WeakNet: A tool for studying the attack vulnerability of biological networks

Gayathri .S. Athreya

Submitted to the faculty of the University Graduate School in partial fulfillment of the requirements for the degree in
Master of Science - Bioinformatics
in the School of Informatics,
Indiana University
December 2005

ACKNOWLEDGEMENTS

First and foremost, I wish to acknowledge the unstinting support of my advisor **Dr. Filippo Menczer**. He has been a great advisor, providing encouragement and constructive criticism throughout this endeavor.

Through my interactions with him, I gained valuable experience in using my programming skills to solve biological problems. Most importantly, his ability to leverage my skills to satisfy the project requirements has been instrumental in fulfilling the requirements for my Master's thesis. I would also like to thank **Dr. Haixu Tang, Dr. Alessandro Vespignani, Dr. Alessandro Flammini and Dr. Matthew Hahn** for their useful suggestions on the project.

I thank my husband Krishna Muralidharan for supporting me throughout this endeavor. It is because of his continuous support and encouragement that I have been able to successfully complete my thesis. I thank him for his confidence in me and for all the support he has provided me in all these years and many more to come.

I have had the good fortune of working under excellent Professors here at the School of Informatics. My research work with Dr. Jean Camp and Dr. Javed Mostafa has provided me immense confidence and experience and I am greatly indebted to them for giving me a chance to work with them and supporting me throughout my graduate studies.

I would like to extend special thanks to **Dr. Gary Wiggins, Dr. Sun Kim, Dr. Mehmet Dalkilic, Dr. Marty Siegel** and other faculty members of the School of Informatics that have been instrumental in designing the curriculum for the bioinformatics program at Indiana University and for granting me permission to choose **Dr. Filippo Menczer** as a primary thesis advisor.

I would also like to acknowledge the timely help and guidance that I have received from **Linda Hostetter, Bob Konicek** and other staff members who have helped me all along and have gone out of their way sometimes to get things done for me.

Last, but not the least, I wish to acknowledge the moral support that my family and friends in India and here, in USA, have provided from time to time, thus making the past two years at Indiana University, full of wonderful memories.

Table of Contents

List of figures

1. Introduction

Complex networks are the backbone of complex systems and their degree of connectivity is neither regular nor random. Most complex networks of the real world have a number of characteristics and the ones mostly studied are the scale-free and the small world networks [11, 12]. Biological networks that represent biological processes like gene interactions, protein interactions and metabolic pathways fall under the category of scale-free networks in which the degree distribution of the nodes follows a power law [9]. Some nodes known as 'hubs' have very high degree while the other nodes have very low degree. The degree distribution of this type of network is inhomogeneous [3].

The degree of a node is defined as the number of edges that are adjacent to the node. In Fig. 1, the degree of C and D is 3, whereas all other nodes are of degree 2. Another quantity used to measure the importance of an edge or vertex is betweenness [4]. It is defined as the number of shortest paths between all pairs of vertices that go through a node or an edge. In Fig. 1, the edge CD has high betweenness [4] because it connects two clusters, ABC and DEF and has the maximum number of shortest paths passing through it.

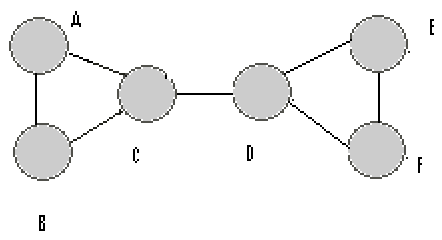


Fig. 1 Example of a network. The edge C D has high betweenness [4]

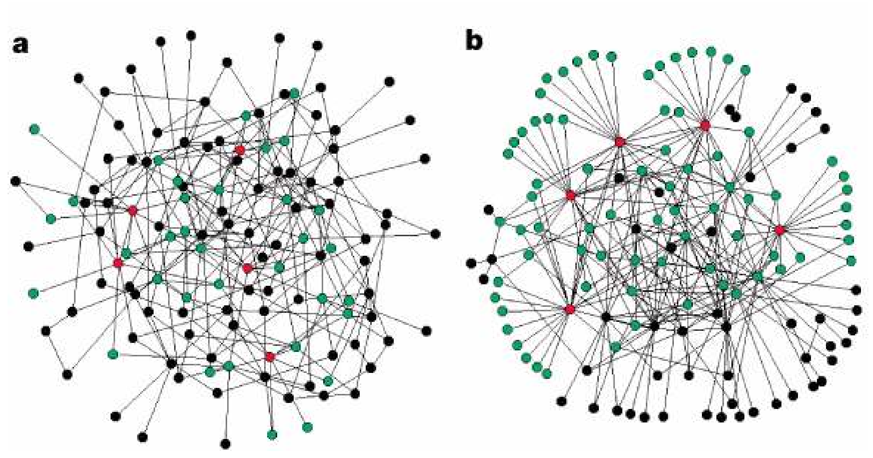


Fig. 2 (a) random (b) scale-free networks

Reference: Error and attack tolerance of complex networks – R. Albert, H. Jeong, and A- L Barabasi. (2002) Nature 406, 378 - 382

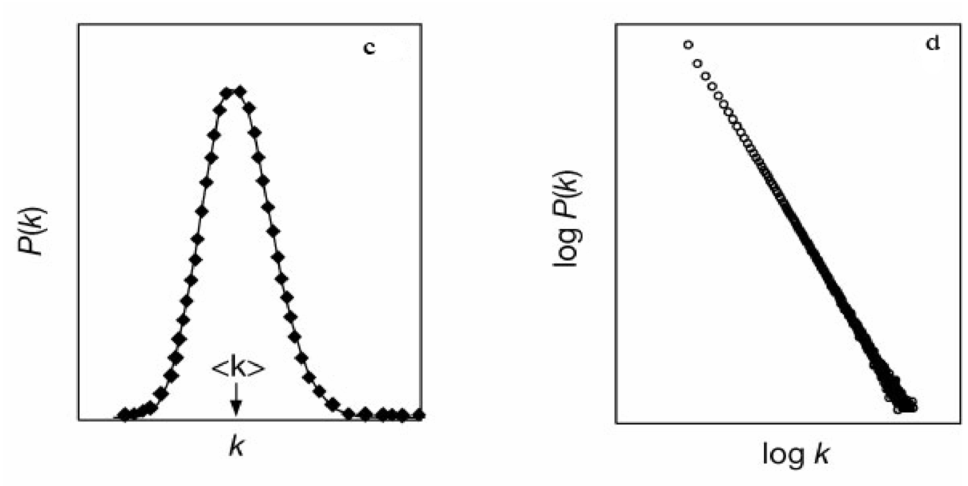


Fig. 3 The network connectivity can be characterized by the probability, *P*(*k*), that a node has *k* links. (c) The distribution of the probability *P(k)* follows a poisson distribution for random networks. (d) The distribution of the probability follows a power law for the scale-free networks. Reference: Jeong H, Tombor B, Albert R, Oltvai ZN, Barabasi AL. 2000. The large-scale organization of metabolic networks. Nature. 407(6804):651-4

The other type of complex network that is widely studied is the random network [1, 2, 3]. They are networks whose connectivity follows a Poisson distribution. Most of the nodes in the network have a similar connectivity and this distribution is homogeneous.

One of the main characteristics that differentiate these two types of networks is attack vulnerability [1, 2]. Attack vulnerability [9] denotes the decrease in performance of a network due to the selected removal of nodes or edges [2]. In the context of metabolic networks, it may mean the prevention of a metabolic reaction due to the removal of an enzyme or substrate.

Studying the attack vulnerability of networks is very important for identifying the weak/strong ‘links’ in the network. Subsequently, this knowledge can be used to

(1) Protect the network from outside attacks [2]

In the case of biological networks, these attacks could be from pathogens.

(2) Engineer more robust networks [2]

This pertains to other kinds of networks like computer networks which can be engineered such that its vulnerability can be lowered.

