Visualization of heterogeneous networks

Analysis report

Problem Description - Introduction

For this visualization, the given data are relational data which are networks or graphs. The networks are heterogenous which means that there are several types of the nodes and/or edges. The types of the nodes and edges are stored as a categorical attribute. And this attribute should be mapped on the visual representation in the visualization.

An example of such data is a network that represents interactions in a software repository. There are several node types such as People, Tickets, Commits and Wiki pages and several edge types that carry information about authorship (i.e., person is author of a commit, a ticket, or a wiki page) and assignment (i.e., person is assigned to a ticket or is responsible for a ticket).

Additionally, nodes and edges can have attributes that can be included into visualization.

- Nodes
  - Categorical (nominal) attributes
    - Name of a person, ticket, commit of a wiki page.
    - Email and role of a person
    - Status of a ticket (e.g., opened, in progress, done)
  - Ordinal attributes
    - Priority of a ticket (e.g., low, middle, high)
    - Creation date of a ticket
    - Due date of a ticket
  - Quantitative attributes
    - Time spent on a ticket (in hours)
    - Progress of a ticker (in percent)
- Edges
  - Categorical (nominal) attributes
    - Name of a connection

The goal of the visualization is to help users to **identify clusters of the network** and provide a **detail of the subnetwork** for the selected clusters. As a cluster, one can imagine, for example, a group of people interacting between each other but not the others, or a person working on a subset of tickets. So, it can be interesting to obtain insights from the data and make useful conclusions.

It is a top-down approach, which means that users should be able to see an overview of the data and then further explore parts of the network that are of interest. This can be done either in a single static view that shows all clusters with associated details or as an interactive view using **overview and detail technique**. This technique creates a view made

of two panels where one panel shows overview of all clusters and the other shows a detail of a selected cluster based on a request from the user.

Existing Visualization Techniques

There are several approaches for visualizing networks. But not all of them are suitable for the given data. For instance, visualization of hierarchies (such as a treemap algorithm) is not suitable because the given data contains cycles, while hierarchy visualization expects trees or DAGs (directed acyclic graphs).

A useful group of visualizations are **node-link diagrams** (see fig. 1) that map nodes to points in a space (often in 2D but also in 3D) and map edges to lines that connect two points that represent nodes connected by the respective edge. This is one of the most frequently used visualizations of networks. However, it has some drawbacks. Node-link diagrams do not scale well with growing numbers of nodes and edges (see fig. 2). And there may arise issues with edge crossings and occlusion of nodes (see fig. 1) [1][2]. To tackle this, there was introduced an interesting approach called **adjacency matrix visualization**.

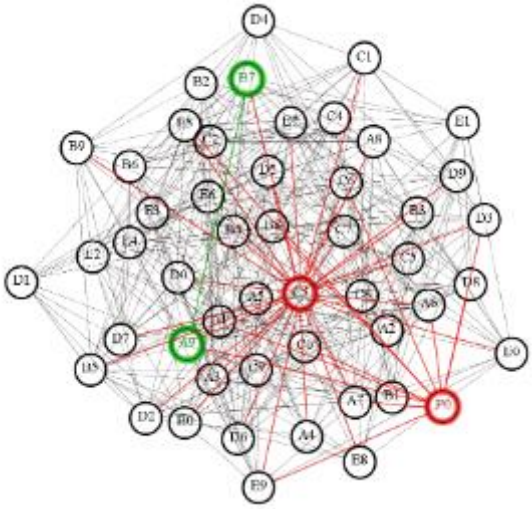

Fig. 1: Node-link visualization example. Source [1]

