and the new value was 1.0. In the case of values above 1, they will also be given an increase by 1.0 in order to keep it consistent if the scores vary between 0 to n. Normalizing every value before running the algorithm contributes to keep all of the values between 0 and 1, and in that way prevent scaling problems when adding 1.0 to a score over 1.0. The whole reason to add 1.0 was to counteract the problem with the score decreasing when it should be increasing.

6.10.3   PageRank (PR) and Personalized PageRank (PRWP)

A Random Walks[63] algorithm which used priors, called seed-weighted random walks ranking [45], proved to be effective at prioritizing biomarker candidates. PageRank (PR) is an algorithm based on the Random Walks principle, and it is contained inside the Java library *JUNG*[11]. PageRank was previously used by Google to rank webpages [21]. Personalized PageRank (PRWP)[51] is a modified version of PR, where nodes and edges can be assigned a score prior to the PR's traversal of the network. PR can have values assigned to the edges, but it does not require any scores in order to rank clusters, which PRWP does. Personalized PageRank is abbreviated PRWP because of the Java classname it has in the Java JUNG library - *PageRankWithPriors*.

The difference between MAA and MAM compared to PR and PRWP is how the network is scored. MAA and MAM calculates the score for each cluster by summing up the attributes in edges and nodes according to the cluster attribute. PRWP scores the current network regardless of the clustering attribute.

MCL gives the option of creating a clustered network, which opens up the possibility of working with two types of the same network, non-clustered and clustered. They both have the clustering attribute in the network, edge and node table, so that the ranking algorithms are able to score the clusters.

PRWP scores the currently selected network in Cytoscape, resulting in the option of scoring the non-clustered network or clustered network. The last option gives the clustering algorithm a bigger impact on the score, because the clustered network has perturbated edges between nodes that is not in the same cluster. MCL in Cytoscape can show the "inter-cluster" connecting edges, which is the egdes that was perturbed during clustering. This last option is a combination of the two others. It will be visually close to the clustered network, but algorithms that run on the network with inter-cluster edges will have the same result as the network only containing the cluster attribute.

Implementation wise, there is also a difference in how the scores are stored in the network after the algorithm is finished executing. All the scores are stored in the nodes. To give information to the user about the scores the edges received, each edge will display the total score for the cluster it is a member of, just like the nodes. Only PRWP and PR will also display the single node score.

6.10.4 Hyperlink-Induced Topic Search

Hyperlink-Induced Topic Search, HITS, an algorithm that is similar to PageRank, was developed around the same time [22][23] and is also contained within the Java JUNG library. HITS will not be used to calculate cluster scores and ranks, due to the fact that it does not require any form of weighting in nodes or edges. Though, it can be used in combination with other ranking algorithms through the use of MAM or MAA.

An example of this could be running PRWP with a score attribute, then HITS on the same network. The next step would be to combine the two scores from PRWP and HITS with MAA. Though, the idea of achieving novel information on network biomarkers for prostate cancer through this workflow is pure speculation, and we have chosen to use each ranking algorithm separately, in order to limit the amount of datasets to analyze.

6.10.5 PR, PRWP and HITS

Because PR, PRWP and HITS rely on an alpha variable controlling the probability of a reset in the traversal of the network. This "traversal reset" is automatically triggered if the algorithm hits a node with no outgoing edges. As the clustering algorithms might change the edges in the network, the outcome of performing ranking on a cluster-created network, versus a network with only a cluster attribute to identify the clusters, can potentially be vastly different. I have decided to not use the cluster-created network, and instead use the network with cluster attributes. The perturbation of edges in the network and separation of nodes into a more disconnected

graph might hide interactions in the network that can provide useful information to PRWP. On the other hand, it might produce more noice in the network and skew the ranking of the clusters through interactions that are false positives, regarding protein complex interactions.

The alpha value parameter will be set to 0.3. This specific alpha value has been proven to be effective in identifying candidate genes in diseases[33]. Another parameter for PR, PRWP and HITS is the iteration parameter. For each iteration of these algorithms, they assign a value to each node. Performing multiple iterations contribute to stabilize these values. In the previously mentioned report from Google [29], a *n x n* matrix where *n* was about 25 billion, required about 50 - 100 iterations in order to stabilize the node values. Iterations needed to stabilize increase with the size of the network, therefore the iteration value to start with will be 30. It is a little less that what Google has specified, but in contrast, the iRefWeb network is 9500 nodes big, with approximately 43,000 edges, which is a considerably smaller network than "the internet", which Google tried to rank.

Chapter 7

Programming specifics

7.1 Tables vs pure OO

One thing in particular that deserves to be mentioned is the way networks are handled. The *CyNode* objects, which represents a single node, does not contain much. This is because all of the information is saved in the form of plain cells in a spreadsheet. This may at first seem like a way to make it easy to show information to the user, but it is also a way of working more efficient with network graph data. Creating objects with attributes for each node in a huge network will increase the amount of overhead. Working with all of the information in the way of a spreadsheet with rows and columns results in a decrease in overhead. A new node is a new row in the node table, so in relation to building the network structure, it is not a complex abstraction. The difference comes in when the nodes have several attributes.

In a table or spreadsheet, attributes can be represented as a single column and be created once for the whole network, instead of once for each node object created. This assumes that getting the objects out of the table is possible by either indexing on a number for arrays, or a unique key for map structures. The result is both a lookup, insertion and deletion time of O(1). These times is as low as the Big-O notation goes in terms of speed related to the size of a collection inside a data structure. So both the creation of objects goes down, and the retrieval of attribute information is as low as it is possible to get, when I choose to represent the time by Big-O notation. The only drawback with this implementation is that there is no current type of wrapper around the row and column system. So the retrieval of information is not done the most intuitive way. This is the way Cytoscape works as a whole, so changing the way this works can either be done through changing the Cytoscape source code, or implementing a wrapper as a standalone function inside clusterMaker2 or as a standalone plugin. In Ranklust, an extension to the existing *NodeCluster* class was

used to keep scoring information about nodes in clusters.

7.2 Changes in existing classes

7.2.1 NodeCluster

```
/**
 * In it's simplist form, a Cluster is a group of nodes that represents the
 * nodes that are grouped together as the result of a clustering algorithm
 * of some sort.  A more complicated form of a cluster could include clusters
 * as part of the list, which complicates this class a little....
 */
```

This comment is in the NodeCluster class. In the Ranklust implementation, several attributes and methods has been added to this class. Saving the temporary state of the scores to the clusters could have been done in the node and edge-tables in Cytoscape, but it was faster both implementation- and performance-wise to extend the NodeCluster class. Also, a criteria for extending this class was that the existing API of this class should not change, in order to avoid unneccessary changes for other classes in cluster-Maker2, that is not a part of Ranklust.

Additions to NodeCluster contains variables representing cluster *score*, *rank* and a *HashMap* from a cluster-node's SUID to its score. The map and rank variables is part of a previous implementation and can be removed. Adding node scores to the cluster and normalizing it afterwards.

Common responsibilities introduced in Ranklust

- Adding node scores to the cluster (ranking algorithm related)

- Normalizing cluster scores for a list of clusters (static method)

- Calculating min/max/avg scores for a list of clusters (static method)

- Ranking a list of clusters by their score and set their rank accordingly (ranking panel related)

7.2.2 GraphicsExportPanel

Minor changes were done to GraphicsExportPanel when a new class for handling PDF documents was introduced. This was the result from the need to change the maven dependency to use the "com.itext"[71] repository location instead of a location from "com.lowagie" because the library was relocated [56].

7.3   New classes

7.3.1   ClusterUtils

ClusterUtils is a new class implemented in the Ranklust contribution. It is used by every cluster ranking algorithm implemented in Ranklust. To some degree it has a common responsibility with the *ModelUtils* class from clusterMaker2. The difference being that ClusterUtils is focused on inserting scores in tables related to cluster ranks. It also normalizes scores not directly related to a NodeCluster, but rather a group of *CyNodes*, that has mapped their SUID in Cytoscape to a score received from a ranking algorithm.

Common responsibilities introduced in Ranklust

- Getting cluster attribute based on CyNetwork

- Getting ranking attribute based on CyNetwork

- Ranking clusters firstly by score, secondly by cluster number (assuming a small cluster number is a bigger cluster)

- Fetching ranking results based on CyNetwork (ranking panel utility)

- Fetching clusters based on CyNetwork (ranking algorithm utility)

- Add score to a NodeCluster based on specified attribute column and CyRow to get it from

- Insert cluster scores into the default node- and edge-tables

- A simplified way of creating columns based on a specific column name, attribute class, table to create column in and immutability status

7.4   Handling nodes and edges

Recipe for working the nodes and edges: The steps are almost equal for calculating both node and edge scores. How the edges are handled depends on what direction they have. Are they undirected, directed, or unidirected. In the current version of Ranklust, *PageRankWithPriors* assumes undirected, *PageRank* and *HITS* assumes directed. *MAA* adds the score from the edge to the cluster and *MAM* multiplies the highest score found as an edge attribute with the current cluster score. MAM and MAA assumes the score is directed corresponding with source and target nodes

in the edge table of Cytoscape. If an edge is a part of several clusters, each cluster will add the score of the edge to its average cluster score. For PR and PRWP, the score will always reside within nodes and not edges at the end of execution. To distribute the scores among the edges, a traversal through all of the edges, which is being matched against all of the nodes in each cluster is performed. When the traversal for an edge hits a cluster with a node that matches either the target or source node in the edge, the cluster that has this matched node adds its cluster rank score to the edge.

1. Add score to the cluster

2. Repeat step 1-2 for every edge

3. Sort clusters based on rank and create a column to represent the cluster score

4. Create single node score from cluster

Chapter 8

Data pre- and post-work

Interpreting these results was done solely by Python scripting, from formatting the data retrieved from databases in order to construct networks in Cytoscape, to cleaning the results exported from Ranklust, and analyzing the cleaned data.

8.1 Before using Ranklust

8.1.1 Network creation

The network was created with protein interaction data from iRefWeb [17].

Figure 8.1: iRefWeb network query

Source Database
- ☐ bind (1253)
- ☐ bind_translation (674)
- ☐ biogrid (82433)
- ☐ corum (72)
- ☐ dip (946)
- ☐ hprd (8278)
- ☐ innatedb (2062)
- ☐ intact (13171)
- ☐ matrixdb (106)
- ☐ mpact (0)
- ☐ mpidb (0)
- ☐ mppi (8)
- ☐ ophid (0)

Can match ANY of these ▼

- ☑ seen by 1 DB (109003)
- ☐ seen by 2 DBs (0)
- ☐ seen by 3 or more DBs (0)

Organism **
- ☑ single organism interaction (109003)
- ☐ cross organism interaction (0)

- ☑ Homo sapiens (109003)
- ☐ Saccharomyces cerevisiae S288c (0)
- ☐ Drosophila melanogaster (0)
- ☐ Mus musculus (0)
- ☐ Arabidopsis thaliana (0)
- ☐ Escherichia coli K-12 (0)
- ☐ Caenorhabditis elegans (0)
- ☐ Campylobacter jejuni subsp. jejuni NCTC 11168 (0)
- ☐ Schizosaccharomyces pombe 972h- (0)
- ☐ Rattus norvegicus (0)

More filters hidden

Must match ALL of these ▼

Nature of Interaction
- ☐ unary (0)
- ☑ pairwise (109003)
- ☐ multi-subunit (0)

- ☐ predicted (0)
- ☑ experimental (109003)

- ☐ genetic (36)
- ☐ physical (108977)

MI (MINT-Inspired) Score

MI (MINT-Inspired) Organism Percentile

Interaction Detection Method *

Interaction Type *

Number of Publications
- ☐ no valid PubMed ID (0)
- ☑ 1 or more publications (109003)
- ☐ 2 or more publications (12232)
- ☐ 3 or more publications (6184)
- ☐ 4 or more publications (3980)

Must match ALL of these ▼

* Filter counts for interaction method and type propagate to their parent terms as defined by the PSI-MI ontology.
Thus a selected method or type filter will return all of its child terms as well.
** Organism filter counts (for instance yeast and its subspecies) do not propagate.

iRefWeb

It was downloaded in the MITAB-MINI format. A python script cleaned up the file, leaving only the aliases for the source and target proteins. This result was matched up against genes from HGNC. The HGNC data represents genes, and their protein product. This way, the network would consist of genes instead of proteins. This conversion was done in order to compare the end results from ranking the clusters with genes that has connections to prostate cancer. A criteria for the interactions retrieved from iRefWeb when converted from proteins to genes was that both the source and target protein had to have a match in the HGNC data, to not create any combinations of gene to protein or protein to gene interactions. Duplicate interactions was not taken care of and they did not occur in the network file either.

