

DESIGN AND IMPLEMENTATION OF THE WEB INTERFACES AND MIDDLEWARE FOR SYNERGY PROJECT

By

Juntao Zhuang

A Thesis Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
Major Subject: COMPUTER SCIENCE

Examining Committee:

Boleslaw K. Szymanski, Thesis Adviser

Mukkai S. Krishnamoorthy, Member

Bülent Yener, Member

Rensselaer Polytechnic Institute
Troy, New York

March 2017
(For Graduation May 2017)

© Copyright 2017
by
Juntao Zhuang
All Rights Reserved

CONTENTS

LIST OF FIGURES	iv
ACKNOWLEDGMENT	iv
ABSTRACT	v
1. INTRODUCTION	1
2. RELATED WORK	4
2.1 Multi-layer Networks	4
2.2 Collaboration Networks	4
2.3 Heterogeneous Information Networks	5
2.4 Ranking Methods	5
2.5 Reverse Proxy (Nginx, n.d.-a)	6
2.6 Locally Distributed Web Server	7
3. SYNERGY LANDSCAPE ARCHITECTURE	8
3.1 General System Organization	8
3.2 Data Storage	9
3.3 Queuing and Caching System	10
3.4 API Warpper	11
3.5 User Registration	12
3.6 Multi-layer Network Organization	12
3.7 Web Server Implementation	14
4. WEBSITE APPEARANCE	19
5. CONCLUSION AND FUTURE WORKS	23
LITERATURE CITED	24

LIST OF FIGURES

2.1	Architecture of a Locally Distributed Web Server (Cardellini, Casalicchio, Colajanni, & Yu, 2002)	7
3.1	System Organization of Synergy	8
3.2	Molecule TREM2 and its neighboring molecules, authors and publications	13
3.3	Flow Chart of doGet() and doPost() Function	17
4.1	Main Page	19
4.2	Waiting Page	20
4.3	Search Result Page	21
4.4	Registered User's Profile Page	22
5.1	Extended Web Server Organization	23

ACKNOWLEDGMENT

For the completion of my master thesis, first, I would like to express my gratitude to my advisor Professor Boleslaw Szymanski for his patience and guidance throughout my master research.

I would also like to thank Konstantin Kuzmin and all other members in our research group. The Synergy Project would not have been completed without the effort and co-operation from our group members.

Finally I would like to thank all my friends and family for supporting me throughout my study.

ABSTRACT

Recent biomedical researches usually involve novel combination of molecules. The goal of the Synergy Project is to facilitate biomedical researchers to find hidden relation among molecules and their potential collaborators. By gathering the data of publications from researchers, information such as authors, molecules, research papers title etc. are stored layer-by-layer in our multi-layer graph network. For each search focusing on a small set of molecules, the scientists whose research is most related to these molecules are listed by our algorithms after processing information within and across the network layers.

In this M.S thesis, we create a website for scientists to enter the list of molecules they are researching on. We also implement a middleware connecting this website with frontend interface and backend graph database. Another contribution of this thesis is a caching and queueing system implemented for concurrent query. For further increase of access, the middleware is scalable across multiple machines.

CHAPTER 1

INTRODUCTION

In recent years, there are calls for re-investment in biological research. Instead of just curing some diseases, the main focus shifts to improving patient care and their quality of life. Big data processing in biological research is one of the ways to approach this issue, revealing valuable research directions. The ELIXIR Project from Europe (Crosswell & Thornton, 2012) and BD2K from National Institutes of Health(NIH) (Margolis et al., 2014) are examples of applying big data for disease treatment. On the other side, when facing uncertainty, researchers usually prefer a "safe" search topic which follows main trend of research of the field.

Smalheiser, Perkins, and Jones (2005) illustrate two extreme situations of establishing collaboration: one is a totally active approach in which both sides are fully involved in the whole research process, taking equal responsibility and receiving equal credit. The other approach is mostly passive: one side acts as a supplier providing knowledge and resources and the other side is receiver who started the collaboration. In real world, most collaborations are between these two extremes. These collaborations could be productive yet are difficult to initiate due to uncertainty and communication overhead. For the solution, Smalheiser et al. (2005) propose a set of guidelines defining several levels of engagement of collaboration which can be a reference for both sides of collaboration. Similarly, our goal is to find possible collaborations in the spectrum between the two extreme situations from biological publications.

Since 2003, NIH funding has dropped 23%, resulting in fear of continuous budget cutting. Scientists are more conservative than ever about initiating innovative yet risky research (Alberts, Kirschner, Tilghman, & Varmus, 2014). On the other hand,

Portions of this chapter previously appeared as (Konstantin Kuzmin, Xiaoyan Lu, Partha Sarathi Mukherjee, Juntao Zhuang, Chris Gaiteri, Boleslaw K Szymanski, Supporting Novel Biomedical Research via Multilayer Collaboration Networks, Applied Network Science, 2016, 1:11)

journals and grant applications are now having a higher requirement for experimental support (Vale, 2015). Statistical data from NIH Research Project Grant(R01) reveals the trend: the number of recipients of R01 funding whose age is over 65 surpasses those under 35. The share of younger recipients’ funding from R01 also drop from 18% to 3% (Daniels, 2015). These numbers indicates that most of the research funding flows into existing projects. Only a small portion of the funding is supporting transformative research (Nicholson & Ioannidis, 2012).

Suggestions have been raised to encourage innovative biomedical research by modifying funding incentives: funding should be distributed over a broader range (Germain, 2015). When peer review cannot judge precisely about the quality of a project, random selection could be a good solution (Fang, Bowen, & Casadevall, 2016; Danthi, Wu, Shi, & Lauer, 2014). However, even if these suggestions are adopted immediately, it may still take as long as a whole generation to alter the current trend. In contrast, our Synergy tool can reverse the conservative trend efficiently without requiring researchers to take extra risk. In order to search rational innovative partnership, we dive into our multi-layer network generated by publications, authors and molecules, which has already been shown to gain meaningful scientific findings (Wuchty, Jones, & Uzzi, 2007). Currently, most of the connections from publications mapped to popular, well-studied molecules. Less popular ones are rarely linked (Rzhetsky, Foster, Foster, & Evans, 2015). Thus, trying to explore additional subjects will be more beneficial than keep focusing on hot topics. Furthermore, influential researchers are in favor of emerging topics (Uzzi, Mukherjee, Stringer, & Jones, 2013) and their novel relationship (Foster, Rzhetsky, & Evans, 2016). Generally, we adjust our molecular network to promote innovation while decreasing the risk: we recognize and list researchers whose research topics have intersections. Specifically, if molecule A and B has connection, we conclude it is likely that the collaboration between researchers of molecule A and B is going to be helpful.

