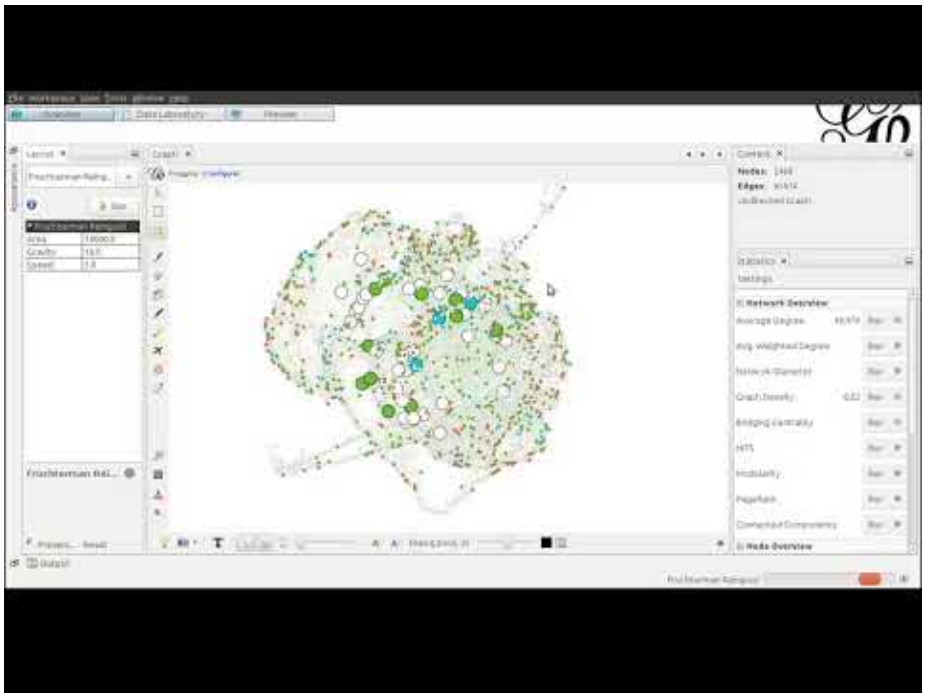
GENPPI: Ab initio protein interaction network generator

GENPPI is a software for creating complex networks of predicted proteins from tens or hundreds of genomes. The software is now available through an easy-to-use **Python interface** that automatically handles downloads and installation, making it accessible to researchers without Linux expertise.

I have a hands-on youtube video explaining how to use it. Click the below image to watch it.



Quick Start with Python Interface (Recommended)

The easiest way to use GENPPI is through the Python interface:

Installation

```
pip install genppi-py
```

Download Model and Sample Data

```
# Download the machine learning model (required for -ml option)
genppi-download-model
```

```
# Download sample data for testing
genppi-download-samples
```

Basic Usage

```
# Run GENPPI on sample data
genppi.py -dir samples

# Run with gene fusion analysis
genppi.py -dir samples -genefusion

# Run with machine learning (requires model download)
genppi.py -dir samples -ml
```

The Python interface automatically: - Downloads the appropriate executable for your operating system - Handles model file extraction - Provides sample data for testing - Works on Windows, macOS, and Linux

Your main limitation is the amount of RAM in your machine/server. For instance, using a conventional computer of 8 Gigabytes of RAM, the software can deal with at least 10 genomes, each possessing on average 2200 proteins. However, this same machine cannot process 50 genomes with an average of five thousand proteins (*Escherichia coli*). To achieve this last task well done, we need to compile GENPPI to use at least 32 Gigabytes of RAM or use the program version capable of storing the bulk data on disk instead of memory, the GENPPIDB executable. For now, I had compiled all versions available in the binaries folder to use different sizes of RAM.

SBCL includes the whole core of libraries for each executable, and it explains the size of the software in megabytes.

Manual Installation (Advanced Users)

For users comfortable with Linux commands who prefer manual installation:

Download Executables

You can download pre-compiled executables from the binaries folder or from our web interface at genppi.facom.ufu.br.