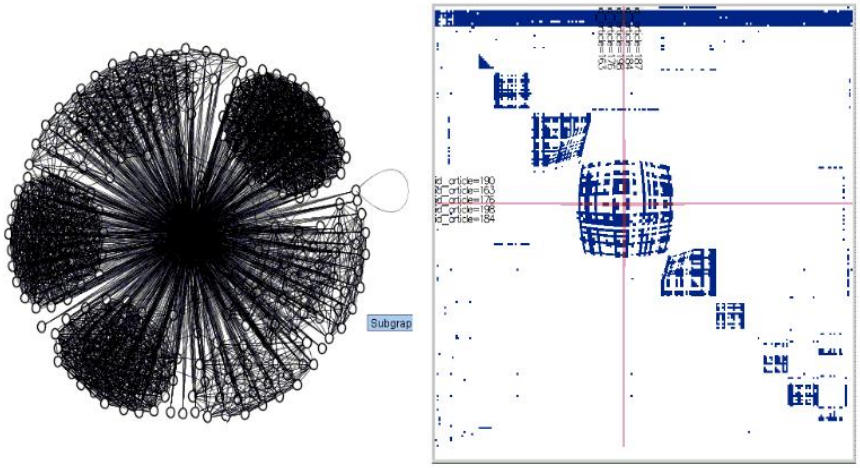

Fig. 2: Comparison of Node-link and Adjacency matrix visualizations on a large graph. Source [2]

Adjacency matrix visualization shows the network as a table in a form of adjacency matrix (see fig. 3). In the original form, the visualization does not have to bear any information. And the key challenge is to reorder rows and columns to obtain patterns along the diagonal of the matrix that form clusters (see fig. 4). However, the drawback is that the visualization can be hard to interpret.

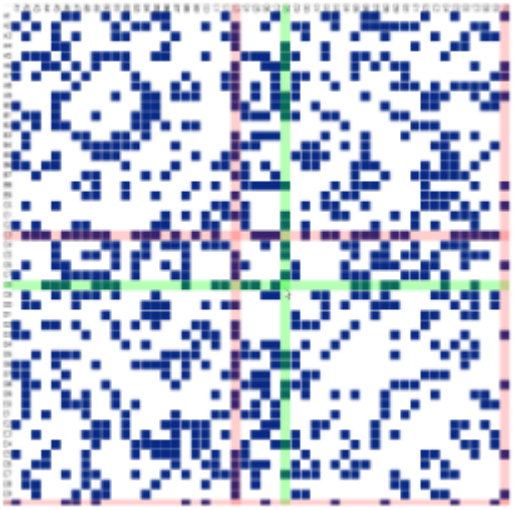

Fig. 3: Adjacency matrix visualization example. Source [1]

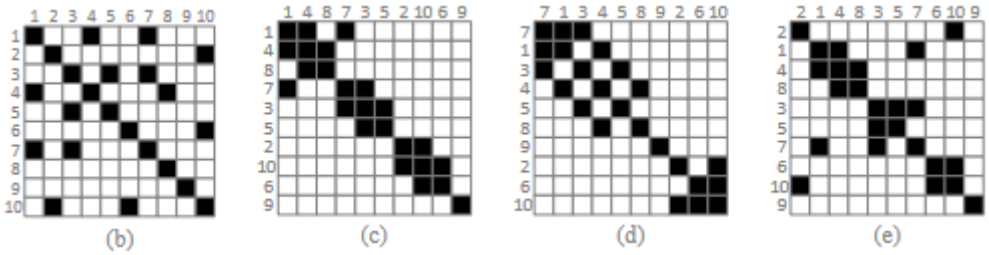

Fig. 4: Adjacency matrix reordering example where (b) shows random order of rows and columns while (c-e) have orders created by algorithms. Source [3]

Node-link vs Adjacency matrix visualizations

To conclude the description of node-link and adjacency matrix approaches, there is the following comparison of advantages and disadvantages of both approaches and a list of tasks that each approach solves.

	Node-link	**Adjacency matrix**
Advantages	Familiar	No edge crossing
	Compact	No node overlapping
	Effective for small or sparse	Fast navigation

	graphs	
		Effective for large graphs
Disadvantages	Edge crossing	Less familiar
	Node overlapping	Uses more screen space
		Path following can be difficult
Challenge / Task	Layout needs to be created	Rows and columns need to be reordered

Adjacency matrix visualization

The semester project will be focused on adjacency matrix visualization and therefore the rest of the analysis report will cover details about this technique.

