# Visualizing patterns in Node-link Diagrams

Antoine Lambert, François Queyroi, Romain Bourqui

▶ **To cite this version:**

Antoine Lambert, François Queyroi, Romain Bourqui. Visualizing patterns in Node-link Diagrams. 16th International Conference on Information Visualisation (IV'12), Jul 2012, Montpellier, France. pp.48-53. hal-00744965

**HAL Id: hal-00744965**
**https://hal.science/hal-00744965**

Submitted on 24 Oct 2012

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Visualizing patterns in Node-link Diagrams

A. Lambert, F. Queyroi, R. Bourqui
*LaBRI, University of Bordeaux 1 and Inria Bordeaux Sud-Ouest*
*Bordeaux, France*
{*antoine.lambert, francois.queyroi, romain.bourqui*}*@labri.fr*

***Abstract*—Pattern discovery plays an important part in the graph analysis process. Good examples are the detection of communities in social networks or the clustering into pathways of metabolic networks.**

**However, elements may be shared by several clusters, making the patterns entangled. When mining such data, experts are usually interested in both each individual cluster and their overlaps. Dedicated visualization methods are therefore necessary to efficiently support their exploration process.**

**In this article, we propose a new method that emphasizes patterns in a node-link diagram representation and allows to easily identify overlaps between these patterns as well. Our technique combines graph topology and embedding to compute concave hulls with holes surrounding the patterns of interest.**

***Keywords*-pattern visualization; graph analysis; overlapping clustering**

## I. Motivation

Analyzing large data sets often relies on the discovering of different patterns. It is the case with discrete data structures such as graphs which are essential to the study of relational data. A popular example of pattern discovery in graph analysis is the detection of *communities*, which corresponds to a partition of the nodes of the graph in highly connected sub-graphs. Community structure is an important component of social networks, web graphs or protein interactions networks[1]. Revealing this structure provides a better understanding of its dynamic. One can for example consider nodes having links with other different communities as mediators in the network. Community detection has then been very investigated over the last few years[2].

However, partitioning may be a simplistic way to analyze networks. For example, a person can be part of several groups such as those formed by their family or their co-workers and those relationships should be accessed by community detection methods. However, a simple graph partition is not likely to capture this hidden multi-modality that often appears in social networks. Many procedures[3], [4], [5] allow then to discover overlapping communities where a node can be shared between several clusters.

The visualization of such overlapping sets has been widely investigated in the past few years. One method is to create a spatial layout of the elements according to set membership. Simonetto *et al.* [6] use such an approach for generating topologically Euler-like diagrams where sets are then emphasized through the use of possibly concave, colored and textured hulls. Riche and Dwyer [7] also generate such kind of diagrams, arranged using a constraint-based graph layout technique, that only uses convex rectangular shapes for representing the sets. Their technique also address the issue of improving the readability of set intersections through dedicated visual encodings. But these techniques can not be applied when the set elements have a semantically spatial organization, for instance a geolocalized layout. Methods have also been proposed to represent sets over existing visualizations and thus avoiding any layout adjustments. Collins *et al.* [8] use continuous, possibly concave, isocontours to delimit set memberships. Another visual metaphor is used by Alper *et al.* [9] where they generate a curve connecting all the elements of a set.

Concerning the visualization of clusters in node-link diagrams, the common and simple approach is to surrounding them with convex hulls ([10], [11]). However, this method can lead to ambiguities as items that are not set members can lie in the corresponding convex region. Techniques introduced in [7] and [9] can be used but they only focus on representing a secondary relationship on the nodes of a network. Consequently they are not adequate for visualizing the intersections of clusters defined by connected sub-graphs. Only the method in [8] could be used to fulfill that task but it does not scale well where there is a large number of overlapping clusters as it can be tedious to distinguish the multiple generated set boundaries.

We present here a new method of complex patterns visualization in node-link diagrams. Our technique provides both the ability to have a good overview of the different patterns over the network as well as an effective detailed analysis. Moreover this technique does not modify the actual layout given to the graph.