Download Model Files

For machine learning features, download and extract the model files: - model.7z.001 - model.7z.002
- model.7z.003

Extract these files in your working directory to enable the `-ml` option.

Basic Usage

GENPPI is a command-line tool. Calling the software without any arguments results in it printing all the possible parameters and their combinations. It also happens when starting the program with the -help option.

Usage Examples

Python Interface Examples (Recommended)

The Python interface provides the same functionality with easier setup:

```
# Install and setup
pip install genppi-py
genppi-download-model
genppi-download-samples

# Basic run with sample data
genppi.py -dir samples

# Advanced configuration
genppi.py -ppcomplete -expt fixed -w1 7 -cw1 4\
-ppdifftolerated 1 -pphistofilter -dir samples

# With machine learning
genppi.py -dir samples -ml
```

Manual Installation Examples

For users using manually downloaded executables:

(i) First of all, create a folder containing multi-fasta files of predicted proteins for your genomes. We will treat each file as a unique genome. I recommend you name these files significantly because a lot of reports will mention these names. I also suggest keeping the protein names as simple as possible and short. I have another program that you can use for this task: I called it valifasta. Valifasta tries to figure out a combination of strings identifying without redundancy all protein within a multi-fasta file. If such an assignment does not succeed, valifasta creates a numerical and sequential identification for each protein and uses it instead of the original proteins.

(ii) To run the program, there are a lot of parameters and combinations of those. I recommend you to start simple. Probably you will not reach out to a comprehensive interaction network on your first try. As you master the simple, do further steps. As a rule of thumb, a good interaction network should represent the majority of your proteins (number of vertices), possess at least two or three thousand interactions (number of edges), and on

average a few hundred (<200) interactions per protein. Some proteins can have thousands of connections, but these should not be in a representative number.

Starting from version 1.5, you can just call the program with the default parameters. For instance, considering all the fasta files of proteins, each representing one genome, are inside a target dir called 'mygenomes':

```
genppi -dir  mygenomes/
```

(iii) A configuration that I use to frequently is this one:

```
genppi -ppcomplete -expt fixed -w1 7 -cw1 4\
-ppdifftolerated 1  -pphistofilter -dir  mygenomes/
```

Meaning:

- "-ppcomplete" (default) is an excellent parameter to run at least once for each set of genomes. It does not restrict the number of interactions concerning
  Phylogenetic Profile (PP); it just let it go. Depending on a small number of genomes in your folder, the consequence is a massive and undesirable number of edges as a final result. The reason is that for a few genomes, very closely related, the majority of the phylogenetic profiles will be very conserved. Try to analyze related genomes but not necessarily very similar to diminish the number of possible expected interactions. The phylogenetic profile report only is created when we settle this parameter. If you decide to restrict the number of connections, do not use this parameter but limit the interactions. Optional parameters to limit edges are -ppiterlimit, -trim, and -threshold, just citing some of them. Check the -help options for a full list.

- "-expt fixed" (default) set the program for comparing neighbors genes at most "-w1" genes, and using "-cw1" as the smaller number of genes for concluding a Conserved Neighbourhood (CN). CN and PP are the primary methods to map significant connections in an interaction network. The counterpart of "-expt fixed" is "-expt dynamic -ws 3", for instance. It conducts a systematic expansion of a windows' limits on analyzing a CN. However, dynamic growth has time-consuming proportional to -ws value.

- "-ppdifftolerated" give you more interactions as a final result of the Phylogenetic Profile (PP) analyses. This parameter's integer value defines the level of tolerance to accept two PPs as similar between genomes.

- "-pphistofilter" set the GENPPI to be too restrictive when deciding the similarity between two proteins, but only during the PP analyses. The lack of this parameter can also create a much larger number of edges at the final interaction network.

- "-dir" is the location and name of the folder where you deposited the multi-fasta files to be processed.