In this section, there are described visual patterns and anti-patterns that can arise when reordering rows and columns and an explanation how to interpret them. And there are listed reordering methods.

Patterns and anti-patterns

When reordering rows and columns in an adjacency matrix, there can arise four patterns and two anti-patterns [3]. Naturally, visual patterns are preferred, but even anti-patterns can be helpful and reason that the network is missing topological structure.

Block pattern - pattern that indicates clusters or groups. For this visualization task, it is the most wanted pattern. In a graph theory such groups are referred to as strongly connected components and in mathematics such a matrix is called block-diagonal matrix.

The example below (see fig. 5) indicates four clusters with nodes [1, 2, 3, 4], [5, 6, 7], [8, 9] and [10]. This example has perfect clusters, which means that each node in a cluster has an edge to other nodes in the same cluster. However, in real data block patterns can have missing cells (i.e. missing edges between nodes in a same cluster) or nodes connected to other clusters.

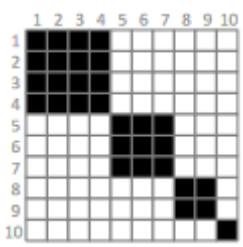

Fig. 5: Block pattern example. Source [3]

Off-diagonal Block pattern - pattern that indicates clusters in a bipartite graph or sub-patterns that arise in addition to block pattern (see fig. 6).

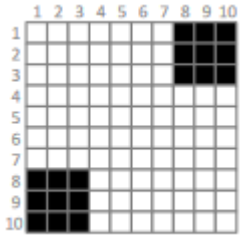

Fig. 6: Off-diagonal Block pattern example. Source [3]

Line/Start pattern - pattern in a form of continuous straight horizontal and vertical lines (see fig. 7). It indicates that the node is strongly connected to several other nodes.

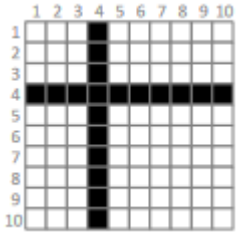

Fig. 7: Line/Start pattern example. Source [3]

Bands pattern - pattern in a form of continuous straight oblique lines (see fig. 8). It refers to paths and cycles in the network.

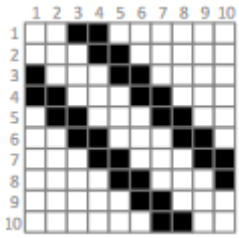

Fig. 8: Bands pattern example. Source [3]

Noise anti-pattern - anti-pattern (see fig. 9) that shows that the network does not have any underlying structure or the matrix reordering method is not able to reveal patterns.

This anti-pattern can arise globally in an entire matrix (and graph) or only locally in a submatrix (and subgraph).

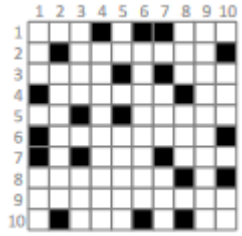

Fig. 9: Noise anti-pattern example. Source [3]

Bandwidth anti-pattern - anti-pattern in a form of continuous curved oblique lines (see fig. 10).

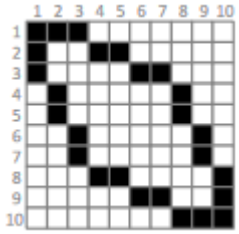

Fig. 10: Bandwidth anti-pattern example. Source [3]

More details and explanations about visual patterns and anti-patterns can be found in a paper **Matrix Reordering Methods for Table and Network Visualization** [3]. Furthermore, the paper contains an overview of matrix reordering methods and examples which methods can lead to which patterns or anti-patterns.