With our molecular network, the selection bias of molecules and the cost of collaboration are both reduced. For example, if a person has negative findings with

a molecule related to cancer, his discovery may be helpful for another researcher whose major study is schizophrenia. Even if there is no common keyword in the publications written by these two researchers, we are still able to find the subtle relationship with our molecular network. Our solution not only utilize the idea of big data but also avoid historical bias. Along paths in molecular data, rational scientific communities can be constructed and potentials from various researches can be discovered.

In order to identify innovative collaborations, the molecular interaction network is merged with authorship network, forming a multi-layer network structure. Next, the path density between researchers are calculated as collaboration potential. When new data is added into the multi-layer network, distances among researchers will be updated, showing new possible collaborations.

The rest of this thesis contains the following content. Relevant previous researches and publications are described in Related Work. The main architecture of this project including sources of biomedical data and technical solution for our multi-layer network are in Synergy Landscapes Architecture. The section of Website Appearance shows the functionalities of our website. Finally, we summarize our work with a glimpse towards the future in Conclusion and Future Work.

CHAPTER 2

RELATED WORK

One of our major tasks is to set up collaborative paths among various types of information such as molecules, publications, authors, etc. (Kuzmin, Gaiteri, & Szymanski, 2016). We need to find a technical solution that can store and analyse our multi-layer network. The straightforward way is to store all these information as node and connections as edges in a graph. We will briefly go through several existing approaches to build the network and show our choice.

2.1 Multi-layer Networks

De Domenico et al. (2013) states that a multi-layer network is a network that consists of entities of different categories. Each type of entities is stored in a separate layer with edges connecting to nodes within the same or to other layers. Boccaletti et al. (2014) describes another type of multi-layer network in which different relationships are represented as layers. In our Synergy network, interactions among molecules are shown as edges within the layer of molecules.

2.2 Collaboration Networks

The earliest work on collaboration network was done by Newman (2004). Each scientist is represented as a node. An unweighted edge between two scientists represent their relationship as co-author. There are various studies and calculations about this type of network such as average distance between two scientists, distribution of edges, centrality measure, etc (Newman, 2001a; Newman, 2001b).

Bian et al. (2014) takes one step further into the study of collaboration network. In their scheme, edges between scientists are weighted with collaborative grants. Thus

Portions of this chapter previously appeared as (Konstantin Kuzmin, Xiaoyan Lu, Partha Sarathi Mukherjee, Juntao Zhuang, Chris Gaiteri, Boleslaw K Szymanski, Supporting Novel Biomedical Research via Multilayer Collaboration Networks, Applied Network Science, 2016, 1:11)

Sections 2.5 and 2.6 constitute original contributions of the author of the thesis.

the most important scientist in a co-authorship can be found with some centrality measures or ranking methods. Moreover, Random Walk with Restart(RWR) algorithm can be used to detect new collaborations. One of the weakness of this scheme is connections between scientists who do not collaborate but work on similar topics cannot be identified.

In Guimera, Uzzi, Spiro, & Amaral’s model of collaboration network (2005), there are two types of members in a team: newcomers and incumbents. Studying distribution of edges can tell us a team’s characteristic, such as diversity and degree of innovation. In such network, a large cluster is formed by small clusters, reflecting a large social and professional connection.

2.3 Heterogeneous Information Networks

Informally, a Heterogeneous Information Network(HIN) is a network consisting of different types of nodes and edges. HINs are widely used in all kinds of subjects and fields. The major difference between multi-layer network and HIN is HIN emphasizes the different relationships among nodes. In the past several years, works on major topics such as link prediction, ranking, entity matching, etc. have been done in the context of HINs.

2.4 Ranking Methods

We need to use ranking algorithms to recommend users with a list of potential collaborators. One of the popular ranking algorithm in this field is PageRank (Page, Brin, Motwani, & Winograd, 1999) with random walk: A surfer starts from a random node, traveling between nodes from their out-going edges. When the surfer goes to some sink, nodes with no out-going edges, the surfer jumps to some random node and continue traveling. After the desired number of rounds/steps are reached, the algorithm stops and the importance of each node is represented as the probability of being visited by surfer.

The major disadvantage of applying PageRank algorithm on our network is the

execution speed. It cannot guarantee to finished in limited time. Thus we use matrix factorization algorithm instead. This algorithm factorizes the adjacency matrix of a large graph. The resultant matrices store the attribute vector of one node. By calculating the dot product of these resultant matrices, we can compare the similarity of two nodes and use this information for recommendation. In practice, we can generate fast recommendation by storing intermediate result of the algorithm.

2.5 Reverse Proxy (Nginx, n.d.-a)

A reverse proxy is part of the server that redirects requests from clients or other parts of the server to an appropriate location. There are multiple advantages of deploying a reverse proxy in our network. Load balancing: in the scenario where multiple servers are running in the backend, the reverse proxy can track the running status of each server, distributing tasks efficiently thus maximize the overall capacity. When some of the web servers crash, reverse proxy can detect and skip the erroneous machines, ensuring the whole system does not get out-of-service. Security: reverse proxy is the only "bridge" between internal servers and clients to and from the Internet. The topology details and identity of internal servers are hidden from clients. To further increase security, ones can set up access control within reverse proxy or map it to a firewall server. Caching and accelerating: reverse proxy can be used to cache network traffic. It can also compress the data from inbound and outbound traffic to save bandwidth.