Detailed Examples

Using Python Interface (Recommended)

The easiest way to test GENPPI is with the Python interface:

```
# Install and download sample data
pip install genppi-py
genppi-download-samples

# Run basic analysis
genppi.py -dir samples

# Run with machine learning (download model first)
genppi-download-model
genppi.py -dir samples -ml
```

Manual Installation Example

To test GENPPI manually, inside the GENPPI GitHub *test* folder, we provided subfolders containing two genomes: *Buchnera aphidicola* and *Corynebacterium pseudotuberculosis*. For *Buchenera* genomes, we have an instant test; It should execute speedily without the -ml parameter. The *Corynebacterium* genomes take a little bit longer to run.

Let's through a step-by-step using the *Buchenera* genomes:

1) Clone the genppi git repository to obtain the entire project on your computer. Having the git installed and using a command-line terminal, you can type:

   ```
   git clone https://github.com/santosardr/genppi.git
   ```

2) From a command-line terminal, change the cursor to the folder:

   ```
   cd genppi/test/Buchnera_aphidicola
   ```

3) There are two folders inside Buchnera_aphidicola, the *assemblies* and the *refer*. The *assemblies* contains the protein multifasta files for five genomes. A *grep* looking for the greater than (>) signal tell us the number of proteins per genome:

   ```
   grep -c '>' assemblies/*.faa
   ```

Resulting in:

```
assemblies/Ba_Ak.faa:559
assemblies/Ba_Bp.faa:553
assemblies/Ba_G002.faa:621
assemblies/Ba_Sg.faa:620
assemblies/Ba_Ua.faa:591
```

However, we recommend working with the *refer* folder. The *refer* contains links for these genomes instead of a raw copy for each file. The purpose is to avoid creating multiple copies of all genomes because GENNPI writes his results in the form of subfolders and files within the folder passed as a parameter at runtime. If you execute GENPPI twice in the same folder, GENPPI will overwrite previous results. Unfortunately, for MS Windows users, the *refer* folder is useless because there is no way to create links like Mac and Linux do; copy the assemblies folder or use the current one (GENPPI do not alter the fasta files).

4) Make a copy of the data source folder. Here the destination are the test1 and test2 folder. For Linux and Mac:

```
cp -r refer test1
cp -r refer test2
```

The "-r" should guarantee link preservation during copies, saving your disk space. Please, pay attention to the fact that offered links are relative to one folder up. For Windows (manual installation):

```
xcopy assemblies test1\
xcopy assemblies test2\
```

Note: Windows users are recommended to use the Python interface (`pip install genppi-py`) which handles these complexities automatically. 5) If the folder copying was successful, now it's time to execute GENPPI.

5.1) *Buchenera* genomes are one of the smallest genomes we have ever known; we expect a fast GENPPI execution and a few hundred edges for each genome. Let's make it happen:

```
genppi -dir test1/
```

Checking the numerical results:

```
wc -l test?/ppi-files/*.sif
```

Resulting in:

```
8140 test1/ppi-files/Ba_Ak.sif
1792 test1/ppi-files/Ba_Bp.sif
9414 test1/ppi-files/Ba_G002.sif
5742 test1/ppi-files/Ba_Sg.sif
6286 test1/ppi-files/Ba_Ua.sif
```

For standard GENPPI run (default parameters), the Ba_G002 has the more extensive interaction network with 9414 edges comprising 529 out 621 proteins. Just looking for this output on the screen, we cannot know about the number of unique proteins. However, using our countdots program, we can make such a count.

```
countdots test1/ppi-files/
```

Resulting in:

```
File    Nodes   Edges
Ba_Ak.dot   448 7917
Ba_G002.dot 529 9166
Ba_Ua.dot   450 6115
Ba_Bp.dot   217 1746
Ba_Sg.dot   430 5583
```

5.2) I will relax the GENPPI parameters to obtain more interactions in the final networks. When looking for conserved phylogenetic profiles, I meant to do such a relaxing parameter telling the GENNPI algorithms to accept as similar proteins those with at least 65% of identity (-ml). Additionally, I will ask for a dynamic expansion in the conserved neighborhood algorithm; it will start with a minimum window size of four to infer conservation (-ws 4). If the algorithm is successful for an initial *ws*, it will expand the window size by four units for subsequent well-success expansions. After all, this is our command:

```
genppi -ml -expt dynamic -ws 4 -dir test2/
```