The paper is organized as follows : a quick overview of our method is given in Section II. In Section III we provide precise computational details. We present applications to real world examples in Section IV. Finally, we give conclusion and suggestions for future work in Section V.

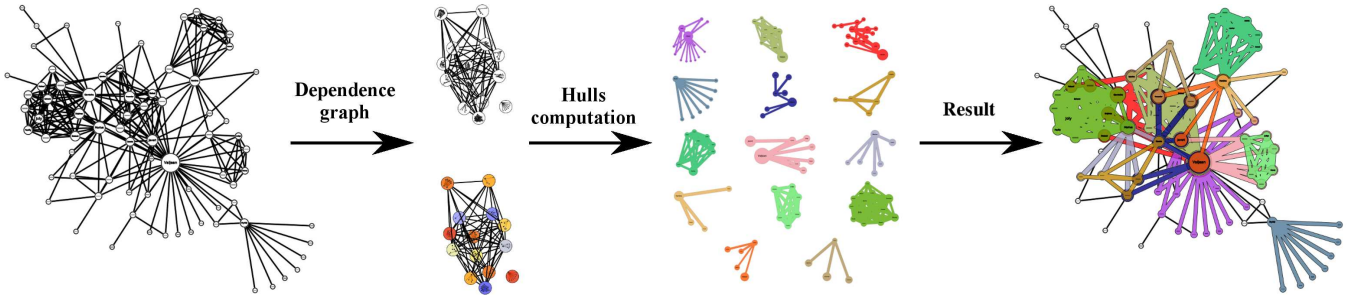



Figure 1: Illustration of the method pipeline: starting from a prescribed fuzzy clustering, we compute distinguishable hulls.

## II. Method overview

Figure 1 shows the main steps of our technique. On the left, one can see the co-occurrence network of the characters of *Les Misérables*[12] written by V. Hugo where the patterns of interest have been computed by the *Link Communities* algorithm[5].

First of all, a *dependence graph* such as defined in [13] is computed to model how the clusters or patterns to visualize overlap. In such graph, each node corresponds to a cluster/pattern in the original graph and two nodes are linked by an edge if and only if the corresponding clusters overlap.

Then, a dedicated graph coloring algorithm is applied to the dependence graph to assign to each cluster a positive value. When computing the concave hulls, that value is used to determine the distance in the plan between the cluster elements (nodes and edges) and its surrounding hull. By applying the coloring algorithm on the dependence graph, we can ensure that the hulls of overlapping clusters will be distinguishable by rendering them in decreasing order of their coloring value.

Even if our technique allows to visualize efficiently overlapping clusters in a network, due to the prescribed layout of the graph some ambiguities can remain. Indeed, when a cluster is dense and its layout implies a lot of edge crossings, it can be tedious to determine if a node lying in its hull really belongs to it. To solve these ambiguous cases, we had some simple interaction techniques. The first one consists in clicking on an element of the network and then displaying only the hulls associated to the clusters the element belongs to. The second one only displays the hull of a particular cluster after clicking on it. Figure 3 illustrates these techniques.

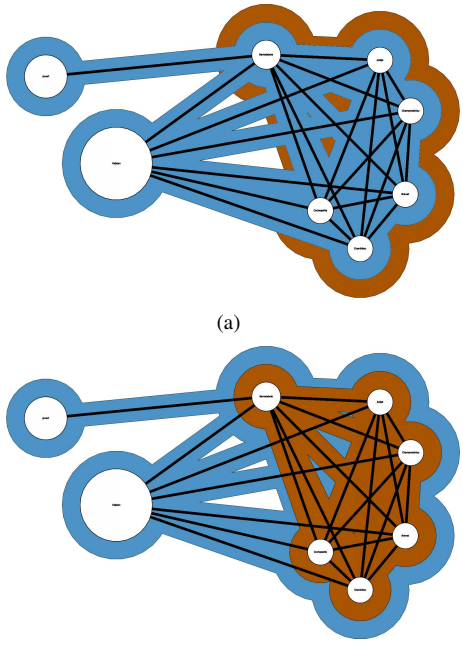
## III. Hulls computation

### A. Coloring the patterns

As mentioned above, we need to assign to each cluster a positive value so that two overlapping clusters have different values.