(3) Select the optimal target in the network for an efficient attack.

These nodes or edges could form effective drug targets.

In the context of metabolic networks, these vulnerable points can serve as potential drug targets. The systematic approach to studying the attack vulnerability of complex networks has been developed and applied to several complex networks [1, 2]; however metabolic networks have not been studied using this approach yet. While researchers have mainly used the degree values of the nodes or edges to attack networks [1], betweenness [4] based strategies haven’t been utilized in studying biological networks.

The algorithms used to study attack vulnerability can also be applied to protein interaction networks to cluster the network into functional units with the help of which functions of proteins with unknown function can be predicted based on the way they cluster [6]. Hence a tool that will aid in studying the attack vulnerability of biological networks would be very useful for further analysis and understanding of the topology of these types of networks. In the remaining part of the thesis, we will describe such a tool.

2. Previous work

Attack vulnerability of networks has earlier been studied by many researchers and two research groups, Reka Albert et al [1] and Petter Holme et al [2] have done pioneering work in this field.

Reka Albert and her group [1] studied the error and attack tolerance of two kinds of networks, the scale-free and the random networks. In order to study the attack tolerance, they artificially created the networks such that the connectivity of the random network followed a Poisson distribution and that of the scale-free network followed a power law. They removed a fraction of nodes from the networks and studied the effect of this removal on the diameter. They defined diameter as the average length of the shortest path between any two nodes in the network. The nodes were selected based on the initial degree values of the nodes and removed according to the decreasing value of degree.

They simulated error by randomly removing nodes from these two networks and made the following observations.

(1) Due to the homogeneity of the random network, all nodes contribute equally to the diameter, so the removal of each node causes the same effect.

(2) Because of the extremely inhomogeneous degree distribution, many nodes have only a few links and nodes with small connectivity will be selected with a much higher probability and these removals do not change the diameter drastically.

By simulating attacks, they made the following observations.

(1) When under attack by an informed agent that targets nodes with high degrees, the random network does not show any difference whether the nodes are select randomly or based on the decreasing values of degree.

(2) Due to the small number of nodes with very high connectivity, the scale-free networks show a drastic change in diameter when the nodes are targeted for removal. The diameter almost doubles when 5% of the nodes are removed.

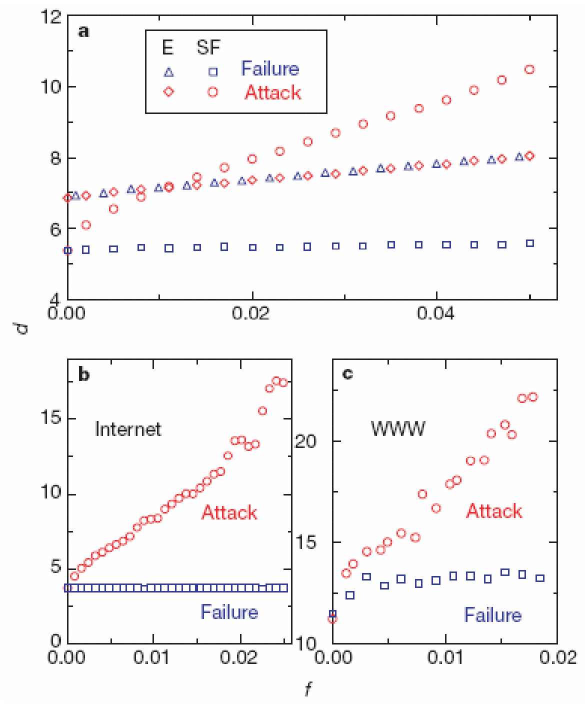


Fig. 4 shows the effects of error and attack on the scale-free and random network models and real network.
Reference: Error and attack tolerance of complex networks – R. Albert, H. Jeong, and A- L Barabasi. (2002) Nature 406, 378 – 382

Petter Holme et al [2] studied the attack vulnerability of real world network and some network models. Among the real world networks were the social network and internet traffic network and the network models consisted of the Erdos – Renyi (ER) random network [10], Watts – Strogatz [11, 12] model of small world network, scale-free network [1], and the clustered scale-free network [2]. The clustered scale-free network is a modification of the scale-free network and exhibits high clustering [2]. They simulated attacks by using algorithms that removed nodes and edges based on degree and betweenness [4]. They used eight algorithms for this purpose. Four of these were node based and four of these were edge based. The algorithms they used were the following.

(1) ID Vertex – This algorithm removes vertices based on the initial degree values of the vertices.

(2) IB Vertex – This algorithm removes vertices based on the initial betweenness [4] values of the vertices.

(3) RD Vertex – This algorithm removes vertices based on the degree values but recalculates the degrees of the nodes after each removal.

(4) RB Vertex – This algorithm removes vertices based on the betweenness [4] values but recalculates the betweenness [4] after each removal.

The above algorithms were also used to target edges and hence 4 more types of algorithms were used. However, in order to define edge degree, the researchers considered different equations for the degree based on the adjacent vertices such as,

(1) ke = kv * kw
(2) ke = kv + kw
(3) ke = min(kv, kw)
(4) ke = max(kv, kw)

where ke is the edge degree, kv is the degree of one of the adjacent vertices and kw the degree of the other. In order to find the most relevant definition, they compared these values with the corresponding betweennness [4] values of the edges and found that ke = kv * kw had the most correlation with betweenness [4]. Hence they defined edge degree as the product of the degree of the adjacent vertices.

They measured the attack based on a number of criteria like the size of the largest connected graph, path length etc. On using these algorithms, they found that

(1) Among edge attacks, the repeated betweenness [4] attack was the most efficient.
(2) The change of network structure during removal is substantial and hence the repeated calculation strategies are more efficient.
(3) The ER model [10] or the random network model was found to be the most robust among all networks.
(4) None of the network models showed similarity in behavior with the real world networks.

While both the research groups studied attack vulnerability, Reka Albert et al [1] confined their research to studying degree based attacks and ignored the effects of attacking targets based on betweenness [4]. Similar approaches were adopted by Petter Holme's group [2] but they did not study biological networks and only concentrated on different network models, social and computer networks. In this work, we design a tool to study the attack vulnerability of biological networks using the algorithm described by Petter Holme et al [2] and examine the performance of our approach by approach by applying it to real world networks like biological networks.

3. Current work – WeakNet

3.1 Description

In this work, we have designed and created a tool called WeakNet for studying the attack vulnerability of biological networks. However, it can also be used to study other types of networks like social networks, computer networks etc. WeakNet is a web based tool written in Perl with a MySQL database backend and is currently hosted at http://biokdd.informatics.indiana.edu/~gathreya/thesis

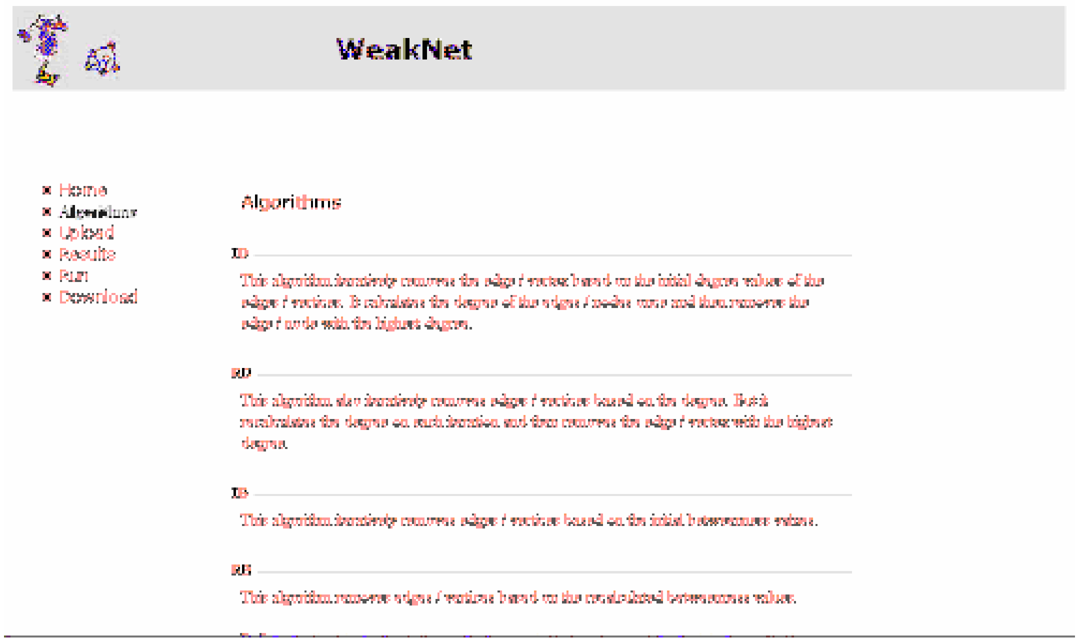
3.2 Organization of WeakNet

WeakNet is organized into 5 web pages representing the following functionality:

1. Algorithms
2. Upload
3. Results
4. Run
5. Download

3.2.1 Algorithms

WeakNet has the facility to run one of the eight algorithms on the network specified. The algorithms available ID, IB, RD [2] and RB (see section 2) can be directed on edges or vertices. F. Radicchi et al [5] have found that the betweenness [4] algorithm used by Girvan and Newman to identify communities in networks [4] is computationally expensive because it requires the repeated evaluation, for each edge in the system, of a global quantity, the betweenness [4] and has complexity O (M^2N)[5]. M here represents the number of edges and N the number of nodes. Since biological networks are typically large with more than 1000 nodes, we decided to use the algorithm described by F. Radicchi et al [5]. The algorithm is faster and is shown to have complexity O (M^2) [5].


WeakNet

- Home
- Algorithms
- Upload
- Results
- Run
- Download

Algorithms

ID

This algorithm iteratively removes the edge / vertex based on the initial degree values of the edges / vertices. It calculates the degree of the edges / nodes once and then removes the edge / node with the highest degree.

RD

This algorithm also iteratively removes edges / vertices based on the degree. But it recalculates the degree on each iteration and then removes the edge / vertex with the highest degree.

IB

This algorithm iteratively removes edges / vertices based on the initial betweenness values.

RB

This algorithm removes edges / vertices based on the recalculated betweenness values.

Fig. 5 Snapshot of the WeakNet algorithm page

3.2.2 Upload

The upload page lets users upload network data according to a specific format. Format: Two nodes separated by a space denote an edge connecting the first node to the second. Each edge should be separated by a UNIX new line character.

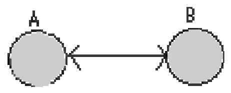


Fig. 6 Network example 1

In the example of Fig.6 the format is a sequence of lines separated by new lines where each line represents an edge.

A B

In example 2 (Fig. 7) we show a network that represents

reaction number 12: A + B → C + D

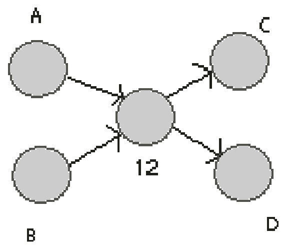


Fig. 7 Network example 2 showing the reaction 12: A + B -> C + D

We would represent this network shown in Fig. 7 by the format,

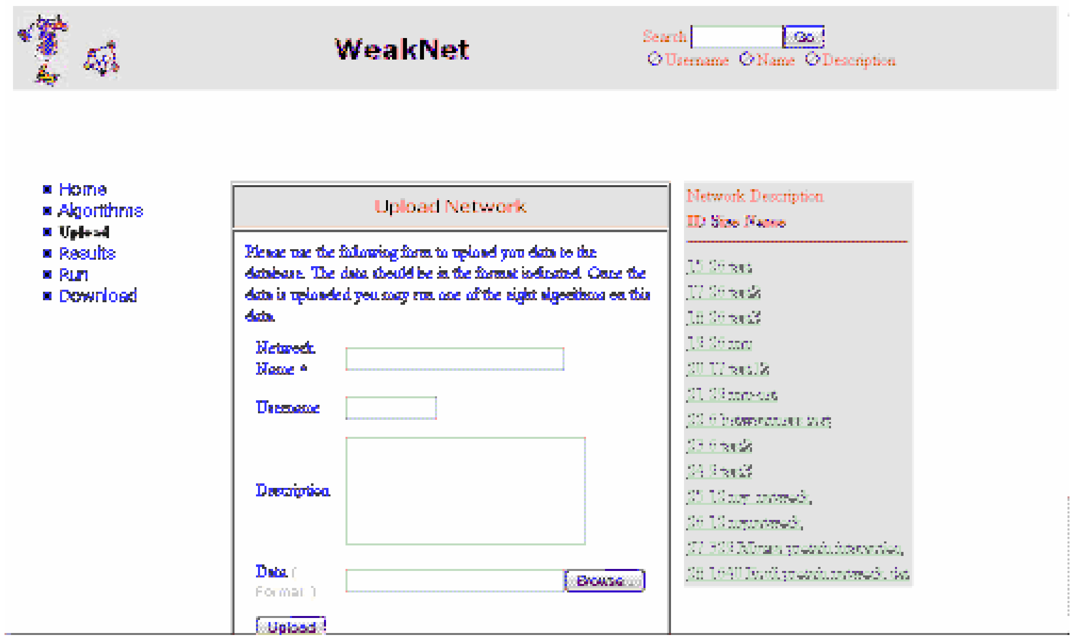
A 12

B 12

12 C

12 D

Once a file is created in the above format, it can be uploaded using the upload button. A unique network name and a clear description are needed so that other users do not upload the same network again using precious server space. A username is also required in order to keep track of the networks submitted by a specific user so that a user can search for networks submitted by him using the search facility which will be describe later. Fig. 8 is a screenshot of the upload page.



Please use the following form to upload you data to the database. The data should be in the format indicated. Once the data is uploaded you may run one of the eight algorithms on this data.

Fig. 8 Snapshot of the WeakNet upload page

3.2.3 Run

Once the network has been uploaded, algorithms can be run on this network by using the run page. The network of interest can be chosen by selecting the network IDs that appear as a dropdown list and the algorithm required can be selected. The algorithms are run until one of two criteria is satisfied.

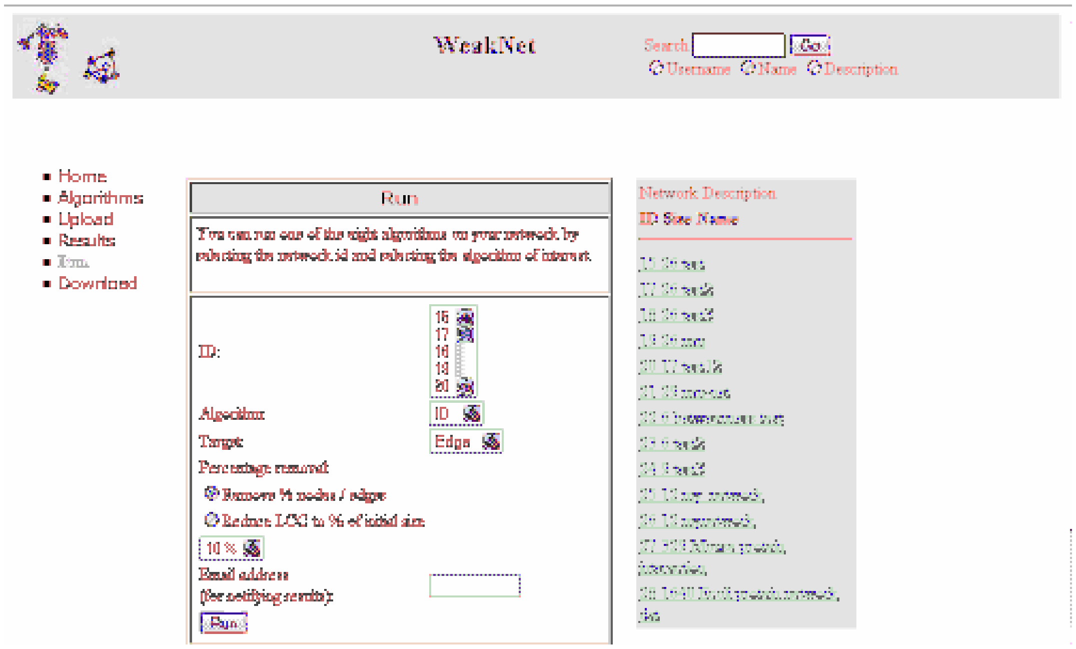
(1) Percentage of nodes / edges to be removed: This specifies the percentage of nodes or edges that need to be removed from the network. The algorithm runs until the specified number of elements is removed.