Checking the numerical results:

```
wc -l test?/ppi-files/*.sif
```

Resulting in:

```
   8140 test1/ppi-files/Ba_Ak.sif
   1792 test1/ppi-files/Ba_Bp.sif
   9414 test1/ppi-files/Ba_G002.sif
   5742 test1/ppi-files/Ba_Sg.sif
   6286 test1/ppi-files/Ba_Ua.sif
  89570 test2/ppi-files/Ba_Ak.sif
  60897 test2/ppi-files/Ba_Bp.sif
  87826 test2/ppi-files/Ba_G002.sif
  86647 test2/ppi-files/Ba_Sg.sif
  80808 test2/ppi-files/Ba_Ua.sif
```

Our new interaction network for Ba_G002, compared to the one obtained with GENPPI default parameters, has 87826 edges comprising all 621 proteins or a nine-fold in the number of interactions.

Exploring GENPPI results

Once we have some networks created by GENPPI, what can we further do with those? Well, this answer will depend on your researching needs. However, the most common requirement is to visualize the results. Our purpose was to produce textual outputs easy to deal with and primarily used. That's the case for the DOT format, broadly used in the GEPHI tool, SIF extensively used by Cytoscape, and R plugins capable of reading Cytoscape and other tabular formats. **GENPPI outputs two data formats, DOT and SIF, for each genome (multifasta file) analyzed**. In the case of R as the visualizing/processing

tool for interaction networks, one should remember that the SIF format is a three-column file where the first and third columns are the interacting protein identifications. So, if one needs a Tab Separated Value (TSV) file or Comma Separated Value (CSV) file, import the SIF file in a spreadsheet program and export only the first and third columns of the SIF file. Another possibility is to run a command line like this one:

```
cut -f 1,3 test2/ppi-files/Ba_G002.sif
```

Resulting in:

```
"BUMPG002_CDS00574" "BUMPG002_CDS00573"
"BUMPG002_CDS00574" "BUMPG002_CDS00572"
(and so on)
```

To redirect the TSV for a file, just type:

```
cut -f 1,3 test2/ppi-files/Ba_G002.sif > Ba_G002.tsv
```

We will give you basic instructions to load the interaction networks created by GENPPI in some visual tools. Please, check the appropriate documentation about how to install each software. In advance, I can tell, you will need at least Java 11 installed in your machine.

GEPHI https://gephi.org/

A fast way to open a DOT file with GEPHI is by calling it in the command line.

```
gephi test2/ppi-files/Ba_G002.dot &
```

The GEPHI interface will ask you some basic questions about open a new workspace or append the data to an existing one. It also will ask you about the edges merge strategy. I use to sum several edges of the same pair of interacting proteins. For GENPPI, which could create three edges for a couple of interacting proteins, GEPHI will summarize in only one connection arrow. The *Ok* button will open the interaction network. Initially, the interaction network appearance is a mass, a completely aleatory distribution of vertices and connections. A more elegant view, for instance, is obtained by the options Window->Layout->Yifan Hu. After that, you can see clusters of vertices. Another possibility is to calculate statistics according to the topology. The Window->Statistics open a lateral menu of possible calculations. For instance, the *Network Diameter* option allows us to calculate the Betweenness Centrality and other measures. All the calculated data is available in a spreadsheet-like format in the *Data Laboratory* view. Figure 1 depicts the GENPPI interaction network to genome Ba_G002.

