Ranklust

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

Henning Lund-Hanssen

Ranklust

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

TODO: Better abstract!

- Motivation

- Goal

- Methods

- Conclusion

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 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

List of Figures

List of Tables

Listings

Glossary

DDPC Dragon Database of Genes associated with Prostate Cancer. 35, 36, 57, 70, 76

DISEASE DISEASE database used for retrieving z-values, p-values and manually curated disease-gene associations. 67, 70, 77, 78, 81, 83, 86, 98

golden standard The prior scoring standard evolved from the DisGeNET and Dragon Database of Genes associated with Prostate Cancer. 35, 38, 39, 57, 70, 75, 76

HITS The Hyperlink-Induced Topic Search method described to score clusters. 22, 25, 26, 28, 31, 49, 91

Integrated Development Environment A development environment specialized in developing software products. 20

MAA The Multiple Attribute Additive method described to score clusters. 22, 25–28, 31, 35–38, 49, 58, 61, 67, 70, 73, 75–77, 88, 92, 93, 97, 98

MAM The Multiple Attribute Multiplicative method described to score clusters. 22, 25–28, 31, 49

Maven The Java build tool used to compile and build clusterMaker2 and Ranklust. 19

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. 22, 25, 27, 28, 36, 49, 51, 91

PRWP The PageRank With Priors method described to score clusters. 22, 25, 27, 28, 35, 37, 38, 49, 51, 53, 58, 61, 64, 67, 70, 73, 75–77, 88, 91–93, 97, 98

R-squared The coefficient of determination when talking about the grade of fitness of a linear regression. 70, 73, 92, 93

Part I

Introduction

Chapter 1

Background

1.1 Motivation

TODO: Motivation in general here!

1.2 Biomarkers

A biomarker is a "biological measure of a biological state" [16]. 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: [60]

- Disease disposition
  - What is a patient's risk of developing cancer in the future?
- Screening
  - 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."[22, 66]. 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[66]. 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[66]. 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[66]. 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 [34].

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 [34]. 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[56].

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 [18]. 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 [24]. 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 [60]. 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[13]. 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 technologies

Today, rapid analysis of genes and proteins are for example made available through Next-Generation Sequencing (NGS)[1]. 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 [17, 40].

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 [26]. 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 [9]. 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[10], 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[8, 15]. 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[45]. But there is faster cluster-creating algorithms than those implemented in ClusterMaker2, and an example is the SPICi [37] 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* [12] and sklearn[55], 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 [29].

Formatting and cleaning the data that Ranklust will provide and export out of Cytoscape is done in Python 2.7[63]. 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[47] and numpy[14] will support the formatting and cleaning of data.

4.3 OSGi and design patterns

Developing OSGi software promotes modularization [25] 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[49], 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 [33], but more of a collection of them, and it is the clean code principles by Robert C. Martin [2]. 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[48].

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 [61], GeneMania [77] and STRING [19]. 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 [62] for iRefIndex and GeneMANIA [43] for GeneMania. The drawback with using apps for retrieving data from databases is that they have to be updated. The Cytoscape app of iRefIndex[62] 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[74]. 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[21] 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 because the protein identifiers from iRefWeb was listed in this exact format.

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[73] 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[44] 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[52] 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[51](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[4]. 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
```

The JAR-file that is being produced after the line over is executed in a terminal or through the build tool of an Integrated Development Environment. 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:

```
<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[48] 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[3]. 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

