Assignment “Graphs II: network mining and module identification”

1. Introduction

As previously seen (Graphs II: network inference) a central goal of systems biology is to derive models of interactions between the components of the living system. In some cases detailed models, for example using ODEs, can be built but in other cases, we have to satisfy ourselves with a top-down approach were the structure of the interactions is inferred from experimental data.

In the framework of system biology, network representations provide a unique view to understand biological systems and the interconnections between the individual components. In biological networks, the molecular components (such as genes, proteins, or cells) are represented as nodes and their interactions as links (or edges). The exact meaning of the edges depends on the studied networks; for instance, in gene regulatory networks, genes are represented as nodes, and the edges represent regulatory interactions where the protein product of a given gene directly modulates the expression of a target gene. In co-expression networks, the connecting edges represent significant co-expression levels of the connected genes, and, in this case, no a priori hypothesis is cast on the cause of the correlation. In protein–protein interaction networks, the nodes represent proteins and edges the presence of physical interactions between them. Furthermore, the edges can be assigned a weight that might represent the strength or the likelihood of the interactions; in these cases, the networks are said to be weighted.

When the networks are analyzed, the interest usually relies on the identification of sets of functionally related genes or proteins that are involved in a common biological process and therefore show distinct patterns of interconnectedness in the network: they form clusters or modules. Other approaches combine molecular profiles, for example the output of a differential expression measurement, with the already known network. In these cases the goal is to derive context-dependent active modules that are associated with the systems' response to the analyzed perturbation.

Clusters or densely connected regions in the graphs can be identified by unsupervised clustering methods. In this assignment you will implement one of the first developed algorithms to identify these clusters: hierarchical clustering. Hierarchical clustering is applied in multiple contexts and is a technique to partition objects into optimally homogeneous groups, i.e. to identify clusters of objects according to some similarity measurement. In this case the objects are the nodes of a network and the similarity measurement is derived from their relative locations in the network. You will have to implement a) a way to *measure the similarity* between the nodes in the network and b) *hierarchical clustering.*

2. Assigment

Here we will use hierarchical clustering to identify modules within networks, this was first implemented in (Rives and Galitski 2003)

The network is represented by an *adjacency matrix.* For a network with N nodes the adjacency matrix is an N x N matrix so that A_{ij} is 1 if there is an edge going from i to j and 0 otherwise. Here we will only use undirected networks, so that the adjacency matrices are symmetric: $A_{ij} = A_{ji}$. You can see an example in Figure 1.

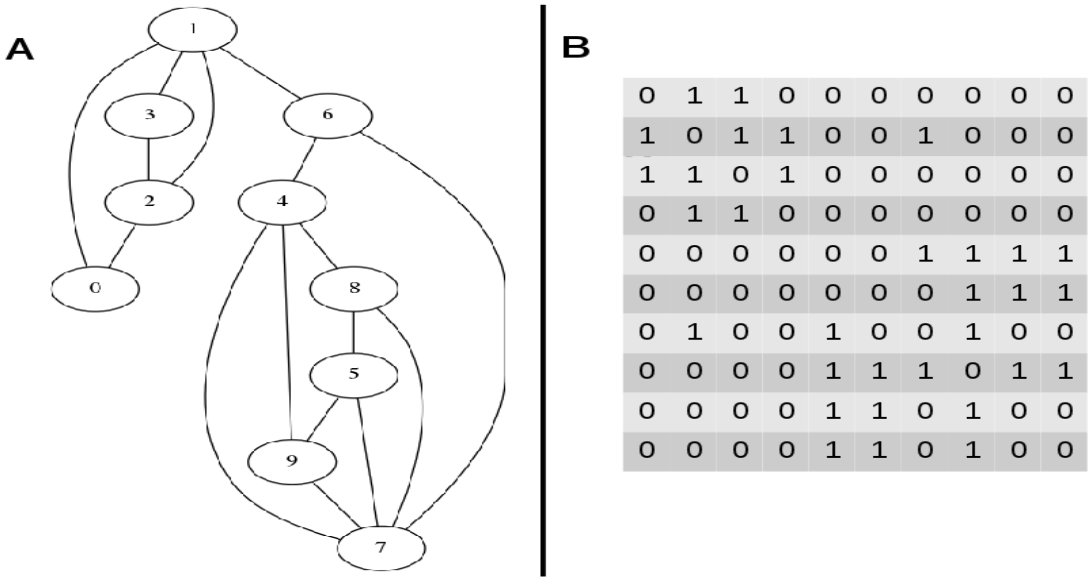

Figure 1: Toy example network: A) graphical representation and B) adjacency matrix.

