GOlorize: a Cytoscape plug-in for network visualization with Gene Ontology-based layout and coloring

O. Garcia, Cosmin Saveanu, M. Cline, M. Fromont-Racine, A. Jacquier, B. Schwikowski, T. Aittokallio

BIOINFORMATICS APPLICATIONS NOTE Vol. 23 no. 3 2007, pages 394–396
doi:10.1093/bioinformatics/btl605

Systems biology

GOlorize: a Cytoscape plug-in for network visualization with Gene Ontology-based layout and coloring

Olivier Garcia[1], Cosmin Saveanu[2], Melissa Cline[1], Micheline Fromont-Racine[2], Alain Jacquier[2], Benno Schwikowski[1] and Tero Aittokallio[1,*]

[1]Systems Biology Unit and [2]Macromolecular Interaction Genetics, Institut Pasteur, 75724 Paris Cedex 15, France

Received on October 18, 2006; revised and accepted on November 21, 2006
Advance Access publication November 24, 2006
Associate Editor: Alvis Brazma

ABSTRACT

Summary: We have implemented a graph layout algorithm that exposes Gene Ontology (GO) class structure on the network nodes. It can be used in conjunction with BiNGO plug-in to Cytoscape, which finds the GO categories over-represented in a given network. Our plug-in, named GOlorize, first highlights the class members with category-specific color-coding and then constructs an enhanced visualization of the network using a class-directed layout algorithm

Availability: http://www.cytoscape.org/plugins2.php

Contact: teanai@pasteur.fr

Supplementary information: Installation instructions and tutorial at http://www.cytoscape.org/plugins/GOlorize/GOlorizeUserGuide.pdf

1 INTRODUCTION

The widespread availability of large-scale experimental data sets has increased the need for software tools for network analysis. Such networks are typically represented as graphs, where the nodes correspond to cellular components (e.g. genes or proteins) and edges to their interactions (e.g. physical binding or statistical relations). Owing to the large number of components and interactions, visualization of these network graphs is non-trivial. While several graph layout algorithms are available in different software packages, such as Cytoscape (Shannon *et al*., 2003), they lack flexibility at incorporating additional class information already in node placement. To this end, we present GOlorize, a Cytoscape plug-in for advanced network visualization, which uses Gene Ontology (GO) annotations as a source of external class information to direct the layout process and to emphasize the biological function of the nodes. Implementation of GOlorize is based on BiNGO plug-in, an efficient tool to assess GO category enrichment (Maere *et al*., 2005).

2 SYSTEM OVERVIEW

After specifying the network (or sub-network) to be visualized, the first step involves the selection of the GO terms that will be used in the layout process. In the default mode, the terms are selected from the list of GO categories with significant over-representation in the selected network, as identified by the BiNGO plug-in (Maere *et al*., 2005). Additionally, the user can add GO categories of interest. Each network node can belong to none, a unique or several classes. If a node belongs to a single class it is identified with a corresponding color, which facilitates the visual interpretation of the network. If a node belongs to several classes, the node is colored as a pie chart with the corresponding colors.

In many cases, the bare color-coding of the nodes is not sufficient for discovering whether the network reflects a GO class structure. At the core of the GOlorize plug-in is a layout algorithm that determines the placement of the nodes based on both their connection structure (the edges) and class structure (the categories). To capture these two goals, we have extended the original force-directed graph layout algorithm by Fruchterman and Rheingold (1991) to favor grouping nodes of the same class near each other, as detailed in the next section. Globally, the operation of our class-directed layout algorithm is organized into the following three phases:

(1) Nodes are organized using force-directed optimization, where, in addition to the standard attractive forces between each connected node, an extra attractive force is applied to the nodes belonging to the same class. The extra attraction is directed through additional (virtual) edges between all class members and a virtual class node (an extra node not included in the network), which represents a particular class. This phase finds good initial positions for the class nodes.

(2) Classes are separated by moving the class nodes away from the center of the network. This phase aims at creating space between the different classes, while preserving the relative placement of the nodes obtained in the initial layout phase.

(3) A final layout is generated using the same force-directed optimization process as in the phase (1), but with class nodes fixed at the locations determined in the phase (2). The aim of this phase is to fine-tune the placement of the actual nodes, not only according to the class information, but also using their underlying pattern of interactions. Neither the virtual edges nor the class nodes are shown in the final visualization.

