BIOINFORMATICS APPLICATIONS NOTE

Vol. 29 no. 19 2013, pages 2509–2511
doi:10.1093/bioinformatics/btt430

Systems biology

Advance Access publication August 13, 2013

SubNet: a Java application for subnetwork extraction

Quanwei Zhang and Zhengdong D. Zhang*

Department of Genetics, Albert Einstein College of Medicine, 1301 Morris Park Avenue, Bronx, NY 10461, USA

Associate Editor: Martin Bishop

ABSTRACT

Summary: The extraction of targeted subnetworks is a powerful way to identify functional modules and pathways within complex networks. Here, we present SubNet, a Java-based stand-alone program for extracting subnetworks, given a basal network and a set of selected nodes. Designed with a graphical user-friendly interface, SubNet combines four different extraction methods, which offer the possibility to interrogate a biological network according to the question investigated. Of note, we developed a method based on the highly successful Google PageRank algorithm to extract the subnetwork using the node centrality metric, to which possible node weights of the selected genes can be incorporated.

Availability: http://www.zdzlab.org/1/subnet.html

Contact: zhengdong.zhang@einstein.yu.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on May 2, 2013; revised on June 28, 2013; accepted on July 20, 2013

1 INTRODUCTION

Biological systems are complex entities composed of interacting components and can be modeled as different types of biological networks. Recently developed high-throughput genomic methods have generated large biological networks on the whole genome level, which are in general too complex, if possible at all, to be analyzed as a whole. Such analytical intractability makes it necessary to focus only on parts of the large networks—the subnetworks—that are most relevant to the research question. Not only do they make network analysis feasible and more focused, subnetworks can also help to predict pathways, discover network components (e.g. genes or proteins) not known to belong to a functionally related group and assign function to novel genes by revealing their neighbors in the networks.

Several algorithms have been developed to extract subnetworks, given a basal network and a set of selected nodes (the 'seeds'). Although the majority of them—the Takahashi–Matsuyama (Takahashi and Matsuyama, 1980), the Klein–Ravi (Klein and Ravi, 1995) and the pairwise *K*-shortest path (Faust *et al*., 2010) algorithms—are based on the shortest paths between nodes in the network, one of them, the *k*-walks algorithm, uses random walks on the network to find the relevant nodes and edges to generate the subnetwork. Although these algorithms can extract subnetworks based on well-established operations on networks, they cannot use informative node weights of seeds, if present, derived from prior knowledge. Here, we present SubNet, a Java-based application with a graphical user-friendly interface combining four different methods to identify and extract subnetworks of interest. We developed one of the extraction methods based on the highly successful Google PageRank algorithm (Page *et al*., 1999). This method is rigorous, efficient and can incorporate prior knowledge about the seeds.

2 METHODS

In SubNet, we provide four different methods to extract relevant subnetworks from a basal network given a list of preselected genes. Such a selection of methods offers the user the flexibility to extract a subnetwork most relevant to the question under investigation. Formally, given a network $G = (V, E)$ with its ensemble of n nodes V and edges E and a set of preselected nodes S, SubNet constructs a subnetwork W based on S, so that $W \subset G$.

Extraction by shell. A node in a network has neighbors on different levels: one edge away, on the so-called first shell, are the direct neighbors; two edges away are the indirect neighbors on the second shell and so on. Directly based on the notion of 'guilt by association'—the closest molecules are highly likely to be related to each other and involved in the same pathways and mechanisms—this method extracts the neighbors of each of preselected nodes within a certain distance. Given the set of selected genes $S = \{v_i\}$, the set of nodes within the k^{th} shell neighborhood, $T = U_{i=1}^{m} U_{j=1}^{k} \{v_i\}_j$, in which $\{v_i\}_j$ is the set of neighbors of v_i on the j^{th} shell. The subnetwork W is thus composed of the nodes in T and edges among them transferred from G.

Extraction by shortest path. To make use of indirect interactions, one can take higher-order neighborhoods into consideration. In a complex molecular interaction network, given a pair of reachable biomolecules, there are almost always a large number of routes to reach one node from the other. The shortest path between them is of particular interest and significance, as it is assumed to be the route used most for information communication between the two molecules and has been shown to provide a good measure for functional relatedness among genes. We use the shortest path in our second method—if the shortest path between two genes includes at least one transitive gene, it is postulated that the transitive genes on the path will be involved in the same biological process as the terminal genes. To find the shortest path between two nodes, we use Dijkstra's algorithm (Dijkstra, 1959). Because the shortest path between two nodes may not be unique, we will recover all the shortest paths if there is more than one. Given the set of selected genes, $S = \{v_i\}$, the set of nodes on the shortest paths among them, $T = U_{i=2}^{m} U_{j=1}^{i-1} \{v_i \leftrightarrow v_j\}$, in which $\{v_i \leftrightarrow v_j\}$ is the set of nodes on the shortest path between and including v_i and v_j. The subnetwork, W, is thus composed of the nodes in T and edges among them transferred from G.