The first step is to compute a dependence graph corresponding to the original network and the prescribed fuzzy clusters (that may not cover the whole network). Running a *node proper coloring* algorithm on the dependence graph is clearly equivalent to assigning positive values to each cluster. Therefore any (node proper) coloring algorithm can solve this issue.

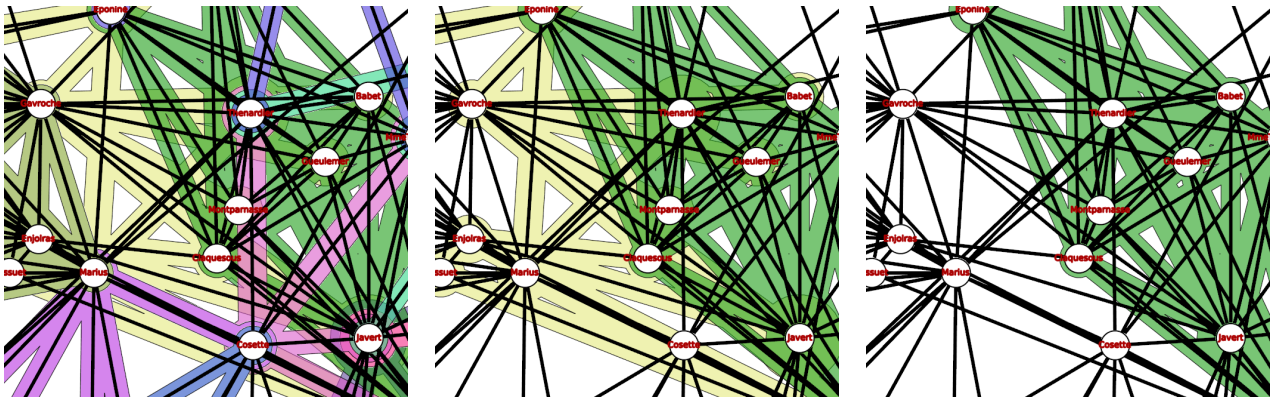
However, to ease the visualization of overlapping clusters but also to enable the visualization of nested clusters, a dedicated coloring algorithm is needed. For instance consider two nested clusters, to easily identify that one of the cluster is contained into the second, its hull must also be contained in the second hull (see Figure 2).



(a)

(b)

Figure 2: Illustration of nested clusters. (a) The value assigned to the deepest cluster is higher than the value of the second cluster. Thus the hull of the deepest cluster is larger than the other one making harder its identification. (b) When the deepest cluster has a lower value, containment of hulls shows clearly that one cluster is contained into the other.



(a) (b) (c)

Figure 3: Illustration of simple interaction techniques to solve possible ambiguous cases. (a) Original clusters visualization. On that example, it is hard to say if the "Montparnasse" node belongs to the green cluster. (b) After clicking on the "Montparnasse" node, only the hulls associated to the clusters it belongs to are displayed. We are now sure that the node belongs to the green cluster. (c) After clicking on the green hull, only it is displayed. We can see that the previous ambiguity is due to the density of cluster edges induced by the layout.

This is achieved by adapting the coloring algorithm so that the order induced by the coloring respects the order induced by the complexity of the clusters. In our case, we used the number of nodes to evaluate how complex a cluster is.

Given dependence graph $G = (V, E)$ and a node weight function $P$ (here, $P(u)$ is the number of elements in the cluster corresponding to node $u$), the following algorithm provides a coloring $C$ such that $\forall (u, v) \in E$ if $P(u) < P(v)$ then $C(u) < C(v)$.

This algorithm is based on a *breadth first search* (BFS) procedure and then runs in linear time.