(2) Reduce largest connected component to percentage of initial size: The algorithm runs until the largest connected component reduces to this percent of its original size.

One of the above criteria can be specified in the upload page and a percentage value can also be selected. The tool keeps check of the algorithms that have been run on a network and prevents users from running an algorithm again on a network. One needs a valid email address to run these algorithms on any network. Since these algorithms are computationally intensive, once the user starts the algorithm, the CGI script forks to churn a new process that runs in the background even after the browser is closed. Once the process is complete, it notifies the user by sending an email to the address he / she specified. Fig. 9 is a screenshot of the run page of WeakNet.

3.2.4 Results

Once the process has started, the user can close the browser and can return to view the results from the results page once he / she receives an email confirming the completion of the process. All results are available for viewing at the results page. Here the user needs to select the id of the network whose results he needs to view. This will show the results of all the runs that have been performed on the network. The user can either order the results by speed or algorithm effectiveness. Fig. 10 shows a screenshot of this feature. The results show the algorithm used and the time taken for the algorithm to complete if the results are ordered by speed and the number of elements removed if they are ordered by effectiveness.


WeakNet

Search Go
Username Name Description

- Home
- Algorithms
- Upload
- Results
- Run
- Download

Run

You can run one of the eight algorithms on your network by selecting the network id and selecting the algorithm of interest.

ID: 16 17 18 19 20

Algorithm: ID

Target: Edge

Percentage removed:

Remove % nodes / edges

Reduce LCC to % of initial size

10 %

Email address (for notifying results):

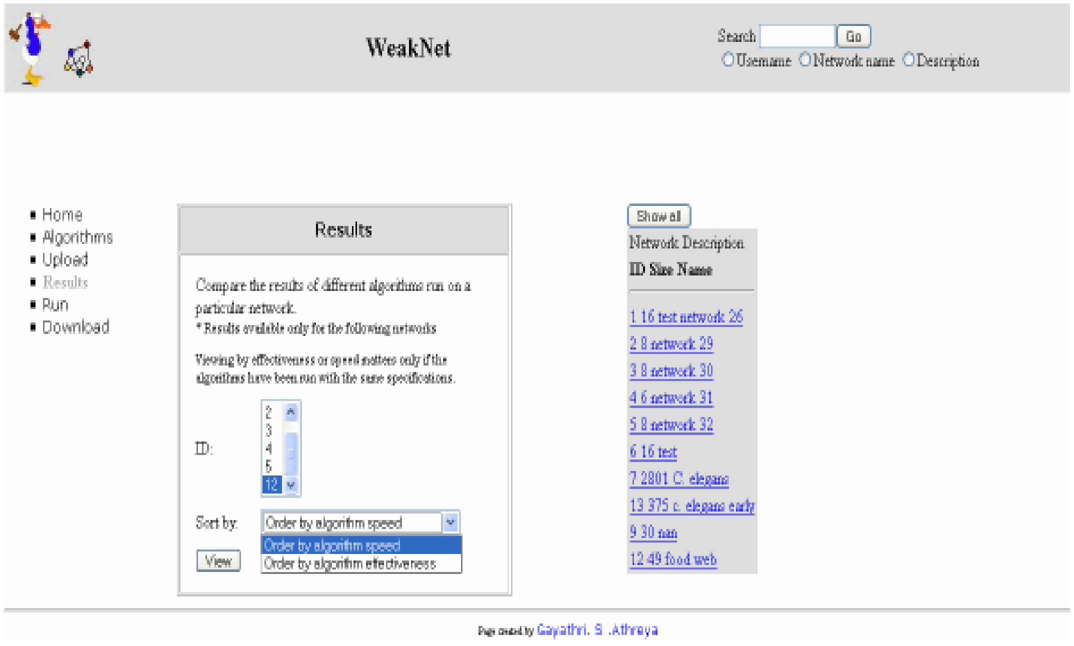
Run

Network Description

ID Size Name

[illegible]

Fig. 9 Snapshot of the WeakNet run page


WeakNet

Search Go
Username Network name Description

- Home
- Algorithms
- Upload
- Results
- Run
- Download

Results

Compare the results of different algorithms run on a particular network.
* Results available only for the following networks

Viewing by effectiveness or speed matters only if the algorithms have been run with the same specifications.

ID: 2 3 4 5 12

Sort by: Order by algorithm speed

Order by algorithm speed
Order by algorithm effectiveness

View

Show all

Network Description

ID Size Name

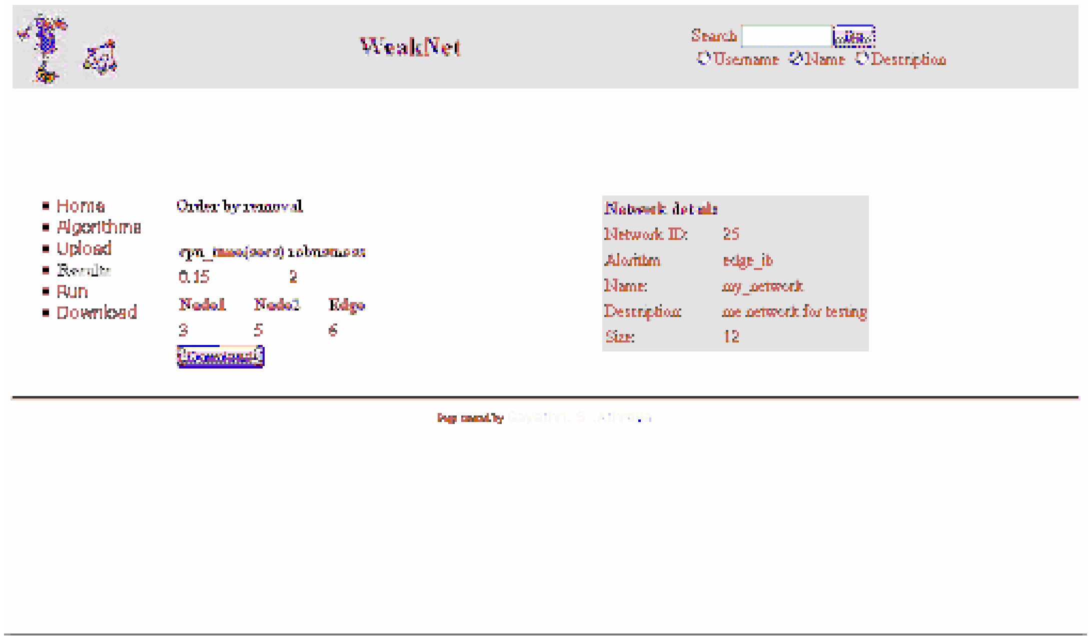
1 16 test network 26
2 8 network 29
3 8 network 30
4 6 network 31
5 8 network 32
6 16 test
7 2801 C. elegans
13 375 c. elegans early
9 30 nan
12 49 food web

Page created by Gayathri. S .Athreya

Fig. 10 Snapshot of the WeakNet results page 1

The user can view further details of a run by clicking on details as shown in Fig. 11. This brings up details of the run such as speed, robustness, the order of elements removed and a button to download the resulting network. This network is presented in

GraphViz format which can be viewed by the user using GraphViz or can be parsed and converted to any desired format. The screenshot of this functionality is shown in Fig. 12.