COSMIC

A network with only prostate cancer related genes from COSMIC[50] was constructed with interaction information from the iRefWeb network. This network was much smaller in terms of both nodes and edges (interactions).

36

The COSMIC network was created and ranked to see how ranking a network with only cancer relevant genes would result in when data from jensen was used to compare it to a much larger and generalized network. Filtering the genes from COSMIC was done the same way as filtering the scores for the genes; using only the COSMIC genes that contained genes from both ends, source and target gene, of an interaction in the network.

8.1.2   Score creation

The golden standard for scoring that I used was created by data from DisGeNET[65] and DDPC[55].

DisGeNET is a database that is one of todays largest repositories of information between diseases and genes. It integrates expert-curated data with text mined data. DisGeNET provides a score for genes related to a specific disease, in this case prostate cancer. This score is based on "the supporting evidence to prioritize gene-disease associations"[65]. It also has a Cytoscape plugin, but like iRefScape, it is outdated and can not be used together with Cytoscape version 3.4.0. DisGeNET was chosen because of its comprehensive collection of prostate cancer gene scores that was used as prior scores in both PRWP and MAA.

DDPC is a knowledgebase of genes that is been experimentally verified as an implicator in prostate cancer. It offers a bigger variety of information on prostate cancer in general, as opposed to DisGeNET, but it does not have scores. Therefore, to be used in the golden standard, the genes from DDPC had to receive a constructed score comparable to the scores from DisGeNET.

The DisGeNET data came with decimal scores from 0 to 1 and contained data about several diseases, not only prostate cancer. Therefore, the disgen data had to be filtered for prostate cancer relevant scores. However, the data from DDPC did not have any scores and was just a list with genes relevant to prostate cancer. To combine the two lists, the DDPC scores had to be converted to match the disgen scores. The method for converting the scores was to let the disgen scores be left as they were and try to add scores to the genes from the DDPC data list. An average was created from the maximum and minimum value from the disgen scores. Then some sort of "sane" max value was created by adding the average to a quarter of the value of the average value subtracted from the maximum value. This created a final value between the average and maximum value that all of the DDPC genes received. The DDPC genes receives on average a higher score than the DisGeNET genes because of the way the lists are constructed. DDPC has no score, therefore they signalize a proof of relevance to prostate cancer, while DisGeNET has a score for how relevant the gene may be.

Finally, this golden standard of scores was filtered by removing any genes that did not occur in the network. Because of the cross-validation performed after the ranking of the clusters, the scores could not contain any genes not in the network in order to not remove genes from the files with scores that would not occur in the network. This would have resulted in a cross-validation where a 100% match never could happen.

8.1.3 Creating scores for cross-validation

A form of cross-validation was performed to assess how the ranking algorithms performed alone and compared to each other. Creating the cross-validation scored data sets required information about the clusters. Running MCL clustering on the network and then export the node table in Cytoscape created the cluster basis needed. From these clusters, it was possible to identify which clusters contained scores that could be removed during cross-validation. A random 10% of the genes used for scoring was removed. Though, the amount of genes removed was not totally random. There was set a rule that every cluster that contained genes with a score from the golden standard, had to end up with keeping atleast 1 of those genes. So in a cluster of 4, where 2 of them were scored in the golden standard, that cluster could only lose 1 of those scored genes when the 10% was removed. Had this rule not been applied to the cross-validation, there would not be a guarantee that 100% of the removed genes would be found as candidate genes. For MAA ranking, it would be impossible, and for PRWP it would be possible, but much less likely to find them. This is due to the simple fact that clusters that go from having weighted nodes (genes), to having no weighted genes, should not be ranked very high. The point of having weights is to rely on prior information about cancer relation in genes and find cancer candidate gene clusters by applying reason for "guilt-by-association" to the genes that are not weighted.

8.2 After running Ranklust

8.2.1 Exporting the data

The data from performing Ranklust on the networks was exported as node tables through Cytoscape. These tables contains information about each node, but not any information about the edges. The scores added to the networks did not involve any edge information, and for both of the algorithms used, MAA and PRWP, the calculated scores are stored in the cluster, so there was no use at this point to use the information in the edge table to find the cancer candidate genes.

8.2.2   Cleaning the data

Cleaning the data was done in Python with its table manipulation library, Pandas[58]. The cleaning removed unnecessary data from the node table files from exported from Cytoscape. The data still represents the same before and after the cleaning, the difference is that the cleaned version is has tab separated data, for better visual interpretation and it has less columns.

Before cleaning:

```
"SUID","__mclCluster","name","PRWP","PRWP_single","score","selected","shared name"
"72","25","FAM160B1","0.04478373407949736","7.818032875772688E-4",,"false","FAM160B1"
"73","6","APOPT1","0.04119274479140687","9.848244844960219E-4",,"false","APOPT1"
"74","6","FAM84A","0.04119274479140687","9.848244844960219E-4",,"false","FAM84A"
```

After cleaning:

```
__mclCluster name PRWP score PRWP_single
1136.0 CDC25C 1.0 0.272714418721 0.225926983193
1136.0 LZTS1 1.0 0.122995792115 0.197642927074
690.0 CREB3L4 0.918902810704 0.272714418721 0.189744545774
690.0 SCX 0.918902810704 0.0 0.102170140032
```

- __mclCluster

    – Which cluster the gene/node in this row belongs to

- name

    – The HGNC representation of this gene/node

- PRWP

    – The score of the cluster to this gene/node belongs to

- score

    – The prior score assigned to this gene/node

- PRWP_single

    – The PageRank score assigned to this gene/node (not a column in MAA)

8.2.3   Recreate the clusters

The cleaned data was then aggregated into clusters represented in text format.

Combined clusters:

```
clusters score genes
1136 1.0 LZTS1:0.122995792115,CDC25C:0.272714418721
690 0.918902810704 SOX9:0.272714418721,SCX:0.0,CREB3L4:0.272714418721
1004 0.758524309337 MMP14:0.272714418721,MMP13:0.12
```

Here, the *clusters*-column are the unique identifier, and not the *HGNC* identifier or the *SUID* from the previous data. The score column does not represent the same data as before, but rather the same as the PRWP column displayed in the cleaning examples, the total score of the cluster. The genes column is double separated. The column itself is tab separated from the other columns, the genes in it is separated with commas and the prior scores belonging to each gene is delimited by a colon, not the PRWP/MAA or any other algorithmic related score. This column contains all of the genes in the cluster and aids in distinguishing the already proven biomarkers for cancer, and the candidates.

8.2.4   Cross-validation comparison

Both PRWP and MAA had 10 cross-validated results each. Each individual result from both of the algorithms was compared with the golden standard result created by each of the algorithms. This procedure was done to identify the cancer biomarkers that existed in the golden standard but ended up being candidate cancer biomarkers in the cross-validation.

8.2.5   Comparing with other testdata

– Movember –

8.3   Plot creation

The plots was created with an online tool named Plotly[67]. Throughout the whole process of creating plots, the X-axis represented the cluster ranks, ordered from best to worst scored cluster, with ascending values representing the ranks. The Y-axis represented in all cases an average. This

average was always calculated by dividing the value or score under scrutiny, by the size of the cluster.
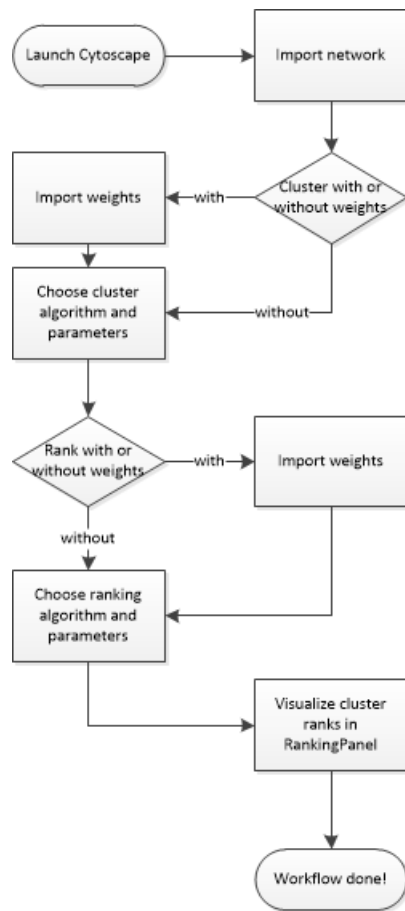
Part III

Results

Chapter 9

Ranklust as a tool for researchers

9.1 The planned workflow in general

First off is the general workflow:

9.2 Detailed walk-through of using Ranklust

To go into more detail, here is the startup screen that the user is met with after launching Cytoscape. This assumes that clusterMaker2 with the Ranklust contribution is already installed into Cytoscape.

