NetworkBuilder: Simple, lightweight Java 8 application for basic undirected network analysis

Lucy Vass

Introduction

Protein-protein interaction (PPI) networks are mathematical representations of transient or stable physical interactions between proteins[1]. Undirected PPI networks can enable the elucidation of the role of uncharacterised proteins or discovery of signalling pathways and multimolecular protein complexes[2]. In biomedicine, PPI networks can offer insight into disease mechanisms, allow discovery of new biomarkers and be used to assess potential off-target drug effects *'in silico'*[3].

Screening techniques such as high-throughput affinity purification are being combined with mass spectroscopy techniques, driving an increase in reliable, high coverage data of the interactome of many species[2]. Much of this data is available to researchers through protein interaction databases. For example, the primary EMBL-EBI hosted database IntAct[4] and the curated Human Protein Reference Database[5].

Tools for studying PPI networks

There are a variety of open-source tools available to biologists to study both directed and undirected PPI networks. These include non-programmatic tools GUI tools such as Cytoscape[6] or non-biologically focused tools such as Gephi[7], which allow non-expert users with no programming experience to carry out complex analyses. Cytoscape is particularly popular and has a wide-range of user curated plugins. Additionally, scripting packages are available for a variety of commonly used languages, for example NetworkX[8]. These are powerful, easy to automate and simple to integrate into existing pipelines.

Development of a lightweight analysis tool

NetworkBuilder is a simple, lightweight Java 8 application developed in Eclipse IDE (4.7.2) [9] which can be easily downloaded and implemented by a non-expert user through a GUI (graphical user interface) developed in Netbeans IDE 8.2 [10]. It aims to provide a quick summary of the network which can be used to inform preliminary or early stage investigations.

Methods

Functionality

NetworkBuilder takes a text file of undirected interactions. It removes duplicate interactions (eg. A:B and B:A) but allows up to one self interaction (eg. A:A). The degree of a node is defined as the number of interactions with other nodes, with self-interactions resulting in the addition of just 1 degree.

The program calculates and produces summary statistics, such as number of nodes and interactions in the network, average degree and hubs of the network. Hubs are defined as nodes in the network with the highest degree.

The program can produce two output files. Firstly, the user can choose to save the network as a text file which might be beneficial if they have added interactions manually. Secondly, the user can also save the full degree distribution as a text file. This can then be used to construct a histogram and differentiate between a random and scale-free networks.

Program structure

NetworkBuilder contains 5 classes, 2 of which encode the GUI. The remaining 3 classes are; Network, Node and Edge. Figure 1 gives an overview of the package structure and functionality.