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

Added these dependencies

Listing 6.2: POM-file JUNG changes

```
<dependency>
    <groupId>net.sf.jung</groupId>
    <artifactId>jung-graph-impl</artifactId>
    <version>2.1</version>
</dependency>
<dependency>
    <groupId>net.sf.jung</groupId>
    <artifactId>jung-algorithms</artifactId>
    <version>2.1</version>
</dependency>
<dependency>
    <groupId>net.sf.jung</groupId>
    <artifactId>jung-api</artifactId>
    <version>2.1</version>
</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[5], 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[75], and it can return different types of a class that has the same java superclass[50]. 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 Algorithms for ranking clusters

Each ranking algorithm has atleast three classes, like the clustering algorithms. A context class responsible for handling GUI elements visible to the user and to convey the parameters and attributes the user sets. A TaskFactory class responsible for registering the algorithm with the CyActivator class and to encapsule the class executing the ranking algorithm in a TaskIterator class. The TaskIterator class is required for every task that is to be executed in Ranklust. This is to have control over the sequence that tasks was added to in the Cytoscape environment. The third required class is the algorithm class itself. It has to contain certain methods in order to compile, but the most important one is the run() method, which is the method that is called in order to initialize the execution of this class, and as a result of that, the algorithm itself.

For every ranking algorithm in Ranklust there is some steps that every ranking algorithm has in common:

1. Initialize progress bars displayed to the user

2. Retrieve the clusters through the ClusterUtils.fetchClusters() method
3. Initialize other variables needed
4. Retrieve node and edge attributes from the Context class
5. Insert nodes and edges into a graph (only for PR, PRWP and HITS)
6. Execute graph algorithm in JUNG (only for PR, PRWP and HITS)
7. Collect and summarize values, create average cluster values and normalize the values

6.6 The results panel from ranking clusters

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[27], followed by the RGB value that was calculated using a site that converts from hex to RGB values[7]. *#ADD* $rgb170,221,221$ for a blueish and cold color representing low score. *#FEC* $rgb255,248,204$ represents off-brown color that is a step warmer than the blueish. The warmest color representing the highest scored clusters is *#F99* $rgb255,153,153$. 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[31] made from the Ranklust contribution of this thesis has been accepted[64], 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[76]. 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)[15] clustering algorithm in *clusterMaker2*, without any limitations to the amount of clusters or weighting of the nodes. I prototyped Ranklust with *Affinity Propagation*(AP)[20] 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[76] 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, Multiple Attribute Additive Method (MAA), Multiple Attribute Multiplication Method (MAM), PageRank (PR), PageRank with Priors (PRWP and Hyperlink-Induced Topic Search (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[48] 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 PageRank with Priors (PRWP)

A Random Walks[54] algorithm which used priors, called seed-weighted random walks ranking [28], 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*[48]. PageRank was previously used by Google to rank webpages [53]. PageRank with Priors (PRWP)[36] 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. PageRank with Priors 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 (HITS)

Hyperlink-Induced Topic Search, HITS, an algorithm that is similar to PageRank, was developed around the same time [38][68] 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 both identifying and prioritizing candidate genes in diseases[11]. 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 [6], 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 clusterMaker2, 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"[65] repository location instead of a location from "com.lowagie" because the library was relocated [42].

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.

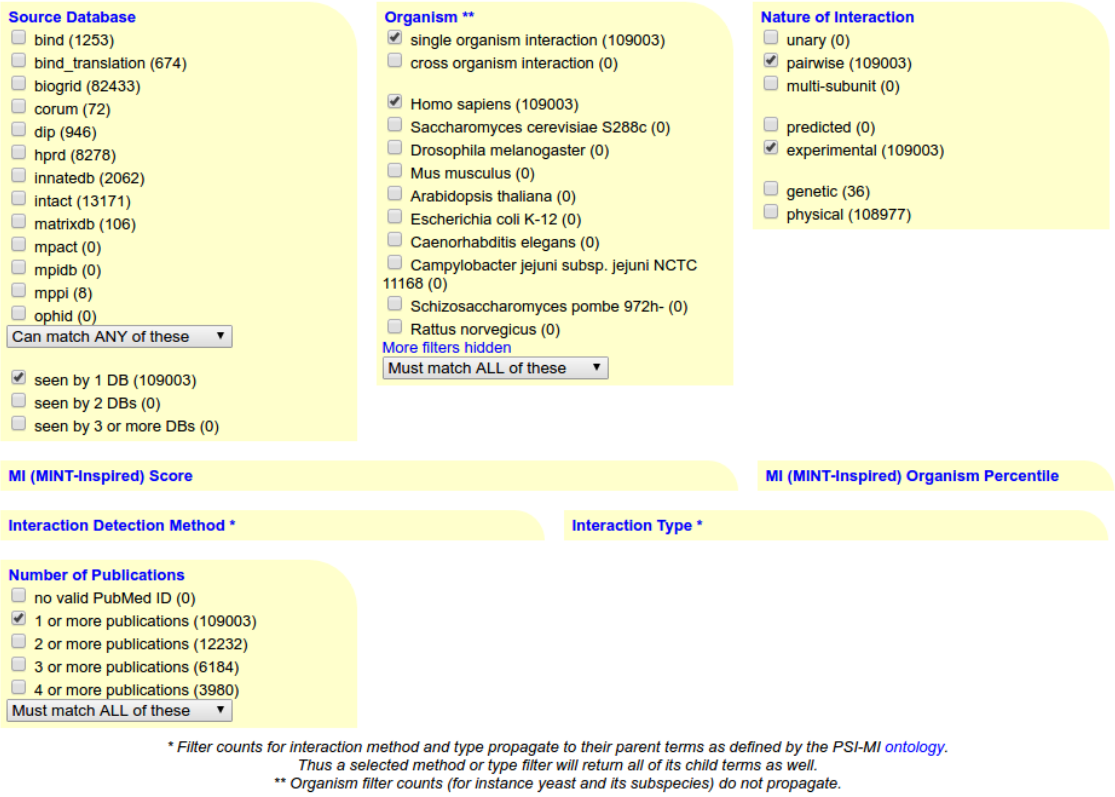
8.1 Before using Ranklust

8.1.1 Network creation

The network was created with protein interaction data from iRefWeb [74]. The query was constructed with the idea of having as large a network possible of human-only interacting proteins, that at a later stage could be converted into genes. The tabs that is not expanded was not changed from the standard settings when creating such a query, which starts off with all of the checkboxes unchecked (8.1). The picture of this query states that there was 109003 iterations. However, this number is subject to change several minutes after querying iRefWeb. This picture of the query was taken before the database search for interaction had finished, which is the reason for displaying 273 interactions less than the actual amount used in the network that Cytoscape ran the algorithms on.

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



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.*

Figure 8.1: iRefWeb network query

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[35] was constructed with interaction information from the iRefWeb network. This network was much smaller in terms of both nodes and edges (interactions). 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[57] and DDPC[41].

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"[57]. 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, and multiple instances of the same gene could have multiple scores for different types of prostate cancer. The duplication of gene entries with scores related to prostate cancer resulted in a solution where the average of all of the prostate cancer scores created a single score for a single gene. 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 DisGeNET scores. The method for converting the scores was to let the DisGeNET 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 DisGeNET 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[47]. 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 test data

Matching the cluster ranks against test data is the last step in terms of creating results that could reveal candidate network biomarkers for prostate cancer. The procedure for matching cluster ranks with test data will consist of inserting the test data genes with the cluster ranks. The test data genes will be a single column list which contains unique genes proven to be prostate cancer biomarkers. Creating a table with an overview of the test genes that match the genes inside the ranked clusters, and whether the golden standard classified these ranked genes as prostate cancer genes or prostate cancer candidate genes, is the final goal of this thesis. Traversing the ranked clusters and ascertain which genes from the test data is inside which clusters and if they are candidate biomarkers or already existing ones will end up producing the final table used to represent the core result in this thesis. From creating and scoring a network, cluster it, ranking it, assert its suitability to rank clusters and present a final list of network candidate biomarkers for prostate cancer.

The test data will come from three different sources. Manually expert curated genes from the movember group, prostate cancer genes from experiments recorded in COSMIC[35], and proven lethal prostate cancer genes from an experiment that tried to reduce the amount of over-treatment of prostate cancer based on screening from PSA[56]. The three data sets will be mentioned as "movember" for first set, "COSMIC" for the second, and "Lethal prostate cancer" or some other context where "lethal" and "prostate" is mentioned together.

8.3 Plot creation