This screen is what opens up after opting to import a network and choose a network file consisting of a header that represents source and target node columns, a_alias and b_alias.
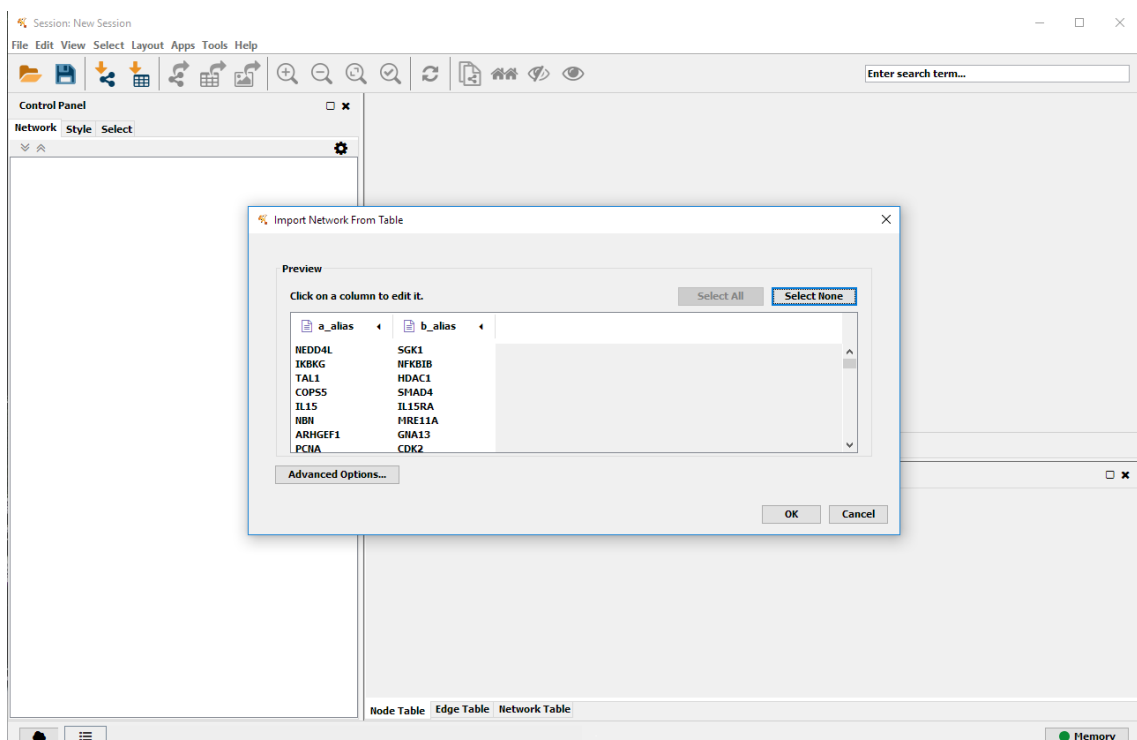
It is important to tell Cytoscape which column is considered as the source and target column, as shown here. In this thesis, genes has been the primary key identifier in both source and target column.
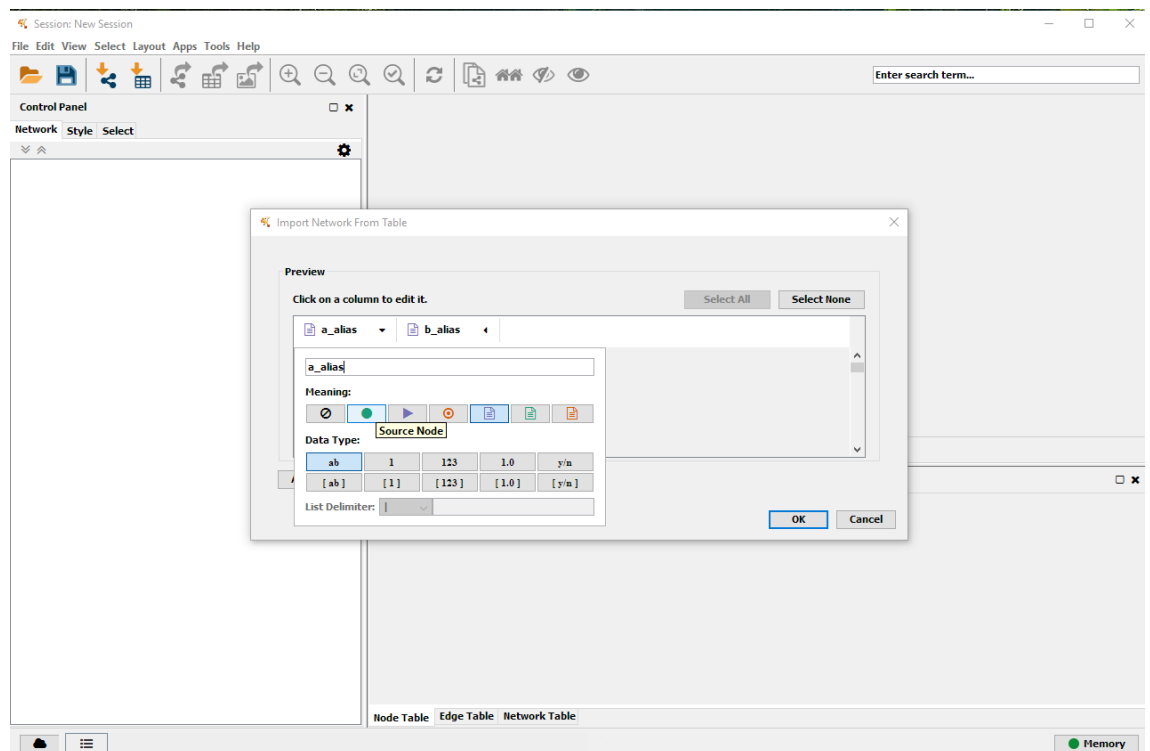
This is how Cytoscape looks after importing the network and it has finished its standard style layout algorithm, which happens automatically after clicking "OK" in the previous screen - having set everything to the correct settings.
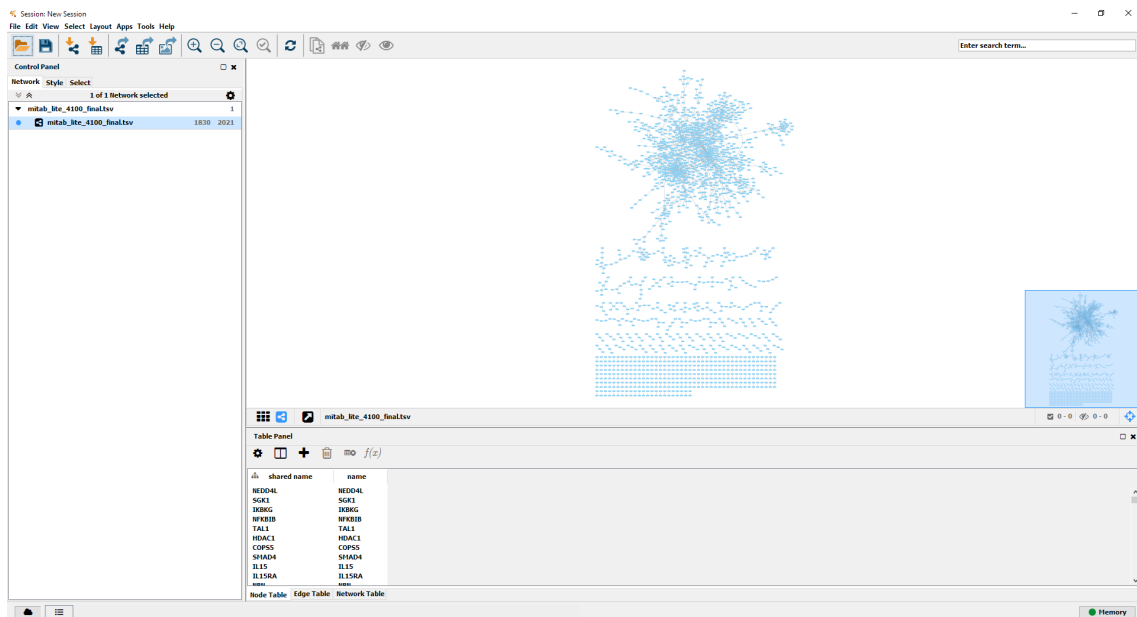
If the user wishes to weight nodes for clustering or the ranking of them, it is possible to do this with the "Import Columns From Table" function.
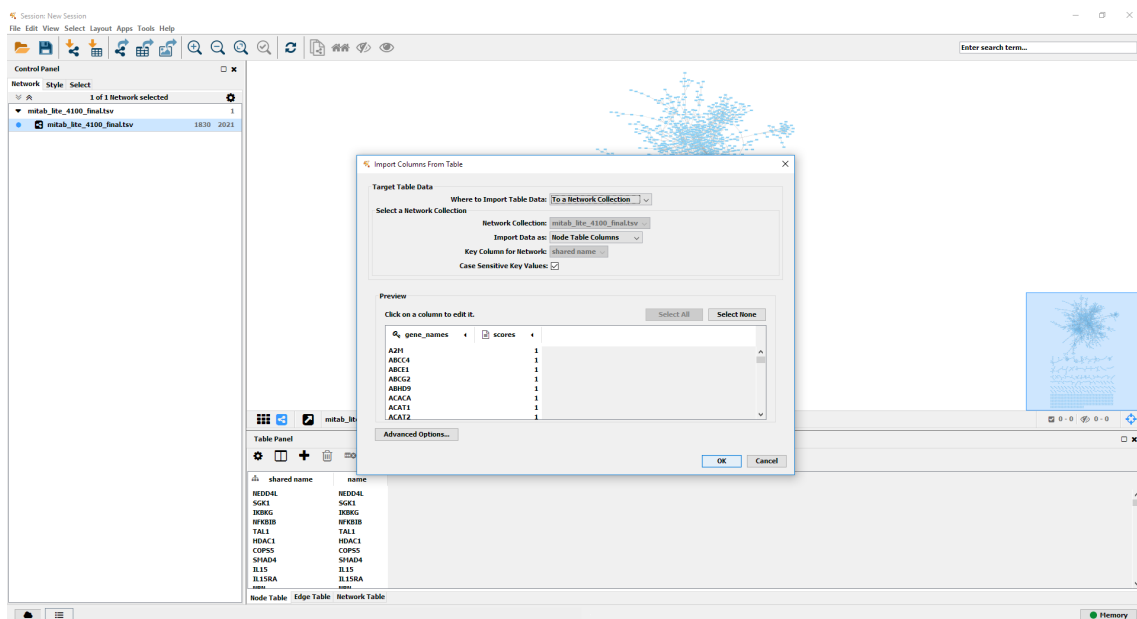