Cytoscape https://cytoscape.org/

After opening Cytoscape, generally via mouse action, you should access *File->Import->Networking from File*. Navigate to the folder used for GENPPI output and load the file "Ba_G002.sif". The *Style* menu (In general, located at the left

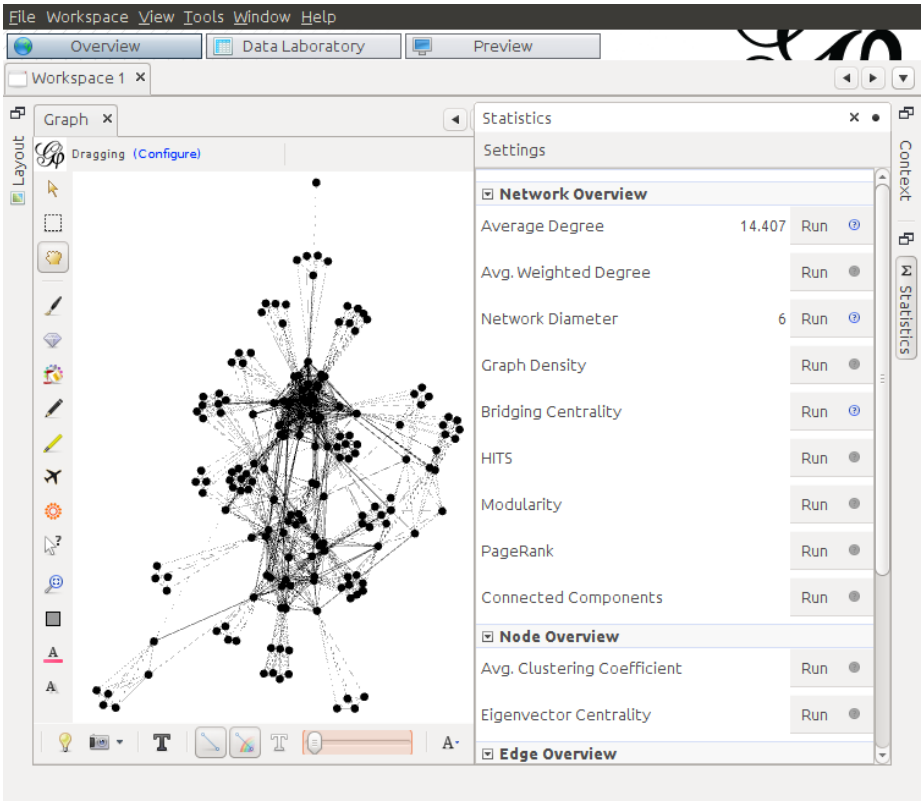


Figure 1: GEPHI sample for Ba_G002

lateral menu) allows you to customize your view. To make a graph similar to the GEPHI style, you can change the Shape and Width options to *Ellipse* and 30, respectively. The *Tool->Analyse Network* menu allows for several topology measures at once, including Betweenness Centrality. Running this option can be very time-consuming, depending on the number of edges and vertices in your network. Figure 2 depicts the Cytoscape interaction network to genome Ba_G002.