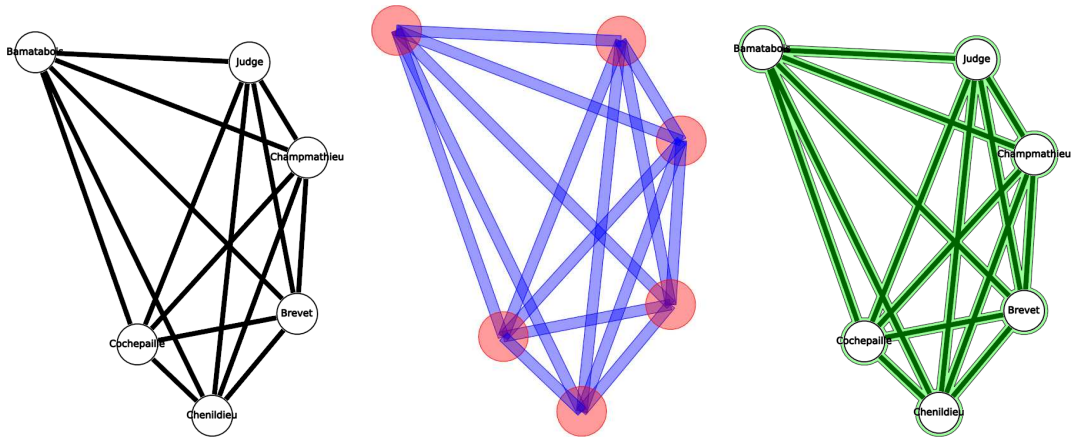
1) Start with $\forall u \in V, C(u) = P(u)$.
2) Do a BFS from an unvisited node $u$ adding in the queue only nodes $v$ having $P(v) = P(u)$. During this phase keep track of:
   - $\sigma(u)$ which is the connected component (maximal under inclusion) formed by $u$ and all nodes $v \in V$ having $P(u)$ as weight.
   - $maxL$ (resp. $minG$) which is the maximum value of $C$ lower than $C(u)$ (resp. the minimum value of $C$ greater than $C(u)$) in the direct neighborhood of $\sigma(u)$
3) Assign to all nodes of $\sigma(u)$ different real values in the range $]maxL, minG[$ if $minG \neq maxL$, in the range $[1, |\sigma(u)|]$ otherwise.
4) Mark all the nodes $v \in \sigma(u)$ as visited.
5) Repeat Step 2 until all nodes are marked as visited.

### B. Building a hull from graph topology and embedding

To visualize the different clusters in the global network context, we generate concave hulls that will surround them. As previously explained, the width of each hull has to be modulated with respect to the value given by the coloring algorithm described in the previous subsection.

Consequently, we need a hull generation method that can take precisely into account a spacing value between the skeleton of a cluster, defined by the layout of its nodes and edges, and the outer border of the hull. We first investigate the technique proposed in [8], which computes concave hulls through iso-surfaces extraction in an image. If that method can effectively produce aesthetic hulls, it does not fulfill our requirement as there is a lack of flexibility for modulating precisely the width of a hull.

Our solution is based on polygon clipping and works in the topological space. The idea is to compute the union of polygons built from the layout of the nodes and edges belonging to the subgraph to emphasize. The polygon associated to a node can be for instance a circle whose center is the node position and radius is defined from the node bounding box and the desired width of the hull to compute. The polygon associated to an edge consists in the extrusion of the polyline representing it parametrized by the desired hull width. We then compute the union of all these polygons through the help of the `Clipper` library [14], an efficient implementation of the Vatti polygon clipping algorithm [15]. Illustrations of that process are introduced Figure 4.




(a) (b) (c)

Figure 4: Illustration of the process for generating a concave hull that will surround a sub-graph. (a) Sub-graph to surround with concave hull. (b) Set of polygons on which to compute the union. Red circles are the polygons computed from the nodes layout, blue quads the polygons computed from the edges layout. (c) Resulting concave hull obtained after the process.