3 CLASS-DIRECTED LAYOUT ALGORITHM

Let n_e and n_v be the number of edges and nodes, respectively, and d denote the Euclidean distance between two nodes. The following

*To whom correspondence should be addressed.

repulsive and attractive forces are applied in the force-directed optimization process:

$$f_{\text{rep}}(d) = -k_{\text{rep}}\frac{k^2}{d}, \quad f_{\text{att}}(d) = k_{\text{att}}\frac{d^2}{k}. \tag{1}$$

The two parameters, k_{att} and k_{rep}, control the relative contribution of each force; the first is adjustable by the user while the second one is fixed to $k_{\text{rep}} = n_e/n_v$; and they both are relative to the original factor $k = D/\sqrt{n_v}$ proposed by Fruchterman and Rheingold (1991). Here, D is the layout dimension, proportional to n_v and to the node size S. The repulsive force is calculated for all the node pairs within a distance $d < 3k$. The identification of such pairs can be implemented efficiently by using grid-based hash lists. The attractive force is calculated for all the node pairs connected by an edge. The extra attraction for edges between class members and a virtual class node is defined as $k_{\text{class}} \cdot f_{\text{att}}(d)$, where the factor k_{class} controls the effect of class-mediated attractive force relative to the standard attraction.

Starting from their initial placements, the nodes are moved iteratively according to the attractive and repulsive forces until their changes become negligible or the maximal number of iterations is achieved. A temperature function controls how much any node can move during one iteration. Initially, a temperature of D/10 is used to allow greater movements, gradually decreasing to a constant temperature of N/30 during the late iteration steps. Poor layout solutions originating from early local minima can be avoided by using a modified distance $d' = \max\{d, 2S\}$ in the repulsive force of Equation (1) during the first half of the iterations. A stable solution, even for larger networks, can be obtained in a few seconds on a standard PC.

In the phase (2), the separation shift for a class node at location $\mathbf{p}$ is performed via the linear transformation:

$$\mathbf{p}' = \mathbf{p}_0 + k_{\text{sep}}(\mathbf{p} - \mathbf{p}_0), \tag{2}$$

where $\mathbf{p}_0$ is the center of gravity calculated over all the nodes. The factor k_{sep} is relative to the position of the class node furthest from $\mathbf{p}_0$ determined in the initial layout phase (1). The non-virtual nodes are fixed during the separation phase, but their final placement is determined in the layout phase (3). As the layout process involves randomized starting positions and shifts during the iteration steps, its results may differ between runs. However, the main characteristics of the layout solutions are reproducible.

4 USAGE AND APPLICATION

Besides the main layout parameters, k_{att} (called density in the graphical user interface), k_{class} (intra-group attraction), k_{sep} (inter-group distance), the user can freely adjust several other parameters, such as the number of iterations performed in the two layout phases and whether the location of a class node is free or fixed in the final layout phase (3). The high speed of the layout algorithm and the convenient graphical user interface makes it easy to experiment with different parameter settings. All the advanced filtering and viewing options of Cytoscape are also available in the process to produce customized, visually pleasing network layouts.

The layout algorithm is demonstrated on an example network, recovered from two protein interaction databases (Guldener *et al.*, 2006; Stark *et al.*, 2006). The network nodes shown in Figure 1A were first used to search for enriched GO terms with BiNGO, and a