To cluster the network, the user has to access the *Apps* menu at the top in the toolbar of Cytoscape. In clusterMaker2 there are three rows belonging to the plugin. *clusterMaker*, where the clustering algorithms are located. *clusterMaker Ranking*, where the cluster ranking algorithms MAA,MAM,PR,PRWP and HITS are. Finally, the *clusterMaker Visualizatons*, where different visualizations in clusterMaker2 can be queried. This row is also where the visualizaton of the ranked clusters option resides.
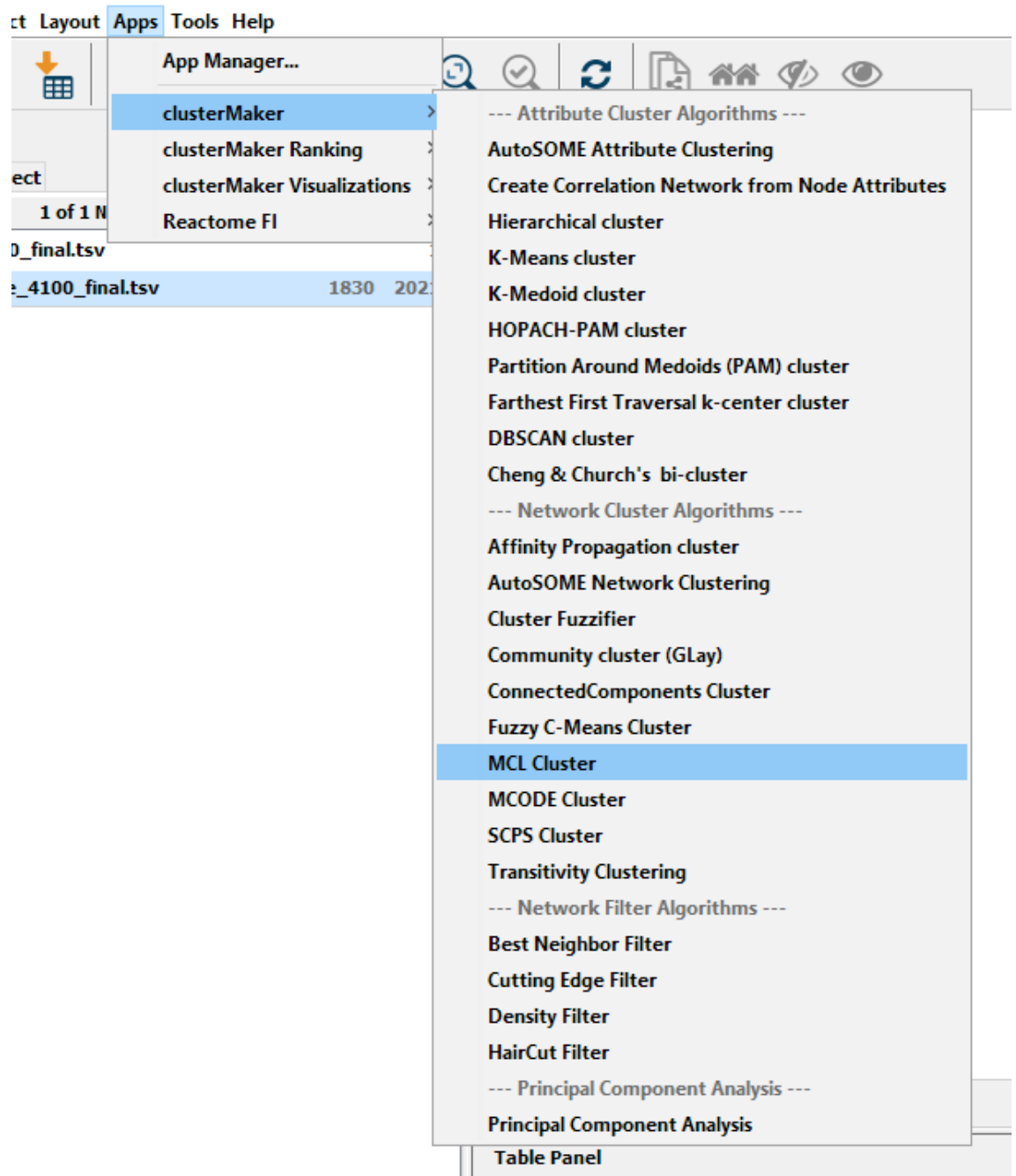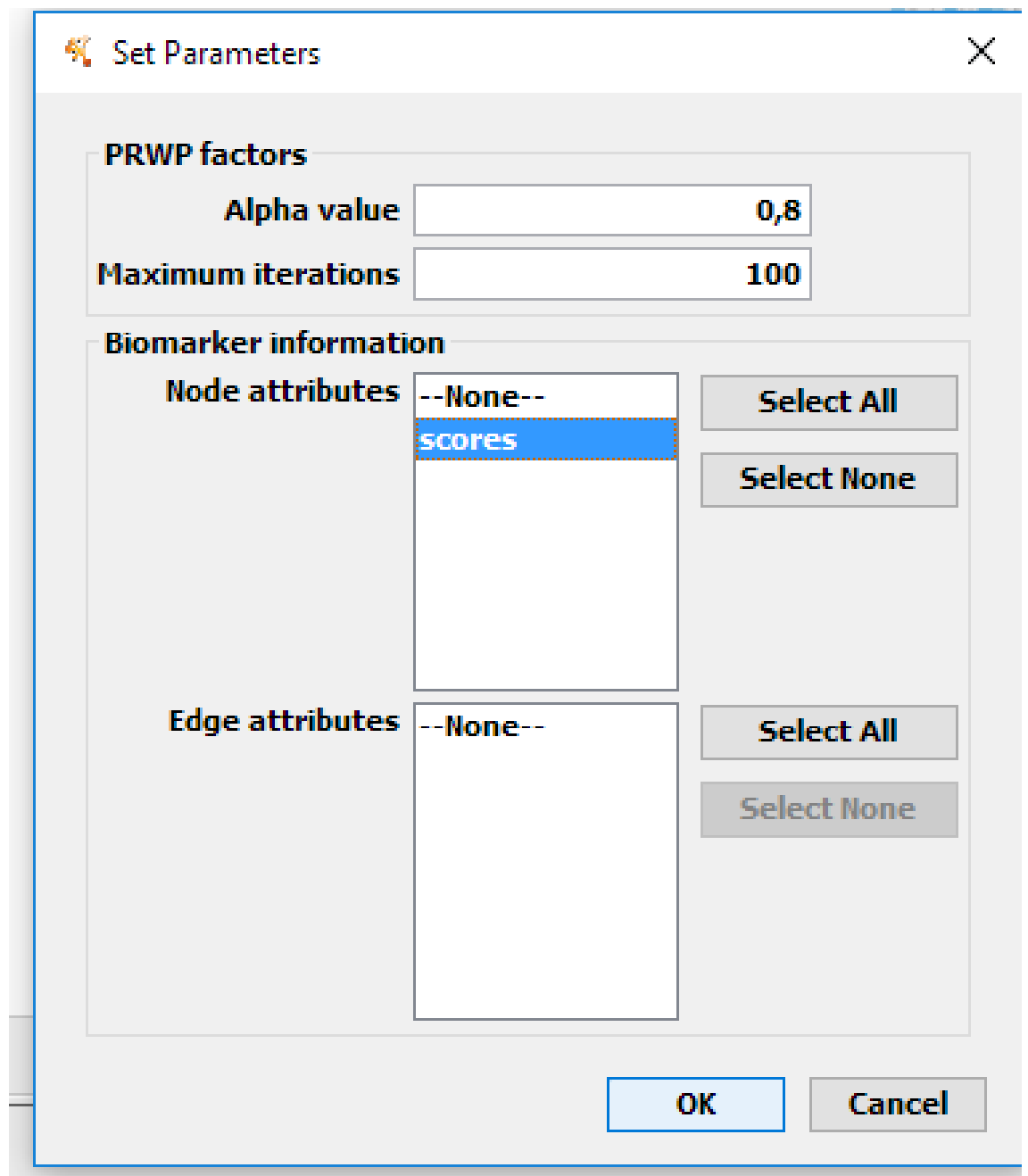
This is the view after clusterMaker2 has run the *MCL cluster* method on the current network. As seen on the left side menu, a new network has been listed. This network only appears if the user sets the "Create new clustered network"-option in the clustering parameter dialog that appears after choosing a clustering algorithm from the clusterMaker2 menu.
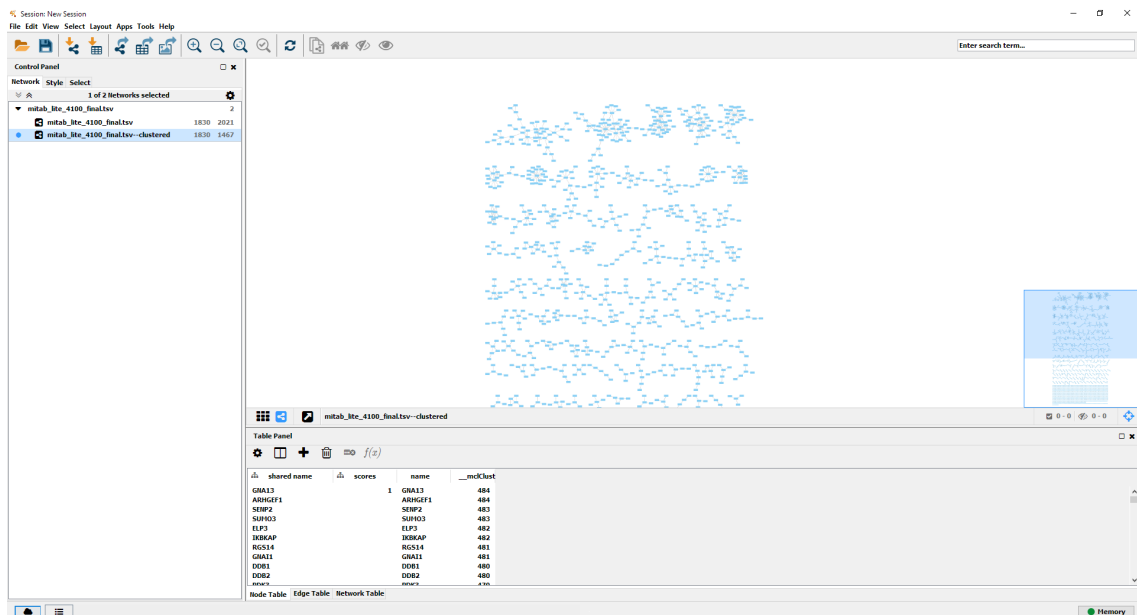
Here the cluster ranking algorithm menu is shown.
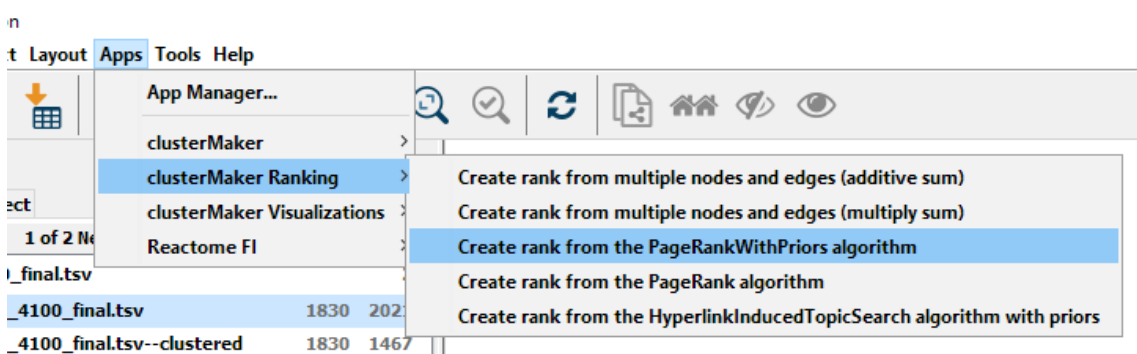
The PRWP was chosen as the ranking algorithm. Node and edge attributes can be combined by the user in any way.

If the user is interested in visualizing the ranks, it is possible to click the "Show results from ranking clusters" option in the visualization menu. No menu will appear after that, but rather will Cytoscape open a loading dialog similar to when clustering and ranking algorithms has been tasked to start.
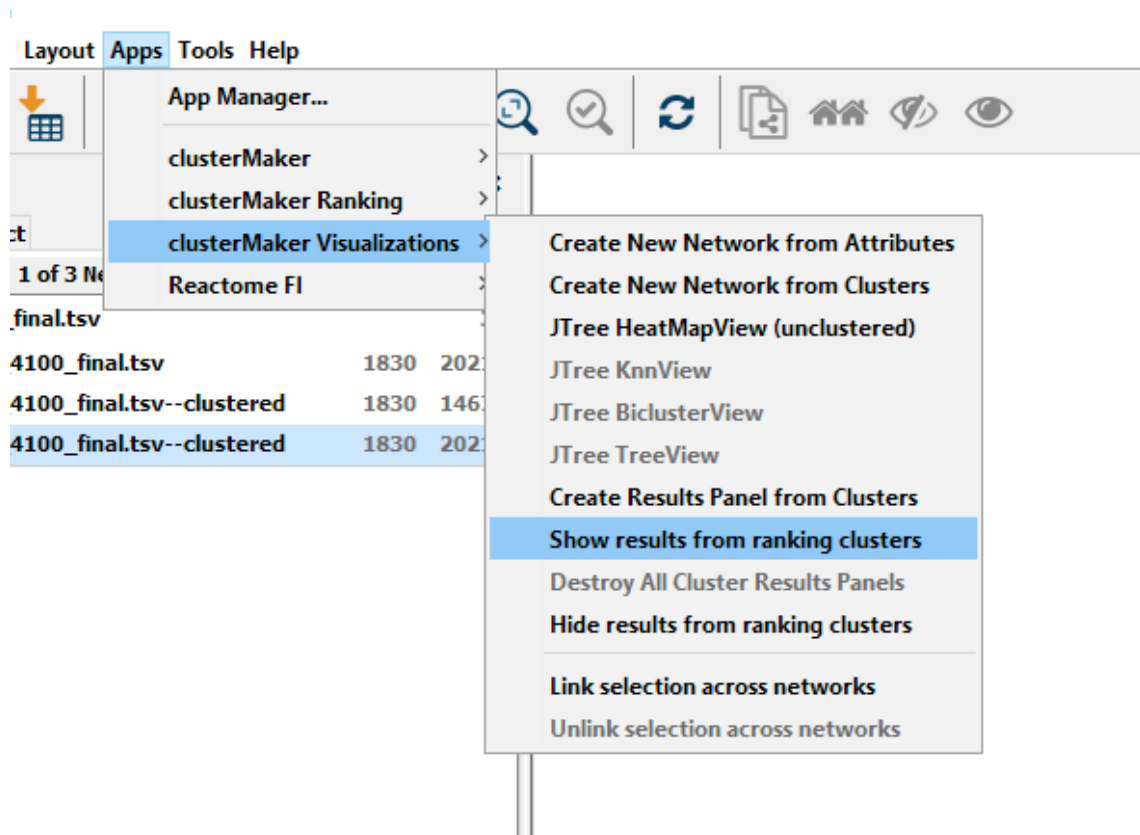
Here is the results panel displaying the ranked clusters from top to bottom, descending scores. The title for each results panel is formatted as in the example.

```
[<clustering algorithm>]{<ranking algorithm>}(<network name>)
```

So with MCL clustering, PRWP ranking and the "mitab_lite_4100_final.tsv–clustered" network, this is what the title becomes:

```
[mcl]{PRWP}(mitab_lite_4100_final.tsv--clustered)
```

As seen in the title. The coloring has been discussed earlier.
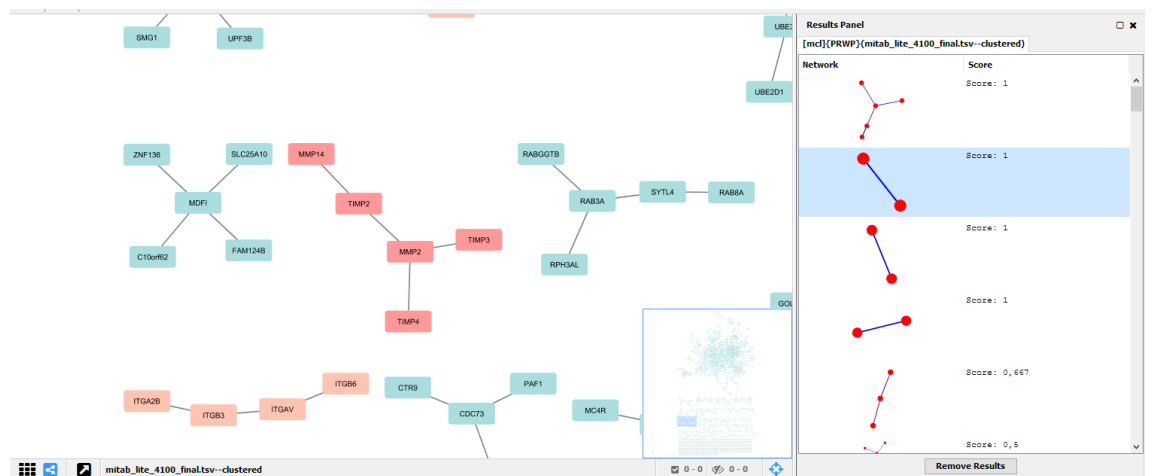
Here is an example of how the clusters change color when selected from the results panel menu.
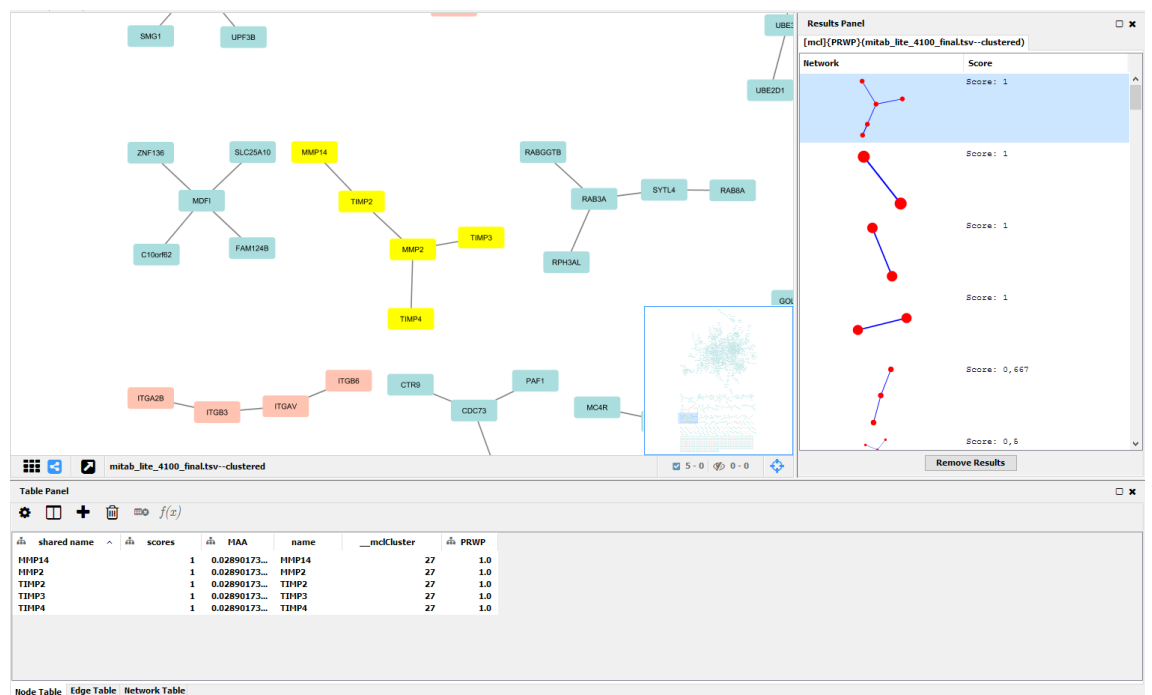
9.3 Design