2.6 Locally Distributed Web Server

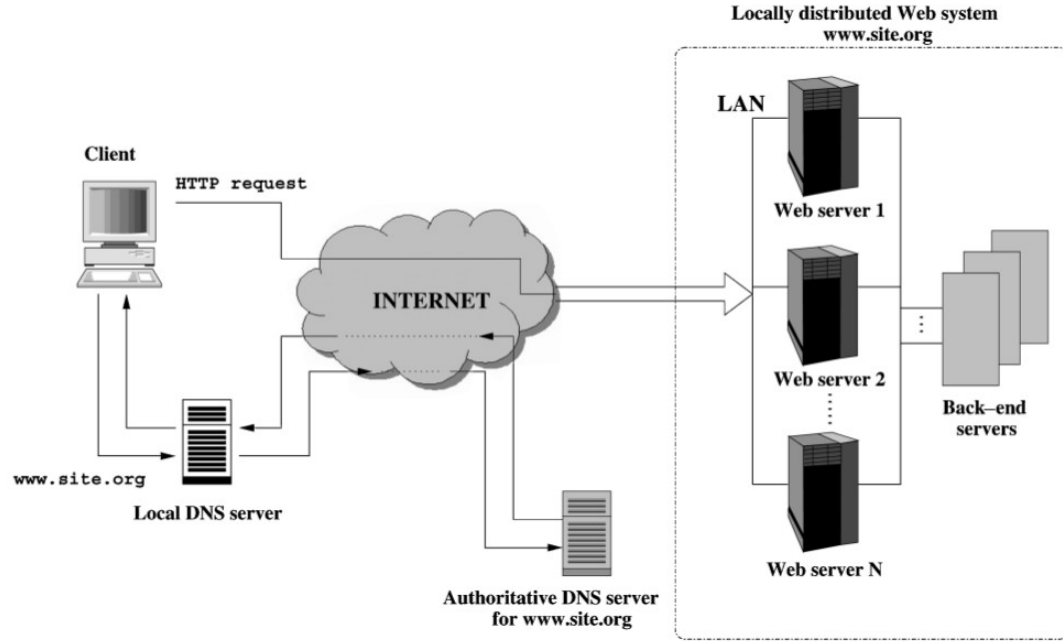


Figure 2.1: Architecture of a Locally Distributed Web Server (Cardellini, Casalicchio, Colajanni, & Yu, 2002)

Similar to Moore's Law (Moore, 1998) which states that for integrated circuit, the number of components is doubled every 18 months, the need for network bandwidth is also growing at a similar, if not higher, speed. Workload for many web services is far over the processing capacity of a single machine. A scalable web system across multiple machines is designed for handling the excessive workload. A simple model suggested by Cardellini et al. (2010) is to deploy multiple identical web servers with a single virtual interface holding the HTTP Address for access from Internet. Our project uses a reverse proxy as virtual interface.

CHAPTER 3

SYNERGY LANDSCAPE ARCHITECTURE

All parts of Synergy Project follow a modular design. All components including web interface, middle-ware, searching algorithms and database, are highly independent from one another. A self-defined, standardized protocol is used for communication among these components. Such architecture reduces the overall complexity and is scalable and extensible.

3.1 General System Organization

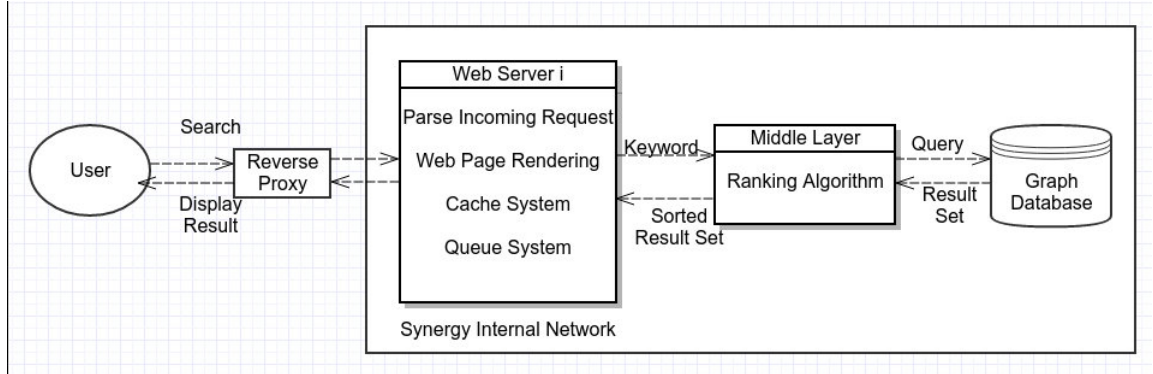


Figure 3.1: System Organization of Synergy

The structure of our Synergy network is shown above in Figure 3.1. The HTTP address of our website is hosted by the reverse proxy. When a user initiates a search, the request is sent to our reverse proxy first. Once the reverse proxy checks its validity, the URL of the request is re-written and redirected to our web server. The web server extracts keywords from the request and gives them to the middle layer

Portions of this chapter previously appeared as (Konstantin Kuzmin, Xiaoyan Lu, Partha Sarathi Mukherjee, Juntao Zhuang, Chris Gaiteri, Boleslaw K Szymanski, Supporting Novel Biomedical Research via Multilayer Collaboration Networks, Applied Network Science, 2016, 1:11)

Sections 3.1, 3.3, 3.4, 3.5 and 3.7 constitute original contributions of the author of the thesis, any portions of this chapter previously appeared in the above paper were written by the author.

with ranking algorithm. The middle layer communicates with the graph database and computes the result set.

After the computation is finished, the middle layer sends data back to the web server. The web server renders the data into human-readable web page and finally sends it back to user through the proxy.

We choose Nginx (Nginx, n.d.-b) as our reverse proxy. Nginx is a open sources server software written in C. Its stability, high performance and low resource consumption makes it suitable for doing simple tasks such as redirecting and forwarding requests. Besides, static contents such as pictures, CSS file, etc. from our website are also served by this reverse proxy, leaving our web server only for dynamically generated content. In real network traffic, a HTTP request may be split into several packages when being transmitted on Internet. The asynchronous, event driven model of Nginx has advantage in processing these fragmented packages, reducing the network traffic overhead. Lastly, Nginx can also be deployed for load-balancing which is necessary for the scalability of our project.

We use Tomcat (Apache Software Foundation, n.d.) as our web server. Tomcat is a open source implementation of Java Servlet from Apache Foundation. Since the programming language for the middle layer is also Java, data exchange between web server and middle layer is convenient: conversion of data structures is not need.

3.2 Data Storage

The selection for data storage scheme is critical for the Synergy project. Almost every function and operation involves the use of these data. We need to download the data from source and re-organize it into our multi-layer graph network. For query processing, we fetch all the relevant data in our local database before conducting our algorithms.