Order by removal

cpu_time(secs)	noofremoves	
0.15	2	
Node1	Node2	Edge
3	5	6

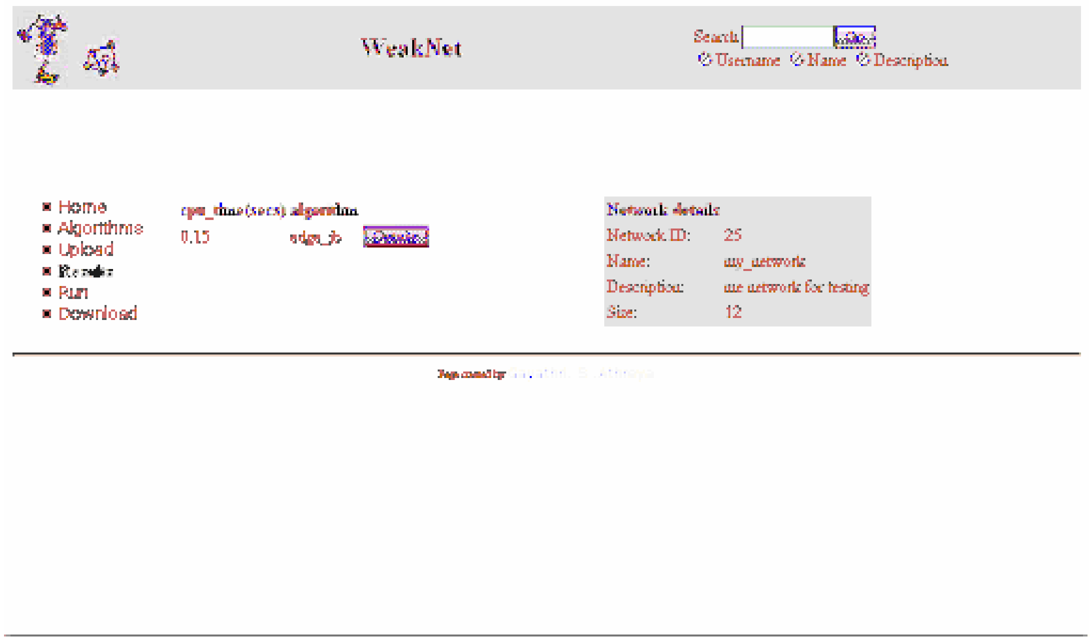
Network details	
Network ID:	25
Alorithm	edge_ib
Name:	my_network
Description	one network for testing
Size:	12

Page created by [illegible]

Fig. 11 Snapshot of the WeakNet results – page 2

3.2.5 Search

Once a network is uploaded, it can be retrieved by searching for it using the search box present on the top right corner of the tool. One can search for a network using network name, description or the username. These do not have to be complete; the user can search the database with partial descriptions. On almost all pages, the tool lists all the networks that are available in the database alongside the respective input forms. The user can click on these names to view descriptions of the network. However, when the search feature is used, this list displays only those entries that match the search criteria.



cpu_time(secs)	algorithm
0.15	edge_jb

Network details

Network ID:	25
Name:	my_network
Description:	one network for testing
Size:	12

Page created by [illegible]

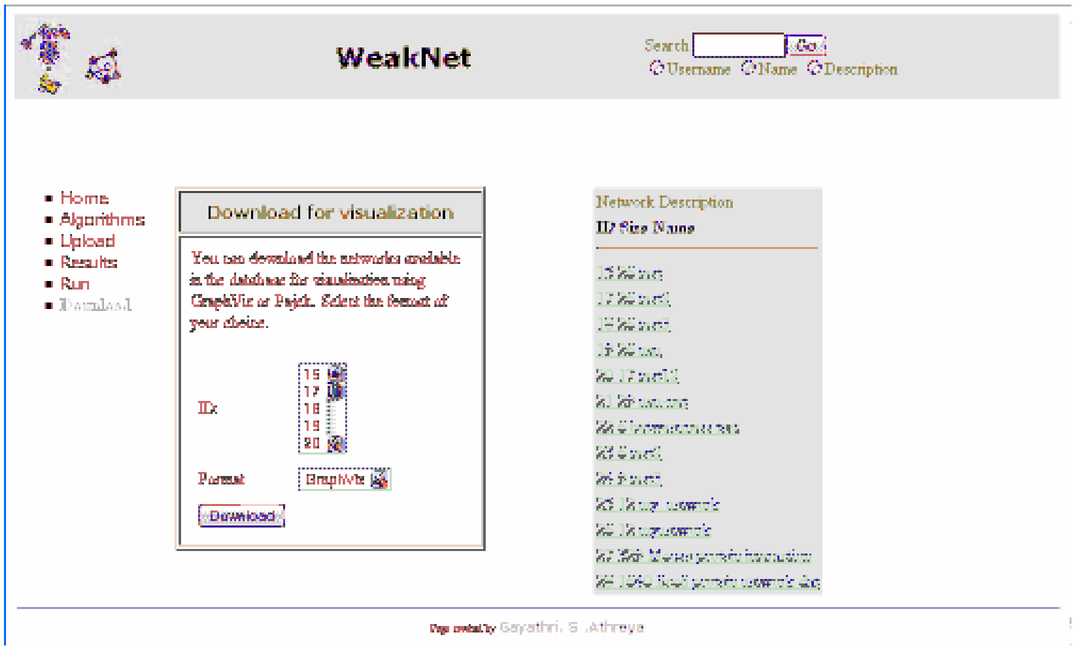
Fig. 12 Snapshot of the WeakNet results – page 3

3.2.6 Download for visualization

The networks present in the database can be downloaded using the download button. This lets the user download the network in either the GraphViz format or the Pajek format. The user can then use GraphViz or Pajek to view the network. This feature is shown in Fig. 13.

3.3 Database model

All the algorithms used by WeakNet are computationally intensive which makes it difficult to compute and display results on the fly. Hence a database is required in order to store data and let users view the results when the computation is complete. A MySQL database was created to store network data and the results generated by the different runs. Fig. 14 is the Entity - Relationship diagram of the database used.


WeakNet

Search [ ] Go
○ Username ○ Name ○ Description

- Home
- Algorithms
- Upload
- Results
- Run
- Download

Download for visualization

You can download the networks available in the database for visualization using GraphViz or Pajek. Select the format of your choice.