The similarity measurement between the nodes is related to the shortest distance between the nodes in the network. This is briefly described in the second paragraph of Rives and Galitski (2003). However, two important elements are missing in their description: first, they do not state how the find the shortest path between two nodes; second, they say that they will use the “uncentered correlation coefficient” as the distance metric, but provide no further information on which correlation they use. Here we will use the shortest path algorithm (already provided in the skeleton) and Pearson's correlation coefficient.

We will use hierarchical agglomerative *average-linkage* clustering, described in a totally different context by(Eisen et al. 1998) in the materials and methods sections. The main elements of the algorithm are depicted in Figure 2:

1. initially, each element is in its own cluster.
2. we go through each pair of elements to identify the closest/most similar one.
3. the most similar pair is merged into a new cluster, to which we assign the average values of its constituent elements.
4. we iterate until only one large cluster remains that contains all the elements.
5. the output can be visualized as a dendrogram.
6. We select the desired number of clusters and we “cut” the tree at the proper height so that we recover the desired number of clusters.

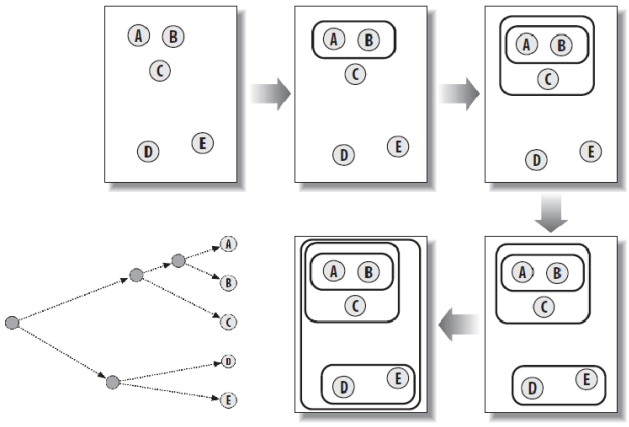

Figure 2: Schematic description of hierarchichal clustering.

You will have to:

- Implement a function **`association_matrix(matrix`**) that computes the association matrix from an initial matrix that contains the shortest-path distances between the nodes, as described in (Rives and Galitski 2003).
- Implement a function **`correlation(x,y)`** that calculates Pearson's correlation coefficient between two vectors:

$$r_{xy} = \frac{\sum x_i y_i - n\bar{x}\bar{y}}{n s_x s_y} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\ \sqrt{n\sum y_i^2 - (\sum y_i)^2}}.$$

- Implement a function **`dissimilarity_matrix_from_correlation(association)`** that produces a symmetric matrix with the distance between two nodes *x* and *y* defined as 1 - correlation(*x*,*y*).
- Implement the **`hclust(dissimilarity)`** function that produces the hierarchical clustering.

In the skeleton provided you will find functions to read adjacency matrices, convert them into graphs and compute the matrix that contains the shortest-path length between each pair of nodes.

Note/Hint
To perform the hierarchical clustering a **`Clusterset`** class has been defined, containing for each cluster an identifier or ID, the clusters that fall to the left and to the right in the dendrogram and the similarity between these nodes. The ID of the cluster is either -1 if it represents a branching point or contains the IDs of the nodes in the network. If you use this class, you will be able to use the functions **`get_ordered_elements`** (the elements as ordered in the dendrogram) and the **`cut_tree`** function that produces the network subsets by cutting the dendrogram at a chosen height. Finally, the function **`print_tree`** produces a very simple representation of the dendrogram. *You don't need to use it*

Questions:

1. Analyze the network given by the adjacency matrix **`toy_example.csv.`** Print
   - the distances between the nodes
   - the association values between the nodes
   - the dissimilarity values between the nodes.
     Compare them to the ones in **toy_example_output.pdf**

2. Perform hierarchical clustering and choose different height cut-offs to extract the groups of nodes from the hierarchical tree. Which height values do you have to choose so that you are able to extract the “correct” modular structure of this network, depicted in Figure 1?