By referencing some studies (Miller, 2013; Angles, & Gutierrez, 2008; Vicknair et

al., 2010), we found that a graph database meets all of our requirements. It is able to naturally represent graph networks while providing efficient access and modification. We choose Neo4j graph database as our storage subsystem. It is a popular solution with active community of users. Neo4j provides a convenient programming model as well as a robust query model, supporting both native and RESTful call. The community edition of Neo4j is free of charge.

3.3 Queuing and Caching System

Solving a query is computational intensive and needs massive amount of memory. A typical query takes around one and a half minute to process and occupies around 800MB of RAM. It is unrealistic to execute all incoming queries in real-time. Thus we design a queuing and caching subsystem in order to relieve the server's pressure and enhance overall efficiency.

```

public LinkedBlockingQueue<String> workingQueue;
public ConcurrentHashMap<String, Status> cacheTable;

public enum Status { OK, UPDATING }

```

Data Structures and Enumeration Type

We implement a first-in first-serve task queue, processing queries one by one. Each task is identified by the list of molecule names, sorting in alphabetical order. Such as if the user searches for molecule "TREM2" and "INPPL1", the task identifier will be "INPPL1TREM2". The status of a task in working queue is synchronized with the record in cache table.

The HashMap "cache table" is for managing cache files. The key of this map is the same as task identifier in working queue while the value is either "OK", or "UPDATING." Every time we finish computing a query, the status of this query is marked as "OK" in cache table and the result with rendered web page is saved as a

single HTML file in the Cache Folder. The naming of the cache file is similar to the naming in task queue. Besides, we hash the name string and append a timestamp to the hashed string. Below is an example:

$$\begin{aligned} INPPL1TREM2 &\xrightarrow{MD5} fec45b0a9cb0b0ae6f9913d3b304555d \\ &\xrightarrow{AppendTimestamp} fec45b0a9cb0b0ae6f9913d3b304555d.1488442245.html \end{aligned}$$

After receiving user's query, the server firstly checks if cache is available and the cache is within the expiration time. If either of the condition is not met, the query will be added to working queue and the corresponding entry in cache table will be marked as "UPDATING".

Tomcat works in a multi-thread model: each incoming connection is handled by a thread. As the working queue and cache manager are public to all threads, we use the concurrent version of corresponding data structures to avoid racing condition.

3.4 API Warpper

In the early version of Neo4j Graph Database, the only way to access data remotely is through RESTful API. The query statement need to be packed into a JSON(JavaScript Object Notation) format ("Introducing JSON", n.d.) first. The packed JSON object need to be further packed into a HTTP POST request with authentication pair as well as some other information as header. In order to facilitate the calling process and enhance security, these procedures are warped into a single function. The input is simply the query statement. The output is a JSON Object in String obtaining a response sent from Neo4j.

```
public String executeQuery(String input) throw Exception;
```

Warped Database Query Function

3.5 User Registration

Beside search, a user can register to our website to enable some extra functionalities. Currently, a registered user can save lists of molecules he is interested in. When visiting with our website in the future, a search can be performed in one click.

3.6 Multi-layer Network Organization

The first layer is for molecules. Each node represents one molecule with a list of aliases. Edges indicate the already known relationship between the pair of molecules. Note that some molecules are indirectly linked to each other through other layers. In order to keep the flexibility of our network, we do not connect these molecules explicitly but these indirect relationships are still being considered while running our algorithms.

The second layer is for publications. Each node represents one publication. Currently, we use PubMed (PubMed, n.d.) database as sole source. When adding a new node to this layer, firstly we check the keywords and meta-data of the publication, finding all molecules that are mentioned and set up corresponding edges between molecules and this node.

The third layer is for authors. Each node represents one author. When adding publications into our multi-layer network, author information is extracted from publications and then insert into this layer and linked to the corresponding publications.

Right now our network has three layers while there is no edges within the layer of publication or the layer of author. This is because the relations between publications and relations between authors cannot be deduced from publication data exclusively. However, we have vertical edges across layers that connect all these entities together. Figure 3.6 shows the edges and neighbors from different categories that are related to molecule "TREM2".