The plots was created with an online tool named Plotly[59]. Throughout the whole process of creating plots, the X-axis represented the cluster ranks, ordered from best to worst scored cluster, left to right, with ascending values representing the ranks. The Y-axis represented in all cases an average. This average was always calculated by dividing the sum of a specific value for each gene, by the total number of genes in 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: (9.1)

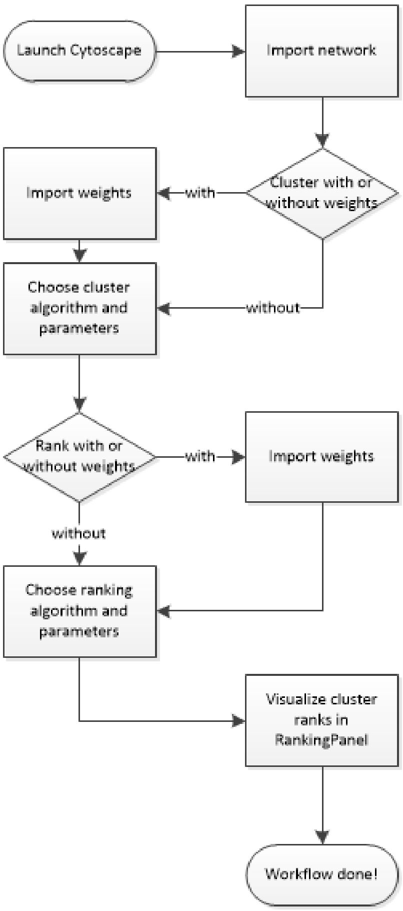
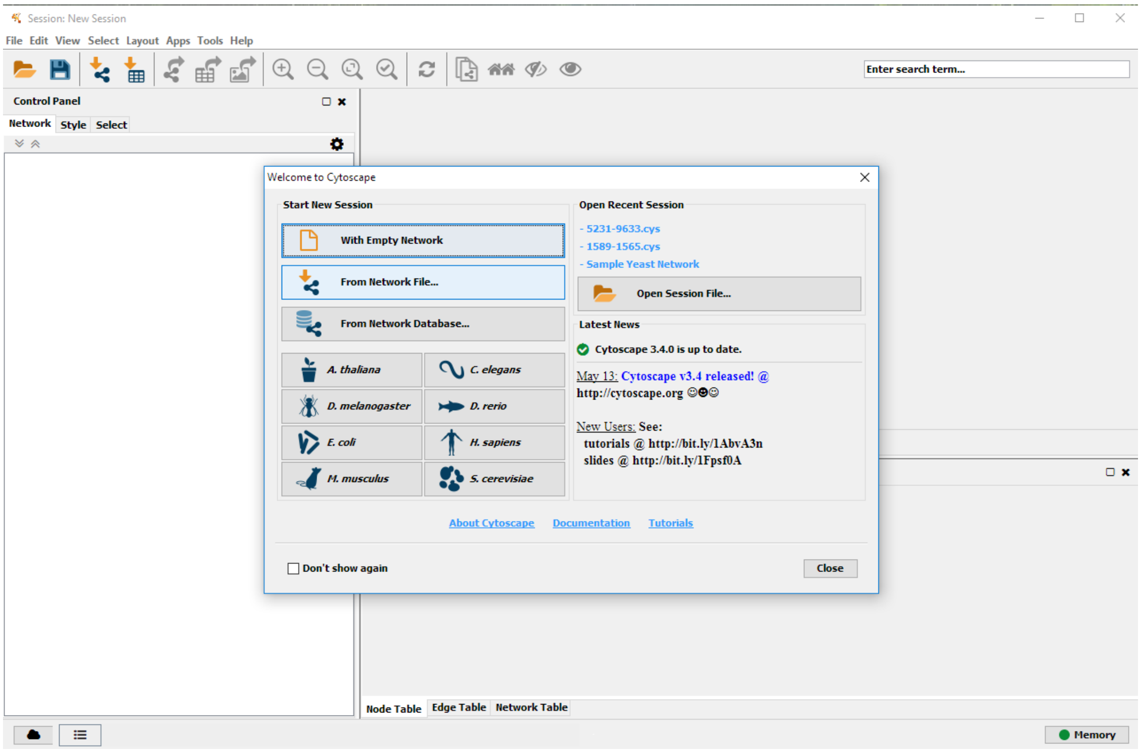

Launch Cytoscape
Import network
Cluster with or without weights
with
Import weights
without
Choose cluster algorithm and parameters
Rank with or without weights
with
Import weights
without
Choose ranking algorithm and parameters
Visualize cluster ranks in RankingPanel
Workflow done!

Figure 9.1: General workflow of using Ranklust to rank clusters

9.2 Detailed walk-through of using Ranklust

To go into more detail, here is the startup screen (9.2) that the user is met with after launching Cytoscape. This assumes that clusterMaker2 with the Ranklust contribution is already installed in Cytoscape.



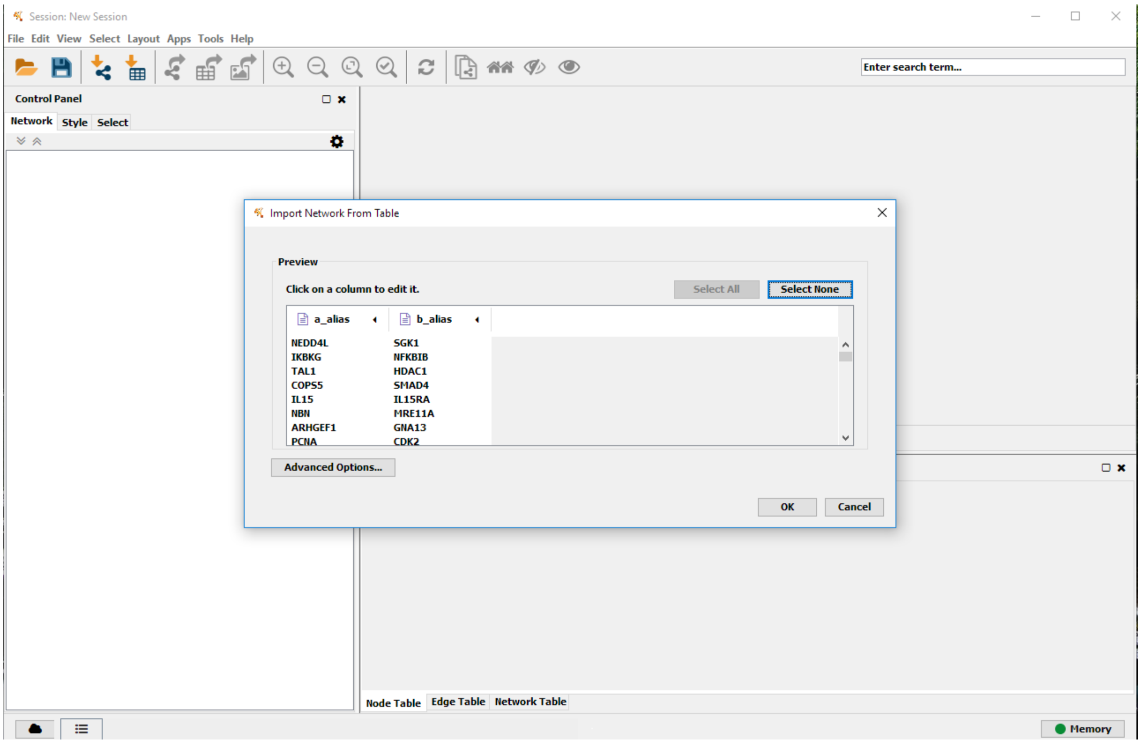
Cytoscape 3.4.0 is up to date.

May 13: Cytoscape v3.4 released! @ http://cytoscape.org ☺☻☺

New Users: See:
tutorials @ http://bit.ly/1AbvA3n
slides @ http://bit.ly/1Fpsf0A

Figure 9.2: Startup screen greeting the user at Cytoscape startup

This screen (9.3) 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. The names for the columns does not have to be exactly a_alias and b_alias, but there should be some header information indicating a source and target gene/node. Having no header in the data imported into cytoscape is not a problem, but it requires the user to explicitly specify this in the "Advanced Options" menu and check off the "Use first line as column names" option.



a_alias	b_alias
NEDD4L	SGK1
IKBKG	NFKBIB
TAL1	HDAC1
COPS5	SMAD4
IL15	IL15RA
NBN	MRE11A
ARHGEF1	GNA13
PCNA	CDK2

Figure 9.3: Importing a network into Cytoscape

It is important to tell Cytoscape which column is considered as the source and target column, as shown here. In this thesis, approved gene symbols from HGNC has been the primary key identifier in both source and target column.

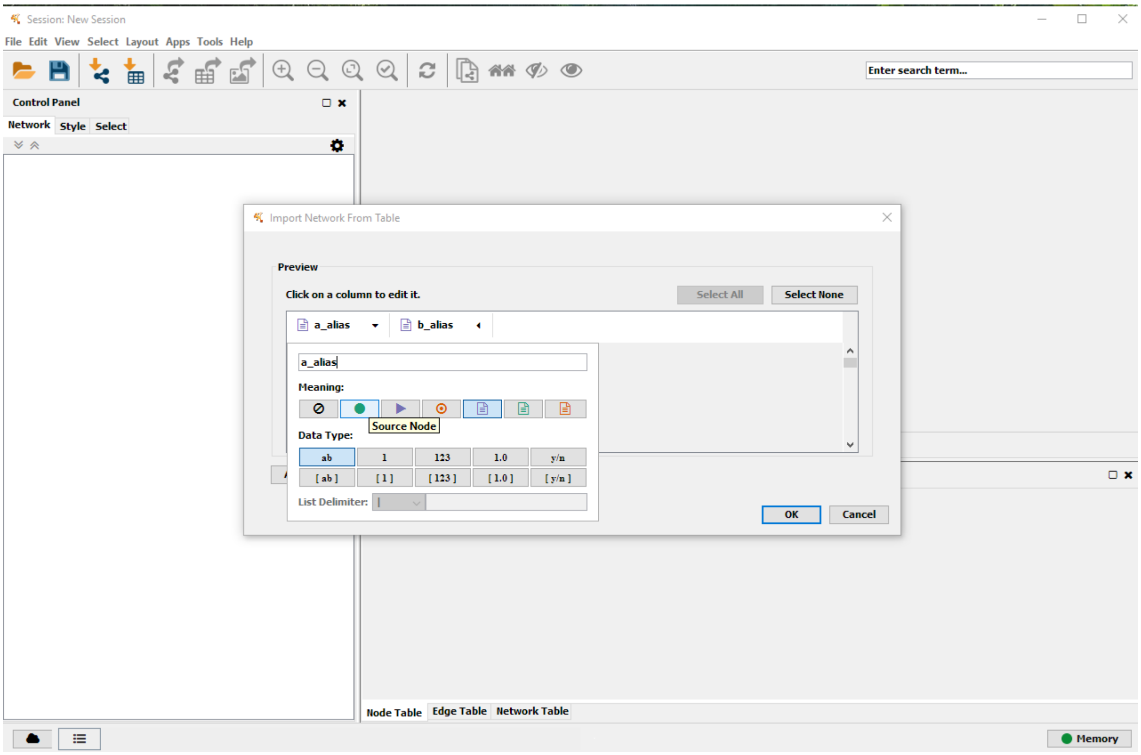
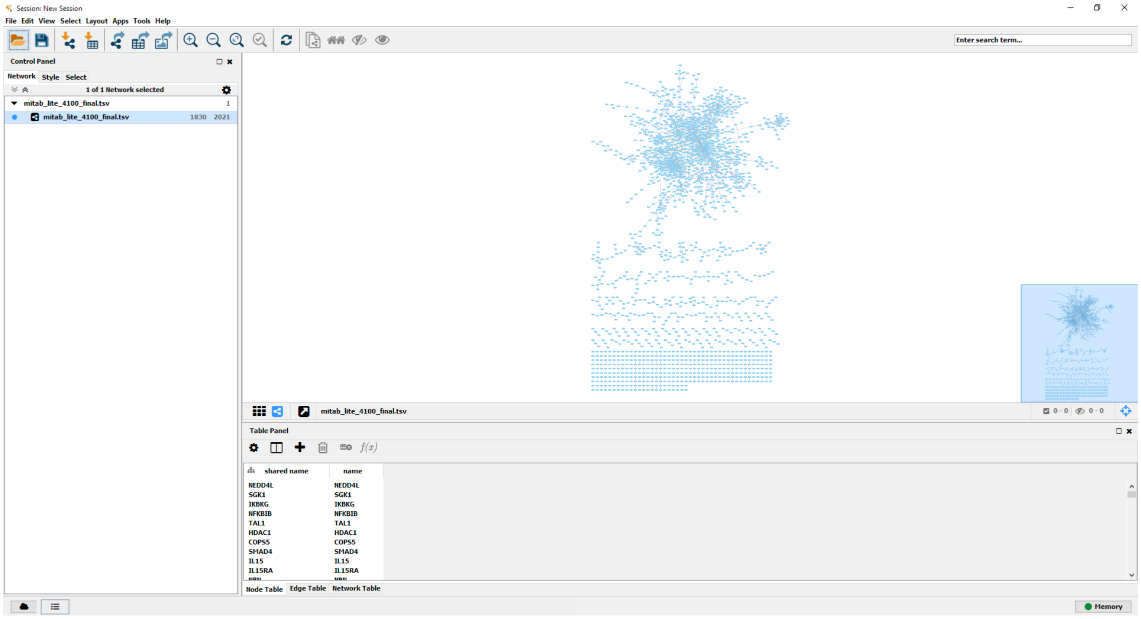


Figure 9.4: Choosing source and target nodes of the interections in the network

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.



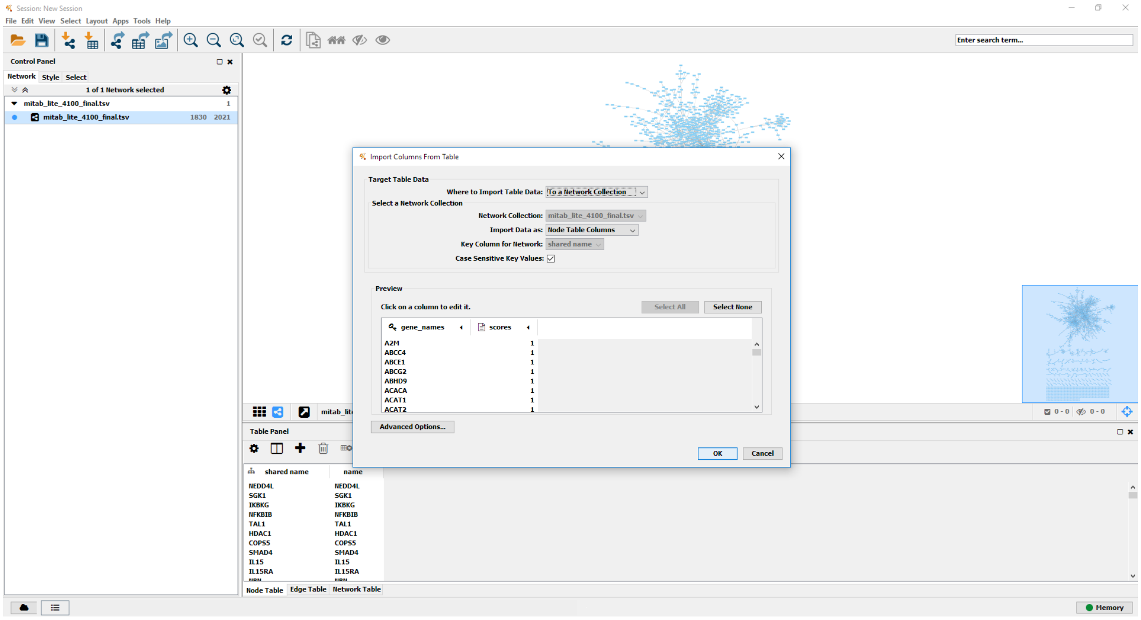
mitab_lite_4100_final.tsv		1
mitab_lite_4100_final.tsv	1830	2021

mitab_lite_4100_final.tsv

shared name	name
NEDD4L	NEDD4L
SGK1	SGK1
IKBKG	IKBKG
NFKBIB	NFKBIB
TAL1	TAL1
HDAC1	HDAC1
COPS5	COPS5
SMAD4	SMAD4
IL15	IL15
IL15RA	IL15RA

Figure 9.5: Cytoscape view after network import has finished

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. Adding prior scores to the iRefWeb network that was created, the file representing the table to import scores from had two columns, "geneName", which representet the gene symbol and was the primary key for matching the score with the right row in the Cytoscape tables, and "score", which contained scores between 0 and 1.



Import Columns From Table

Target Table Data

Where to Import Table Data: To a Network Collection

Select a Network Collection

Network Collection: mitab_lite_4100_final.tsv

Import Data as: Node Table Columns

Key Column for Network: shared name

Case Sensitive Key Values: ☑

Preview

Click on a column to edit it.

gene_names	scores
A2M	1
ABCC4	1
ABCE1	1
ABCG2	1
ABHD9	1
ACACA	1
ACAT1	1
ACAT2	1

shared name	name
NEDD4L	NEDD4L
SGK1	SGK1
IKBKG	IKBKG
NFKBIB	NFKBIB
TAL1	TAL1
HDAC1	HDAC1
COPS5	COPS5
SMAD4	SMAD4
IL15	IL15
IL15RA	IL15RA

Figure 9.6: Adding prior scores to the network

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* row, where the clustering algorithms are located. *clusterMaker Ranking* row, where Ranklust's cluster ranking algorithms MAA,MAM,PR,PRWP and HITS are. Finally, the *clusterMaker Visualizatons* row, where different visualizations in clusterMaker2 can be queried. This row is also where option to visualize Ranklust's ranked clusters resides.

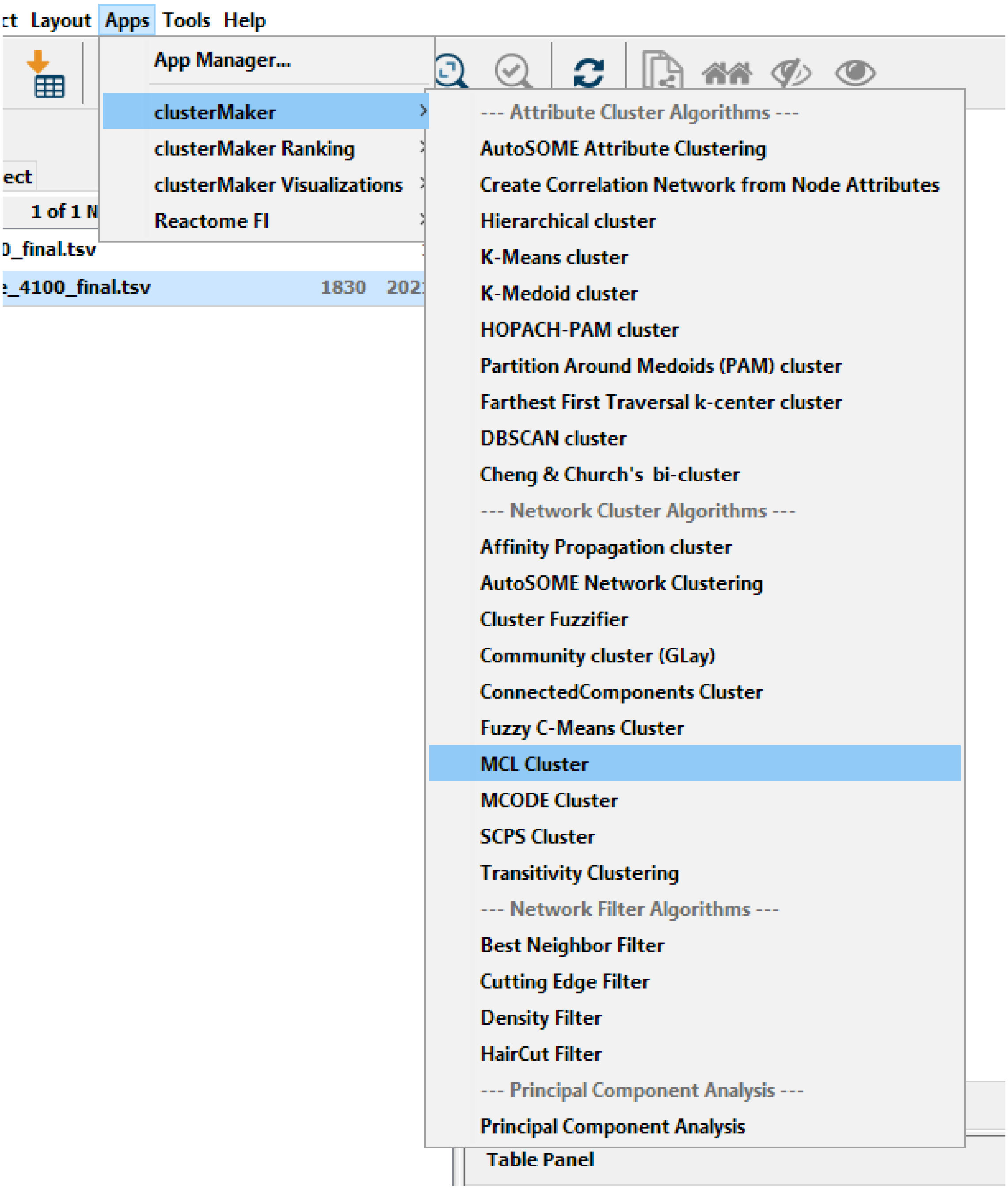
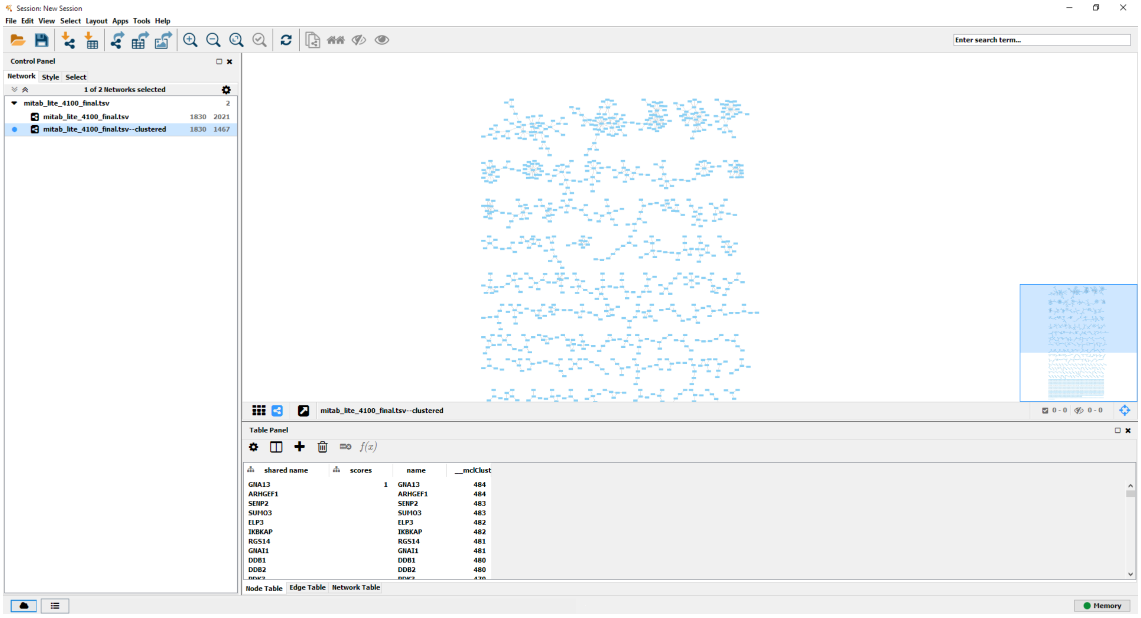

ct Layout Apps Tools Help
App Manager...
clusterMaker
clusterMaker Ranking
clusterMaker Visualizations
Reactome FI
ect
1 of 1 N
D_final.tsv
e_4100_final.tsv 1830 202
--- Attribute Cluster Algorithms ---
AutoSOME Attribute Clustering
Create Correlation Network from Node Attributes
Hierarchical cluster
K-Means cluster
K-Medoid cluster
HOPACH-PAM cluster
Partition Around Medoids (PAM) cluster
Farthest First Traversal k-center cluster
DBSCAN cluster
Cheng & Church's bi-cluster
--- Network Cluster Algorithms ---
Affinity Propagation cluster
AutoSOME Network Clustering
Cluster Fuzzifier
Community cluster (GLay)
ConnectedComponents Cluster
Fuzzy C-Means Cluster
MCL Cluster
MCODE Cluster
SCPS Cluster
Transitivity Clustering
--- Network Filter Algorithms ---
Best Neighbor Filter
Cutting Edge Filter
Density Filter
HairCut Filter
--- Principal Component Analysis ---
Principal Component Analysis
Table Panel

Figure 9.7: Choosing the MCL cluster algorithm in clusterMaker2

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.



1 of 2 Networks selected

mitab_lite_4100_final.tsv		2
mitab_lite_4100_final.tsv	1830	2021
mitab_lite_4100_final.tsv--clustered	1830	1467

mitab_lite_4100_final.tsv--clustered

shared name	scores	name	__mclClust
GNA13	1	GNA13	484
ARHGEF1		ARHGEF1	484
SENP2		SENP2	483
SUMO3		SUMO3	483
ELP3		ELP3	482
IKBKAP		IKBKAP	482
RGS14		RGS14	481
GNAI1		GNAI1	481
DDB1		DDB1	480
DDB2		DDB2	480

Figure 9.8: Network view of the clustered network created by MCL - note the added column in the Node Table

Here the cluster ranking algorithm menu is shown.

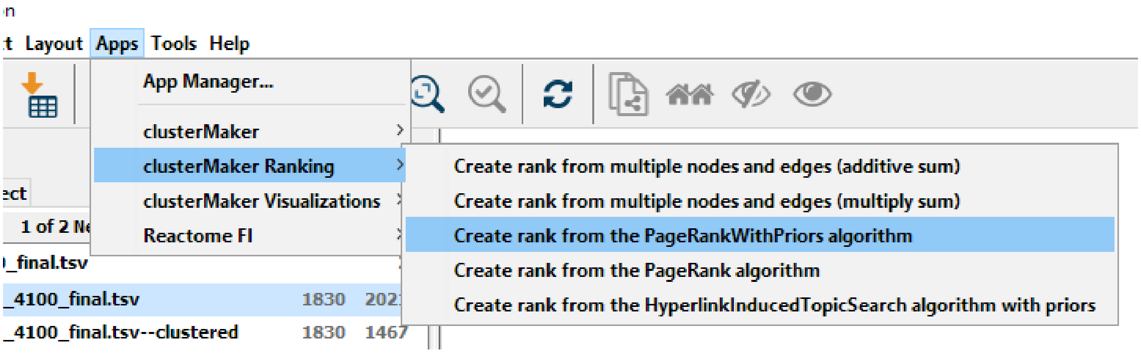
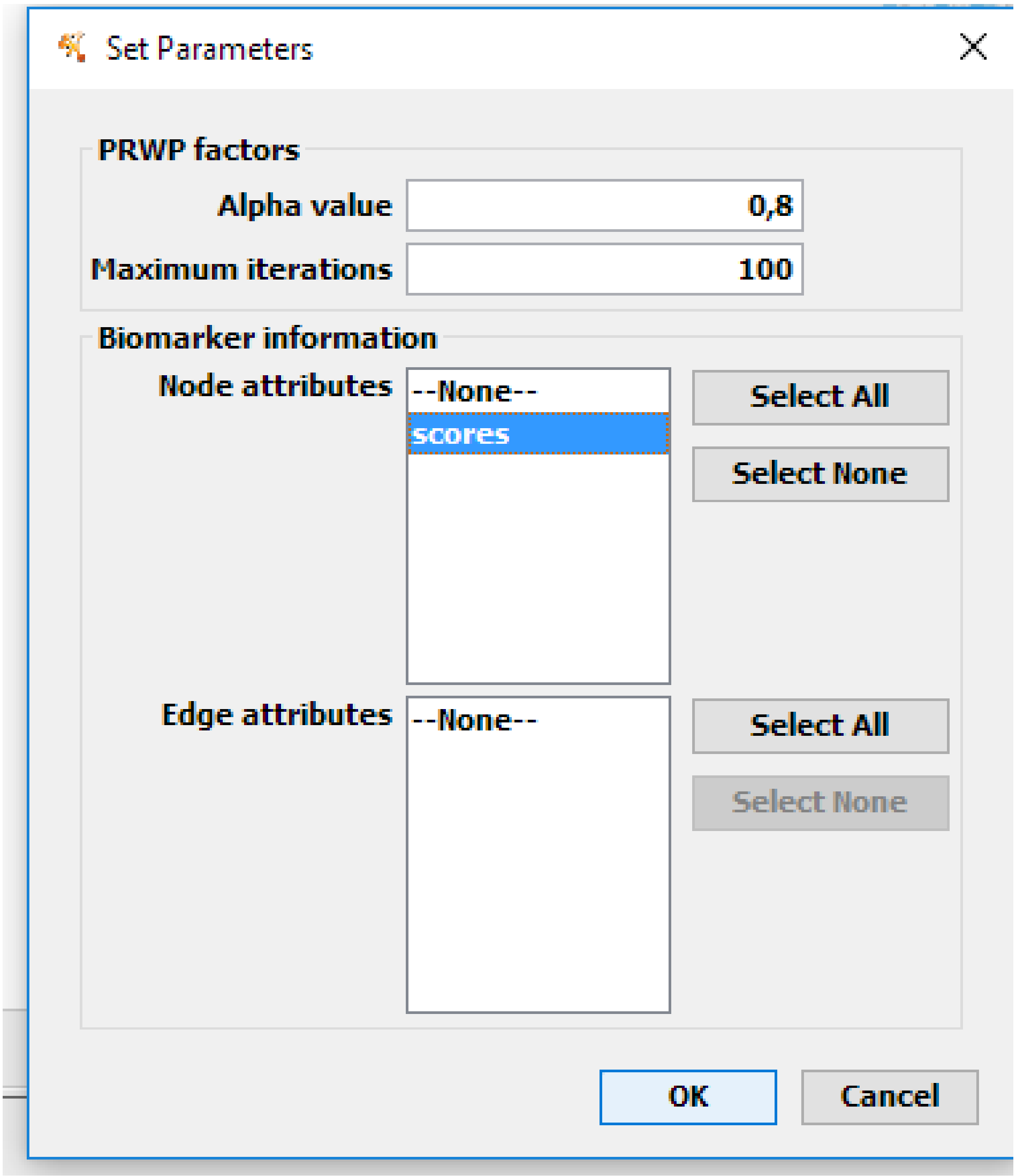

t Layout Apps Tools Help
App Manager...
clusterMaker
clusterMaker Ranking
clusterMaker Visualizations
Reactome FI
Create rank from multiple nodes and edges (additive sum)
Create rank from multiple nodes and edges (multiply sum)
Create rank from the PageRankWithPriors algorithm
Create rank from the PageRank algorithm
Create rank from the HyperlinkInducedTopicSearch algorithm with priors
ect
1 of 2 Ne
_final.tsv
_4100_final.tsv 1830 202
_4100_final.tsv--clustered 1830 1467

Figure 9.9: Choosing the PageRankWithPriors (PRWP) ranking algorithm to rank the clusters in the network

The "Create rank from the PageRankWithPriors algorithm" (PRWP) was chosen as the ranking algorithm. Node and edge attributes can be combined by the user in through simple selection. Selecting no attributes will cause PRWP to not calculate scores at all. If the user does not want to use attributes to rank the clusters, but still use PageRank, the "Create rank from the PageRank algorithm" (PR) algorithm should be used.



Set Parameters

PRWP factors

Alpha value: 0,8

Maximum iterations: 100

Biomarker information

Node attributes: --None--, scores

Edge attributes: --None--

Figure 9.10: Setting the PRWP parameters before executing the algorithm

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 of clustermaker2. 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. After the loading dialog is finished, the results panel for the ranked clusters will be shown.

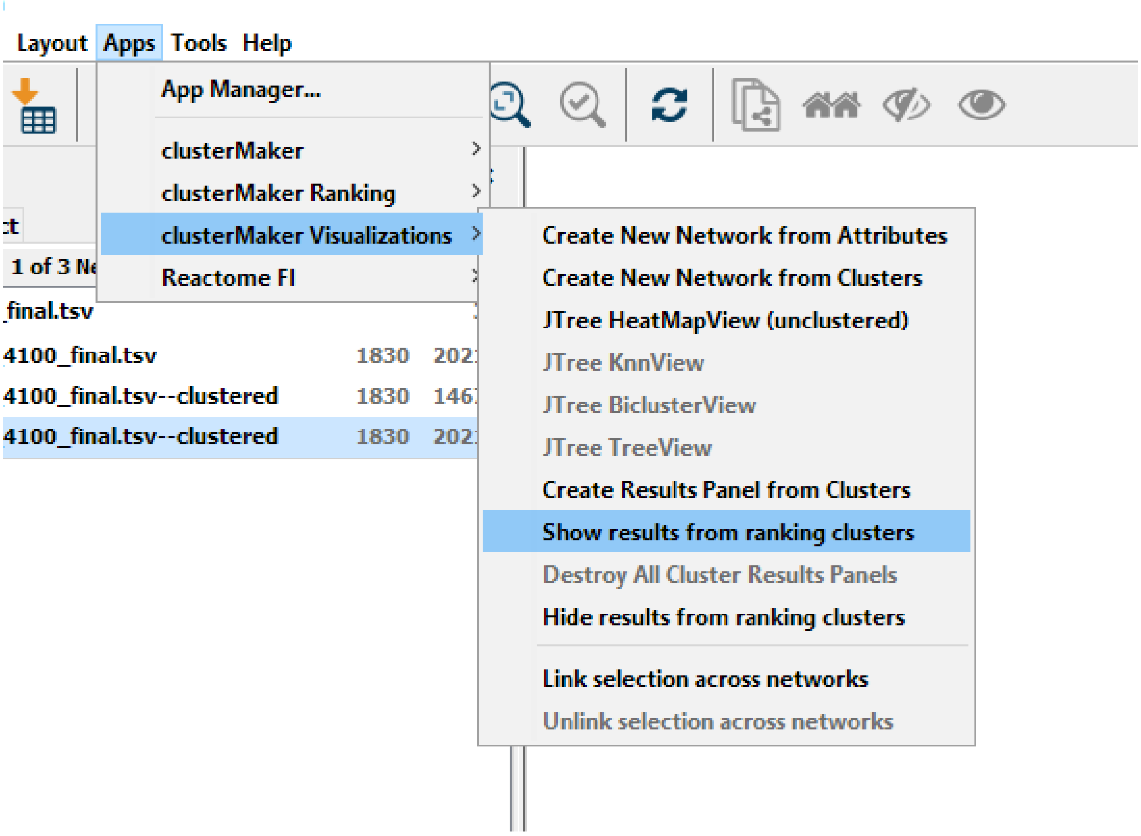

Layout
Apps
Tools
Help
App Manager...
clusterMaker
clusterMaker Ranking
clusterMaker Visualizations
Reactome FI
Create New Network from Attributes
Create New Network from Clusters
JTree HeatMapView (unclustered)
JTree KnnView
JTree BiclusterView
JTree TreeView
Create Results Panel from Clusters
Show results from ranking clusters
Destroy All Cluster Results Panels
Hide results from ranking clusters
Link selection across networks
Unlink selection across networks
1 of 3 Ne
final.tsv
4100_final.tsv 1830 202
4100_final.tsv--clustered 1830 146
4100_final.tsv--clustered 1830 202

Figure 9.11: Choosing to visualize the results after the PRWP algorithm has finished

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.

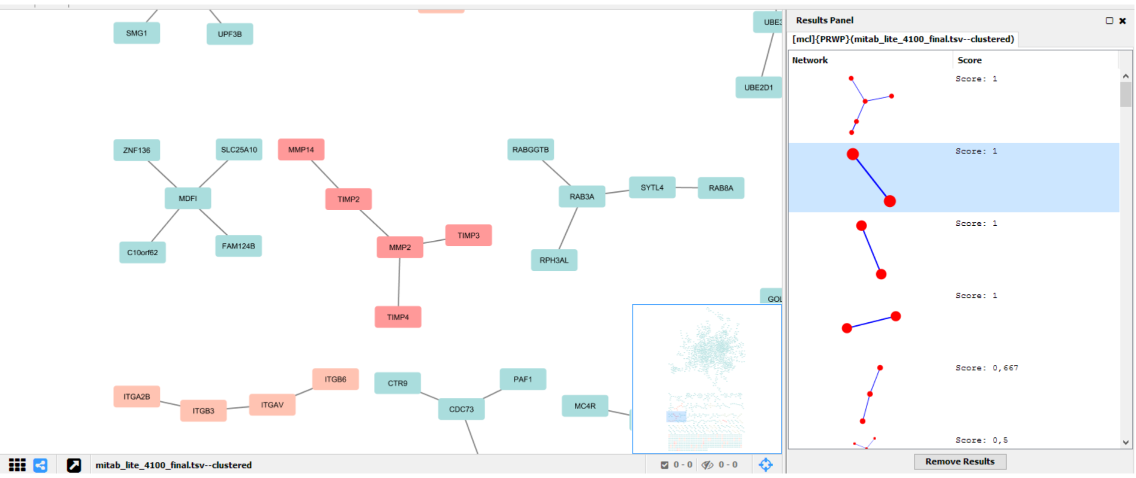
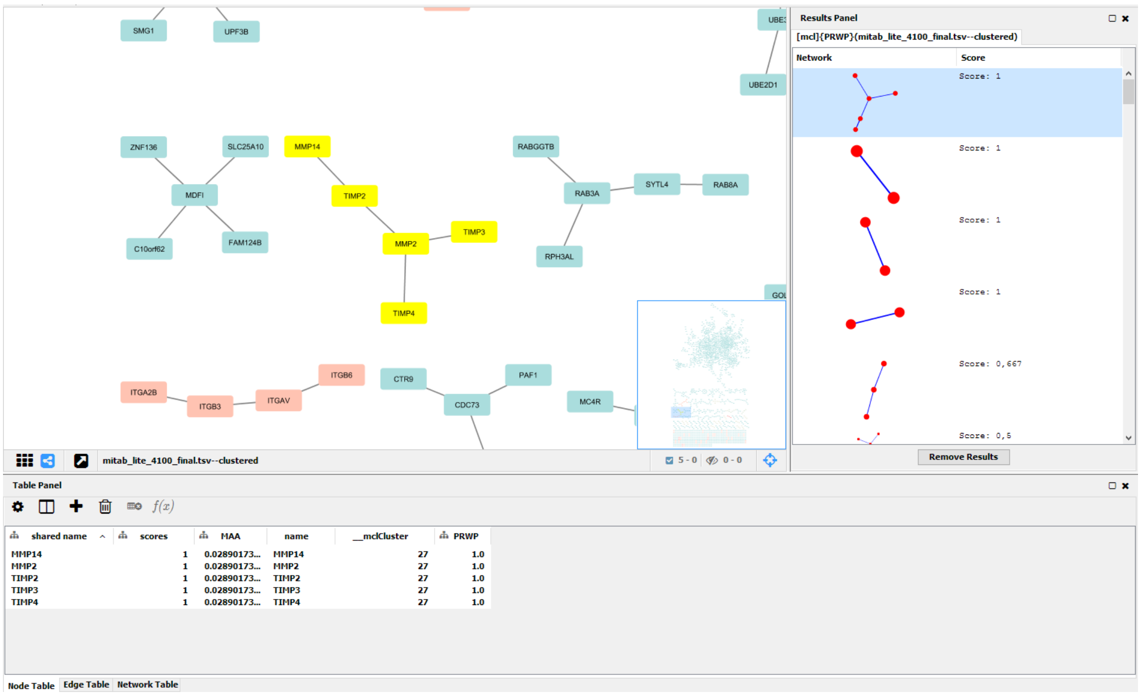


Figure 9.12: The visualization of the ranked clusters are finished, both colored nodes in the network and the results panel on the right side is displayed to the user