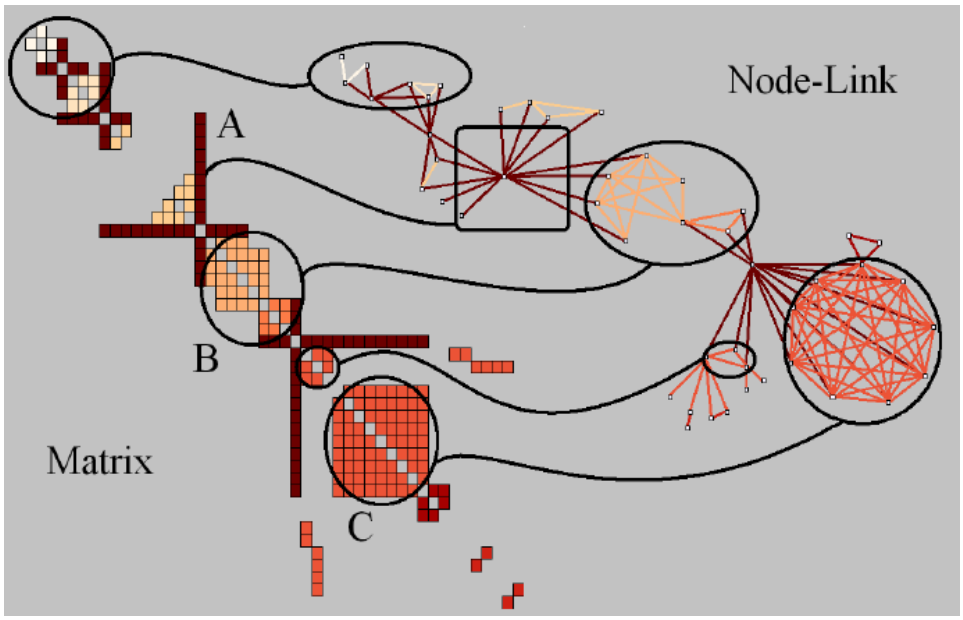

Fig. 11: Visual patterns in adjacency matrix shown as node-link diagrams. Source [2]

Matrix reordering methods

The matrix reordering task is needed in different domains and over years researchers from different fields have developed various algorithms and methods. As it can be difficult to navigate between different domains and fields, authors of the paper **Matrix Reordering Methods for Table and Network Visualization** [3] have collected overview of such algorithms and have described how the methods work and for what kinds of data they should be used on. In this section below, there is described a short overview of methods with emphasis on algorithms that should be useful for the given data.

Families of ordering methods [2] [3]:

- Robinsonian (statistics) - idea is to reorder a matrix so that similar rows are arranged close and dissimilar rows are placed farther apart.

    - **Bipolarization algorithm** (greedy algorithm)
    - **Hierarchical clustering algorithms** - e.g., agglomerative clustering with average linkage. Should be used for the clustering tasks.
    - **Optimal-Leaf-Ordering Algorithms** - algorithm tends to produce visually coherent and well-organized block-diagonal forms.
- Spectral methods - utilize eigenvalues and eigenvectors to calculate a reordering, i.e., each row (or column) is projected into the eigenspace where distances between eigenvectors are used to calculate a reordering.
    - **Rank-two Ellipse Seriation**
- Dimension reduction - utilize Principal Component Analysis or Single Value Decomposition
- Heuristic approaches - offer a good tradeoff between time and quality of the visualization.
- Graph theoretic - approaches that exploit the graph structure.
    - **Reverse Cuthill-McKee** - fast algorithm to apply.
    - **Traveling Salesperson Problem**
- Bi-clustering - apply clustering in of rows and columns simultaneously.

Path finding visualization for Adjacency Matrices

One of the disadvantages of adjacency matrix visualization are difficulties in path following. I.e., seeing visual patterns in the adjacency matrix is not enough to explore paths between nodes and for humans it can be very confusing. There is an interesting paper called **Path Visualization for Adjacency Matrices** [4] where authors try to tackle this issue.

Possible solution for path visualization is to add additional visual elements such as lines (straight or curved) between cells of the adjacency matrix that indicate path between nodes. As an example, authors of the paper present social (friendship) network data and the task is to find friendship connections between Irv and Walt (see fig. 12).