Extraction by emission decay. Given the network connectivity, the preselected seeds can be considered as heat sources, emitting functionality along the edges to other nodes in the network. Such emission decays as the distance increases, and the total retainment at each node (excluding seeds) is the sum of functionality transmitted from all seeds: $F = [w_i / (d_i)^2]^\lambda$, in which d_i and w_i are the distance and the total edge weight,

*To whom correspondence should be addressed.

respectively, between the node and the seed i, and λ is the exponent parameter whose value ($\lambda = 1$, 2 or 3) is selected by the user. The subnetwork, W, is thus composed of the nodes whose functionality retainment is greater than a preset threshold and edges among them transferred from G.

Each node sees its decay value updated after each seed is being considered, with the initial decay value at 0 before the first seed. Once every seed has been explored, the final decay value of each node is then compared with a threshold τ, which is initially set by the user. The resulting subnetwork W consists of the nodes, with decay value higher than τ and the edges connecting them.

Extraction by PageRank. We also developed a method based on the node centrality/influence and adapted the Google PageRank algorithm to incorporate preselected nodes. Let **A** be the adjacency matrix of network G:

$$a_{ij} = \begin{cases} w_{ij} & \text{if } i \leftrightarrow j \in E \\ 0 & \text{otherwise} \end{cases},$$

in which w_{ij} is the weight of the edge between nodes i and j in an undirected network, or from node i to j in a directed network ($w_{ij} = 1$ if the weight is unspecified). For nodes that are preselected as seeds, values in their corresponding columns in **A** are increased: $a_{ij} = c \cdot w_{ij}$ ($c > 1$).

The centrality/influence of each node (p_i) is the sum of the influence it receives from each of its source node (p_j) divided by the number of outedges from that source node (o_j): $p_i = \sum_{j \to i \in E} p_j / o_j$. It can be written in the matrix form as $\mathbf{P} = \mathbf{B}^T\ \mathbf{P}$, in which **B** has $b_{ij} = a_{ij} / \sum_j a_{ij}$ and $\mathbf{P} = [p_1, p_2, \ldots, p_n]^T$. This is the characteristic equation of the eigensystem, and the solution to vector **P** is an eigenvector of **B** with the corresponding eigenvalue of 1. However, if **B** is a stochastic matrix that is also irreducible and aperiodic, 1 is the largest eigenvalue and **P** is the principal eigenvector (Perron–Frobenius theorem). To make such a matrix, we first replace every cell of a row with $1/n$ if that row contains only 0 and then add a link from each node to every node with a small transition probability. This operation generates a new matrix **C**:

$$\mathbf{C} = (1 - d)\mathbf{E}/\mathrm{n} + d\mathbf{B}^T (0 < d < 1),$$

in which **E** is an $n \times n$ matrix of 1's and d is a parameter controlling the weight of the **E** component in **C**. To obtain **P**, we use the power iteration method (von Mises *et al.*, 1929), which can be applied to large sparse matrices of biological networks:

$$\mathbf{P} = \mathbf{P}^{(t)} = \mathbf{C}\mathbf{P}^{(t-1)} \text{and } \mathbf{P}^{(0)} = [1/\mathrm{n}, \ldots, 1/\mathrm{n}]^T.$$

The resulting subnetwork W consists of the nodes with values in **P** higher than a preselected threshold and the edges connecting them.

Implementation and user interface. SubNet was entirely implemented in Java for more streamlined software engineering, better platform portability and easier future development. Depending on the user's need, it can be run either with a graphical user interface (Fig. 1A) or on the command line. When data processing is done, SubNet outputs the subnetwork information in several files for nodes, edges and interactions. These files can be subsequently imported into programs such as Cytoscape (Shannon *et al.*, 2003) for subnetwork visualization and further analyses. We benchmarked the performance of SubNet (Supplementary Table S1) and compared various aspects of the four methods implemented in SubNet (Supplementary Table S2). In its application discussed below, SubNet used ~3.75 GB of memory and finished the job in 90 s on a typical MacBook laptop. Future work will focus on the implementation of a web-based version and a Cytoscape plugin.