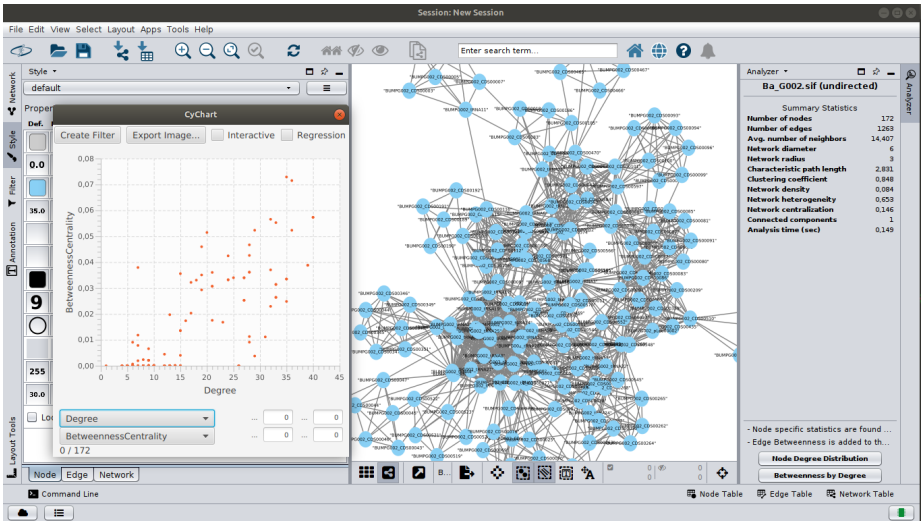


Figure 2: Cytoscape sample for Ba_G002

R https://www.r-project.org/

Installing R packages to handle protein interaction networks sometimes requires previous dependencies. For instance, on installing the package *graph* demands the installer package of the software provider, the *BiocManager*. This process takes several minutes of downloading and installing:

```
sudo R

if (!requireNamespace("BiocManager", quietly = TRUE))\
install.packages("BiocManager")
BiocManager::install("graph", version = "3.8")
q()
```

After finishing the installation process, I will use chapter 11 of "A Little Book of R for Bioinformatics!" written by Avril Coghlan from the Trust Sanger Institute, Cambridge, to demonstrate how to use GENPPI output in R. The main trick to compute interaction networks by GENPPI in R statistical software is creating a data frame with a GENPPI output. Fortunately, Mr. Coghlan provided us an **R function** (click the bold text to navigate), named *makeproteingraph*, to read

a two-column file as an R data frame. The only modification I had to perform here to load my data was at line 8. Instead:

```
protnames <- c(levels(proteins1),levels(proteins2))
```

I need to change it to:

```
protnames <- c(proteins1,proteins2)
```

Now, we can follow most of the examples of "A Little Book of R for Bioinformatics!". Let us start by reading the data. In Linux and Mac, we will create a TSV from a SIF file generated by GENPPI:

```
cut -f 1,3 test1/ppi-files/Ba_Sg.sif > Ba_Sg.tsv
```

Please, pay attention that now I'm using a smaller network created when I ran the default parameters of GENPPI. Now let us open the R statistical software in the same folder where you saved the TSV:

```
R
```

Please copy and paste the *makeproteingraph* function in a text editor and update line 8, as I pointed above. After the edit, copy and paste the edited function in the R console. Now, load the data file:

```
BaSg <- makeproteingraph("Ba_Sg.tsv")
```

After you navigate to the top of the web page of "A Little Book of R for Bioinformatics!" you will find several cool things to make with our network. For instance, this function shows the adjacent nodes to one specific:

```
adj(BaSg,"BUsg_616")
```

Resulting in:

```
$BUsg_616
[1] "BUsg_613" "BUsg_573" "BUsg_574" "BUsg_575" "BUsg_576"
```

We can show the degree distribution of the graph:

```
mydegrees <- graph::degree(BaSg)
sort(mydegrees)
```

Resulting in:

```
BUsg_018 BUsg_590 BUsg_595 BUsg_019 BUsg_065 BUsg_020 BUsg_591   BUsg20
1        1        1        2        2        2        2        2
(and so on)
BUsg_393 BUsg_394 BUsg_395 BUsg_396 BUsg_397 BUsg_607 BUsg_470 BUsg_471
10       10       10       10       10       10       10       10
BUsg_473 BUsg_613 BUsg_601 BUsg_604 BUsg_603
10       10       11       12       13
```

We can plot a histogram of the degree distribution:

```
hist(mydegrees, col="red")
```

The result plotting is found in Figure 3.