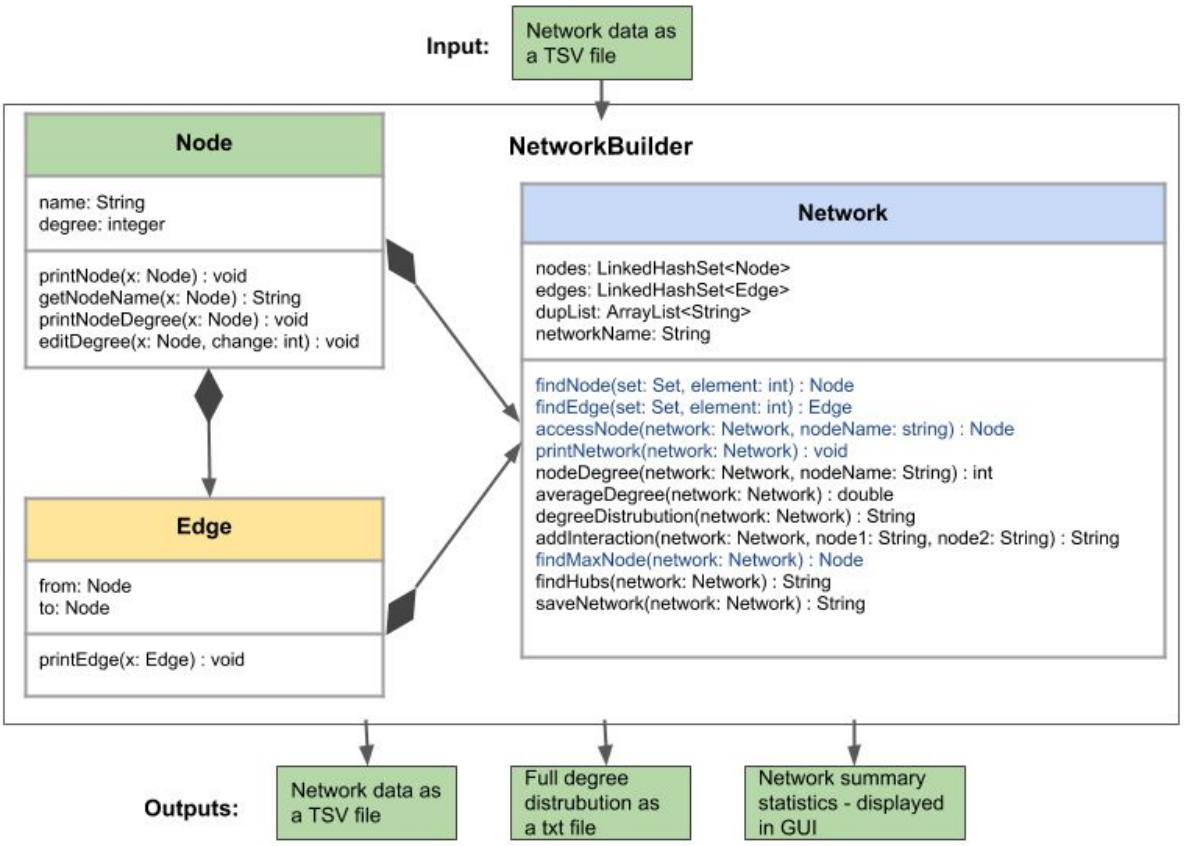

Figure 1: Diagram of classes and relationships in the NetworkBuilder package

The attributes and methods of each class as listed in the 1st and 2nd rows of their table. Arrows between classes represent their composition aggregation relationships. In the Network class, methods in blue are those which are used internally only (not directly actionable through the GUI).

Role of the classes

Node

The node class stores each node in the network. Each node has two attributes; name (a unique identifier such as UniProt accession number) and degree. The ‘equals’ and ‘hashCode’ methods of Node are overridden in order to prevent the adding to duplicate nodes (defined as those with the same name).

Edge

The edge class represents an interaction in the network. It takes two nodes as arguments. For every edge added, the degree of each node involved in the edge is increased by 1.

Network

The Network class is made up of 4 attributes. These are the Nodes and Edge objects stored in linked hashsets. In addition there is the network name stored as a string and array list (dupList) which is used to prevent the addition of duplicate interactions.

StartPageGUI

Launches the first GUI window in which the user enters the file path of the interactions text file.

NetworkViewGUI

Launches a second window for each network created. Within this window the user can produce output files, view summary statistics, add interactions and find out the degree of specific nodes.

Using NetworkBuilder

NetworkBuilder can be compiled and run from the command line with StartPageGUI.java as the main class. On opening, the first window prompts for the file path of the interactions in a tab-separated text file (See example input file ‘genes.txt’).

On creation of a network a second window allows the user to analyse and modify the network. Figure 2 shows this window.

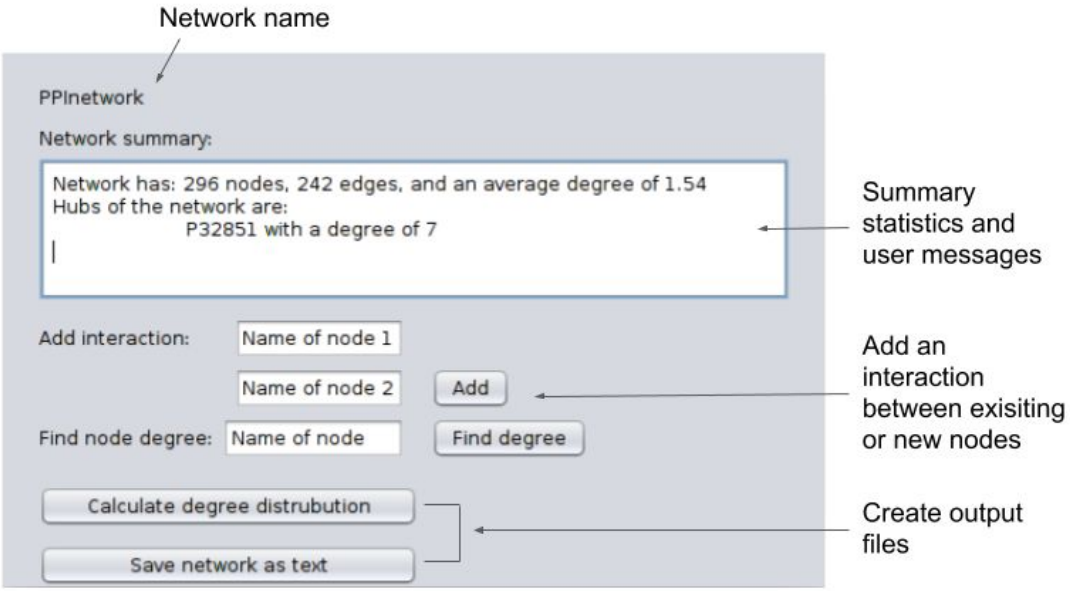

Figure 2: Network analysis and modification GUI window

Results

NetworkBuilder was used to analyse a large PPI network of mammalian proteins involved in neurotransmission and neurogenesis (See PPInetworks.txt). Summary statistics calculated by NetworkBuilder show that there are 296 nodes and 242 interactions or edges in this network. The average degree is fairly low at 1.54, and the hub of the network, UniProt ID P32851, has a degree of 7. NetworkBuilder was also used to produce a degree distribution. Figure 3 presents a histogram of this data.