## IV. Case study

In this section, we present two real world examples. All the figures have been created using the Tulip framework[16].

### A. Co-occurrence network

Here we show the relevance of our visualization method to extract information from a fuzzy clustering of a co-occurrence network. We study the subgraphs created by the algorithm of [5] applied to the network *Les Misérables*[12] introduced earlier.

The algorithm of [5] does not only produce dense connected subgraphs as many clustering algorithms focus on. Here, we classify the subgraphs into three categories:

- **Densely connected subgraphs:** subsets of nodes having a high amount of edges between them.
- **Bipartite subgraphs:** subsets of nodes connecting densely connected subgraphs.
- **Tree subgraphs:** subset of nodes forming a tree, can also connect densely connected subgraphs.

Using our method we can display this classification by simply assigning a color to the hull according to the category the subgraph they surround falls into.

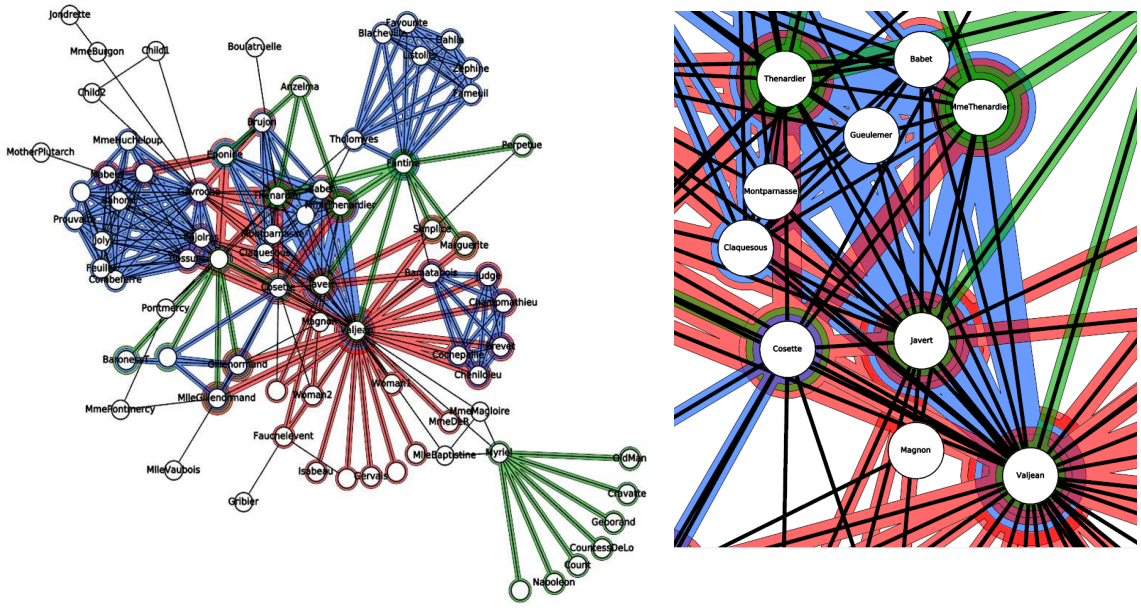
The results are shown in Figure 5. At a larger scale (see Figure 5(a)), we are able to visualize smaller subgraphs that overlap with bigger subgraphs. The different densely connected subgraphs (in blue) or tree subgraphs (in green) are visible. We can see for example that nodes belong to at most one densely connected cluster.

The center of the layout in Figure 5(a) is harder to visualize. It gathers several highly connected nodes belonging to bipartite subgraphs. We make a zoom on this area in Figure 5(b). Remember that adjacent subgraphs (having at least one node in common) are represented using hulls with different spacing. The different categories of subgraphs for each node are then visible. For example, we can see that both "Valjean" and "Javert" belong to four bipartite subgraphs: they play an important role of mediators in the network. We can also see that "MmeThenardier" belongs to the same clusters as "Thenardier" (his husband in the novel).