Figure 9.1: Ranklust ranking algorithm relations
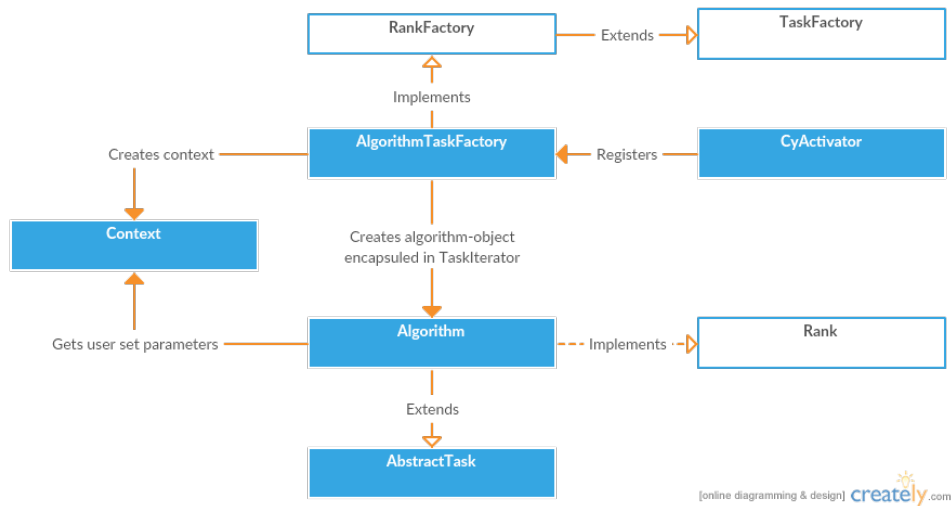
Figure 9.2: Ranklust ranking panel relations
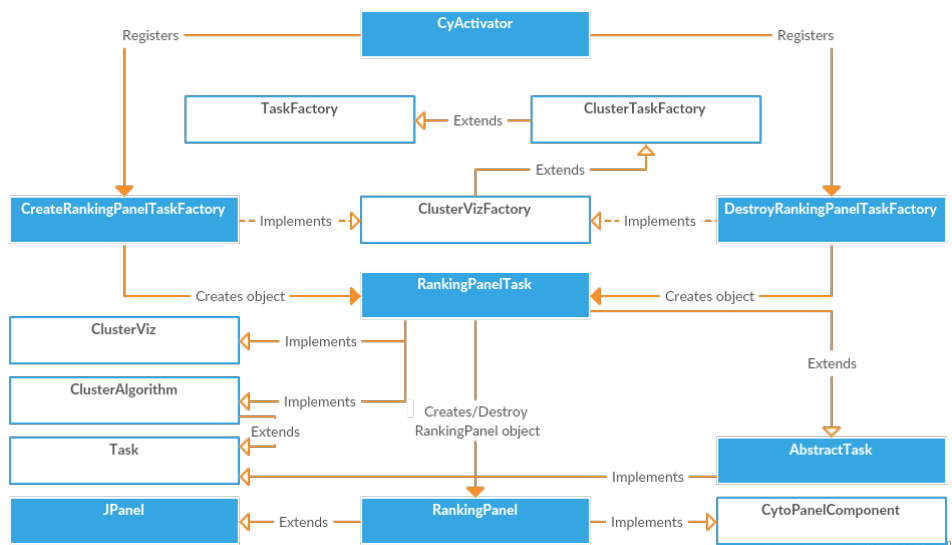
Chapter 10

Graph analysis

PPI network from iRefWeb. downloaded with these settings: irefweb said 109276 interactions after hgnc it was 43706 (perturbation: 60% from iref) after clustering it was 13183 (perturbation: 87,9% from iref - 69,8% from hgnc)

10.1 Creating a network

Converting from proteins to genes through HGNC data resulten in a 60% data network edge perturbation. From 100k ish links to 40k ish.

10.1.1 Creating connections

Creating the connections was achieved through the interactions data from iRefWeb.

10.1.2 Adding weights

Adding the weights was done through the golden standard.

10.2 Ranking results

Talking about MCL results

Inflation	Clusters	Avg. cluster size	Max. cluster size	Min. cluster size	Mo
1.6	1068	8.88	968	2	
1.8	1400	6.60	660	2	
2.0	1599	5.68	405	2	
2.5	2053	4.20	179	2	
3.0	2210	3.75	122	2	

Table 10.1: MCL clustering parameter and statistic results

The modularity of the clustered networks gives an indicator of how well the process of creating the clusters went. Modularity is given as a score from 0 to 1. A score closer to 1 is more preferrable, as this indicates that the clusters created have a good degree of separation to the other clusters in the network. The preferred score to end up with would be around 0.8, but in this network there has been a good amount of perturbation through the protein-to-gene process. Modularity is not the only indicator of how well a network was clustered, hence the choice of not setting the inflation value in MCL to 1.6, but rather 1.8. When a lower inflation value is set, MCL does not separate edges between nodes as vigorously and as a direct cause, inflation will go up. Taking the other attributes in the table 10.1 into consideration, 1.8 seemed like the best inflation value. An inflation value of 1.8 has also been proved to be good for large high-throughput constructed protein-protein networks with a large amount of alterations[30].

The amount of iterations used for MCL ended up being 200. It started out at 1000, but the results converged somewhere between 170 and 200 iterations, so it was decreased form 1000 to 200 to speed up the time used in the pipeline.

10.3   Cross-validation

Executing a cross-validation on the iRefWeb with PRWP and MAA rankings had two purposes. The first being to prove the fact that every gene that had its prior score removed by the cross-validation, should be found in the results of the cluster ranking and identified as candidate biomarkers.

The second purpose was to analyze the distribution of the genes with prior scores removed by the cross-validation in terms of how they placed in the cluster ranking. The analysis of this distribution was done by dividing the amount of genes removed by cross-validation in a cluster by the total amount of genes in the same cluster. This operation was repeated for every cluster in the ranking, resulting in a distribution of the average amount of genes removed by cross-validation, that was detected by Ranklust together with post-processing of data, as cancer candidate biomarkers. A distinct descending distribution from high to low ranked clusters would indicate

that most of the genes, with a prior score of their relevance to prostate cancer, was ranked in a way that achieved the goal of Ranklust; ranking clusters in biological network according to network structure and prior knowledge of relevance to diseases.
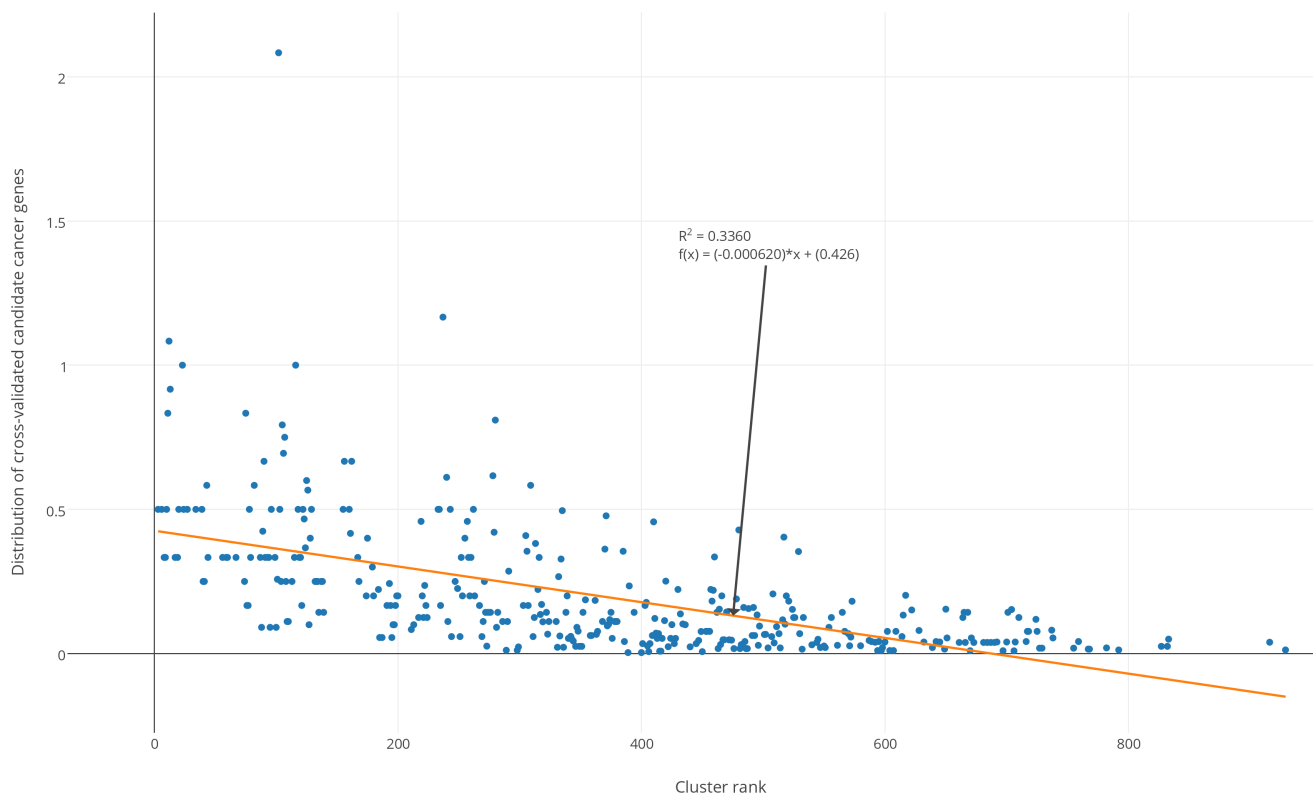
10.3.1 Cross-validation in PRWP

Figure 10.1: Cross-validation distribution in clusters (PRWP)

This plot 10.3.1 is developed from the 10 random cross-validation runs ranked with PRWP. The results from this cross-validation shows that the higher the rank of the cluster, the more candidate cancer genes the cluster had.