Here is an example of how the clusters change color when the same cluster is selected from the results panel menu.



shared name	scores	MAA	name	__mclCluster	PRWP
MMP14	1	0.02890173...	MMP14	27	1.0
MMP2	1	0.02890173...	MMP2	27	1.0
TIMP2	1	0.02890173...	TIMP2	27	1.0
TIMP3	1	0.02890173...	TIMP3	27	1.0
TIMP4	1	0.02890173...	TIMP4	27	1.0

Figure 9.13: Change the color of the nodes in a cluster when the cluster is selected from the results panel

9.3 Design

The class relations in Ranklust's ranking algorithms and ranking results panel are described below in simple UML diagrams. The goal of this design was to have an implementation of the ranking algorithms and the panel as close to the existing clustering algorithms and results panel.

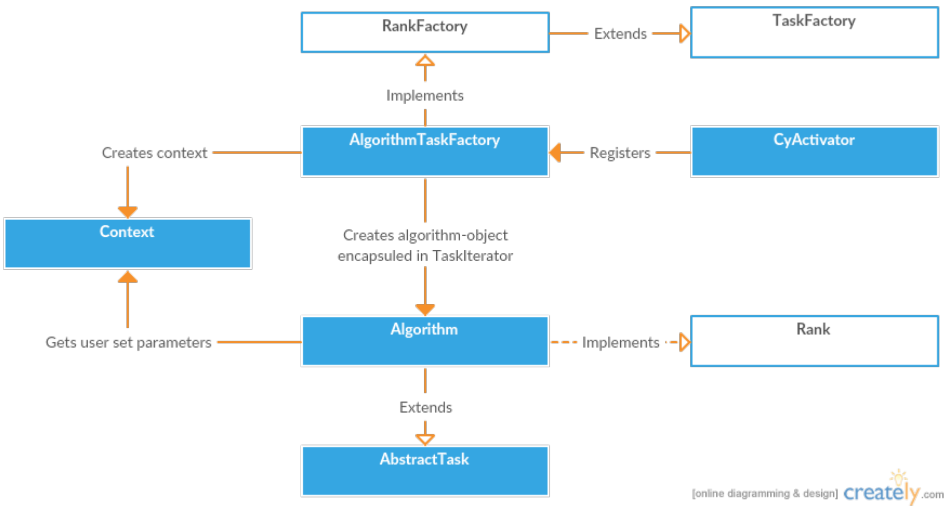


Figure 9.14: UML diagram for Ranklust's class relations for the ranking algorithms

The ranking algorithm relations (Figure: 9.14) shows how the classes are connected together and what they offer. The context is not self-explanatory, but it has responsibility for the GUI component that represents each specific ranking algorithm. Each algorithm in Ranklust has its own instance of the following classes:

- AlgorithmTaskFactory
- Context
- Algorithm

The rest of the classes are are not unique to any of the ranking algorithms.

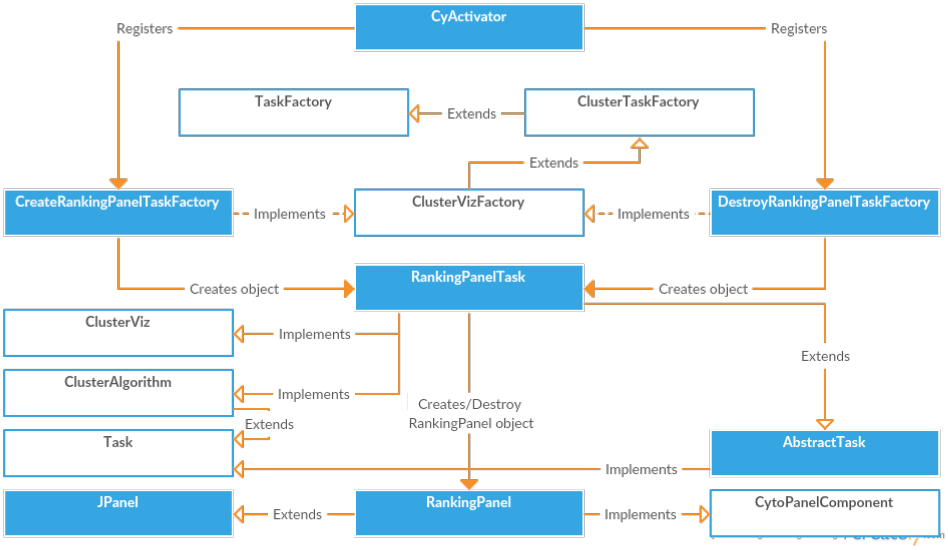


Figure 9.15: UML diagram for Ranklust's class relations for the ranking results panel

The ranking panel relations (Figure: 9.15) shows how the classes are connected to the ranking panel. The "Create" and "Destroy" task factories exists to be able to destroy all of the panels ranking clusters, or to create a new one. They are two separate classes to appear in the Cytoscape menu as two separate options. Both of these two classes returns a RankingPanelTask object, where the arguments sent to its class constructor defines whether if the task should create or destroy a panel. An advantage to have two separate classes responsible for creating or destroying a panel is the ability to make them unavailable. For example, it is not possible for the user to select the "Destroy All Cluster Results Panels" in the "clusterMaker Visualizations" menu, if no ranking results panels exists.

Chapter 10

Prioritizing network biomarkers in prostate cancer through graph analysis

10.1 How the network was created

After querying iRefWeb for the PPI-network with the query displayed earlier (figure 8.1) the resulting network consisted of 109276 interactions. After filtering the same network through the protein-to-gene mapping constructed from HGNC, the final network consisted of 9500 nodes (genes) and 43706 edges (interactions). At this point, all of the nodes and edges were undirected and unweighted. Converting from proteins to genes ended up with a 60% perturbation of the network in the form of removed edges. Clustering the network resulted in a further perturbation of 69.8% of edge-removal when compared to the HGNC-filtered network, 87.9% when compared to the unfiltered iRefWeb network.

The creation of the golden standard through combining DisGeNET and DDPC resulted in a file with two columns. One representing a gene, the second one representing the score of a gene, so a single row would contain a unique gene in the list and the score it received.

10.2 Which parameters for clustering was used and why

Talking about MCL results

Inflation	Clusters	Avg. cluster size	Max. cluster size	Min. cluster size	Modularity
1.6	1068	8.88	968	2	0.367
1.8	1400	6.60	660	2	0.307
2.0	1599	5.68	405	2	0.269
2.5	2053	4.20	179	2	0.223
3.0	2210	3.75	122	2	0.199

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 preferable, 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 (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[8].

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 Results from cross-validation and what they indicate

The plots in figure 10.1 and figure 10.2 is developed from the 10 random cross-validation runs ranked with PRWP and MAA. The x-axis represents the cluster ranks which is represented as a blue dot in the scatter plot. Each rank consists of a single cluster which has an arbitrary number of genes above 1. The y-axis represents a result from two steps. The first step was to go through each of the 10 cross-validated results from ranking the iRefWeb network with PRWP and MAA. In each of the cross-validated results an average was calculated for each cluster. The average was calculated from dividing the number of genes, that had their prior score removed as a result of the cross-validation, by the total number of genes in the cluster. Each cluster would at this point have an average representing the average of

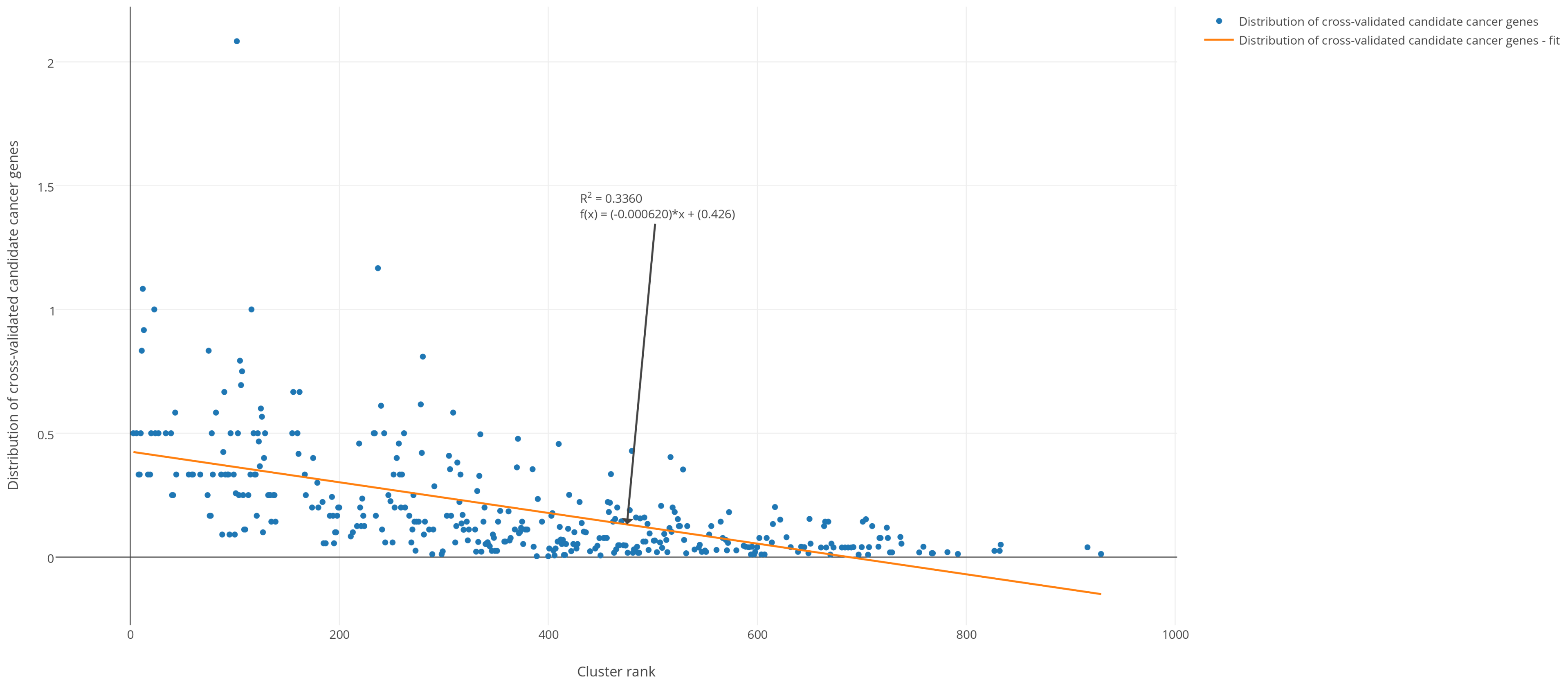


Figure 10.1: Distribution of combined averages of genes, which had their scores removed by cross-validation, ranked by PRWP

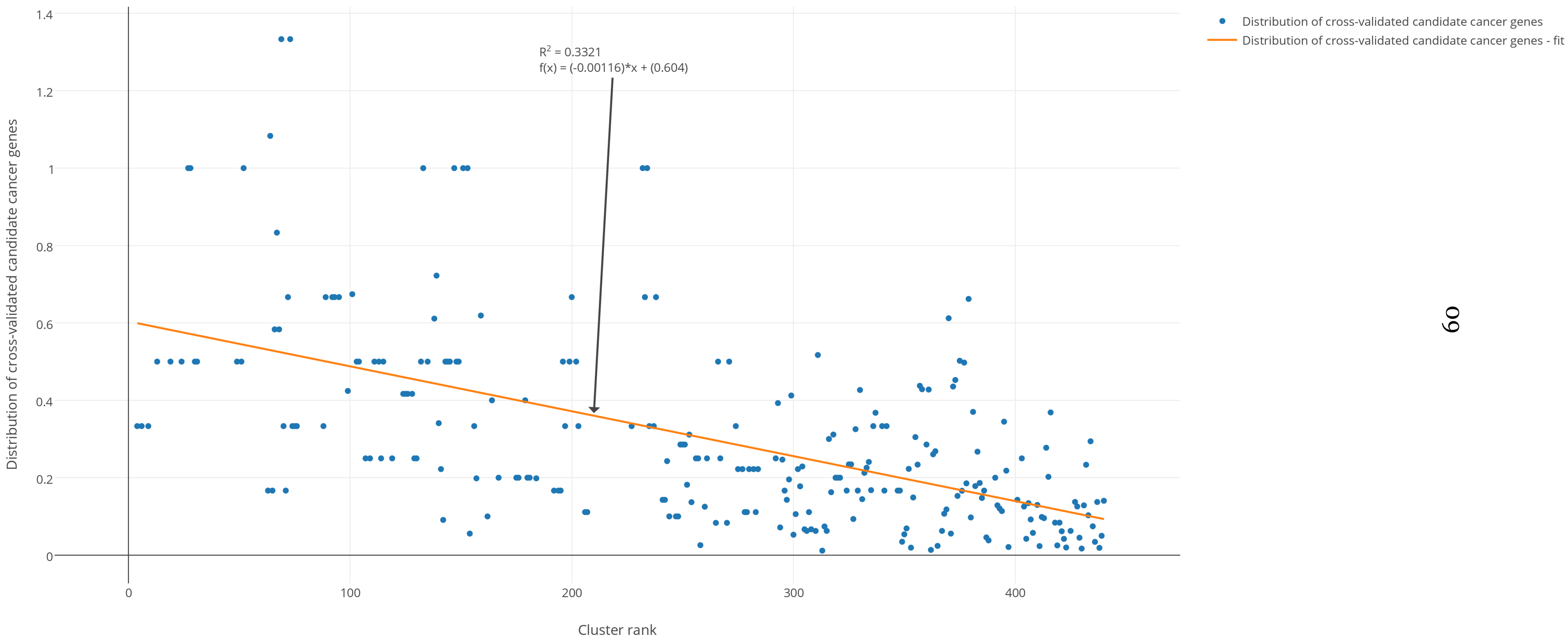


Figure 10.2: Distribution of combined averages of genes, which had their scores removed by cross-validation, ranked by MAA

cross-validated genes in a cluster. The second step completes the values for the y-axis in the plot and represents the score in a cluster when additively combining the averages from each of the 10 cross-validated results that was found by PRWP and MAA. To filter out the uninteresting results, all zero values on the y-axis was removed.

The analysis of which clusters contained the largest combined average of genes, which had their prior score removed by cross-validation, shows that the topmost ranked clusters had the highest combined average. This information indicates that ranking clusters with PRWP and MAA have a tendency towards ranking the larger part of the population of prostate cancer biomarkers at the top of the cluster ranks, and the lower at the bottom.

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, to prove that the distribution of the combined average in the clusters should correlate to the rank they obtained through Ranklust's use of PRWP and MAA.

Chapter 11

Benchmarking Ranklust against text mined, manually knowledge curated and experimental test data

11.1 Retrieving the test data from the DISEASE database

Benchmarking Ranklust is done against three resources of data from a single database called DISEASE[58]. This database can be queried for diseases or genes. The query in this database was limited to searching for a single disease or gene name. No API was found to access the database directly, so to retrieve the gene names related to prostate cancer, whole files was downloaded. There were three files, one for each type of research put into retrieving the data: text mined, manually curated knowledge and experimental data. Each of these files was filled with genes and their relation to different diseases, so they had to be filtered to only contain genes with information about prostate cancer. This was done by only using genes that contained "prostate cancer" in the column indicating which disease the specific gene was related to.

The plots representing values in clusters from each of these three files have split each cluster in two parts, blue and orange. The blue dots in the scatter plot represents the genes in a cluster that has prior scores. The orange dots in the scatter plot represent genes in the same cluster as the blue ones in terms of which rank they are in based on the x-axis, but they do not have prior scores. The blue dots are also mentioned as prostate cancer genes and orange dots as prostate candidate cancer genes, because they have no score, but they are in some cases related to other prostate cancer genes to such a degree that they are susceptible to be candidate cancer biomarkers for prostate cancer. As with the cross-validation plots, the zero values in the plots have been removed.

11.2 Z-scores for text mined genes in clusters

The text mined scores are represented by a z-score. The z-score to a gene in the text mined data from the DISEASE database is a developed from a co-occurrence score, which increased when a gene and a disease was mentioned together, but also decreased when they were mentioned with multiple other genes or diseases. This co-occurrence score was later converted to z-scores to be more robust to changes to the size of the text corpus in the DISEASE database[58]. This results in the average z-score to a cluster, which is based on the average z-score of each gene in a cluster, to be a benchmark as to how high the cluster should be ranked in terms of being a relevant network biomarker for prostate cancer.