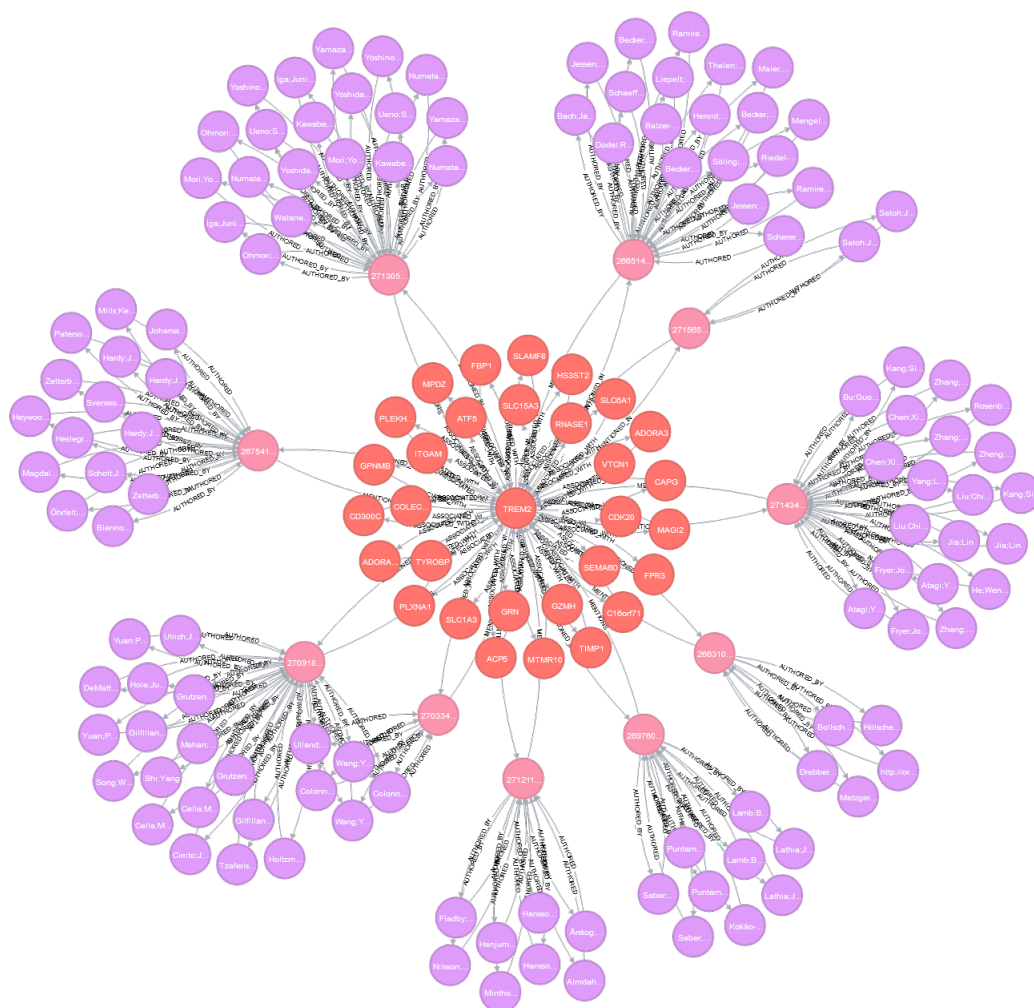


Figure 3.2: Molecule TREM2 and its neighboring molecules, authors and publications

Our multi-layer network is flexible and extensible. With additional sources of data, new layer can be created, providing extra search options for users. For example, we can add a layer of disease in which each node stands for one disease with cross-layer edges connecting to molecules. Thus we can provide a search option for searching researchers that work on the same disease through cross-layer edges.

3.7 Web Server Implementation

The entire software written for the web interface and middleware is organized into three parts: web pages displayed to the user interacting with the system, web server middleware and script for server setup and configuration.

Web page is implemented with HTML as well as JavaScript and CSS. The web interface is designed with HTML. The style of all types of elements in web pages such as button, panel, font, color, etc. is specified with CSS. Some complicated animations and pre-processing of input are done with JavaScript.

As the number of web pages grows, it is very usual that some maintenance problems may occur such as bad links, inconsistency of styles, etc. In order to avoid these issues, firstly, for the hyperlinks between web pages we use relative URL instead of absolute URL.

```
http://www.cs.rpi.edu/synergy/some_page.html
/synergy/some_page.html
```

Difference between absolute URL and relative URL

In the case when the folder directory changes, we can still reach the correct location by performing URL Rewrite with our reverse proxy.

We introduce modular design into the web page implementation. Instead of coding in the traditional way which is placing everything in a web page into a single file, we create separate files for each web page component. All web pages share a common CSS style sheet, JavaScript function list and navigation bar are stored as *styles.css*, *func.js* and *navigation_bar.html*. When user open a page of our website, these components are sent back to user and are combined in web browser.

For users query, input is firstly recorded with a HTML Form and then is translated into a HTTP GET request and is sent to our web server. The following is an example of searching molecule named TREM2 using normalized pagerank algorithm:

```
GET /synergy/post_query/?search_type=pagerank&
mol_name=TREM2&mode=1&mol_names= HTTP/1.1
Host: www.cs.rpi.edu
```

GET Request Header for the search

Note: mode=1 is a defined value for normalized sorting.

All the user's queries are sent with GET request while requests related to user, for example user login and user register, are sent with POST method, since POST method has a higher security level than a GET method. The following is an request for user registration. The user name is test03 and password is 789.

```
POST /synergy/post_query/ HTTP/1.1
Host: cs.rpi.edu

action=user_register&username=test03&password=789
```

POST Request Header for User Registration

We use cookie to record the status of a login user. The cookie consists of two parts, the first part is a unique encrypted identifier for the user and the second part is the expiration date of this cookie. The default timeout is 1800 seconds. If there is no action for 30 minutes, the user will be logout automatically. The following is an example of a cookie:

```
login_token=63a8e2de56aa98b60c1461366848f87;  
Expires=Fri, 13-May-2016 01:45:11 GMT; Path=/  

```

Cookie Given to a Login User

Every incoming request is warped as a class called `HttpServletRequest` in Tomcat. If the incoming request is a GET request, method `doGet()` will be invoked. If the incoming request is a POST request, `doPost()` will be invoked.

```
protected void doGet(HttpServletRequest request ,  
    HttpServletResponse response)  
    throws ServletException , IOException;  
  
protected void doPost(HttpServletRequest request ,  
    HttpServletResponse response)  
    throws ServletException , IOException;  