IDs: 15, 17, 18, 19, 20

Format: GraphViz

Download

Network Description

ID Size Name

[illegible]

Page created by Gayathri. S. Athreya

Fig. 13 Screenshot of the WeakNet Download page

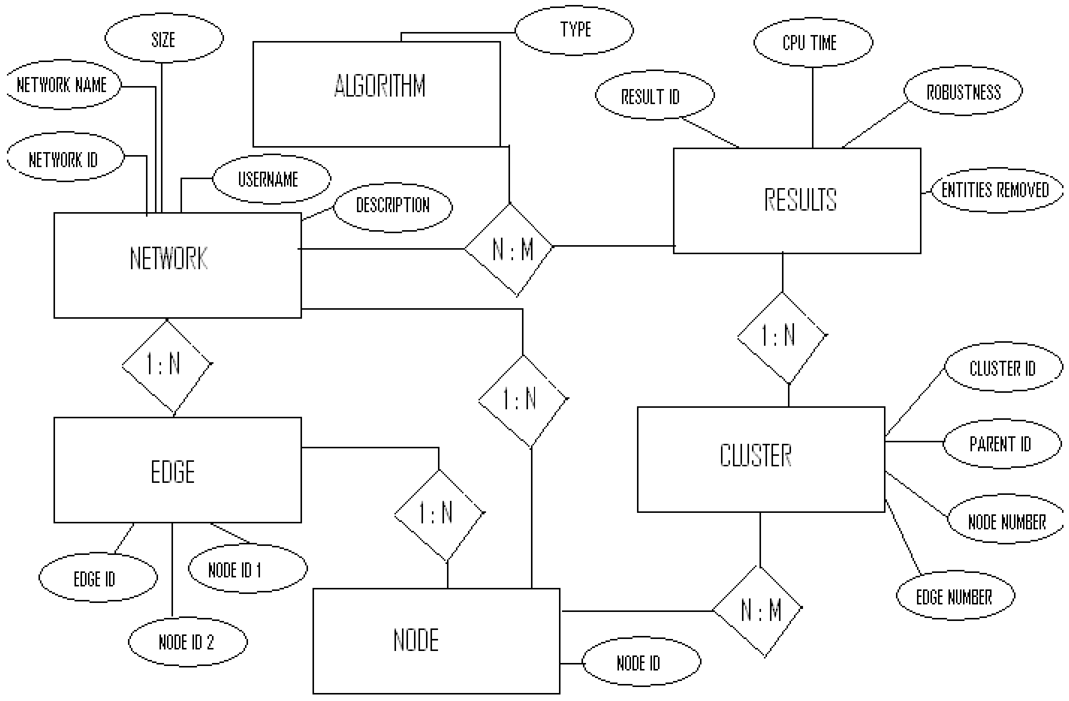


Fig. 14 Entity - Relationship diagram of the WeakNet database

The WeakNet database consists of 6 entities namely network, algorithm, results, edge, cluster and node. Network has attributes such as network ID, network name, size (number of nodes), username and description. The entity edge has attributes edge id, node id1 and node id2. Since a network has many edges and a particular edge can belong only to a single network, these two entities share a 1: N relationship. Edge also shares a 1: N relationship with node whose attribute is node id. The entity cluster is used to describe the clusters formed when a network is broken down by applying different algorithms. Cluster has attributes cluster id, parent id which is the id of the parent cluster, the node number in order to identify the nodes in the cluster and the edge number to identify the edges in it. Node shares an N: M relationship with cluster since a node can belong to many clusters and a cluster can have many nodes. It also has a 1: N relationship with results as a result can have many clusters but a cluster can only belong to one result. Results has attributes such as cpu time, robustness and entities removed. Algorithm is an entity that shares an N: M relationship with both results and network because, an algorithm can have many results depending on the algorithms run and a network can have different results with different algorithms. The only attribute of algorithm is the type.

4. Results

The algorithms were run on some network models in order to verify their accuracy. The results of the run are shown in Tables 1 and 2. In the results to follow, RB denotes the repeated betweenness [4] algorithm, IB [2] denotes the algorithm based on the initial betweenness [4] values, RD [2] the one based on repeated degree calculations and ID [2] the one based on the initial degree values.

4.1 Network example 1

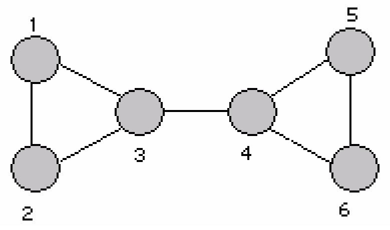


Fig. 15 Network example 1

All the algorithms were run on the network shown in Fig. 15 and results are shown in Tables 1 and 2.

Edges removed by edge based algorithms

Order of removal	RB	IB	RD	ID
1	3 – 4	3 – 4	3 – 4	3 – 4
2	6 – 7	1 – 3	2 – 3	2 – 3
3	4 – 7	2 – 3	1 – 3	1 – 3

Table 1

Vertices removed by vertex based algorithms

Order of removal	RB	IB	RD	ID
1	4	4	3	3
2	3	3	4	4

Table 2

The observations made from the results suggest that the algorithms identify the high betweenness [4] and degree points correctly. However, while the betweenness [4] based algorithms are able to identify the tightly connected clusters, the degree based algorithms are not.

4.2 Network example 2

The same test was conducted with a more complex network shown in Fig. 16 and the results are shown in Tables 3 and 4. Once again we see that the RB and IB [2] algorithms are able to detect the tightly connected clusters while RD [2] and ID [2] are not able to target the edges or nodes that connect different clusters together.

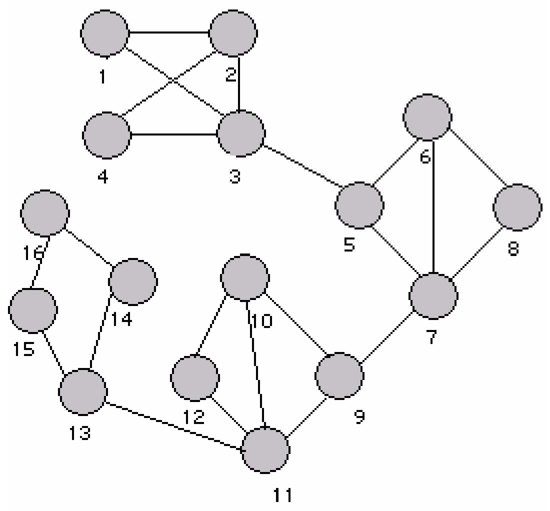


Fig. 16 Network example 2

Edges removed by edge based algorithms:

Order of removal	RB	IB	RD	ID
1	3 – 5	3 – 5	6 – 7	6 – 7
2	7 – 9	7 – 9	9 – 11	7 – 9
3	11 – 13	11 – 13	10 – 11	9 – 11
4	10 – 12	5 – 7	2 – 3	10 – 11
5	11 – 12	9 -11	5 – 7	11 – 13

Table 3

Vertices removed by vertex based algorithms:

Order of removal	RB	IB	RD	ID
1	13	13	11	11
2	9	9	3	3
3	7	7	7	7
4	5	5	10	10
5	3	3	13	13

Table 4

4.3 Network example 3

To test if the algorithms are able to detect functional clusters in real biological networks, we tested our algorithms on the protein interaction network of C. *elegans* obtained from DIP http://dip.doe-mbi.ucla.edu/ [13] which is the Database of interacting Proteins. DIP is an online database that catalogs experimentally determined interactions between proteins. Since the degree based algorithms are straight forward and use inbuilt functions available in the Graph module which is a library of Perl methods available in CPAN (Comprehensive Perl Archive Network http://www.cpan.org), we just tested the repeated edge betweenness [4] algorithm. This algorithm has already been studied by some researchers [6] and has found to be useful in identifying protein clusters with significant correlations to functional annotations. In order to test the algorithm, we downloaded the protein interaction network of the organism C. *elegans* from the DIP [13] website. Due to lack of gene ontology terms (GO) [7] for a majority of genes in the C. *elegans* data that we obtained, we used an earlier network of C. *elegans* with 437 interactions to test the effectiveness of the algorithm. The ontology terms were obtained using BIND http://www.bind.ca/Action which is also a database of protein interactions and the wormbase http://www.wormbase.org/ [8]. We applied the repeated edge betweenness [4] algorithm to the network and obtained a set of clusters. On analyzing some of the clusters by searching for the GO [7] terms of the genes present, we found that almost all of the genes in a cluster had common functions. The genes and the related GO [7] terms obtained for some of the clusters obtained are presented in Tables 5, 6 and 7. It can be observed from the data that most of the genes present in a cluster have similar functions. We also found a few genes (ones highlighted in yellow) whose function did not explicitly fall under the categories mentioned. This could be because these genes probably play an important role in two different processes and because the algorithm is very rigid about assigning a gene to exactly one cluster, it was grouped under the specified category.