10.3.2 Cross-validation in MAA
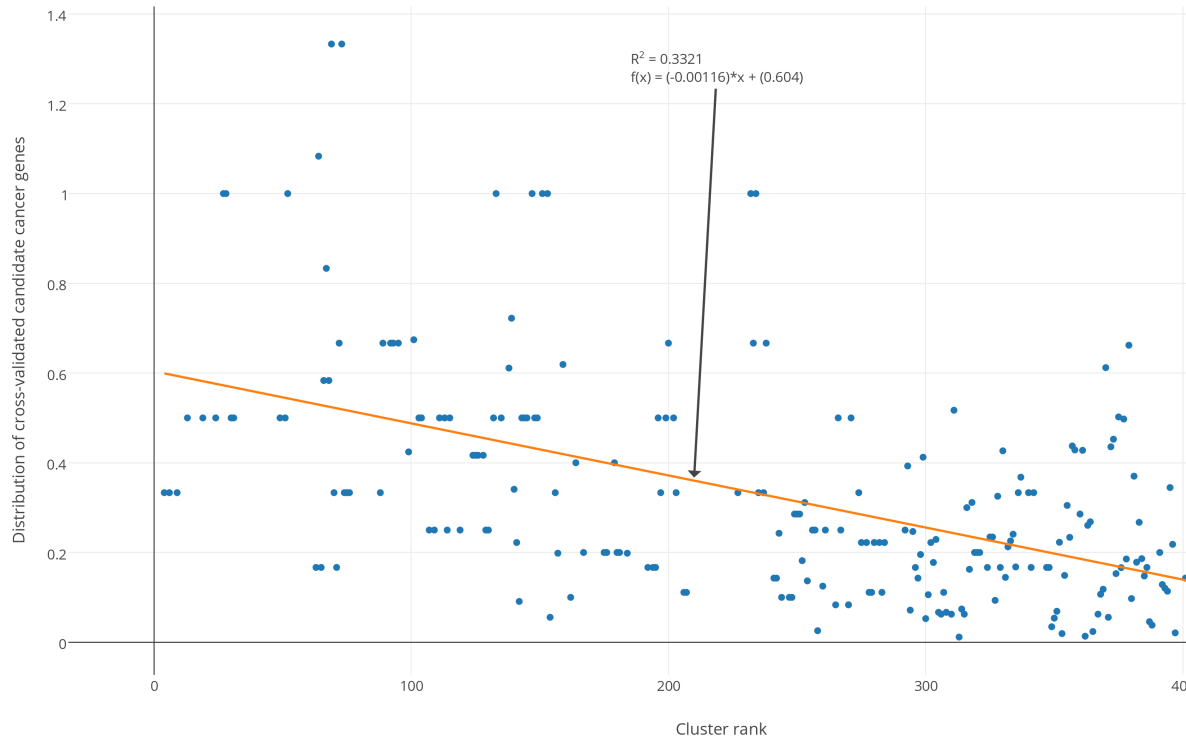
Figure 10.2: Cross-validation distribution in clusters (MAA)

This plot 10.3.2 is developed from the 10 random cross-validatioon runs ranked with MAA. The results from this cross-validation shows that the higher the rank of the cluster, the more candidate cancer genes the cluster had.

10.3.3 PRWP versus MAA

There is clearly a trend in both PRWP and MAA. The difference between them being mainly the amount of clusters found and the coupling of values around the linear regression fit. It is important to point out that this is just a result over the distribution of candidate cancer genes at certain ranks. Where the clusters resides in the rankings may be very different between the PRWP and MAA.

The fact that both of the ranking algorithms managed to get a descending amount of prostate candidate cancer genes in the clusters, which was cross-validated from the same golden standard priors, indicates that a certain similarity of what was mentioned in the previous paragraph. The similarity implicated here is how different the ranking of the clusters are between PRWP and MAA.

The values in PRWP have a tighter coupling, in other words, the distance the coordinates have in the scatter plot deviate less from the fit than the ones in the MAA plot. The difference is not huge, as the coefficient of determination indicates, 0.336 in PRWP against 0.332 in MAA. Both ranking algorithms achieves a descending distribution of cross-validated genes from the topmost to the lowest ranked cluster. But when comparing the cross-validation results between PRWP and MAA, PRWP comes out ahead by a margin in R-squared value.

10.4 Benchmarking Ranklust against biomarker resources

In all of the upcoming plots of benchmarks, blue will represent prostate cancer biomarkers from the golden standard and orange represent prostate candidate cancer biomarkers. The benchmarks are all based on prostate cancer data queried from the DISEASE database[66]. The database has a hierarchy based definition of diseases, so filtering the data on an insensitive-case "prostate cancer"-query with the UNIX grep tool[74] retrieved all of the genes used for the benchmark.

The average z-values in the clusters are split among 2 categories as mentioned above. The z-values of the prostate candidate cancer genes in a cluster is calculated from the average of all of them. The same procedure was done for the z-values of the prostate cancer genes.

10.4.1 Text mined and scored with z-values

These next z-value scored plots represents text mined gene results from the DISEASE database.
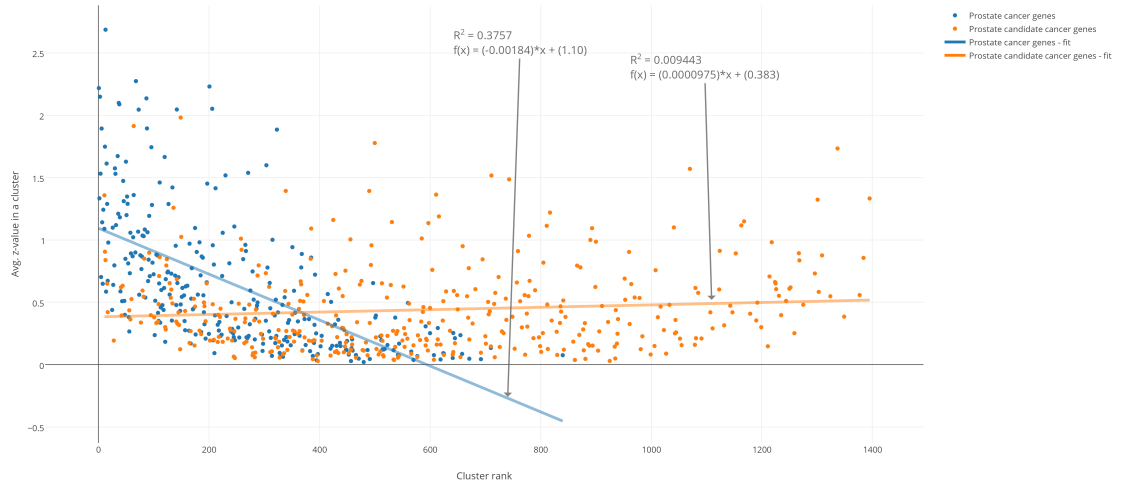
Figure 10.3: Average distribution of z-scores in clusters ranked by PRWP.

For PRWP, the prostate cancer candidate genes is descending in z-values from the topmost ranked cluster to the lowest, which is contributing to showing PRWPs suitability for ranking clusters as candidate biomarkers.

The prostate candidate cancer biomarkers are ascending in z-value from the topmost to the lowest ranked cluster. High z-values would contribute to the fact that ranklust has found actual prostate candidate cancer biomarkers. However, a low z-values does not contradict it. The only fact to deduce from low z-values is that they have not been examined to the degree that they are not mentioned in papers related to prostate cancer.
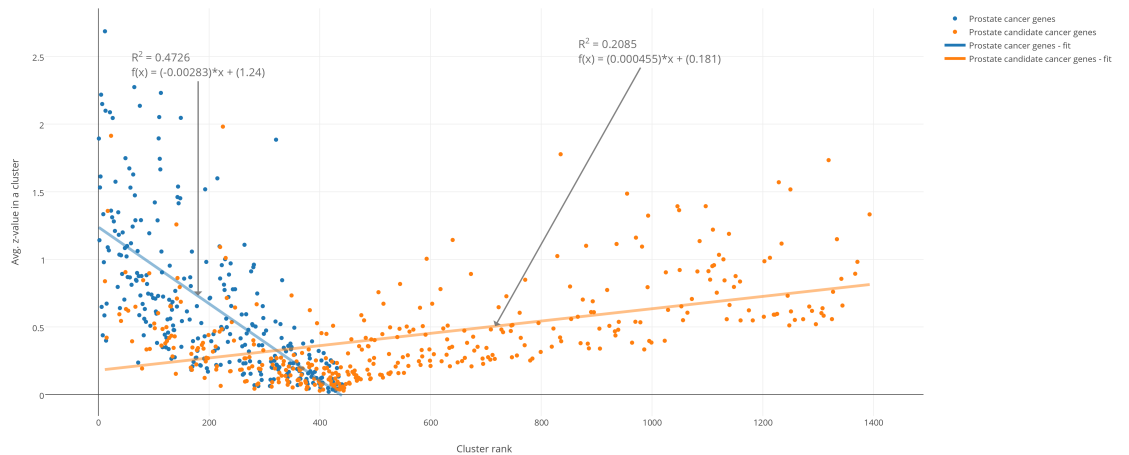

Figure 10.4: Average distribution of z-scores in clusters ranked by MAA.

For MAA, the prostate cancer genes have the same distinct descension in z-values from the topmost to the lowest cluster ranks. The difference from PRWP to MAA being that MAA has a more distinct ascending linear regresstion fit for the z-values in the prostate candidate cancer genes.

This demonstrates the main difference between PRWP and MAA. PRWP takes network structure into comparison as well as the prior scores. MAA focuses purely on the prior scores, and so the network structure of protein complexes are being completely ignored.

10.4.2 Knowledge curated distribution of genes

This data had no score except for a confidence score in the DISEASE database. Due to the low grade of distinctiveness between the different genes in the database, the average amount of genes in a cluster would receive a knowledge score based on if the gene occurs in the knowledge curated part of the DISEASE database or not. So the knowledge curated data is only based on occurence, and not a specific value, in contrast to the text mined and experimentally mined genes in the database.

Another trait the knowledge curated data posssesses is the amount entries in the DISEASE database that has. Text mined data can be seen as the high-throughput technology of retrieving relevant data from papers, while the knowledge curated data is manually curated by researchers. This is why the amount of entries for knowledge curated data is so low when compared to text mined data.
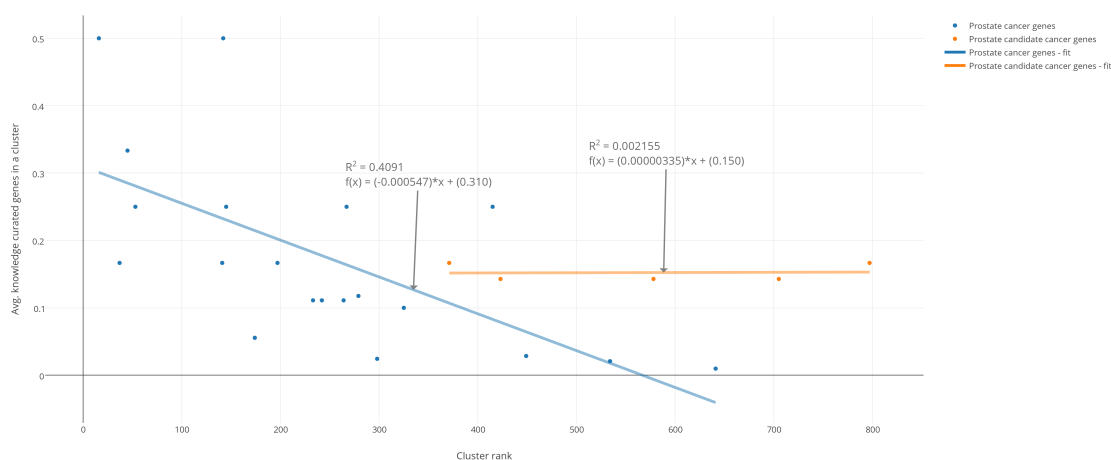
Figure 10.5: Average distribution of curated knowledge mined genes in clusters ranked by PRWP.

– PRWP knowledge curated data discussion –
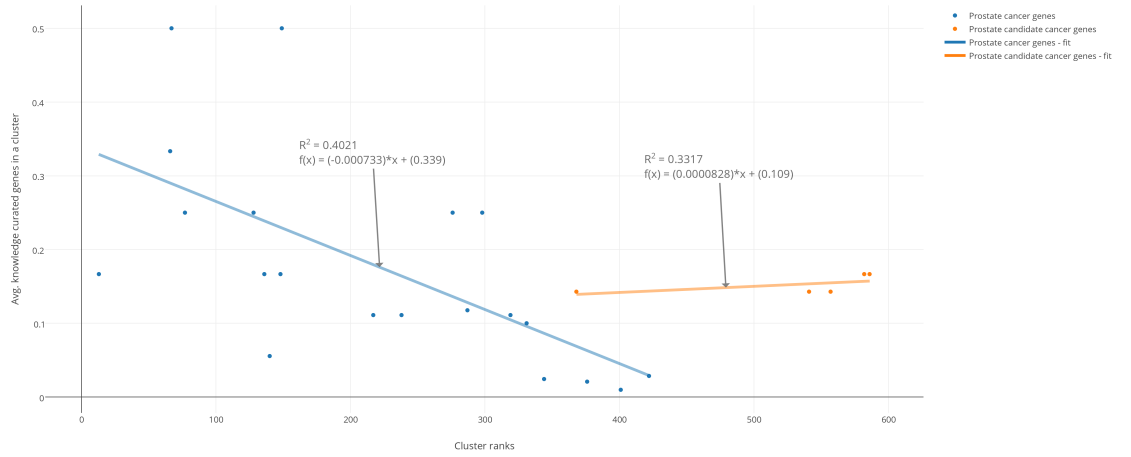
Figure 10.6: Average distribution of curated knowledge mined genes in clusters ranked by MAA.