3. (Saccenti, et al. 2014) described networks of relationships between blood metabolites associated to cardiovascular disease risk. These networks can be used to find process relevant to healthy cardiovascular states. The file **high_risk_network.csv** contains the adjacency matrix of the network associated to high cardiovascular risk. However the calculation of the shortest path distance is very slow. Instead you can read them from **shortest_path_distance_high_risk_network.csv** using the **`csv_to_matrix`** function. The nodes in the network are described in the **metabolite_nodes.csv.** If you explore different height values to cut the tree, you will notice that glucose (node 5) tends to form a cluster with 4 other metabolites. Which are those? Can you find any common characteristic they all share?

4. How do time/memory requirements of each part of the algorithm vary with the number of nodes in the network?
   1. Find shortest distances between the nodes in the network
   2. Association values

3. Dissimilarity values
4. Hierarchical clustering

5. How do time/memory requirements of each part of the algorithm vary with the number of edges in the network?

6. Is this algorithm exact or approximate?

7. How would you modify this approach to find clusters in a protein-protein interactions for which we have experimental measurements of protein expression values in a set of chosen conditions? The goal is to identify clusters that are specific to the selected conditions.

8. **OPTIONAL:** The provided function to calculate the shortest-path distance within the graph is quite slow. There are multiple libraries (like "Another Python Graph Library" http://pythonhosted.org/apgl/index.html that have these functions efficiently implemented. The Floyd-Warshall algorithm http://en.wikipedia.org/wiki/Floyd%E2%80%93Warshall_algorithm would also be an efficient solution. Try your own implementation of this algorithm!

Please, email your Python script and a PDF file containing the answers to the questions to algorithms@bioinformatics.nl no later than 14.00 on the day of the lecture.

Reading material

The review paper by (Mitra et al. 2013) "Integrative approaches for finding modular structure in biological networks" Nat. Rev. Genetics presents an overview of methods for network mining and their application in a biological context.

For the discussion, we have selected ClusterONE (Nepusz, Yu, and Paccanaro 2012) that is an algorithm to detect overlapping protein complexes from protein-protein interaction data. The second algorithm, Matisse, (Ulitsky and Shamir 2007) that uses both a network and expression data with the goal of identifying modules in the network that are active. When presenting the papers, please focus on the algorithmic challenges and implementation rather than the (biological) results. For these methods the applicability to genome-scale data is of interest: do the algorithms scale well? It is also important to address how reproducible are the results, if the starting expression dataset is slightly different, would we get similar results?

References

Eisen, M B, P T Spellman, P O Brown, and D Botstein. 1998. "Cluster Analysis and Display of Genome-Wide Expression Patterns." *Proceedings of the National Academy of Sciences of the United States of America* 95 (25): 14863–68.

Mitra, Koyel, Anne-Ruxandra Carvunis, Sanath Kumar Ramesh, and Trey Ideker. 2013. "Integrative Approaches for Finding Modular Structure in Biological Networks." *Nature Reviews Genetics* 14 (10): 719–32. doi:10.1038/nrg3552.

Nepusz, Tamás, Haiyuan Yu, and Alberto Paccanaro. 2012. "Detecting Overlapping Protein Complexes in Protein-Protein Interaction Networks." *Nature Methods* 9 (5): 471–72. doi:10.1038/nmeth.1938.

Rives, Alexander W., and Timothy Galitski. 2003. "Modular Organization of Cellular Networks." *Proceedings of the National Academy of Sciences* 100 (3): 1128–33. doi:10.1073/pnas.0237338100.

Saccenti, Edoardo, Maria Suarez-Diez, Claudio Luchinat, Claudio Santucci, and Leonardo Tenori. 2014. "Probabilistic Networks of Blood Metabolites in Healthy Subjects as Indicators of Latent Cardiovascular Risk." *Journal of Proteome Research*, November. doi:10.1021/pr501075r.

Ulitsky, Igor, and Ron Shamir. 2007. "Identification of Functional Modules Using Network Topology and High-Throughput Data." *BMC Systems Biology* 1: 8. doi:10.1186/1752-0509-1-8.