One of the possible disadvantages (depends on the application of the visualization) is that the visualization needs to be interactive, and the user needs to select start and end nodes of the path (i.e., apply different filters in the visualization pipeline) and recreate the visualization.

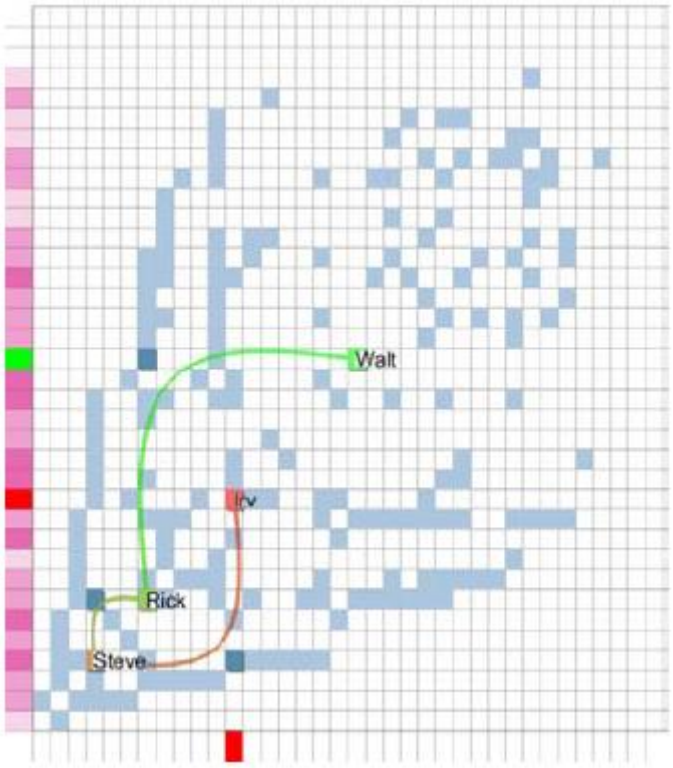

Fig. 12: Example of path finding visualization in adjacency matrix. Source [4]

Path finding visualization will not be included in the semester project, but it can be interesting to review and apply in future projects.

Conclusion

In this analysis report, there is described problematics of visualization of heterogeneous networks with emphasis on clustering.

In the beginning there is an analysis of the input data and the visualization task.

Further, the analysis focuses on existing visualization techniques. It describes node-link diagrams and adjacency matrix visualization and shows advantages, disadvantages, and challenges of both methods.

The semester project will be focused on adjacency matrix visualization. Therefore, the report describes aspects of adjacency matrix visualization in more detail. There are shown patterns and anti-patterns that can arise in the adjacency matrix. And there is listed an overview of the methods that can be employed for reordering of columns and rows of the adjacency matrix.

In the end, this information should be sufficient to implement custom adjacency matrix visualization for heterogeneous networks that would help users to identify the clusters.

References

[1]: Ghoniem, Mohammad & Fekete, Jean-Daniel & Castagliola, Philippe. (2004). **A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations**. IEEE Symposium on Information Visualization. 17-24. 10.1109/INFVIS.2004.1.

[2]: J. D. Fekete. **Matrix-Based Visualization of Graphs** keynote presentation. IEEE VIS 2014 in Paris, https://aviz.fr/~fekete/pmwiki/uploads/Main/keynote-gd2014.pdf

[3]: Behrisch M., B. Bach, N. Henry Riche, T. Schreck, and J. D. Fekete. **Matrix reordering methods for table and network visualization**. In Computer Graphics Forum, 35(3):693-716, 2016.

[4]: Shen, Zeqian & Ma, Kwan-Liu. (2007). **Path Visualization for Adjacency Matrices**. Proceedings of Eurographics/IEEE-VGTC Symposium on Visualization (EuroVis). 83-90. 10.2312/VisSym/EuroVis07/083-090.