```

Two Entry Functions in Server Code

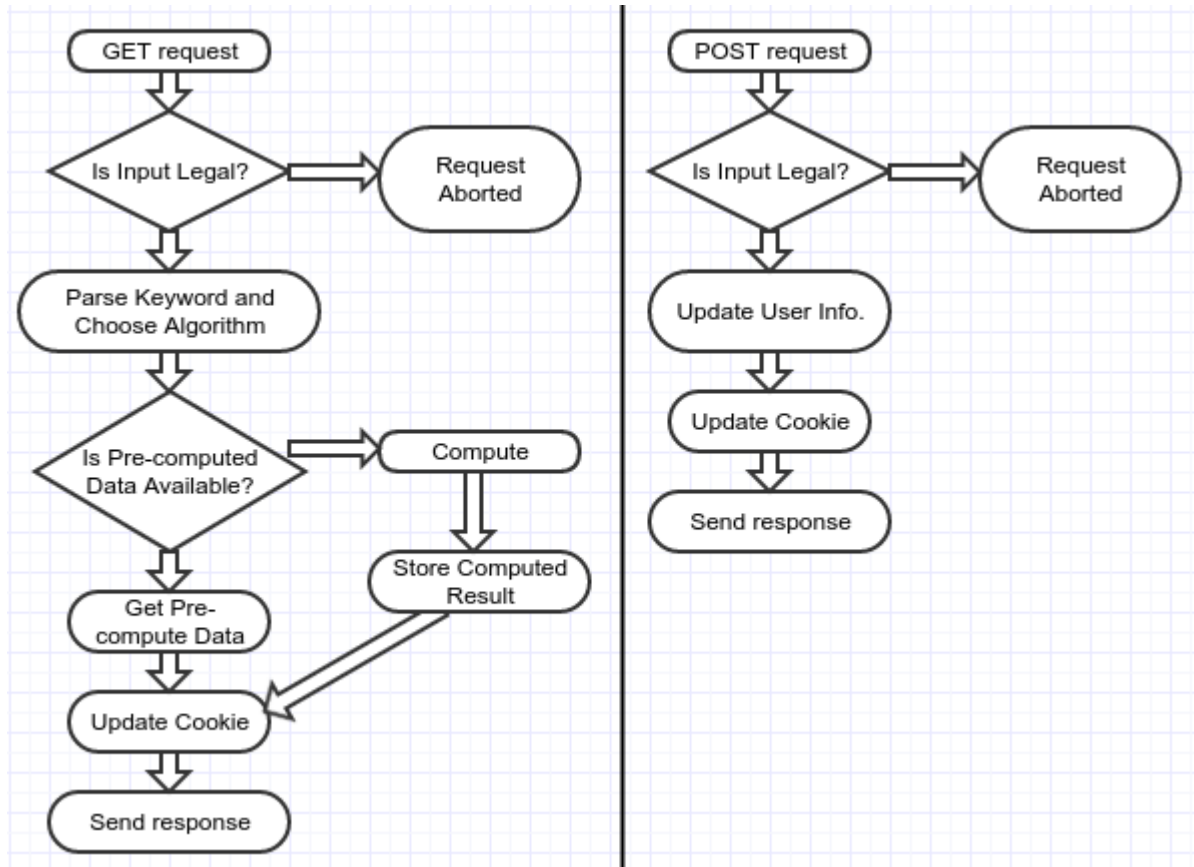


Figure 3.3: Flow Chart of doGet() and doPost() Function

The Nginx reverse proxy is the only bridge connecting the Internet and our web server. The following script specify the behavior of our proxy:

```

.....
listen 2826 default_server;
listen [::]:2826 default_server ipv6only=on;

location / {

    try_files $uri/ =404;
    root /var/www/html;
    index index.html index.htm;
}

location /post_query/ {
    proxy_pass http://127.0.0.1:8080/Synergy/q;
}
.....

```

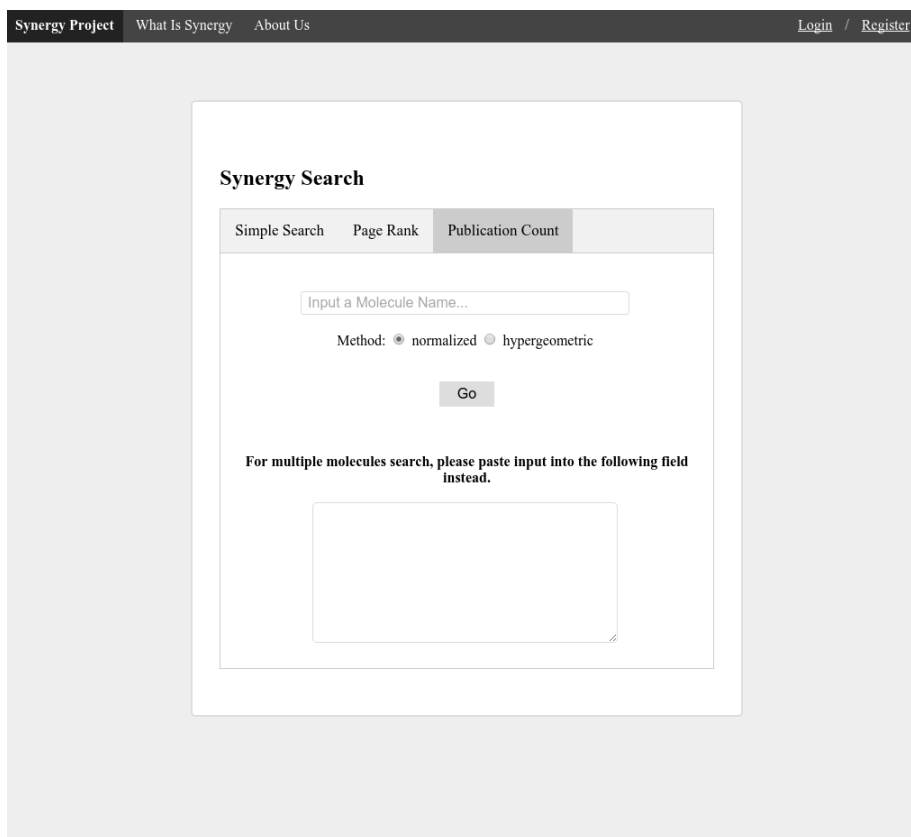
Configuration Script for Nginx(Part)

Since our server machine is physically located in RPI's internal network, the first part set up the connection between RPI's router(the Internet) and this proxy. The second part specifies the directory allowed to access. This directory contains *styles.css*, *func.js* as well as other public static resources. The third part sets up the connection between the user and our web server for processing dynamic requests such as queries, user logins, etc.

CHAPTER 4

WEBSITE APPEARANCE

At this stage, our website (Rensselaer Polytechnic Institute, n.d.) accepts a single molecule name or a list of molecule names as input. The output is a list of researchers ranked by the relevance of the input molecule name(s). According to our algorithms, names appearing in the front have higher possibilities to become potential collaborators with the user.



The screenshot shows the main page of the Synergy Search website. At the top, there is a dark navigation bar with links: "Synergy Project", "What Is Synergy", "About Us", "Login", and "Register". The main content area is titled "Synergy Search" and features three tabs: "Simple Search", "Page Rank", and "Publication Count". The "Simple Search" tab is active. Below the tabs, there is a text input field labeled "Input a Molecule Name...". Underneath the input field, there is a "Method:" label with two radio buttons: "normalized" (selected) and "hypergeometric". A "Go" button is positioned below the method selection. Further down, a text instruction reads: "For multiple molecules search, please paste input into the following field instead." Below this instruction is a large, empty text area for pasting multiple molecule names.

Figure 4.1: Main Page

This chapter constitutes original contributions of the author of the thesis, any portions of this chapter previously appeared as (Konstantin Kuzmin, Xiaoyan Lu, Partha Sarathi Mukherjee, Juntao Zhuang, Chris Gaiteri, Boleslaw K Szymanski, Supporting Novel Biomedical Research via Multilayer Collaboration Networks, Applied Network Science, 2016, 1:11) were written by the author.

There are two algorithms available: page rank algorithm and publication count algorithm. Each of them has two sub-options. For the page rank algorithm the result set can be sorted in either normalized or non-normalized way; for the publication count algorithm, the result set can be sorted in normalized or hyper-geometric way.