The results obtained for all the three networks indicate that the algorithms function as expected and target nodes or edges correctly as per the algorithm type.

Cluster – 1	**GO terms**
C36B1.4	embryonic development, reproduction, positive regulation of growth
F25H2.9	embryonic development, reproduction, positive regulation of growth
ZK945.2	embryonic development, reproduction, positive regulation of growth
CD4.6	embryonic development, reproduction, positive regulation of growth
C06A8.1	positive regulation of growth
EFT-3 (FE1E3.5)	positive regulation of growth
F56H1.4	embryonic development, reproduction, positive regulation of growth
T07F8.4	DNA development
W02G9.2	development
H15N14.1	regulation of protein activity
C56C10.7	interacts with ZK945.2 (helps in embryonic development)
R186.7	cofactor biosynthesis (interacts with ZK945.2, embryonic development)
F23F1.8	embryonic development, ATP biosynthesis, growth, positive regulation of growth rate
ZK1098.4	protein biosynthesis, translational initiation
H28O16.1	embryonic development, ATP biosynthesis, growth, positive regulation of growth rate
Y79H2A.1	regulation of protein activity
C02F5.9	ubiquitin dependant protein, required for meiotic division progression
D1054.2 *	ubiquitin dependant protein catabolism
Y39G10AR	amino acid phosphorylation
C30F8	Unknown
T08A11.1 *	intracellular signaling cascade
D1005.1 *	metabolism lipid biosynthesis
F09E5.7	Unknown
C44B7.1	Unknown
C48B6.3	Unknown (interacts with C36B1.4)
Y42H9AR.f	Unknown

Table 5 Composition of cluster 1 as obtained by applying the repeated edge betweenness [4] algorithm to the C. *elegans* protein interaction network. This cluster has 26 proteins. The GO [7] terms indicate that these proteins are all related to growth and development.

Cluster -2	**GO terms**
T10B10.1	lipid catabolism, phosphate transport
B02O5.3	ubiquitin-dependant protein catabolism
F23F12.6	protein catabolism
R13A5.8	Physiological processes
K11H3.1	carbohydrate metabolism, glycerol-3-phosphate metabolism
C23H3.4	physiological process
F23C8.5	electron transport
Y39B6B.j	proteolysis, peptidolysis
F53G12.10	proteolysis and peptidolysis
B0336.10	signal transduction,
T23B5.1	signal transduction, protein amino acid methylation
B0336.2	small GTPase mediated signal transduction , protein amino acid ribosylation, intracellular protein transport
T24D8.1	ion transport, synaptic transmission
K08A8.1	protein amino acid phosphorylation,
F55D10.2 *	protein biosynthesis(interact with B02O5.3)
W09C5.1 *	protein biosynthesis(interact with B02O5.3)
F20D12.1 *	gametogenesis, embryonic development
K09E2.3	Unknown
F15B9.5	Unknown
T02E9.2	Unknown
C16C2.3	Unknown

Table 6 Composition of cluster 2 as obtained by applying the repeated edge betweenness [4] algorithm to the C. *elegans* protein interaction network. This cluster has 21 proteins. The GO [7] terms indicate that these proteins are all related to metabolism and signaling.

Cluster -3	**GO terms**
C30A5.2	base excision repair, DNA recombination
Y43C5A.6	DNA repair, DNA recombination
Y51A2D.17	regulation of transcription DNA dependant, DNA binding
K07D4.3	ubiquitin dependant protein
T06D8.8	regulation of exit from mitosis
F29B9.6	protein modification, ubiquitin cycle
F20H11.5	electron transport,
C02F12.4	proton transport
F55A11.3	protein ubiquitination
W06D4.6	DNA repair meiosis
C09D4.5	protein biosynthesis, reproduction
Y116A8C.13	DNA repair protein
ZC434.2	protein biosynthesis
R01H10.5	hypothetical protein
K07H8.6 *	lipid transport
R12E2.3 *	embryonic development

Table 7 Composition of cluster 3 as obtained by applying the repeated edge betweenness [4] algorithm to the C. *elegans* protein interaction network. This cluster has 16 proteins. The GO [7] terms indicate that these proteins are all related to protein biosynthesis.

4.4 Performance and Scaling

All the algorithms were run on networks of different sizes, 10% of their vertices were removed and their performances were plotted on a log-log scale. The plots shown in Fig. 17 and Fig. 18 were obtained as a result of these plots. The graph shown in Fig. 17 is a log – log plot of the performance of the vertex based algorithms. From the graph, it can be observed that the degree based vertex algorithms [2] have an exponent of 1.4 and are almost linear. Whereas the betweenness based vertex algorithms [2] have a higher exponent of 1.9. This shows that the betweenness based vertex algorithms [2] are more computationally intense than the degree based algorithms[2]. If the plots are extrapolated, it can be found that the degree based vertex algorithms [2] can handle a network of around 2756 vertices in a minute whereas the betweenness based vertex algorithms [2] can handle 244 vertices in a minute.

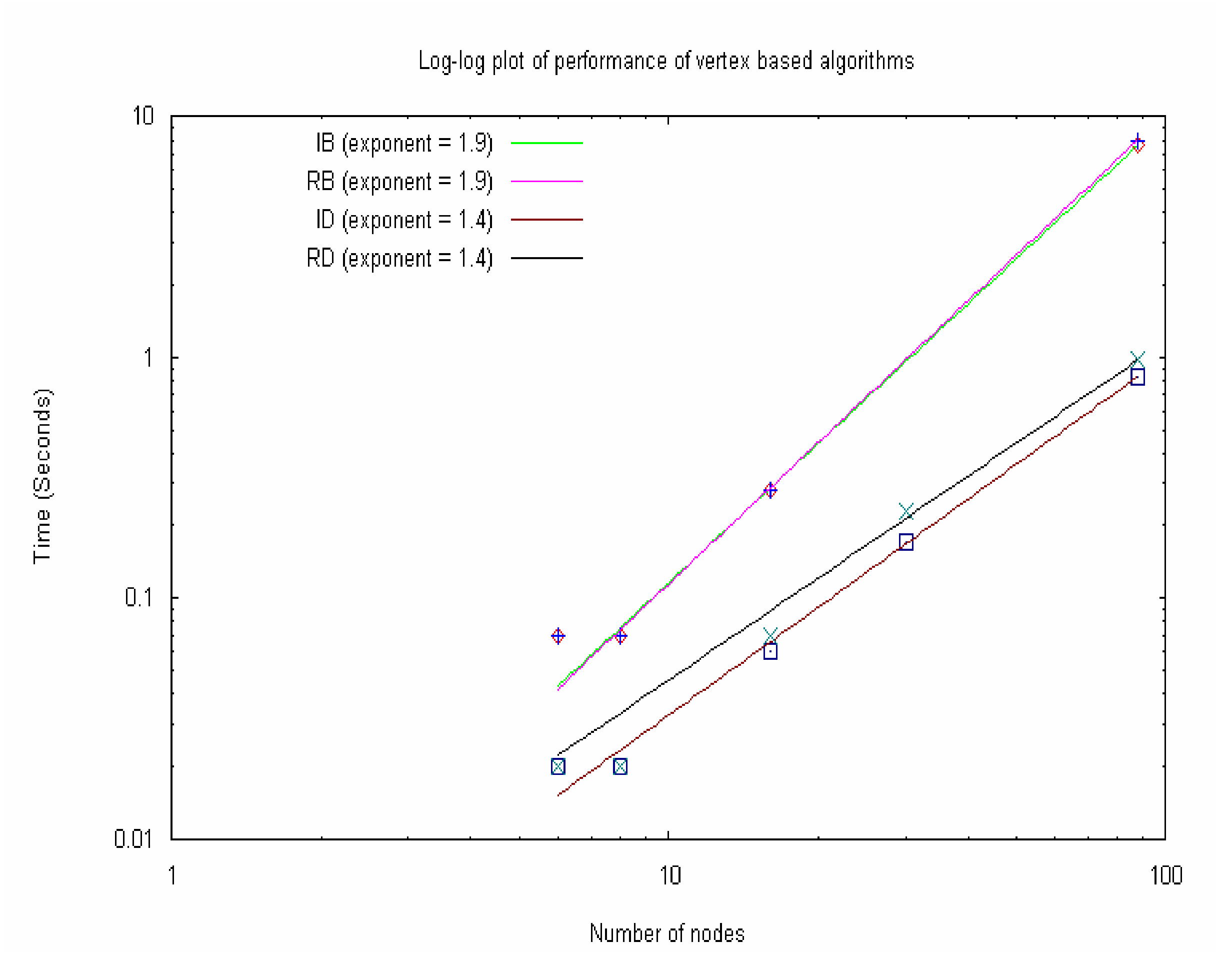


Fig. 17 Log-log plot of the performance of the vertex based algorithms when 10% of their nodes were removed.