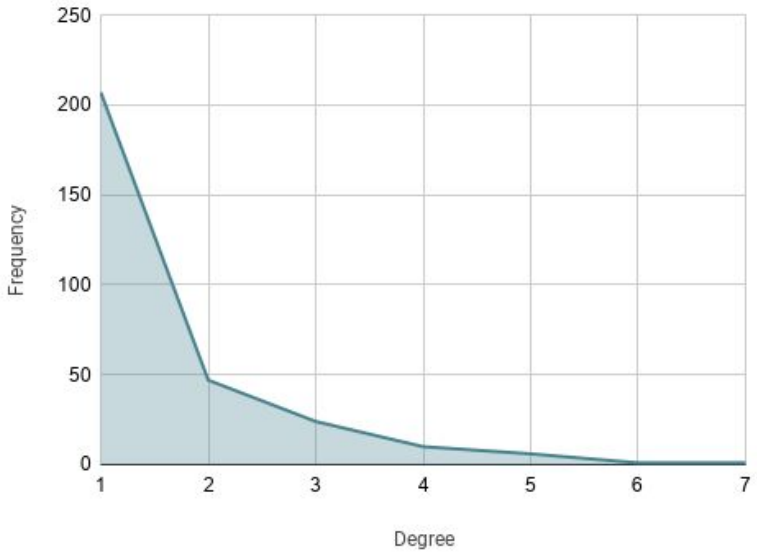

Figure 3: Histogram of the full degree distribution of example network
Frequency is the number of nodes in the network with the given degree

The hub of the network is P32851, corresponds to the rat protein Syntaxin-1A, which is involved in hormone and neurotransmitter exocytosis. The interactions of Syntaxin-1A in this network include Synaptosomal-associated protein 25 (P60881) and Syntaxin-binding protein 1 (P61765), both of which are involved in neurotransmission docking and release in the synapse. Some examples of nodes with a lower degree are Synapsin-3 (O70441) and Syntaxin-2 (P50279) which are also involved in the regulation of neurotransmitters[11].

Discussion and Conclusion

Insights from the neurotransmission network investigation

This small investigation has demonstrated the use of NetworkBuilder to quickly calculate summary statistics and gain an insight into a PPI network.

Using both the degree distribution and the function of looking up the degree of specific nodes, we have been able to acquire enough data to make some predictions about the network.

The degree distribution in Figure 3 shows that the network displays a scale-free distribution, in which there are few well-connected ‘hubs’ and many poorly connected peripheral nodes[1]. After investigating the function of the proteins in these two groups, it would be sensible to predict that in this network, the ‘hubs’ and nodes with high degree are central the regulation and transport of many neurotransmitters in the synapse. Whereas the periphery nodes in this network are likely to be responsible for the transport of a more specific subset of neurotransmitters.

Utility of NetworkBuilder and future development

NetworkBuilder is very simple and has few functionalities, however this might benefit some basic users. More powerful programs such as Cytoscape are highly demanding in terms of computing resources, and basic users may be overwhelmed with the choice of plugins and options.

In the future, the functionalities of NetworkBuilder could be expanded to allow for more in-depth network analysis. For example, the ability to calculate other topological indicators in addition to average degree, such as the average pathlength, clustering and network diameter[1].

The potential applications of Network Builder could be expanded by adding the option to analyse a directed network. Score-flow algorithms could also be easily integrated into the existing program.

The GUI could also be further expanded. For example generation of a histogram could be added to prevent the user from having to use an additional program to visualise the degree distribution. Visualisation of the nodes and interactions in a network diagram would also be a useful feature, especially for smaller networks. More information could be integrated into this diagram by using a ‘heatmap’ like colour coding system to highlight the high and low degree nodes.

References

1. GA Pavlopoulos et al, (2011). **Using graph theory to analyze biological networks.** *BioData Min.* 4: 10.
2. ebi.ac.uk/training/online/course/network-analysis-protein-interaction-data-introduction, 'Network analysis of protein interaction data: an introduction' course, accessed 15/01/18.
3. T Sevimoglu & KY Arga, (2014). **The role of protein interaction networks in systems biomedicine.** *Comput Struct Biotechnol J,* 11(18): 22–27.
4. ebi.ac.uk/intact/, IntAct Molecular Interaction Database, accessed 15/01/18.
5. hprd.org/, Human Protein Reference Database, accessed 15/01/18.
6. cytoscape.org/, Cytoscape: An Open Source Platform for Complex Network Analysis, accessed 30/01/18.
7. gephi.org, Gephi - The Open Graph Viz Platform, accessed 30/01/18.
8. networkx.github.io, NetworkX, accessed 30/01/18.
9. eclipse.org, Eclipse IDE for Java Developers, version: Oxygen.2 Release (4.7.2).
10. netbeans.org, Netbeans IDE version 8.2.
11. uniprot.org, UniProtKB used to search for and research various proteins by their UniProt identifiers, accessed 30/01/18.