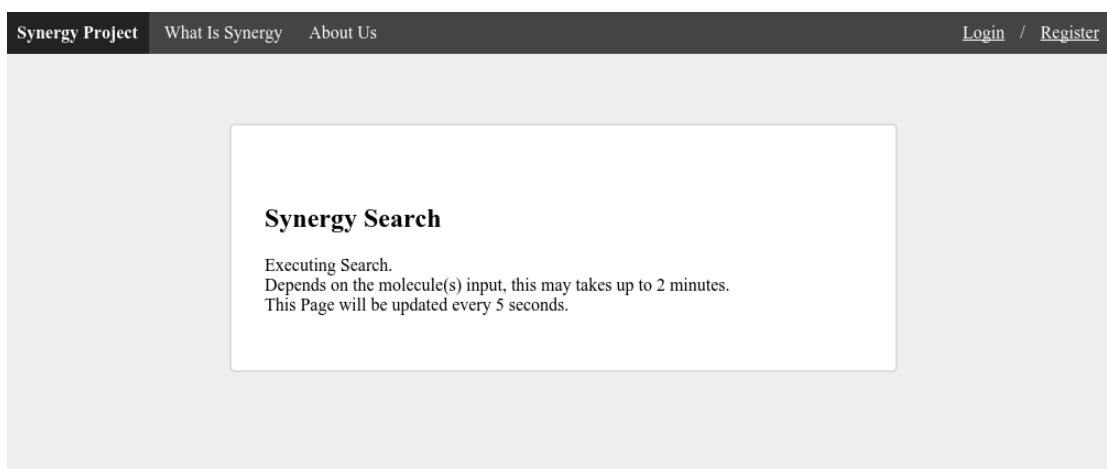


Figure 4.2: Waiting Page

Once the server receives user's request, user will be redirected to a waiting page. This page checks update with server every 5 seconds until the result is finished.

Synergy Project

What Is Synergy

About Us

Login / Register

Search Result for Molecule INPPL1: 181 related authors

Author	Katsuhiko Mikoshiba		
P-Value	3.262149E-14		
Publication Molecules	CSF1R(1), ITPR1(24), ITPR2(3), ITPR3(3),		
Number of Neighboring Molecules	4	Publication Count on Neighboring Molecules	29
Total Publication Count	30	Normalized Publication Count	0.97
Affiliation	Brain Science Institute, RIKEN, Wako, Saitama, Japan.		

Author	Jin Chen		
P-Value	3.543872E-9		
Publication Molecules	EPHA1(2), EPHA2(27), HGF(1), SHC1(1),		
Number of Neighboring Molecules	4	Publication Count on Neighboring Molecules	29
Total Publication Count	37	Normalized Publication Count	0.78
Affiliation	Department of Otorhinolaryngology Head and Neck Surgery, First Affiliated Hospital of Guangxi Medical University, Nanning 530021, China.		

Author	M Takahashi		
P-Value	2.29199E-8		
Publication Molecules	CSF1R(3), HGF(19), ITPR1(1), SHC1(4),		

Figure 4.3: Search Result Page

The screenshot above is the result set of INPPL1, using publication count algorithm.

Synergy Project What Is Synergy About Us [Logout](#)

Hi, test01

Stored Molecule List:

HCK, INPPL2, SORL1, TREM2,

Figure 4.4: Registered User's Profile Page

The above screenshot is the interface of editing list of interested molecules for a registered user.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

The Synergy Project composes a tool for finding innovative collaboration between researchers by investigating inside of our multi-layer network. It satisfies the public need for novel research topics while reduces risk for individual researchers.

For the future work, besides published data, it is also beneficial to include unpublished materials, such as preliminary finding, negative result on certain molecule, etc. into our multi-layer network. These data could be a valuable reference for researchers working in similar field. It is unlikely we can find such a source of data directly online. However, we can manually collect these information from our registered users by providing them an interface for upload.

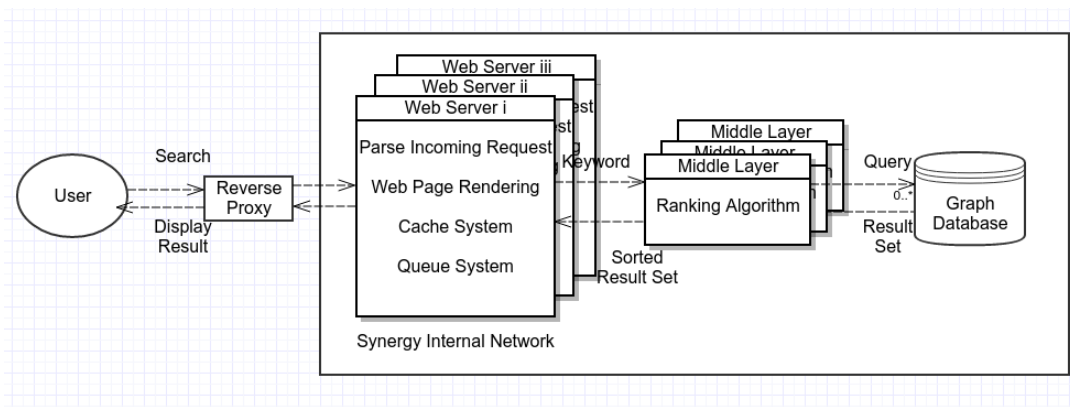


Figure 5.1: Extended Web Server Organization

For the current implementation, we have one instance of web server and the middle layer is integrated as part of the web server. Related setting of extending our web server into a local distributed server group is already finished as part of this thesis

Portions of this chapter previously appeared as (Konstantin Kuzmin, Xiaoyan Lu, Partha Sarathi Mukherjee, Juntao Zhuang, Chris Gaiteri, Boleslaw K Szymanski, Supporting Novel Biomedical Research via Multilayer Collaboration Networks, Applied Network Science, 2016, 1:11)

Figure 5.1 and the chapter text following this figure constitute original contributions of the author of the thesis.

and can be applied at once for future needed.