The performance of the edge based algorithms [2] was also measured by removing 10% of the edges in many networks. The exponents obtained for these lines were as follows. The initial degree edge algorithm [2] behaved similar to the initial degree vertex algorithm [2] and its exponent was 1.4 indicating that it can handle a network of size 2756 in a minute. Whereas the other algorithms namely the repeated degree edge, the initial betweenness edge and the repeated betweenness edge [2] had higher exponents of 2.1, 2.2 and 2.3 respectively. Using this to calculated the size of the network that this algorithm can handle, we found that the repeated degree edge algorithm [2] can handle a network of size 305 edges a minute, the initial betweenness algorithm [2] a size of 178 edges a minute and repeated betweenness edge algorithm a size of 194. These results prove that the algorithms scale as expected (quadratic) as calculated by Radicchi et al [5].

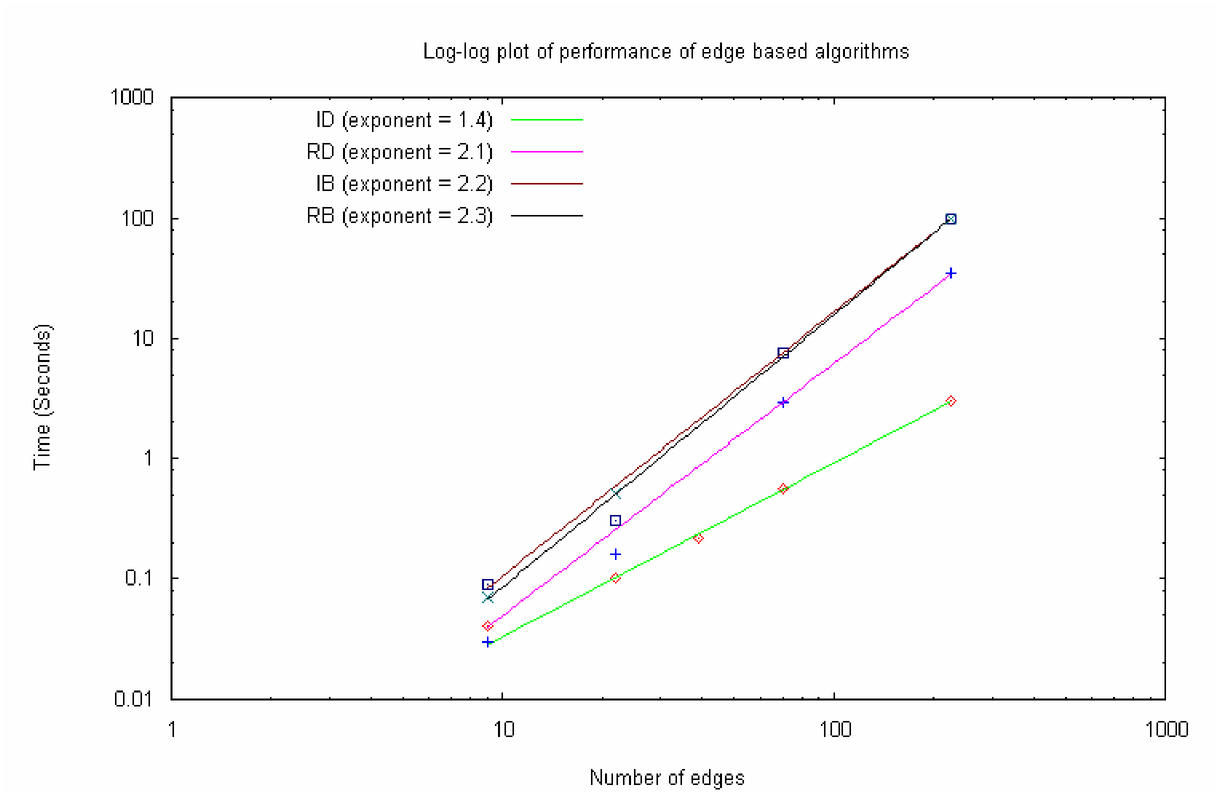


Fig. 18 Log-log plot of the performance of the edge based algorithms when 10% of their edges were removed.

5. Conclusion

The results obtained for all the three networks indicate that the algorithms function as expected and target nodes or edges correctly as per the algorithm type. While the both the betweenness based algorithms are successful at identifying the tightly connected clusters in the network, the degree based algorithms are not.

Functionality can be added so that a person can delete nodes or edges in any order according to his needs instead of using the algorithms to indicate the important nodes or edges. The tool currently generates networks in GraphViz or Pajek formats for visualization. It would be very useful if this functionality is built into the system. Incorporating flux analysis tools would also be very useful for this kind of research.

References

1. Error and attack tolerance of complex networks – R. Albert, H. Jeong, and A- L Barabasi. (2002) Nature 406, 378 - 382

2. Attack vulnerability of complex networks – P. Holme, B. J. Kim, C. N. Yoon, and S. Han (2002) Physical review E, Volume 65, 056109.

3. The large-scale organization of metabolic networks – H. Jeong, M. Tombor, R. Albert, A-L Barabasi (2000) Nature 407, 651 – 654

4. Community structure in social and biological networks – M. Girvan and M. E. J. Newman (2002) PNAS, Volume 99, 7821 – 7826

5. Defining and identifying communities in networks – F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi (2004) PNAS, Volume 101, 2658 – 2663

6. The Use of Edge-Betweenness Clustering to Investigate Biological Function in Protein Interaction Networks – R. Dunn, F. Dudbridge, C. M. Sanderson (2005) BMC bioinformatics, Volume 6, 39

7. Gene Ontology: tool for the unification of biology – M. Ashburner, C. A. Ball, J. A. Blake et al. (2000) Nature Genetics, Volume 25, 25 – 29

8. WormBase web site, http://www.wormbase.org, release WS150, date 11/30/2005.

9. Emergence of scaling in random networks – A – L Barabasi, R. Albert (1999) Science, Volume 286, 509 – 512.

10. On the evolution of random graphs – P. Erdos, A. Renyi (1960) Public Mathematical Institute of Hungary Academy of Sciences, Volume 5, 17 – 61.

11. Small Worlds: The dynamics of networks between order and randomness – D. J. Watts (1999) Princeton University Press.

12. Collective dynamics of 'small – world' networks – D. J. Watts, S. H. Strogatz (1998) Nature, Volume 393, 440 – 442.

13. DIP: Database of Interacting Proteins – I. Xenarios, D. W. Rice, L. Salwinski et al. (2000) Nucleic Acids Research, Volume 28, 289 – 291.

14. The larse-scale organization of metabolic networks – H. Jeong, M. Tombor, R. Albert, A – L Barabasi (2000) Nature, Volume 407, 651 – 654.

15. Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms – H. Ma, A – P. Zeng (2002) Bioinformatics, Volume 19, 270 – 277.