Since the plots have split each cluster into two parts, the cluster part of genes with priors and the ones without priors, the expected outcome, should the ranking algorithms perform as expected, would be to have the blue dots descending and the orange dots ascending, when looking at a linear regression fit going from the topmost ranked cluster to the lowest.

11.2.1 PRWP benchmarked with text mined genes

For PRWP (figure: 11.1), the prostate cancer candidate genes is descending in z-values from the topmost ranked cluster to the lowest, which is contributing to showing PRWP's 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 could 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 examinated to the degree that they are not mentioned as a single gene related to prostate cancer in scientific papers.

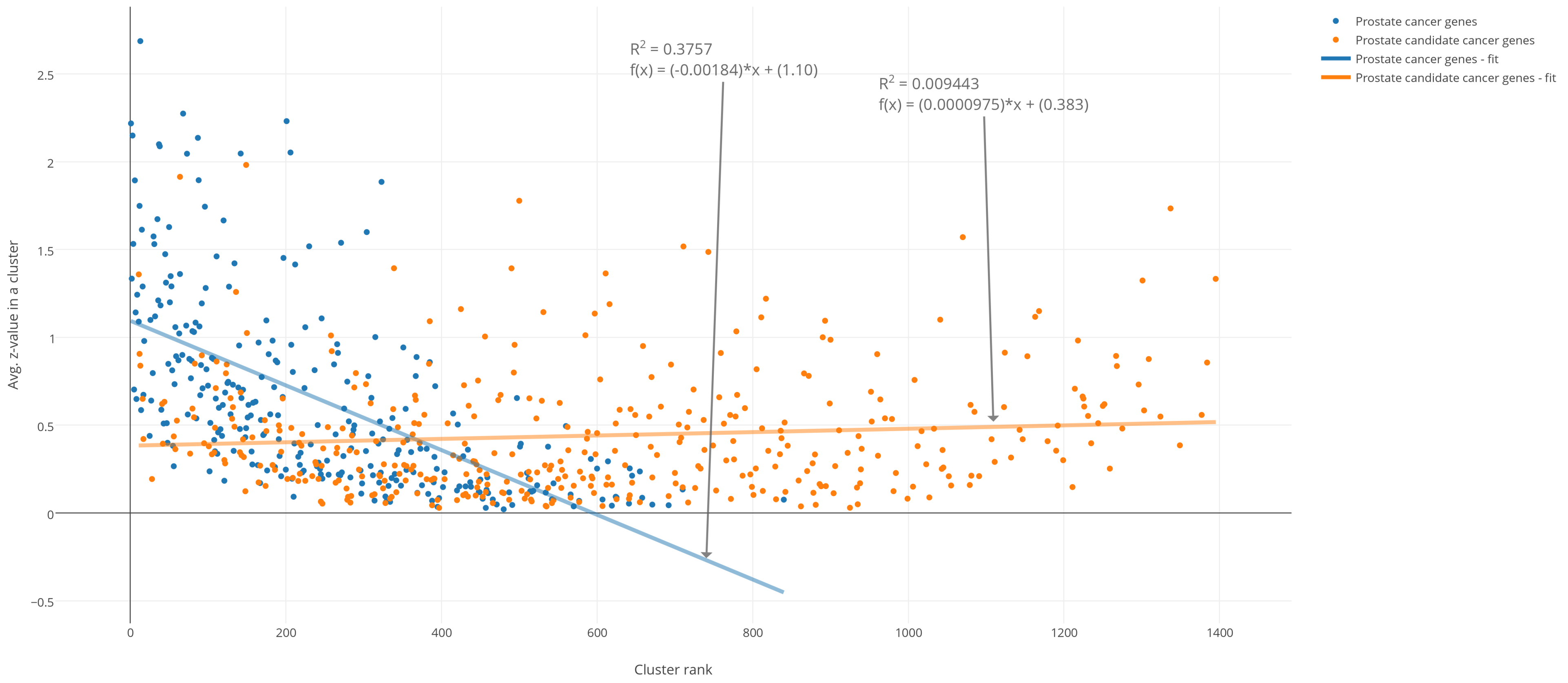


Figure 11.1: Average z-score in a cluster ranked by PRWP

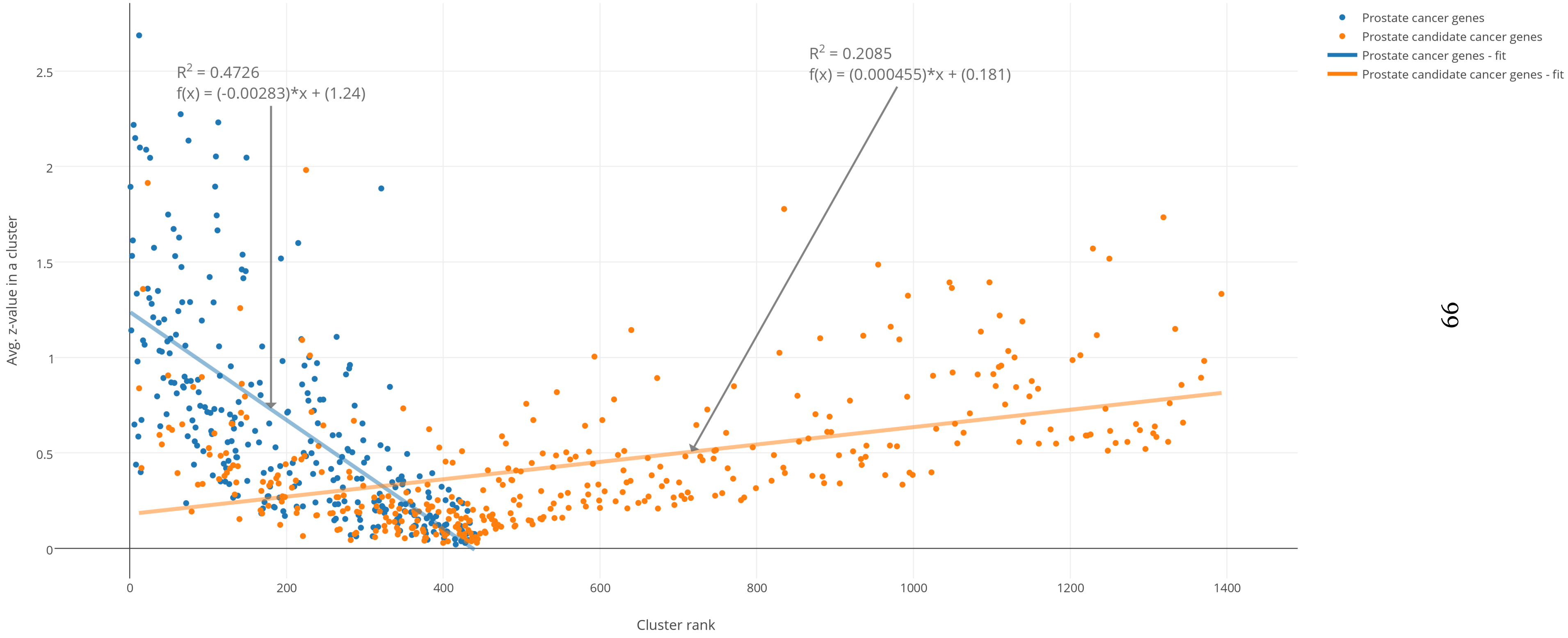


Figure 11.2: Average z-score in a cluster ranked by MAA

11.2.2 MAA benchmarked with text mined genes

For MAA (figure: 11.2), 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 regression fit for the z-values in the prostate candidate cancer genes.

11.3 Manually knowledge curated genes in a cluster

This data had no score except for a confidence score in the DISEASE database. Every gene in the manually knowledge curated file had a confidence score of 5 stars, due to being manually curated by researchers[58]. Therefore, the average number 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 occurrence, and not a specific value, in contrast to the text mined and experimentally mined genes in the database. The clusters are split two-ways, in a similar way that the benchmark with the text mined genes was, blue for genes in the cluster which have priors and orange for the genes that do not have prior scores.

Another trait the knowledge curated data possesses 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 knowledge by researchers. This is why the amount of entries for knowledge curated data is so sparse when compared to text mined data.

For the manually knowledge curated genes to indicate valuable rankings of the clusters, the genes with prior scores should have a descending trend from the topmost ranked cluster to the lowest. If the genes without prior scores, have a clear ascending trend from the topmost to the lowest ranked cluster, it would have been a direct contradiction to the fact that the ranking algorithms should be able to rank genes without prior scores in a reasonable way if they are related to genes with priors. A higher frequency of manually knowledge curated genes would increase the validity of these trends, should they occur, especially if they are subtle.

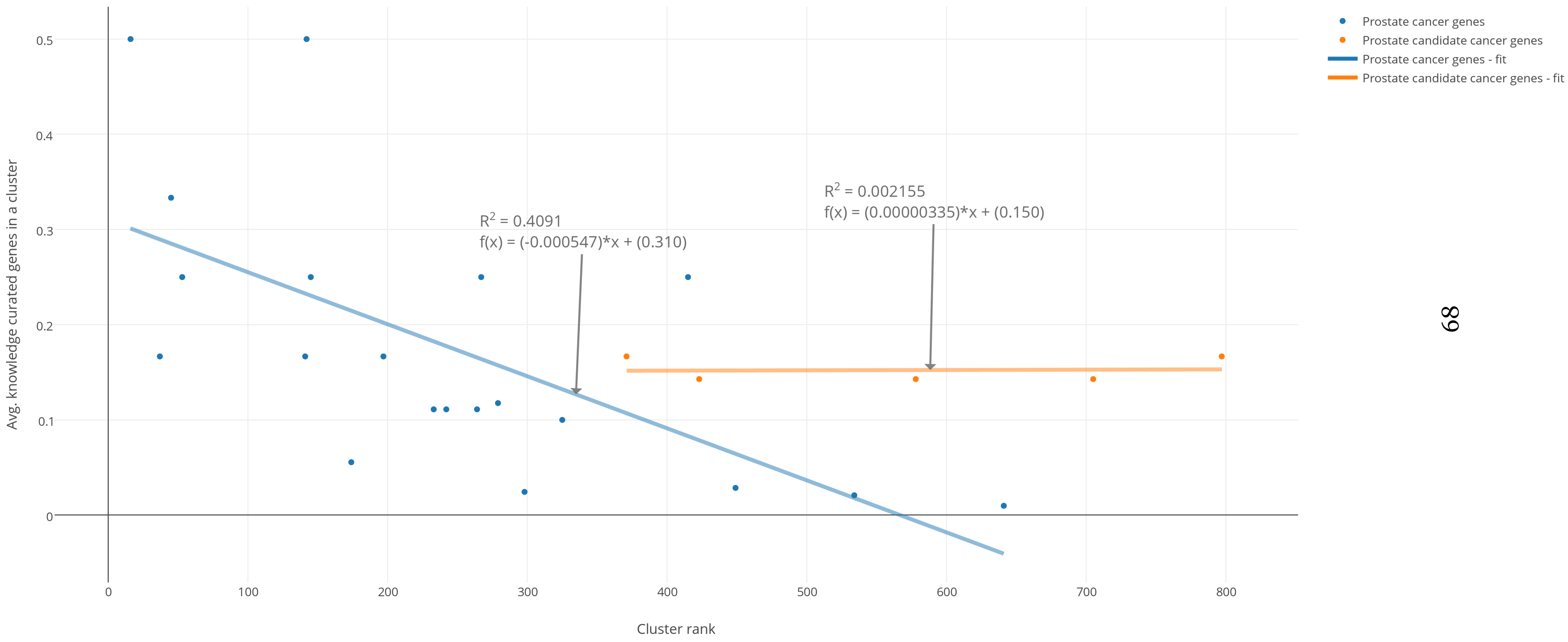


Figure 11.3: Average distribution of curated knowledge mined genes in clusters ranked by PRWP.

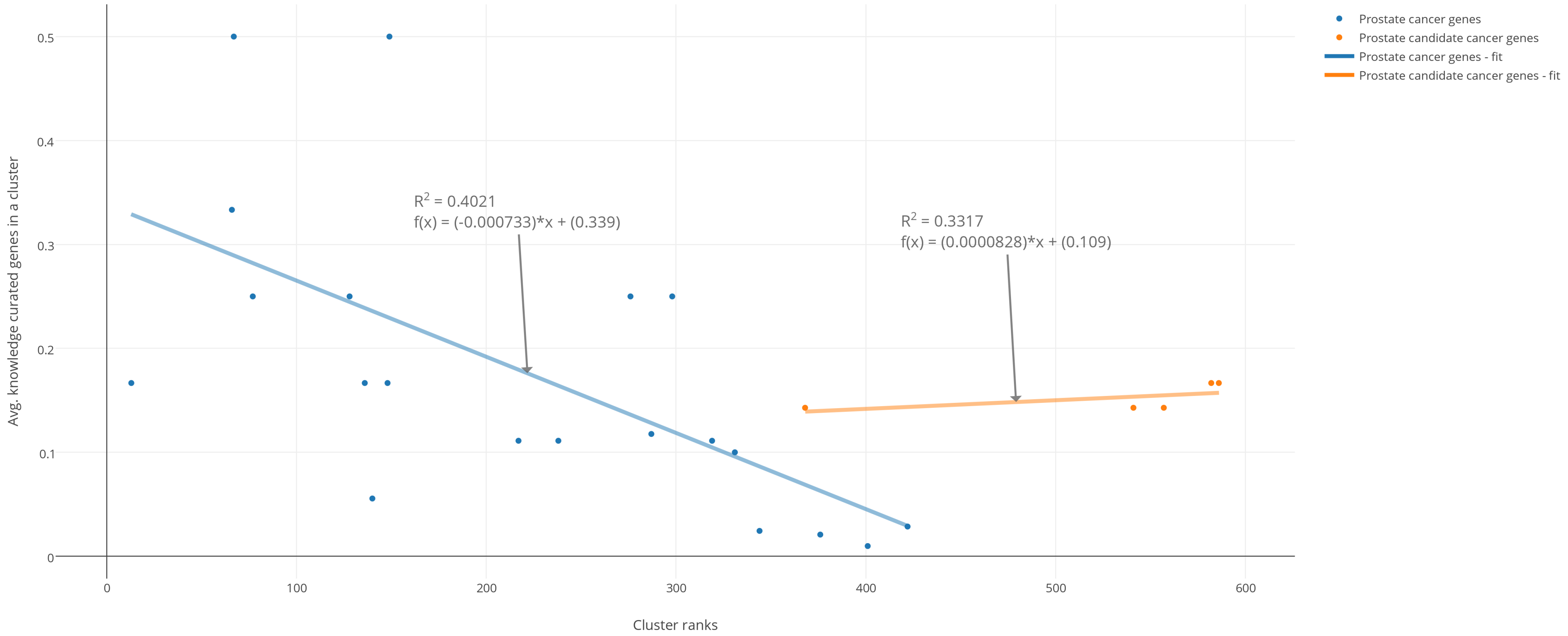


Figure 11.4: Average distribution of curated knowledge mined genes in clusters ranked by MAA.

11.3.1 PRWP benchmarked by manually knowledge curated genes

For PRWP (figure: 11.3), the genes with prior scores in a cluster have a clear descending trend from the topmost to the lowest ranked cluster. This builds up under the validitiy of PRWP being able to rank prior scored genes correctly. The genes without prior scores in a cluster does ascend to a low degree and the R-squared for the fit that has this ascending trend is not deemed as a good fit for the scores, at a R-squared value of only 0.002.

11.3.2 MAA benchmarked by manually knowledge curated genes

For MAA (figure: 11.4), the genes with prior scores in a cluster have the same trend as in PRWP. For the genes without prior scores in a cluster, the trend also is the same as in PRWP, but to a higher degree. The fit for the ascending average in manually knowledge curated genes in a cluster, for the genes in the cluster without prior scores is an R-squared value of 0.33, which is considerably higher than the ascending value for PRWP.

11.4 Experimental genes distribution of p-values in genes

All of the experimental genes are from experiments and are a result from genome-wide association studies (GWAS). Each gene in this test data are scored after p-values from "the most statistically significant SNP within the block."[23] in experiments related to prostate cancer. The DISEASE database has experimental data from both the Catalogue of Somatic Mutations in Cancer (COSMIC) and DistiLD, but for prostate cancer, only experimental data from DistiLD was available.

The score for each cluster is calculated from the average p-value from each gene in the cluster. In contrast to the previous plots, the trend required to validate PRWP and MAA as cluster ranking algorithms for prioritizing network biomarkers in prostate cancer, is an ascending trend in both genes with and without prior scores, when ranking clusters from the topmost to the lowest cluster. An ascending score for the genes with prior scores proves that the highly ranked clusters have low p-values, which is proof of a low chance of the null hypothesis being true. Here, the null hypothesis would be that a gene has no relevance to prostate cancer. The golden standard is based on DDPC and DisGeNET scores. The DisGeNET scores are developed as a score based on the supporting evidence for the context of a gene and a disease being true[57]. Based on this fact, it is feasible if PRWP and MAA

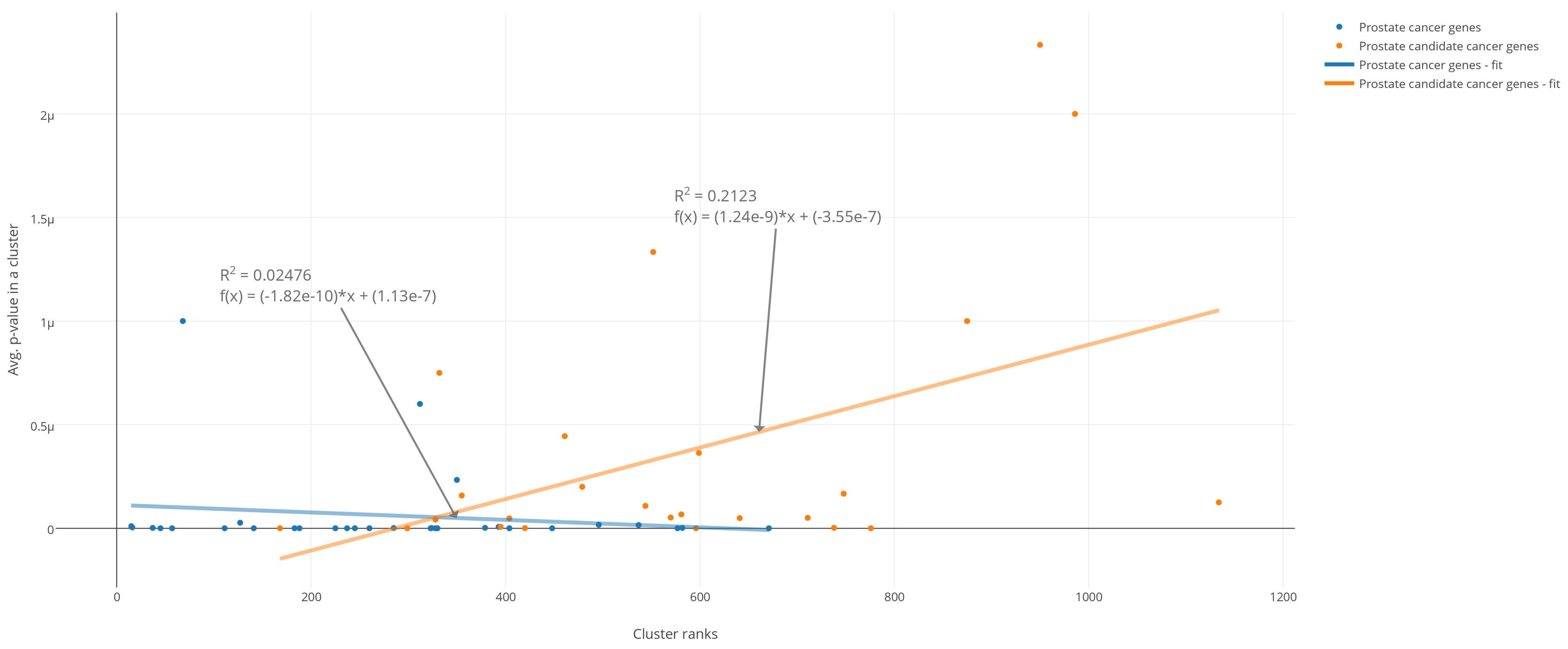


Figure 11.5: Average distribution of p-values in clusters ranked by PRWP.

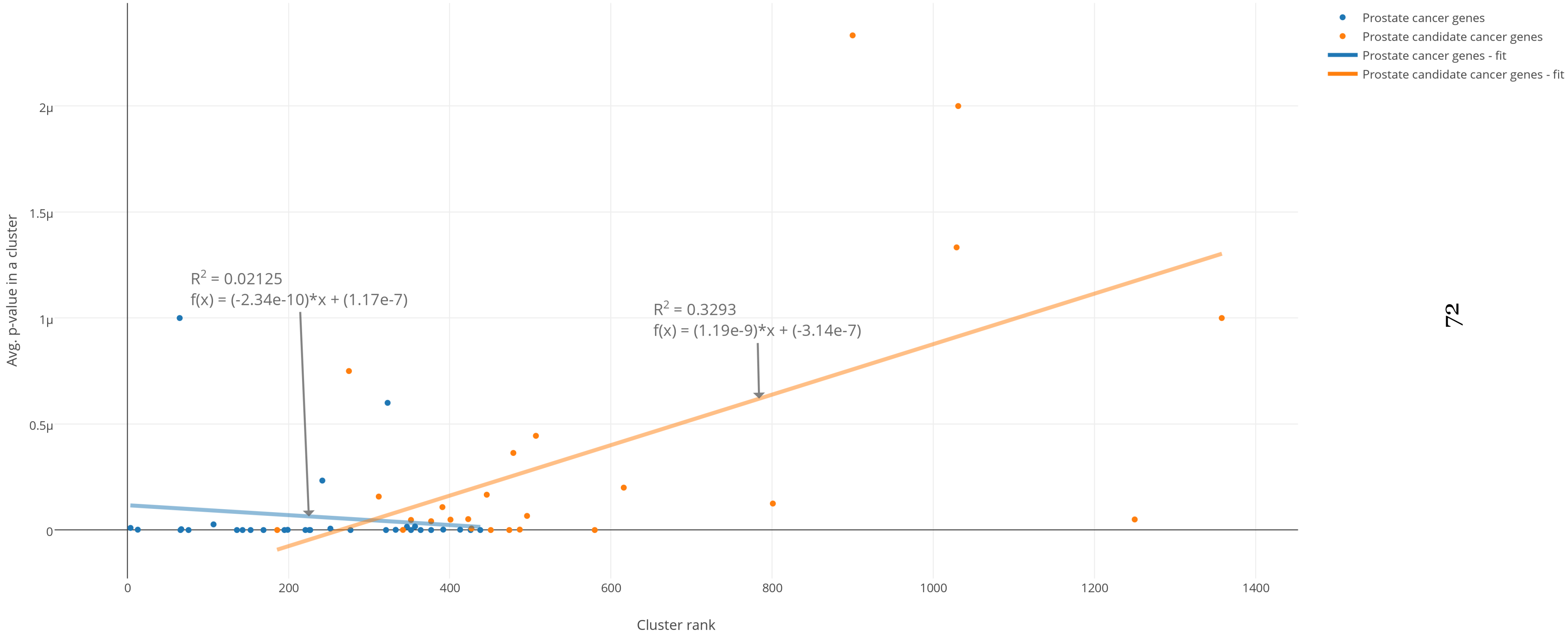


Figure 11.6: Average distribution of p-values in clusters ranked by MAA.

would demonstrate the previously defined trend.

The experimental data has one of the same feats as the manually knowledge curated data, it is very sparse. As with the previous plots, the genes with prior scores in a cluster are colored blue, and the ones without prior scores as orange.

11.4.1 PRWP benchmarked by experimentally mined genes

For PRWP (figure: 11.5), there is a trend for descending p-values for the genes with prior scores in a cluster and an ascending p-value trend for the genes without prior scores in a cluster. For the genes without prior scores, this is a feasible result in order to validate Ranklust as a tool for prioritizing network biomarkers. For the genes with prior scores, there is a single cluster that seems to be the cause of the descending trend but the R-squared value for the fit of the linear regression is not high.

11.4.2 MAA benchmarked by experimentally mined genes

For MAA (figure: 11.6), the trends are the same as with PRWP, but the trends are more noticeable. For the genes with prior scores, the descending p-value trend towards the lower ranked clusters are steeper than the one in the plot for PRWP, but the R-squared score is very low and the as it is in the PRWP plot, it seems to be a single cluster that is responsible for the descending trend in p-values.

Chapter 12

Comparison of PRWP and MAA to prostate cancer relevant genes

Using PRWP and MAA, 6 final lists of prioritized network biomarkers for prostate cancer will be presented. These 6 lists will be the result of 3 PRWP and 3 MAA ranked clusters from the whole iRefWeb filtered network with gene names and the scored with the full golden standard. Each of the 3 ranking results from both algorithms will use the same three sets of data, namely movember, cosmica and lethal.

12.1 Using Ranklust to rank the clusters of a network

The exact workflow of using Ranklust to create the ranks that the test data is compared to is exactly as explained earlier in both the methods chapter and the workflow part of this results chapter.

12.1.1 Step 1 - create the network and fill it with prior scores

The network was downloaded from iRefWeb. This network consisted of protein interactions to create an unweighted and undirected PPI network. These proteins was then filtered with their corresponding gene names taken from HGNC. Most of the proteins in the network did not have a corresponding gene name that HGNC could match, so over half of the interactions was removed.