### B. Metabolic networks

The metabolism is the set of biochemical reactions that occur in a living system. Each reaction transforms a set of molecules (or metabolites) called substrates into other molecules called products. When interested to metabolism, one can consider different scales that vary according to the data and the biological questions. For instance, toxicologists often follow the degradation of a given molecule; in that case they focus only on a very small number of reactions. At a larger scale, biologists studying particular biological functions will focus on few sets of biochemical reactions, called *metabolic pathway*, each of them allowing the organism to perform one of these specific function. Finally the *metabolic network* results from the integration of all the metabolic pathways of an organism into a single network.

Figure 6 shows the yeast metabolic network drawn with the algorithm of Lambert *et al.* [17] with all the pathways emphasized using our method. That network originally contained 838 nodes (423 metabolites, 415 reactions) and 938 edges spread over 164 pathways and all elements belonging to more than 3 pathways have been duplicated resulting in a network of 1360 nodes and 1430 edges. The interest of our pattern visualization technique on that network is that a large number of metabolites/reactions nodes and the edges connecting them are shared by several pathways. One can see in the zoomed view of the Figure that each pathway and their common elements can be clearly identified.




(a) Global View

(b) Zoom on a dense area

Figure 5: Application of our method to the co-occurrence network *Les Misérables*[12] to illustrate the fuzzy clustering given by the algorithm of [5]. The different subgraphs are divided into three categories : the dense subgraphs (blue), the star subgraphs (green) and the bipartite subgraphs (red).

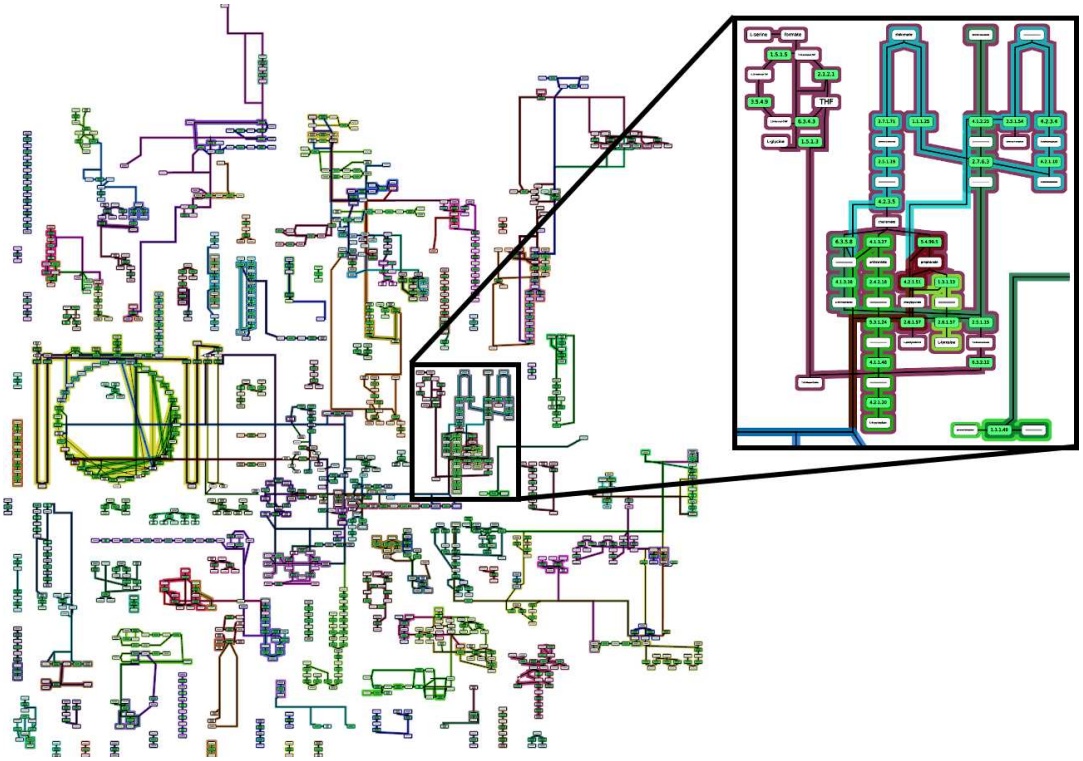


Figure 6: Yeast metabolic network where each of the 165 pathways have been emphasized. In the zoomed view, one can identify the different pathways and the common elements.