– MAA knowledge curated data discussion –

10.4.3 Experimental mined genes distribution of p-values in genes
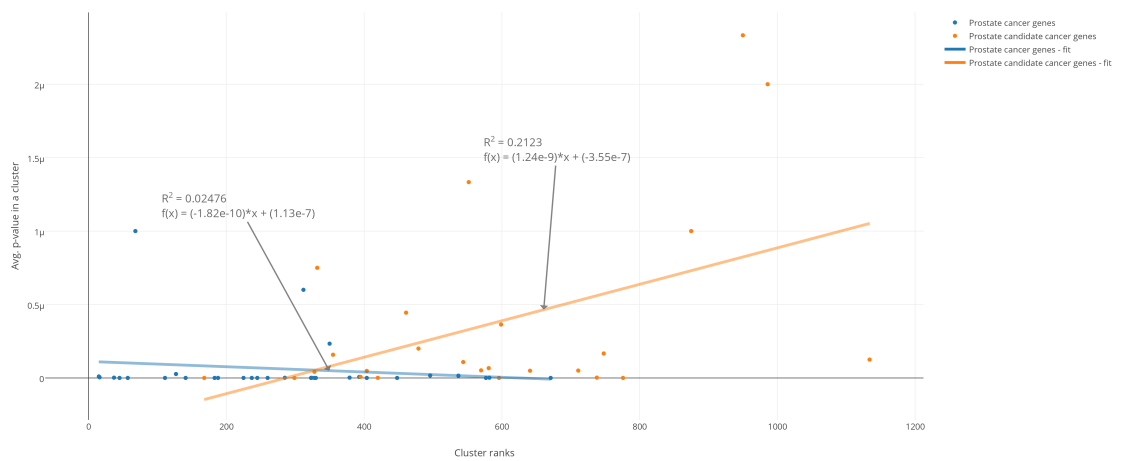
Figure 10.7: Average distribution of p-values in clusters ranked by PRWP.

– PRWP experimentally mined data discussion –
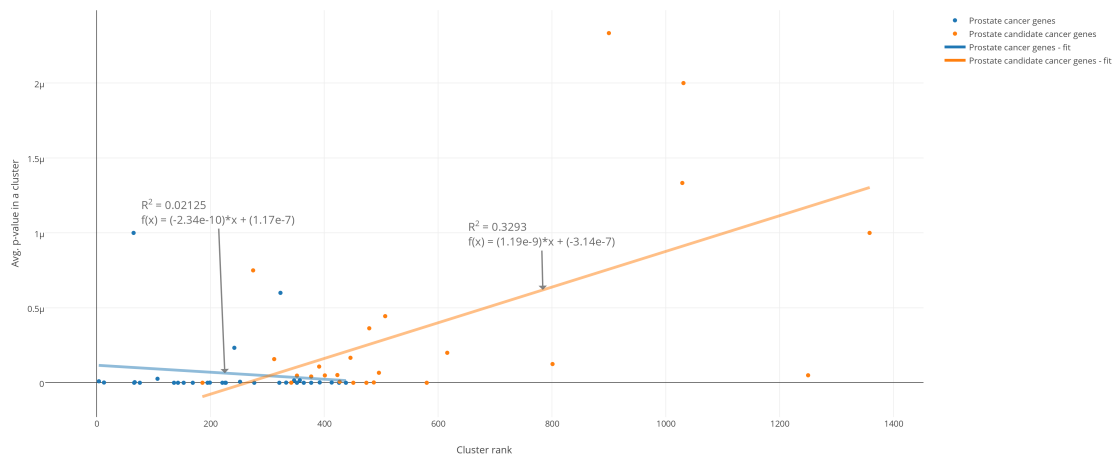
Figure 10.8: Average distribution of p-values in clusters ranked by MAA.

– MAA experimentally mined data discussion –

10.5 Comparison to known biomarkers

– Tests from Movember data etc. –

10.6 Identification of possible cluster biomarkers

– Final list with cluster biomarkers –

Part IV

Discussion and conclusion

Chapter 11

Discussion

11.1 Network handling

If the whole network could receive a change in a single aspect that would make for a better network to rank clusters in, it would be the direction of the edges. The ranking algorithms used can all come to useful results with undirected edges, but directed edges take better advantage of how PR, PRWP, HITS are meant to be used.

11.1.1 Clustering

The clusters could to a greater degree have been filtered more strictly. The average cluster size was around 8.8 nodes, the biggest cluster consisted if over 900 nodes, and the smallest ones were 2.0 nodes in size. Removing the smaller clusters with only 2 genes has been done by other researchers because of the low likeliness of a protein complex with only 2 genes to have a significant impact on disease status. The biggest cluster might also be removed due to its large size when compared to the average. Such a large cluster has a good chance of not being realisticly compartmentalized into a protein complex, and can skew the results in either the direction of false positives or false negatives.

11.1.2 Ranking

Ranking the cluster-created network could have been done instead of or complimentary to the network which only received cluster attributes as a sign of clustering, and had no perturbation of the edges as a result of clustering.

Adjusting the alpha parameter to higher values as 0.8 instead of 0.3 would also give interesting information. Had the cross-validation with PRWP been done with 0.8, and resulted in a linear regression fit 10.3.1 that would have had a less degree of descension of candidate cancer genes throughout the cluster ranks, it would be proof of the network structure having a bigger effect on the rankings than the priors scores.

11.2 Future work

11.2.1 General improvements

We have used an undirected network, which is not the preferred type to use with PageRank, but it works. An idea could be to use KEGG pathways[53], which has the option of downloading directed networks of several types. STRING[14] also has some directed network information, together with weights. Because of STRINGs size, utilizing a Neo4J database to perform the ranking algorithms could be a better option than directly in Cytoscape.

In the future, maybe a database with complete protein complexes could exclude the need for clustering, and provide different ways of ranking them based on query criterias.

Other ranking algorithms than PageRank could also be used. There are numerous variations based off of this well known algorithm, for example NetRank and GeneRank[57, 78]. Not all of these PageRank variants are intended to be used with PPI networks, but they can in many cases be modified to fit this specific purpose.

In Ranklust, the average score in the nodes becomes the cluster score. PageRank was used to combine prior knowledge of cancer genes and network structure to identify cancer candidate genes and clusters. Taking the structure of the network in higher consideration and maybe consider the distance between the nodes to have an effect on the result could be a contribution to the PageRank algorithm. Also, going the other way and increase the significance of the prior scores added to the nodes in the network could help to get a better segregated result of which clusters are related to cancer and not. For example, PRWP used an alpha value of 0.3 because of earlier experiments in ranking biological networks with PageRank had good results with it. What if the alpha value was set to 0.8 and give the prior scores in the node a higher degree of bias?

11.2.2 Minor features to complement Ranklust/cluster-Maker2

A compiled list of all considered features and tweaks in the Ranklust contribution to clusterMaker2

- User can specify what direction the ranking algorithm should consider for the graph

11.2.3 Ranking through Neo4J

ClusterMaker2 does every calculation within Cytoscape. There exists another way of doing large and complex calculations, especially when it comes to algorithms that focus on edge information in the network. CyNeo4j[73] is a Cytoscape App that can realise this idea. It supports import/export of data with a Neo4J[49] database. The original CyNeo4J Github-repository does not support user authentication at the moment, but during the discussion about what ranking algorithm should be used, Neo4J was an alternative. It resulted in a git fork[46] of the CyNeo4J repository and simple user/password authentication was added.

11.2.4 Data communication

Data communication between the Neo4J database server and the Cytoscape instance is done through the CyNeo4J app to Cytoscape [73]. CyNeo4J is a Cytoscape app that was developed during the Google Summer Code 2014 arrangement. It supports connecting to a Neo4J instance, as well as syncing data up and down from and to the database server. One thing it did not support was authentication on Neo4J servers. Not having authentication is a serious problem, so we implemented a simple way of getting access to the database server by providing a possibility to insert username and password at the same time the user has to provide a URL to the Neo4J database server instance. Implementation-wise, this only required an extra header to be included in each http request going to the password protected Neo4J database server instance. Every request used the static **Request** class to Get/Post/Put HTTP requests to the Neo4J database server instance. Except for creating the Auth64 encoded information, the refactor looked something along these lines in all of the files.

Before:

```
Request req = Request.Post(url)
        .bodyString(call.getPayload(), ContentType.
    APPLICATION_JSON);
```

After:

```
1  Request req = Request.Post(url)
2          .addHeader("Authorization", auth64EncodedInfo)
3          .bodyString(call.getPayload(), ContentType.
     APPLICATION_JSON);
```

Synchronization time between Neo4J and Cytoscape through the CyNeo4J app is a huge timesink. As of now, the time it takes to populate an empty Cytoscape network with the gene information from STRING is about 2 hours, though on a slow laptop. This could be shortened by exporting the Neo4J data with GraphML and into Cytoscape. Because after the initial data is inside Cytoscape, updates to the Neo4J instance goes much faster.

The CyNeo4J also uses a legacy HTTP library to get information from the Neo4J database [24]. It is possible that the performance increases with the new library [25]. The new library supports creating transactions, which implicit gives support for rollbacks in case something goes wrong with the query.

A future improvement to the CyNeo4J app could be to change the communication between Neo4J and Cytoscape to be done in GraphML and not Cypher. This is because through this whole process of importing and exporting data from and to Neo4J, GraphML has shown itself to be a superior format over Cypher. Though, to this day, direct queries to a running Neo4J instance has to be done in Cypher, and is not possible in GraphML.

Chapter 12

Conclusion

Glossary

DDPC Dragon Database of Genes associated with Prostate Cancer. 37

DISEASE DISEASE database used for retrieving z-values, p-values and manually curated disease-gene associations. 61, 63

golden standard The prior scoring standard evolved from the DisGeNET and Dragon Database of Genes associated with Prostate Cancer. 37, 40, 61

HITS The Hyperlink-Induced Topic Search method described to score clusters. 24, 27, 29, 30, 33, 49, 69

MAA The Multiple Attribute Additive method described to score clusters. 24, 27–29, 33, 37–40, 49, 60, 61, 63

MAM The Multiple Attribute Multiplicative method described to score clusters. 24, 27–29, 33, 49

Maven The Java build tool used to compile and build clusterMaker2 and Ranklust. 21

MCL Markov Cluster algorithm used in Cytoscape to cluster the networks. 11, 57, 58

pipeline The time used from start to end when analyzing the network in Cytoscape. 58

PR The PageRank method described to score clusters. 24, 27–30, 38, 49, 69

PRWP The PageRank With Priors method described to score clusters. 24, 28–30, 37, 38, 40, 49, 51, 53, 60–63, 69, 70

R-squared The coefficient of determination when talking about the grade of fitness of a linear regression. 61

Bibliography

[1]   URL: http://apps.cytoscape.org/apps/clustermaker2 (visited on 22/05/2015).

[2]   URL: https://github.com/RBVI/clusterMaker2 (visited on 08/07/2016).

[3]   URL: http://www.cancer.gov/cancertopics/types/prostate/psa-fact-sheet (visited on 11/05/2015).

[4]   URL: http://www.illumina.com/technology/next-generation-sequencing.html (visited on 11/05/2015).

[5]   URL: http://biopython.org/wiki/Main_Page (visited on 20/05/2015).

[6]   URL: http://scikit-learn.org/stable/ (visited on 08/07/2016).

[7]   URL: http://wiki.cytoscape.org/Cytoscape_3/AppDeveloper (visited on 08/05/2015).

[8]   URL: http://www.javaworld.com/article/2878952/java-platform/modularity-in-java-9.html (visited on 20/05/2015).

[9]   URL: http://www.techopedia.com/definition/18822/design-pattern (visited on 20/05/2015).

[10] URL: http://www.amazon.com/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882 (visited on 20/05/2015).

[11] URL: http://jung.sourceforge.net/ (visited on 07/07/2016).

[12] URL: http://irefindex.org/wiki/index.php?title=iRefIndex (visited on 07/05/2015).

[13] URL: http://www.genemania.org/ (visited on 07/05/2015).

[14] URL: http://string-db.org/ (visited on 07/05/2015).

[15] URL: http://apps.cytoscape.org/apps/irefscape (visited on 07/05/2015).