The golden standard was created as explained earlier, with prostate cancer data from DisGeNET and DDPC. The network was uploaded into Cytoscape and populated with the prior scores.

12.1.2 Step 2 - Clustering the network

The Markov Cluster algorithm, MCL, was used to cluster the iRefWeb network with an inflation parameter of 1.8 and 200 iterations. This process took about 15 hours on the High-Performance Computing (HPC) instance Invitro at the University of Oslo. It used a total of 64 cores, which averaged at a 90-95% load for the majority of the time the cluster algorithm ran.

12.1.3 Step 3 - Ranking the clusters of the network

PRWP and MAA was ran on the network, choosing only the priors in the nodes given by the golden standard. The alpha value for PRWP was set to 0.3 together with 30 max iterations.

12.1.4 Step 4 - Exporting cluster ranks from Cytoscape and comparison with test data

The node table with the results from PRWP and MAA was exported to csv files, cleaned up with python scripts so it would form clusters containing information about which genes was in which clusters, the rank of the cluster and which genes in the cluster had a prior score or not.

The test data is in the form of a single column text file where each row contains the name of a gene. By traversing the ranked list of clusters, four categories were made from these test data genes.

The first two was named "test biomarkers" and "test candidates". These two have in common that they both list genes from the cluster in its column if it is contained in the test data set of genes. The "test biomarkers" are genes that both have a prior score and are contained in the test data set of genes. The "test candidates" are also in the test data set of genes, but they do not have prior scores.

The next two categories are "remaining biomarkers" and "remaining candidates". These two categories have in common that neither of them will list genes that was in the test data set of genes. The difference is that "remaining biomarkers" have prior scores, and "remaining candidates" have no prior scores.

Each cluster had their genes split into these four categories, represented as four columns in the next upcoming tables (tables: 12.3 12.4, 12.5, 12.6, 12.7, 12.8). The last column represents the rank of the cluster from either PRWP or MAA, depending on the table. From each of these tests, it is displayed the top 10 clusters, which had a combined amount of genes in either the "test biomarkers" or the "test candidates" column above 0.

As a comment to each table, the topmost ranked cluster in each table will be analyzed when it comes to each of the genes it contains. The genes will be assigned their functional classification according to PANTHERDB and DISEASE database[57, 71, 72].

12.2 Top 10 clusters from each ranking algorithm in Ranklust

Cluster rank	**Cluster number**	**Cluster score**	**Genes**
1	1136	1.0	LZTS1, CDC25C
2	690	0.918902810704	SOX9, SCX, CREB3L4
3	1004	0.758524309337	MMP14, MMP13
4	1364	0.69307698666	SSTR2, SSTR3
5	1227	0.689177108028	MLNR, GHRHR
6	721	0.671675886965	TAGLN, RNF14, TNFAIP3
7	1110	0.666772639835	ADAMTS5, TIMP3
8	1359	0.636059086342	GSTA1, GSTA2
9	527	0.585248909289	KLK14, KLK5, SPINK5, CAMP
10	1143	0.576075890388	LYPLA2, ITGA4

Table 12.1: Top 10 clusters from ranking the iRefWeb network with the golden standard priors and PRWP ranking algorithm - total amount of ranked clusters: 1340

Cluster rank	Cluster number	Cluster score	Genes
1	721	1.0	TAGLN, RNF14, TNFAIP3
2	1110	1.0	ADAMTS5, TIMP3
3	1364	1.0	SSTR2, SSTR3
4	1305	0.768245431105	CDK5R1, LMTK2
5	1136	0.725502913802	LZTS1, CDC25C
6	1359	0.725005246913	GSTA1, GSTA2
7	1004	0.720010369387	MMP14, MMP13
8	680	0.666666666667	PRTN3, F2R, F2RL1
9	690	0.666666666667	SOX9, SCX, CREB3L4
10	719	0.666666666667	AGER, RHOA, GMIP

Table 12.2: Top 10 clusters from ranking the iRefWeb network with the golden standard priors and MAA ranking algorithm - total amount of ranked clusters: 440

12.3 Prostate cancer genes manually curated from the movember group

Top ranked cluster ranked with PRWP and tested with Movember data (table: 12.3)

- TNFRSF11B
  - PantherDB subfamily - Tumor necrosis factor receptor superfamily, member 11b
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 4.2)
- THBS1
  - PantherDB subfamily - Thrombospondin-1
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 5.0)
- VEGFA
  - PantherDB subfamily - Vascular endothelial growth factor A
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 7.5)

Top ranked cluster ranked with MAA and tested with Movember data (table: 12.4)

Rank	Test biomarkers	Test candidates	Remaining biomarkers	Remaining candidates
13	TNFRSF11B	THBS1	VEGFA	-
19	-	F12	MMP12	-
25	F2R	-	F2RL1	PRTN3
28	-	RRM2	MAGEA3,MAGEA1	DNM1L,PGAM5,SCG3
29	CEACAM1	-	-	CLEC4M
31	ALOX15B	-	-	ERAL1
46	SMAD4	-	-	ZMIZ1
48	STAT6	-	ACSL3,IFI16	TRIM56,TMEM173,SLC39A14
53	RNASEL	IQGAP1	-	GSPT1,NPHS2
55	DPP4	-	VIP,GHRH,ADCYAP1	PYY,AVPR1A,GCG,GIP,TAC1,FAP,NPPB

Table 12.3: iRefWeb network ranked with PRWP and movember data - matched 254 test genes from movember data set out of 271 possible

Rank	Test biomarkers	Test candidates	Remaining biomarkers	Remaining candidates
8	F2R	-	F2RL1	PRTN3
12	TNFRSF11B	THBS1	VEGFA	-
14	STAT6	-	ACSL3,IFI16	TRIM56,TMEM173,SLC39A14
20	-	F12	MMP12	-
25	SMAD4	-	-	ZMIZ1
26	CD44	-	-	SCYL3
35	CEACAM1	-	-	CLEC4M
57	BIRC5	-	-	KCNJ6
58	ALOX15B	-	-	ERAL1
69	-	CRIP2	UXT,RELA,ALPL	NR1H4,NME5

Table 12.4: iRefWeb network ranked with MAA and movember data - matched 172 test genes from movember data set out of 271 possible

- F2R
  - PantherDB subfamily - Proteinase-activated receptor 1
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 3.3)
- F2RL1
  - PantherDB subfamily - Proteinase-activated receptor 2
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 2.4)
- PRTN3
  - PantherDB subfamily - Myeloblastin
  - DISEASE relation - Relation to several diseases, cancer is not among them

12.4 Curated prostate cancer genes from the COSMIC database

Top ranked cluster ranked with PRWP and tested with COSMIC data (table: 12.5)

- TNFAIP3
  - PantherDB subfamily - Tumor necrosis factor alpha-induced protein 3
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 3.3)
- RNF14
  - PantherDB subfamily - Ubiquitin-protein ligase
  - DISEASE relation - Relation to several diseases, among them specifically prostate cancer (z-score of 3.6)
- TAGLN
  - PantherDB subfamily - Transgelin
  - DISEASE relation - Relation to several diseases, among them

Rank	**Test biomarkers**	**Test candidates**	**Remaining biomarkers**	**Remaining candidates**
6	TNFAIP3	-	RNF14,TAGLN	-
11	BIRC3	-	-	BIRC2
12	CXCR4	-	-	CXCL14
17	-	DAXX	TGFBR3,ACVR2A	TCTEX1D4
18	RHOA	-	AGER	GMIP
24	-	ELN	EFEMP2,SOD3,FBLN5	-
37	AR	KMT2A	MAK,TSPY1	PKLR,HHAT
39	TOP1	-	-	RCVRN
44	WIF1	-	-	WNT11
46	SMAD4	-	-	ZMIZ1

Table 12.5: iRefWeb network ranked with PRWP and COSMIC data - matched 423 test genes from the COSMIC data set out of 580 possible

cancer (z-score of 4.2)

Top ranked cluster ranked with MAA and tested with COSMIC data (table: 12.6)

- TNFAIP3
  - PantherDB subfamily - Tumor necrosis factor alpha-induced protein 3
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 3.3)
- RNF14
  - PantherDB subfamily - Ubiquitin-protein ligase
  - DISEASE relation - Relation to several diseases, among them specificly prostate cancer (z-score of 3.6)
- TAGLN
  - PantherDB subfamily - Transgelin
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 4.2)

12.5 Proven prostate cancer genes that resulted in lethal outcome for the patient

Top ranked cluster ranked with PRWP and tested with Lethal prostate cancer data (table: 12.7)

- F2R
  - PantherDB subfamily - Proteinase-activated receptor 1
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 3.3)
- F2RL1
  - PantherDB subfamily - Proteinase-activated receptor 2
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 2.4)

Rank	Test biomarkers	Test candidates	Remaining biomarkers	Remaining candidates
1	TNFAIP3	-	RNF14,TAGLN	-
10	RHOA	-	AGER	GMIP
13	AR	KMT2A	MAK,TSPY1	PKLR,HHAT
14	STAT6,ACSL3	-	IFI16	TRIM56,TMEM173,SLC39A14