A new communication protocol called Bolt (Nigel Small and Stefan Plantikow, 2016) has been released for Neo4j graph database recently. Comparing to the RESTful API we are using right now, Bolt is a binary protocol and has direct support for Java. We expect that by switching the protocol, the Synergy system will benefit from 20 percent increase in the speed for processing query. Another good subject for future work is introducing some automatic stress testing tools for Synergy server. Such a tool would enable users to verify scalability and stability of the Synergy design and find potential bottlenecks of the system under heavy load. This tool would also enable the users to optimize the overall Synergy performance for a specific application.

LITERATURE CITED

- Crosswell, L. C., & Thornton, J. M. (2012). ELIXIR: a distributed infrastructure for European biological data. *Trends in biotechnology*, 30 (5), 241.
- Margolis, R., Derr, L., Dunn, M., Huerta, M., Larkin, J., Sheehan, J., ... & Green, E. D. (2014). The National Institutes of Health's Big Data to Knowledge (BD2K) initiative: capitalizing on biomedical big data. *Journal of the American Medical Informatics Association*, 21(6), 957-958.
- Smalheiser, N. R., Perkins, G. A., & Jones, S. (2005). Guidelines for negotiating scientific collaboration. *PLoS Biol*, 3(6), e217.
- Alberts, B., Kirschner, M. W., Tilghman, S., & Varmus, H. (2014). Rescuing US biomedical research from its systemic flaws. *Proceedings of the National Academy of Sciences*, 111(16), 5773-5777.
- Vale, R. D. (2015). Accelerating scientific publication in biology. *Proceedings of the National Academy of Sciences*, 112(44), 13439-13446.
- Daniels, R. J. (2015). A generation at risk: young investigators and the future of the biomedical workforce. *Proceedings of the National Academy of Sciences*, 112(2), 313-318.
- Nicholson, J. M., & Ioannidis, J. P. (2012). Research grants: Conform and be funded. *Nature*, 492(7427), 34-36.
- Germain, R. N. (2015). Healing the NIH-funded biomedical research enterprise. *Cell*, 161(7), 1485-1491.
- Fang, F. C., Bowen, A., & Casadevall, A. (2016). NIH peer review percentile scores are poorly predictive of grant productivity. *Elife*, 5, e13323.
- Danthi, N., Wu, C. O., Shi, P., & Lauer, M. (2014). Percentile ranking and citation impact of a large cohort of National Heart, Lung, and Blood Institute-funded cardiovascular R01 grants. *Circulation research*, 114(4), 600-606.
- Wuchty, S., Jones, B. F., & Uzzi, B. (2007). The increasing dominance of teams in production of knowledge. *Science*, 316(5827), 1036-1039.

- Rzhetsky, A., Foster, J. G., Foster, I. T., & Evans, J. A. (2015). Choosing experiments to accelerate collective discovery. *Proceedings of the National Academy of Sciences*, 112(47), 14569-14574.
- Uzzi, B., Mukherjee, S., Stringer, M., & Jones, B. (2013). Atypical combinations and scientific impact. *Science*, 342(6157), 468-472.
- Foster, J. G., Rzhetsky, A., & Evans, J. A. (2015). Tradition and innovation in scientists research strategies. *American Sociological Review*, 80(5), 875-908.
- Kuzmin, K., Gaiteri, C., & Szymanski, B. K. (2016, January). Synergy Landscapes: A multilayer network for collaboration in biological research. In *International Conference and School on Network Science* (pp. 205-212). Springer International Publishing.
- De Domenico, M., Sol-Ribalta, A., Cozzo, E., Kivel, M., Moreno, Y., Porter, M. A., ... & Arenas, A. (2013). Mathematical formulation of multilayer networks. *Physical Review X*, 3(4), 041022.
- Boccaletti, S., Bianconi, G., Criado, R., Del Genio, C. I., Gomez-Gardenes, J., Romance, M., ... & Zanin, M. (2014). The structure and dynamics of multilayer networks. *Physics Reports*, 544(1), 1-122.
- Newman, M. E. (2004). Coauthorship networks and patterns of scientific collaboration. *Proceedings of the national academy of sciences*, 101(suppl 1), 5200-5205.
- Newman, M. E. (2001). The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2), 404-409.
- Newman, M. E. (2001). Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical review E*, 64(1), 016132.
- Bian, J., Xie, M., Topaloglu, U., Hudson, T., Eswaran, H., & Hogan, W. (2014). Social network analysis of biomedical research collaboration networks in a CTSA institution. *Journal of biomedical informatics*, 52, 130-140.
- Guimera, R., Uzzi, B., Spiro, J., & Amaral, L. A. N. (2005). Team assembly mechanisms determine collaboration network structure and team performance. *Science*, 308(5722), 697-702.
- Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

- Nginx. (n.d.-a). Reverse Proxy Configuration. Retrieved March, 16, 2017 from <https://www.nginx.com/resources/admin-guide/reverse-proxy/>
- Cardellini, V., Casalicchio, E., Colajanni, M., & Yu, P. S. (2002). The state of the art in locally distributed Web-server systems. *ACM Computing Surveys (CSUR)*, 34(2), 263-311.
- Moore, G. E. (1998). Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86(1), 82-85.
- Nginx. (n.d.-b). nginx. Retrieved March, 18, 2017 from <https://nginx.org/en/>
- Apache Software Foundation. (n.d.). Apache Tomcat. Retrieved March, 18, 2017 from <http://tomcat.apache.org/>
- Miller, J. J. (2013, March). Graph database applications and concepts with Neo4j. In *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA* (Vol. 2324, p. 36).
- Angles, R., & Gutierrez, C. (2008). Survey of graph database models. *ACM Computing Surveys (CSUR)*, 40(1), 1.
- Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., & Wilkins, D. (2010, April). A comparison of a graph database and a relational database: a data provenance perspective. In *Proceedings of the 48th annual Southeast regional conference* (p. 42). ACM.
- PubMed. (n.d.). Retrieved March, 20, 2017 from <http://www.ncbi.nlm.nih.gov/pubmed/>
- Rensselaer Polytechnic Institute. (n.d.). The Synergy Landscapes Project. Retrieved March, 21, 2017 from <http://www.cs.rpi.edu/synergy/>
- Introducing JSON. (n.d.). Retrieved March, 21, 2017 from <http://www.json.org/>
- Nigel Small and Stefan Plantikow. (2016, June 22). A Deeper Dive into Neo4j 3.0 Language Drivers. Retrieved March, 22, 2017 from <https://neo4j.com/blog/neo4j-3-0-language-drivers/>