3 APPLICATION

By extracting subnetworks, SubNet can identify other components involved in the pathways and biological processes of the genes or proteins selected as seeds. Six proteins—HMGB1, HMGB2, LIG1, LIG3, LIG4 and XRCC4—are known to play a major role in DNA ligation as part of the DNA repair (GO:0051103) process. Using them as the seeds and the protein–protein interaction network compiled in the Human Protein Reference Database (release 9) (Peri *et al.*, 2003) as the basal network, we extracted a subnetwork composed of 50 nodes with highest scores and 181 interactions among them (Supplementary Table S3) by using PageRank. Five of the six selected proteins were nodes with highest scores. A manual annotation of the 50 subnetwork nodes indicated that 39 (including the five seeds) of them are directly or indirectly (by physical interaction with other proteins) involved in the DNA repair and DNA damage response processes. Using the same input data, we also extracted a subnetwork composed of 26 proteins (Supplementary Table S4) and 63 interactions (Fig. 1B, Supplementary Fig. S1) by using the shortest path method. Among 14 proteins that are not in the aforementioned PageRank subnetwork are IRF2, PGR and PNKP, which are involved in DNA ligation or DNA binding, and APLF, APTX and PARP1, which play important roles in DNA repair, the parent ontology of that of the queried proteins (Fig. 1C). A previous study showed that combining methods based on random walk and shortest path may give the best result in subnetwork extraction (Faust *et al.*, 2010).

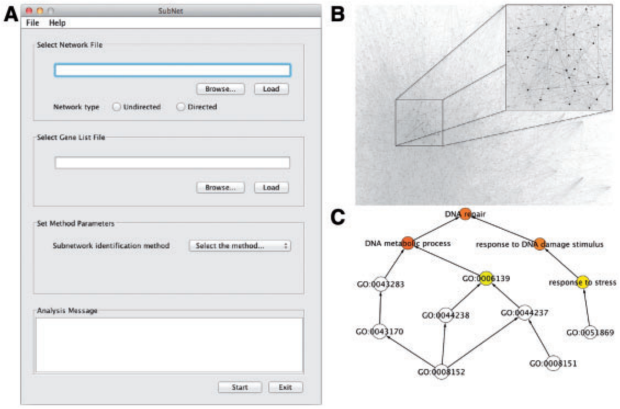

Fig. 1. SubNet. (**A**) Its graphical user interface. (**B**) Subnetwork extraction by SubNet. A subnetwork, highlighted with black nodes and edges in the zooming window, was extracted based on six DNA ligation-related proteins from the basal Human Protein Reference Database protein–protein interaction network (in gray). See also Supplementary Figure S1. (**C**) Functional enrichment in the subnetwork. We analyzed the Gene Ontology annotation of proteins in the subnetwork with the Biological Networks Gene Ontology tool (BiNGO) (Maere *et al.*, 2005). The additional proteins extracted in the subnetwork show significant enrichment (in grey/orange) for DNA repair, response to DNA damage stimulus and DNA metabolic process

Funding: NIH Pathway to Independence Award from National Library of Medicine (5R00LM009770-06 to Z.D.Z.) and the Ellison Medical Foundation New Scholar in Aging Award (AG-NS-0751-11 to Z.D.Z.).

Conflict of interest: none declared.

REFERENCES

Takahashi,H. and Matsuyama,A. (1980) An approximate solution for the Steiner problem in graphs. *Math. Japonica*, **24**, 573–577.

Klein,P. and Ravi,R. (1995) A nearly best-possible approximation algorithm for node-weighted Steiner trees. *J. Algorithms*, **19**, 104–115.

Faust,K. *et al.* (2010) Pathway discovery in metabolic networks by subgraph extraction. *Bioinformatics*, **26**, 1211–1218.

Page,L. *et al.* (1999) The PageRank citation ranking: bringing order to the Web. In: *Technical report*. Stanford InfoLab.

Dijkstra,E.W. (1959) A note on two problems in connexion with graphs. *Numer. Math.*, **1**, 269–271.

von Mises,R. and Pollaczek-Geiringer,H. (1929) Praktische Verfahren der Gleichungsauflösung. *Z. Angew. Math. Mech.*, **9**, 152–164.

Shannon,P. *et al.* (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, **13**, 2498–2504.

Peri,S. *et al.* (2003) Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res.*, **13**, 2363–2371.

Maere,S. *et al.* (2005) BiNGO: a cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks. *Bioinformatics*, **21**, 3448–3449.