15	-	DAXX	TGFBR3,ACVR2A	TCTEX1D4
17	BIRC3	-	-	BIRC2
19	-	SUFU	PIAS1	-
25	SMAD4	-	-	ZMIZ1
29	SET	-	-	TAF1C
37	TOP1	-	-	RCVRN

Table 12.6: iRefWeb network ranked with MAA and COSMIC data - matched 277 test genes from the COSMIC data set out of 580 possible

Rank	Test biomarkers	Test candidates	Remaining biomarkers	Remaining candidates
25	F2R	-	F2RL1	PRTN3
28	-	RRM2	MAGEA3, MAGEA1	DNM1L, PGAM5, SCG3
31	ALOX15B	-	-	ERAL1
55	DPP4	-	VIP, GHRH, ADCYAP1	PYY, AVPR1A, GCG, GIP, TAC1, FAP, NPPB
79	BIRC5	-	-	KCNJ6
88	SERPINA3	-	KLK4	CTRC, GZMM, SGCD
91	-	CRIP2	UXT, RELA, ALPL	NR1H4, NME5
92	CCNB1	UBE2C	-	UBE3D
114	JAG1	-	-	NEURL1, CD46
120	-	CYB5A	CYP17A1, CYP3A4, CYP3A5, CYP2E1	CYP4F2, CYP4A11

Table 12.7: iRefWeb network ranked with PRWP and lethal prostate cancer data - matched 99 test genes form the lethal prostate cancer data set out of 157 possible

- PRTN3
  - PantherDB subfamily - Myeloblastin
  - DISEASE relation - Relation to several diseases, cancer is not among them

Top ranked cluster ranked with MAA and tested with Lethal prostate cancer data (table: 12.8)

- F2R
  - PantherDB subfamily - Proteinase-activated receptor 1
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 3.3)
- F2RL1
  - PantherDB subfamily - Proteinase-activated receptor 2
  - DISEASE relation - Relation to several diseases, among them cancer (z-score of 2.4)
- PRTN3
  - PantherDB subfamily - Myeloblastin
  - DISEASE relation - Relation to several diseases, cancer is not among them

12.6 Results from testing PRWP and MAA against several data test sources with prostate cancer relevant genes

	Data set	Hits	Possible hits	Percentage
PRWP	Movember	254	271	93.7
	COSMIC	423	580	72.3
	Lethal	99	157	63.1
MAA	Movember	172	271	64.5
	COSMIC	277	580	47.8
	Lethal	66	157	43.0

Table 12.9: Test data hits for PRWP and HITS

Rank	Test biomarkers	Test candidates	Remaining biomarkers	Remaining candidates
8	F2R	-	F2RL1	PRTN3
57	BIRC5	-	-	KCNJ6
58	ALOX15B	-	-	ERAL1
69	-	CRIP2	UXT,RELA,ALPL	NR1H4,NME5
79	-	RRM2	MAGEA3,MAGEA1	DNM1L,PGAM5,SCG3
92	CCNB1	UBE2C	-	UBE3D
105	JAG1	-	-	NEURL1,CD46
106	DPP4	-	VIP,GHRH,ADCYAP1	PYY,AVPR1A,GCG,GIP,TAC1,FAP,NPPB
109	SERPINA3	-	KLK4	CTRC,GZMM,SGCD
112	-	CYB5A	CYP17A1,CYP3A4,CYP3A5,CYP2E1	CYP4F2,CYP4A11

Table 12.8: iRefWeb network ranked with MAA and lethal prostate cancer data - matched 66 test genes form the lethal prostate cancer data set out of 157 possible

There are two of the topmost ranked clusters that occur in several of the test data sets across ranking algorithms. The first cluster consist of the following genes: F2R, F2RL1 and PRTN3. This cluster was the topmost ranked cluster by both PRWP and MAA, that contained genes from the Lethal prostate cancer data set. MAA also had it listed as the topmost ranked cluster, that contained genes from the Movember data set.

The second cluster consist of the following genes: TNFAIP3, RNF14 and TAGLN. This cluster was the topmost ranked cluster by both PRWP and MAA, that contained genes from the COSMIC test data set.

The cluster that contained F2R, F2RL1 and PRTN3 was the topmost ranked cluster in 3 out of the 6 cases with test data. The cluster that contained TNFAIP3, RNF14 and TAGLN occured 2 out of 6 times with the test data. The last topmost ranked cluster appeared in the Movember data set and was ranked by PRWP. It contained the genes TNFRSF11B, THBS1 and VEGFA.

Part IV

Discussion and conclusion

Chapter 13

Discussion

13.1 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.

13.1.1 Clustering

The clusters could to a greater degree have been filtered more strictly. The average cluster size was around 8.8 nodes (Table 10.1), 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 realistically compartmentalized into a protein complex, and can skew the results in either the direction of false positives or false negatives.

13.1.2 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 of PRWP 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 (Figure 10.1) that would have had a less descending trend of candidate cancer genes throughout the cluster ranks, it would be significant proof of the network structure having a bigger effect on the rankings than the priors scores.

13.2 Cross-validation between PRWP and MAA

There is clearly a trend in both PRWP and MAA (figures: 10.1 and 10.2). The difference between them being mainly the number of clusters ranked 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 are positioned in the rankings may be very different between the 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. Though, when comparing the cross-validation results between PRWP and MAA, PRWP comes out ahead by a margin in R-squared value.

13.3 Benchmarking through text mined genes from the DISEASE database

This demonstrates the main difference between PRWP and MAA (figures: 11.1 and 11.2). PRWP takes network structure into comparison as well as the prior scores, so it will rank genes without priors higher than in MAA. MAA focuses purely on the prior scores, and so the network structure of protein complexes are being completely ignored.

13.4 Benchmarking through manually knowledge curated genes from the DISEASE database

Benchmarking MAA with manually knowledge curated gene test data has uncovered its weakness of ranking prostate candidate cancer genes correctly (figures: 11.3 and 11.4). PRWP had the same trend, but the fit was not very accurate and the trend was negligible. Though, with such a small sample size of manually knowledge curated genes when compared to text mined genes, MAA is not necessarily excluded as a viable ranking algorithm for prioritizing network biomarkers in prostate cancer.

13.5 Benchmarking through genes from experiments from the DISEASE database

As with the manually knowledge curated genes, the experimental genes have a small sampling size (figures: 11.5 and 11.6). Also, the trends that would discredit Ranklust's ability to prioritize network biomarkers is not backed up by very high values of confidence in terms of R-squared values in the linear regression fits. In this specific case of benchmarking Ranklust, it is of a high probability that the cause of the trend is a single outlier in the plot.

13.6 Testing the ranks of PRWP and MAA with prostate cancer relevant genes

Both PRWP and MAA have demonstrated both the ability of identifying and prioritizing network biomarkers for prostate cancer according to text mined, manually knowledge curated and experimental data. Also, when tested against data from the movember project, the COSMIC database and proven lethal prostate cancer genes, they managed to prioritize network biomarker genes for prostate cancer (tables: 12.3, 12.4, 12.5, 12.6, 12.7 and 12.8).

PRWP has demonstrated throughout the benchmarks and tests to score marginally to significantly better than MAA in all of the tests and benchmarks. PRWP also managed to score a greater amount of clusters compared to MAA (see captions in tables 12.1,12.2).

Chapter 14

Conclusion

14.1 Future work

14.1.1 General improvements

I 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[39], which has the option of downloading directed networks of several types. STRING[19] 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 having Cytoscape do all of the heavy computations.

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 criteria.

Other ranking algorithms than PageRank could also have been used. There are numerous variations based off of this well known algorithm, for example NetRank and GeneRank[46, 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 prioritize network biomarers in prostate cancer. 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?

14.1.2 Minor features to complement Ranklust/clusterMaker2

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

14.1.3 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[67] is a Cytoscape App that can realise this idea. It supports import/export of data with a Neo4J[32] 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[30] of the CyNeo4J repository and simple user/password authentication was added.

Data communication

Data communication between the Neo4J database server and the Cytoscape instance is done through the CyNeo4J app to Cytoscape [67]. 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:

```
Request req = Request.Post(url)
        .addHeader("Authorization", auth64EncodedInfo)
        .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 [69]. It is possible that the performance increases with the new library [70]. 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. No convertion needs to be done in order to have Cytoscape understand this format since it already supports importing networks in this format. Through testing CyNeo4j as an alternative to use together with Ranklust, it was also discovered that GraphML is not only easy to use with Cytoscape, but it required substantially less memory and time while importing GraphML data into a Neo4j database instance. Though, to this day (2016-08-14), direct queries to a running Neo4J instance does not support GraphML.

14.2 Final results from identifying and prioritizing network biomarkers in prostate cancer

In the three unique clusters that had the topmost ranks in the results from testing PRWP and MAA against several data test sources related to prostate cancer, every gene in these clusters was related to cancer, except for 1, PRTN3. MAA ranked the cluster that contained PRTN3 higher than clusters where all of the genes were related to cancer, when checked in the DISEASE database. Comparing PRWP to MAA clearly shows that PRWP is better at identifying and prioritizing network biomarkers for prostate cancer (table 12.9). This conclusion is based on which of the two ranking algorithms that had the highest percentage of genes in the cluster ranks that was found in the test gene sets. The main difference between PRWP and MAA is, as mentioned earlier, that PRWP takes network structure into consideration when ranking nodes in the network, and as a result, manages to give meaningful scores to a greater number of clusters than MAA.

Bibliography

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

[2] Amazon.com. *Clean Code: A Handbook of Agile Software Craftsmanship*. Cite for proof of existence of the book - author of the book is Robert C. Martin. URL: http://www.amazon.com/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882 (visited on 20/05/2015).

[3] 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).

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

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

[6] 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).

[7] Jamie Brittain. *COLORRRS*. URL: http://hex.colorrrs.com/ (visited on 09/06/2016).

[8] 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.

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

[10] 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.

[11] 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.

[12] Peter J.A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski and Michiel J. L. de Hoon. 'Biopython: freely available Python tools for computational molecular biology and bioinformatics'. In: *Bioinformatics* 25 (11 Mar. 2009). URL: http://biopython.org/wiki/Main_Page (visited on 20/05/2015).

[13] 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).

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

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

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

[17] 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.

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

[19] A Francheschini, D Szklarczyk, S Frankild, M Kuhn, M Simonovic, A Roth, J Lin, P Minguez, P Bork, C von Mering and Jensen LJ. 'String v0.1: protein-protein interaction networks, with increased coverage and integration'. In: *Nucleic Acids Research (Database issue)* (Jan. 2013). URL: http://string-db.org/ (visited on 07/05/2015).

[20] 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.

[21] KA Gray, B Yates, RL Seal, MW Wright and EA Bruford. 'Genenames.org: the HGNC resources in 2015'. In: *Nucleic Acids Res.* (Jan. 2015). URL: http://www.genenames.org (visited on 23/02/2016).

[22] 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.

[23] Disease Systems Biology Group. *DistiLD - Diseases and Traits in LD*. URL: http://distild.jensenlab.org/about.html (visited on 13/08/2016).

[24] 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).

[25] Jeff Hanson. *Modularity in Java 9: Stacking up with Project Jigsaw, Penrose ans OSGi*. URL: http://www.javaworld.com/article/2878952/java-platform/modularity-in-java-9.html (visited on 20/05/2015).

[26] 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.

[27] 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).

[28] 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).

[29] Tim Hull. *Cytoscape 3.0 App Development*. URL: http://wiki.cytoscape.org/Cytoscape_3/AppDeveloper (visited on 08/05/2015).

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

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

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

[33] Techopedia Inc. *Design Pattern, Definition - What does Design Pattern mean?* URL: http://www.techopedia.com/definition/18822/design-pattern (visited on 20/05/2015).

[34] The National Cancer Institute. *Prostate-Specific Antigen (PSA) test*. URL: http://www.cancer.gov/cancertopics/types/prostate/psa-fact-sheet (visited on 11/05/2015).

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

[36] 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).

[37] 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).

[38] Jon M. Kleinberg. *Authorative Sources in a Hyperlinked Environment*. URL: https://www.cs.cornell.edu/home/kleinber/auth.pdf (visited on 05/07/2016).

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

[40] 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).

[41] 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).

[42] 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).

[43] Jason Montojo and GeneMANIA TEAM. *GeneMANIA*. URL: http://apps.cytoscape.org/apps/genemania (visited on 07/05/2015).

[44] Scooter Morris. *clusterMaker2 github repository*. URL: https://github.com/RBVI/clusterMaker2 (visited on 08/07/2016).

[45] Scooter Morris, Leonard Apeltsin, Aaron Newman, Jan Baumbach, Tobias Wittkop, Gang Su, Gray Bader and Tomar Abhiraj. *clusterMaker2 - Multi-algorithm clustering app for Cytoscape*. URL: http://apps.cytoscape.org/apps/clustermaker2 (visited on 22/05/2015).

[46] 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.

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

[48] Joshua O'Madadhain, Danyel Fisher and Tom Nelson. *JUNG - Java Universal Network/Graph Framework*. URL: http://jung.sourceforge.net/ (visited on 07/07/2016).

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

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

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

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

[53] L. Page, S. Brin, R. Motwani and T. Winograd. 'The PageRank citation ranking: Bringing order to the Web'. In: *Proceedings of the 7th International World Wide Web Conference*. Brisbane, Australia, 1998, pp. 161–172. URL: citeseer.nj.nec.com/page98pagerank.html.

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

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay. 'Scikit-learn: Machine Learning in Python'. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[56] 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.

[57] 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).

[58] 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.

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

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

[61] Sabry Razick, George Magklaras and Ian M. Donaldson. 'iRefIndex: A consolidated protein interaction database with provenance'. In: *BMC Bioinformatics* 9.1 (Sept. 2008). URL: http://irefindex.org/wiki/index.php?title=iRefIndex (visited on 07/05/2015).

[62] Sabry Razick, Antonio Mora, Paul Boddie, Katerina Michalickova and Ian M. Donaldson. 'iRefScape. A Cytoscape plug-in for visualizatoin and data mining of protein interaction data from iRefIndex'. In: *BMC Bioinformatics* (Oct. 2011). URL: http://apps.cytoscape.org/apps/irefscape (visited on 07/05/2015).

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

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

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

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

[67] 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).

[68] Raluca Tanase and Remus Radu. *Lecture 4: HITS Algorithm*. URL: http://www.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture4/lecture4.html (visited on 05/07/2016).

[69] Neo Technology. *The Neo4j Developer Manual v3.0*. URL: http://neo4j.com/docs/stable/rest-api-cypher.html (visited on 17/03/2016).

[70] Neo Technology. *The Neo4j Developer Manual v3.0*. URL: http://neo4j.com/docs/stable/rest-api-transactional.html (visited on 17/03/2016).

[71] Paul Thomas. *Panther Classification System*. URL: http://pantherdb.org/ (visited on 13/08/2016).

[72] Paul D. Thomas, Michael J. Campbell, Anish Kejariwal, Huaiyu Mi, Brian Karlak, Robin Daverman, Karen Diemer, Anushya Muruganujan and Apurva Narechania. 'PANTHER: A Library of Protein Families and Subfamilies Indexed by Function'. In: *Genome Research* (2003).

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

[74] Brian Turner, Sabry Razick, Andrei L. Turinsky, James Vlasblom, Edgar K. Crowdy, Emerson Cho, Kyle Morrison, Ian M. Donaldson and Shoshana J. Wodak. 'iRefWeb: interactive analysis of consolidated protein interaction data and their supporting evidence'. In: *Database* 16 (Sept. 2010). URL: http://wodaklab.org/iRefWeb/search/index (visited on 08/07/2016).

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

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

[77] D Warde-Farley, SL Donaldson, O Comes, K Zuberi, R Badrawi, P Chao, M Franz, C Grouios, F Kazi, CT Lopes, A Maitland, S Mostafavi, J Montojo, Q Shao, G Wright, GD Bader and Q Morris. *GeneMANIA*. July 2010. URL: http://www.genemania.org/ (visited on 07/05/2015).

[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).