[16] URL: http://apps.cytoscape.org/apps/genemania (visited on 07/05/2015).

[17] URL: http://wodaklab.org/iRefWeb/search/index (visited on 08/07/2016).

[18] URL: http://www.genenames.org (visited on 23/02/2016).

[19]   URL: http://graphml.graphdrawing.org/ (visited on 19/02/2016).

[20]   URL: http://hex.colorrrs.com/ (visited on 09/06/2016).

[21]   URL: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf (visited on 05/07/2016).

[22]   URL: https://www.cs.cornell.edu/home/kleinber/auth.pdf (visited on 05/07/2016).

[23]   URL: http://www.math.cornell.edu/~mec/Winter2009/RalucaRemus/ Lecture4/lecture4.html (visited on 05/07/2016).

[24]   URL: http://neo4j.com/docs/stable/rest-api-cypher.html (visited on 17/03/2016).

[25]   URL: http://neo4j.com/docs/stable/rest-api-transactional.html (visited on 17/03/2016).

[26]   Apache. *Apache Felix Maven Bundle Plugin (BND)*. URL: http:// felix.apache.org/documentation/subprojects/apache-felix-maven- bundle-plugin-bnd.html (visited on 25/07/2016).

[27]   Apache. *Apache Maven Project*. URL: https://maven.apache.org/ (visited on 09/08/2016).

[28]   Apache. *Apache Spark*. URL: http://spark.apache.org/ (visited on 24/07/2016).

[29]   David Austin. *How Google Finds Your Needle in the Web's Haystack*. URL: http://www.ams.org/samplings/feature-column/fcarc- pagerank (visited on 07/07/2016).

[30]   Sylvain Brohée and Jacques van Helden. 'Evaluation of cluster algorithms for protein-protein interaction networks'. In: *BMC Informatics* (2006). DOI: 10.1186/1471-2106-7-488.

[31]   Anne-Ruxandra Carvunis and Trey Ideker. 'Siri of the Cell: What biology Could Learn from the iPhone'. In: *CellPress* 157 (Apr. 2014).

[32]   Hao Chen, Zhitu Zhu, Yichun Zhu, Yunging Mei and Yunfeng Cheng. 'Pathway mapping and development of disease-specific biomarkers: protein-based network biomarkers'. In: *Journal of Cellular and Molecular Medicine* (Feb. 2015), pp. 297–314.

[33]   Jing Chen, Bruce J. Aronow and Anil G. Jegga. 'Disease candidate gene identification and prioritization using protein interaction networks'. In: *BMC Bioinformatics* 10.1 (2009), pp. 1–14. ISSN: 1471- 2105. DOI: 10.1186/1471-2105-10-73. URL: http://dx.doi.org/10. 1186/1471-2105-10-73.

[34]   Pau Creixell, Jüri Reimand, Syed Haider, Guanming Wu, Tatsuhiro Shibata, Miguel Vazquez, Ville Mustonen, Abel Gonzalez-Perez, John Pearson, Chris Sander, Benjamin J Raphael, Debora S Marks, B F Francis Oulette, Alfonso Valencia, Gary D Bader, Paul C Boutros, Joshua M Stuart, Rune Linding, Nuria Lopez-Bigas and Lincoln D Stein. 'Pathway and network analysis of cancer genomes'. In: *Nature Methods* 12.7 (July 2015).

[35] Numpy Developers. *Numerical Python*. URL: http://www.numpy.org/ (visited on 24/07/2016).

[36] Stijn Marinus van Dongen. 'Graph clustering by flow simulation'. PhD thesis. Utrecth of University, 2000. URL: http://micans.org/mcl/.

[37] MD Dr Ananya Mandal. URL: http://www.news-medical.net/health/ What-is-a-Biomarker.aspx (visited on 12/05/2015).

[38] Justin M. Drake, Evan O. Paull, Nicholas A. Graham, Thomas G. Graeber, Owen N. Witte and Joshua M. Stuart. 'Phosphoproteome Integration Reveals PatientSpecific Networks in Prostate Cancer'. In: *Cell* 166 (Aug. 2016), pp. 1–14.

[39] Michael F.Berger et al. 'The genomic complexity of primary human prostate cancer'. In: *Nature* 1 (2011).

[40] Brendan J. Frey and Delbert Dueck. *Clustering by Passing Messages Between Data Points*. 16th Feb. 2007. URL: http://www.psi.toronto. edu/affinitypropagation/FreyDueckScience07.pdf.

[41] Biomarkers Definitions Working Group. 'Biomarkers and surrogate endpoints: Preferred definitions and conceptual framework'. In: *Clinical Pharmacology & Therapeutics* 69.3 (18th Mar. 2001), pp. 89–95.

[42] Alexander Haesea, Alexandre de la Tailleb, Hendrik van Poppelc, Michael Marbergerd, Arnulf Stenzle, Peter F.A. Muldersf, Hartwig Hulandg, Clément-Claude Abboub, Mesut Remzid, Martina Tinzld, Susan Feyerabende, Alexander B. Stillebroerf, Martijn P.M.Q. van Gilsf and Jack A. Schalkenf. 'Clinical Utility of the PCA3 Urine Assay in European Men Scheduled for Repeat Biopsy'. In: *European Urology* 54 (2008).

[43] David Heckerman. *A Tutorial on Learning With Bayesian Networks*. MSR-TR-95-06. Microsoft Research, Mar. 1995. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=69588.

[44] https://commons.wikimedia.org/wiki/User:Grolltech. *Testing the colors of a web chart, (center), to ensure that no information is lost to the various forms of color-blindness*. Grolltech is the username for the person responsible. URL: https://en.wikipedia.org/wiki/Color_ blindness#/media/File:Safe_Chart_Colors-F99-FEC-ADD.jpg (visited on 24/07/2016).

[45] Tianxio Huan, Xiaogang Wu, Zengliang Bai and Jake Y. Chen. 'Seed-weighted Random Walks Ranking Method and Its application to Leukemia Cancer Biomarker Prioritizations'. In: (2009).

[46] GitHub Inc. *Fork A Repo*. URL: https://help.github.com/articles/fork-a-repo/ (visited on 25/07/2016).

[47] GitHub Inc. *Using pull requests*. URL: https://help.github.com/ articles/using-pull-requests/ (visited on 24/07/2016).

[48] Neo Technology Inc. *Intro to Cypher*. URL: https://neo4j.com/ developer/cypher-query-language/ (visited on 24/07/2016).

[49] Neo Technology Inc. *Neo4J*. URL: https://neo4j.com/ (visited on 24/07/2016).

[50] Wellcome Trust Sanger Institute. *COSMIC - Catalogue of somatic mutations in cancer*. URL: http://cancer.sanger.ac.uk/cosmic/download (visited on 07/08/2016).

[51] Gbor Ivn and Vince Grolmusz. 'When the web meets the cell: Using Personalized PageRank for Analyzing Protein Interaction Networks'. In: *Bioinformatics Advance Access* (12th Dec. 2010).

[52] P. Jiang and M. Singh. 'SPICi: a fast clustering algorithm for large biological networks'. In: *Bioinformatics* 26.8 (15th Apr. 2010), pp. 1105–1111. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btq078. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btq078 (visited on 22/05/2015).

[53] Kanehisa Laboratories. *KEGG PATHWAY Database*. URL: http://www.genome.jp/kegg/pathway.html (visited on 26/07/2016).

[54] Rebecca J. Leary, Isaac Kinde, Frank Diehl, Kerstin Schmidt, Chris Clouser, Cisilya Duncan, Alena Antipova, Clarence Lee, Kevin McKernan, Francisco M. De La Vega, Kenneth W. Kinzler, Bert Vogelstein, Luis A. Diaz Jr. and Victor E. Velculesc. 'Development of Personalized Tumor Biomarkers Using Massively Parallel Sequencing'. In: *Science Translational Medicine* 2 (2010).

[55] M Magungo, M Kaur, SK Kwofie, A Radovanovic, U Schaefer, S Schmeier, E Oppon, A Christoffels and VB Bajic. 'DDPC: Dragon Database of Genes associated with Prostate Cancer'. In: *Oxford Journals - Nucleic Acids Research* (Sept. 2010).

[56] matthiaskoenig. *Fixed changed POM dependencies*. Author is a GitHub.com username. URL: https://github.com/RBVI/clusterMaker2/pull/3/commits/42d910b51cb6cae0511379346b8e0b8ac85a124b (visited on 25/07/2016).

[57] Julie L. Morrison, Rainer Breitling, Desmond J. Higham and David R. Gilbert. 'GeneRank: Using search engine technology for the analysis of microarray experiments'. In: *BMC Bioinformatics* 6.1 (2005), pp. 1–14. ISSN: 1471-2105. DOI: 10.1186/1471-2105-6-233. URL: http://dx.doi.org/10.1186/1471-2105-6-233.

[58] NumFOCUS. *Python Data Analysis Libarary*. URL: http://pandas.pydata.org/ (visited on 24/07/2016).

[59] OpenJDK. *Project Jigsaw*. URL: http://openjdk.java.net/projects/jigsaw/ (visited on 24/07/2016).

[60] Oracle. *Using the keyword super*. URL: https://docs.oracle.com/javase/tutorial/java/IandI/super.html (visited on 24/07/2016).

[61] Oracle. *What is JAR?* URL: https://docs.oracle.com/javase/8/docs/technotes/guides/jar/jarGuide.html (visited on 09/08/2016).

[62] Orcale. *Package javax.swing*. URL: https://docs.oracle.com/javase/8 / docs / api / javax / swing / package - summary . html (visited on 24/07/2016).

[63] K. Pearson. 'The Problem of the Random Walk'. In: *Nature* 72 (Aug. 1905), p. 342. DOI: 10.1038/072342a0.

[64] Kathryn L. Penney et al. 'mRNA Expression Signature of Gleason Grade Predicts Lethal Prostate Cancer'. In: *Journal of Clinical Oncology* 29.17 (June 2011), pp. 2391–2396.

[65] J. Piñero, N Queralt-Rosinach, À Bravo, A Bauer-Mehren, M Baron, F Sanz and Li Furlong. 'DisGeNET: a discovery platform for the dynamical exploration of human diseases and their genes.' In: *Oxford University Press* (Apr. 2015).

[66] S Pletscher-Frankild, A Pallejà, K Tsafou, JX Binder and LJ Jensen. 'DISEASES: text mining and data integration of disease-gene associations'. In: *Methods* 74 (Mar. 2015), pp. 83–89.

[67] Plotly. *Plotly - Make charts and dashboards online*. URL: https://plot.ly/plot (visited on 09/08/2016).

[68] John R. Prensner, Mark A. Ruin, John T. Wei and Arul M. Chinnayiyan. 'Beyond PSA: The next generation of prostate cancer biomarkers'. In: *Sci Transl Med* 1 (2012).

[69] Guido van Rossum. *Python*. URL: https://www.python.org/ (visited on 24/07/2016).

[70] scootermorris. *Merge in ranking algorithms from Henning-Lund-Hanssen*. URL: https : / / github . com / RBVI / clusterMaker2 / commit/f055347df9af397b4f15866bc35db55b8bb5b357 (visited on 24/07/2016).

[71] iText Software Corp. *IText*. URL: http://developers.itextpdf.com/ (visited on 25/07/2016).

[72] Kyle Strimbu and Jorge A. Tavel. 'What are Biomarkers?' In: *Current Opinion in HIV and AIDS* (Nov. 2010).

[73] Georg Summer. *cyNeo4j - A Cytoscape app to connect to a Neo4J database and execute extensions of the Neo4j database.* Author name was found on the wordpress page of the app https://cyneo4j.wordpress.com/. URL: https://github.com/gsummer/cyNeo4j (visited on 17/03/2016).

[74] Ken Thompson. *grep(1) - Linux man page*. URL: http://linux.die.net/man/1/grep (visited on 10/08/2016).

[75] TimHull. *CytoScape 3 App Cookbook*. URL: http://wiki.cytoscape.org/Cytoscape_3/AppDeveloper/Cytoscape_3_App_Cookbook (visited on 24/07/2016).

[76] tutorialspoint.com. *Design Pattern - Factory Pattern*. URL: http://www.tutorialspoint.com/design_pattern/factory_pattern.htm (visited on 24/07/2016).

[77]   James Vlasblom and Shoshana J Wodak. 'Markov clustering versus affinity propagation for the partitioning of protein interaction graphs'. In: *BMC Bioinformatics* (30th Mar. 2009).

[78]   Christof Winter et al. 'Google Goes Cancer: Improving Outcome Prediction for Cancer Patients By Network-Based Ranking of Marker Genes'. In: *PLoS Comput Biol.* (May 2012).