## V. Conclusion and future directions

In this paper we introduced a novel technique to visualize complex patterns in a node-link diagram view. It produces a layer of distinguishable hulls without modifying the actual layout of the studied network. Real world examples illustrate the effectiveness for our method to extract information at several levels of details. We believe this method to be of great interest for experts in domains where the layout of the network has a semantic value such as geography or biology. Hulls may overlap heavily if the graph layout is very compact. We proposed simple interaction techniques to address this issue.

Our approach is simple and the procedure is easy to implement. The computation time is relatively short thanks to the effectiveness of the `Clipper` library. Nevertheless, it could be further improved by the use of multi-threading since the computation is done independently for each subset of nodes.

As future work, we plan to conduct an experimental study to compare our method to the other existing ones and validate its effectiveness. Finally, other definitions of cluster complexity could be investigated with respect to domain experts' knowledge in order to tune the hulls coloration step.

## References


[1] M. Girvan and M. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, p. 7821, 2002.

[2] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[3] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[4] A. Lancichinetti, F. Radicchi, J. Ramasco, and S. Fortunato, "Finding statistically significant communities in networks," *PloS one*, vol. 6, no. 4, p. e18961, 2011.

[5] Y. Ahn, J. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, pp. 761–764, 2010.

[6] P. Simonetto, D. Auber, and D. Archambault, "Fully automatic visualisation of overlapping sets," in *Computer Graphics Forum*, vol. 28, no. 3. Wiley Online Library, 2009, pp. 967–974.

[7] N. H. Riche and T. Dwyer, "Untangling euler diagrams," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1090–1099, Nov. 2010.

[8] C. Collins, G. Penn, and S. Carpendale, "Bubble sets: Revealing set relations with isocontours over existing visualizations," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 15, no. 6, pp. 1009–1016, 2009.

[9] B. Alper, N. Riche, G. Ramos, and M. Czerwinski, "Design study of linesets, a novel set visualization technique," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 12, pp. 2259–2267, 2011.

[10] J. Heer and D. Boyd, "Vizster: visualizing online social networks," in *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, oct. 2005, pp. 32 –39.

[11] T. Dwyer, K. Marriott, F. Schreiber, P. Stuckey, M. Woodward, and M. Wybrow, "Exploration of networks using overview+detail with constraint-based cooperative layout," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 1293–1300, November 2008.

[12] D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing*. New York, NY, USA: ACM, 1993.

[13] R. Bourqui, D. Auber, V. Lacroix, and F. Jourdan, "Metabolic network visualization using constraint planar graph drawing algorithm," in *Pro. of the 10th Int. Conf. on Information Visualisation (IV'06)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 489–496.

[14] A. Johnson, "Clipper - an open source freeware polygon clipping library." http://www.angusj.com/delphi/clipper.php.

[15] B. R. Vatti, "A generic solution to polygon clipping," *Communications of the ACM*, vol. 35, pp. 56–63, July 1992.

[16] D. Auber, D. Archambault, R. Bourqui, A. Lambert, M. Mathiaut, P. Mary, M. Delest, J. Dubois, and G. Mélançon, "The Tulip 3 Framework: A Scalable Software Library for Information Visualization Applications Based on Relational Data," INRIA, Technical report RR-7860, Jan. 2012. [Online]. Available: http://hal.inria.fr/hal-00659880

[17] A. Lambert, J. Dubois, and R. Bourqui, "Pathway Preserving Representation of Metabolic Networks," *Computer Graphics Forum*, vol. 30, no. 3, pp. 1021–1030, 2011.