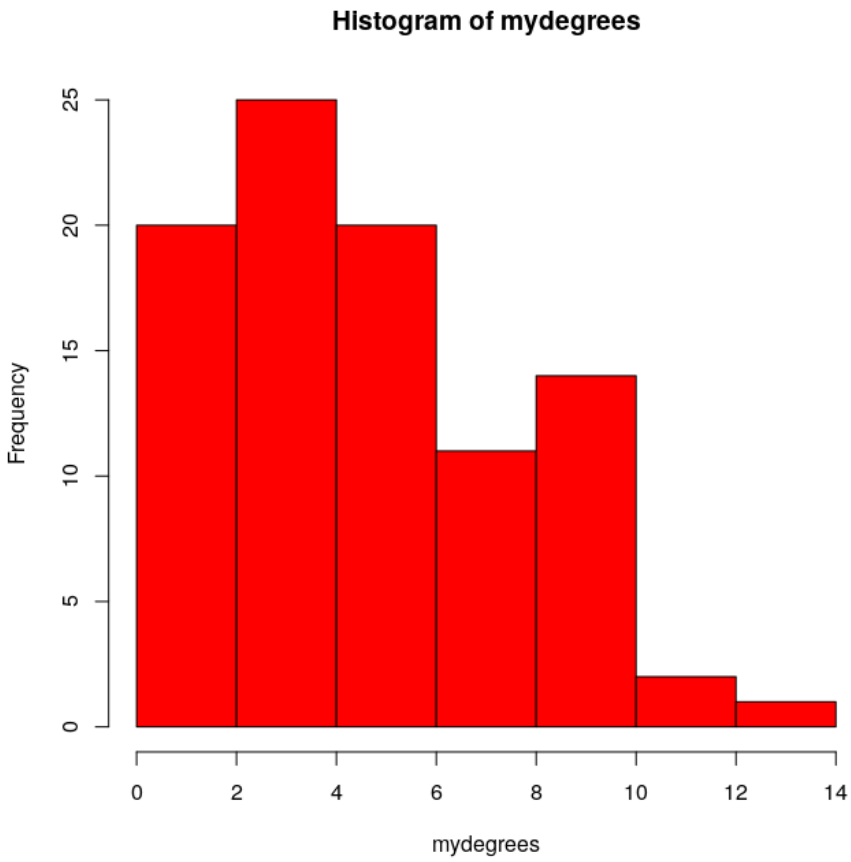


Figure 3: R sample 1

To Finish, we can plot the network interaction on the screen:

```
library("Rgraphviz")
mygraphplot <- layoutGraph(BaSg, layoutType="fdp")
renderGraph(mygraphplot)
```

The result plotting is found in Figure 4.

Report plotting

The github GENPPI *plotting* folder contains a **tutorial** explaining how we can create graphical results from the textual GENPPI reports. Please, check this

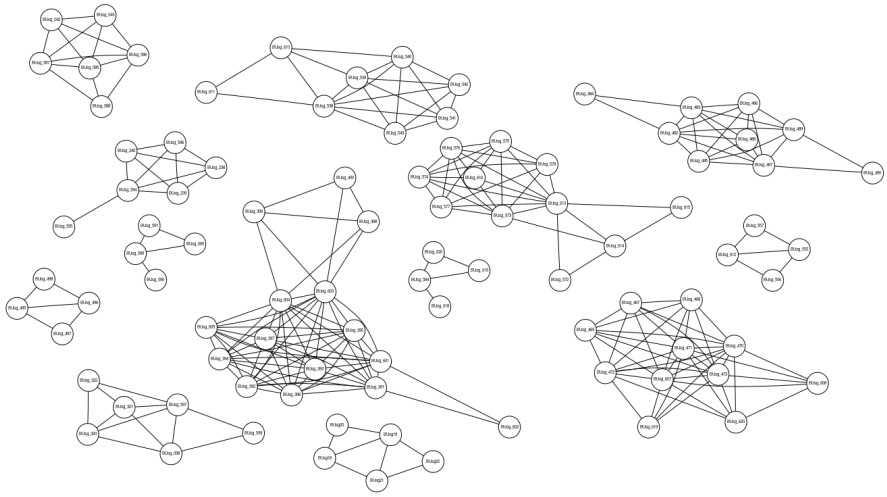


Figure 4: R sample 2

tutorial for further instructions and examples. In advance, I can tell you that we also created the web page **genppi.facom.ufu.br** to facilitate reports generation from files named 'report.txt' located in several GENPPI output folders.

Thank you for your patients.

Enjoy it.

Anderson Santos

santosardr@ufu.br