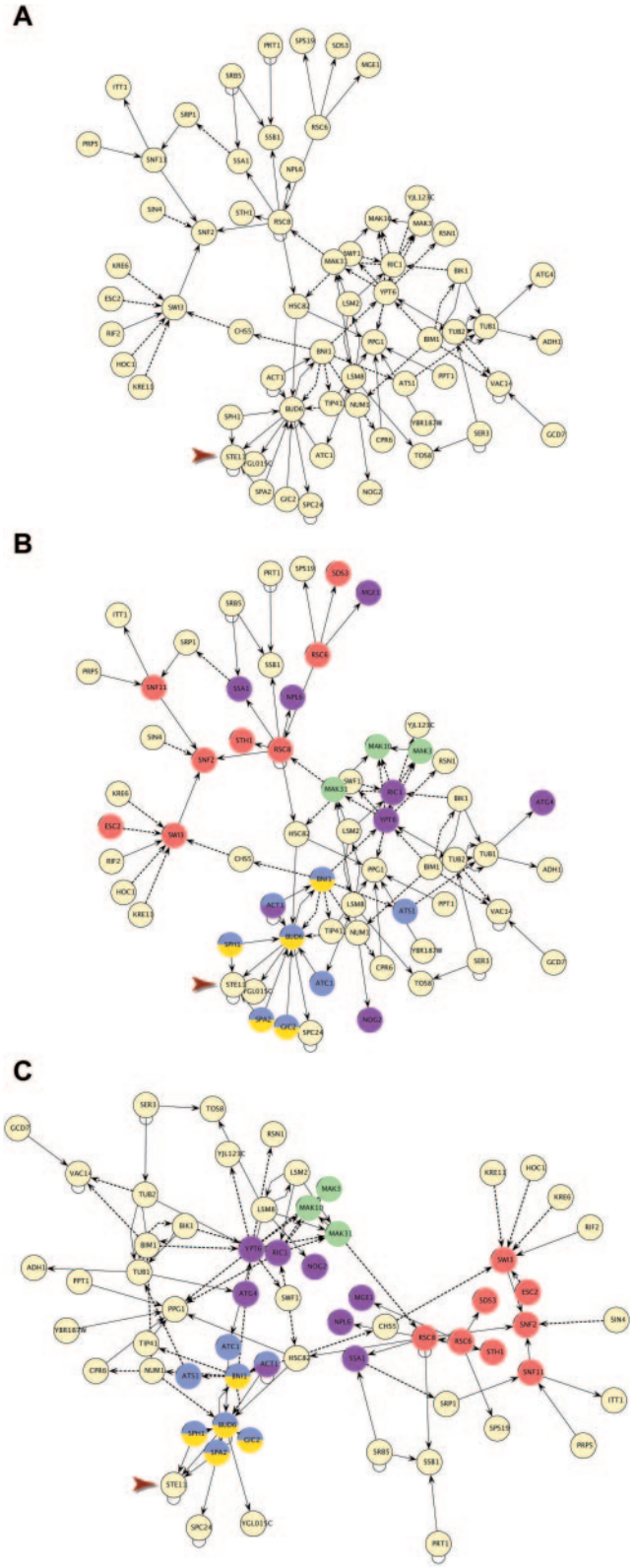

Fig. 1. A network of physical (continuous edges) and genetic (dashed edges) interactions, visualized using (**A**) Cytoscape spring-embedded layout algorithm, (**B**) supplemented with GO category-specific coloring and (**C**) using the GOlorize plug-in to direct the node placement process.

few terms were manually selected in GOlorize. Next, colors were assigned to the nodes in the network as follows: green, N-terminal protein amino acid modification; red, chromatin remodeling; blue, cell budding; yellow, Rho protein signal transduction; magenta, protein localization (Fig. 1B). Finally, the selected GO categories were used to organize the network layout using the class-directed node placement (Fig. 1C).

The example case shows that GOlorize is able to group the nodes of the same class together, while the nodes with multiple annotations lie typically in between their main classes (e.g. ACT1). Unclassified proteins are placed according to their connection structure (white nodes), suggesting hypothetical functional roles. For instance, while there are three interactions of Ste11 to the proteins annotated as involved in Rho protein signal transduction and cell budding (indicated by arrowheads), these associations are obvious only after application of the GOlorize plug-in.

Both Cytoscape and GOlorize are platform-independent Java applications. Further system requirements, installation instructions, and a walk-through tutorial of the GOlorize user interface as well as its future updates can be found from the user guide, which is available at http://www.cytoscape.org/plugins2.php

ACKNOWLEDGEMENT

Funding to pay the Open Access publication charges for this article was provided by NIH Grant 1 R01 GM070743

Conflict of Interest: none declared.

REFERENCES

Fruchterman,T.M.J. and Reingold,E.M. (1991) Graph drawing by force-directed placement. *Software—Practice and Experience*, **21**, 1129–1164.

Guldener,U. *et al*. (2006) MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Res*., **34**, D436–D441.

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

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

Stark,C. *et al*. (2006) BioGRID: a general repository for interaction datasets. *Nucleic Acids Res*., **34**, D535–D539.