Sets Visualization using their Graph Representation

Abdalmunam Abdalla
Faculty of Sciences
Omar Al-Mukhtar University, Libya
abdalmunam.abdalla@case.edu

Mehmet Koyuturk
Faculty of Engineering
Case Western Reserve University, USA
mehmet.koyuturk@case.edu

Abdelsalam M. Maatuk
Faculty of Information Technology
Benghazi University, Libya
abdelsalam.maatuk@uob.edu.ly

Alfaroq O. Mohammed
Faculty of Sciences
Omar Al-Mukhtar University, Libya
fomf200@yahoo.co.uk

ABSTRACT

With the emergence of new data acquisition technologies, large amounts of data are available in many domains. While a significant amount of computational research is dedicated to the analysis of such data, it is needed to be visualized in a way that is easy to analyze and understand. Recently, there have been significant advances in visualizing graphs; however, not enough tools exist for automatic visualization of sets. In this paper, we devise a spectral approach for visualizing overlapping sets, so that the underlying hierarchy and relations of the sets can be easily understood by visual inspection. The algorithm utilizes the spectral decomposition of the graph representation of the sets to compute the best coordinates for all items on the Euclidean plane. The experimental results were very encouraging, and showed positive indication on the efficiency of the proposed method.

Keywords

Sets, visualization, sets visualization, overlapping sets, sets drawing.

1. INTRODUCTION

Visualization is a mean of representing data, so that data can be explored and understood by visual inspection. Visualization utilizes the human visual ability to allow users to understand visualized objects and their underlying relations. Visual representation of data is useful in providing abstract information about the data at once. Visualization is increasingly applied in many applications, including software engineering [1], imaging [2], digital libraries [3] and others.

In this paper, we devise algorithms to visualize overlapping sets, so the users (e.g. researchers) can easily and quickly have information about the underlying relationship between the sets. The visualization of sets can help to classify them and make decisions for further analysis. For example, we can see which subsets of the sets are heavily overlapping, so that we can study them together because they share a lot of information. In contrast, disjoint sets can be analyzed separately.

This problem is closely related to graph visualization and multi-dimensional scaling (MDS). However, this problem is fundamentally different in that we are also interested in visualizing the sets in addition to their items. In MDS, the distances between items are given and a low-dimensional mapping of the items is sought. In graph visualization, the pairwise relationships between items are given and a two-dimensional mapping of the items and the pairwise relationships is sought. On the other hand, this study intends to obtain a two-dimensional mapping of the items in such a way that we can visualize which items belong to which set. This requires optimization of the mapping of the items on the two-dimensional space, as well as optimization of the representation of the sets. Experiments of the algorithms were conducted on both real and synthesis datasets, and showed positive indication on the efficiency of the proposed method.

The rest of the paper is organized as follows. Section 2 provides an overview of related work. The problem is defined formally in Section 3. Section 4 presents our spectral approach for sets visualization based on the Laplacian of co-membership and bipartite graphs generated from the sets. The experimental results, performance and evaluation metrics used are briefly discussed in Section 5. Section 6 concludes the paper.

2. RELATED WORK

A great amount of research has been recently done on information visualization. In this section we will discuss the work related to set visualization, including visualizing sets, graph visualization, and multidimensional scaling.

Euler diagrams can be used to represent the relationships between sets. Many algorithms have been developed for drawing Euler diagrams. Flower and Howse [4] outlined well-formedness conditions on drawn diagrams and presented an algorithm to decide whether an abstract diagram is drawable under those conditions. If a diagram is diagnosed as drawable, then a drawing is produced. Later work [5] aimed to enhance the layout of an already drawn Euler diagram, using a hill-climbing based optimization approach in combination with a range of layout metrics, to assess the quality of the drawing.

Alsallakh et al. [16] proposed a technique for finding and analyzing different kinds of overlaps between sets using frequency-based representations. An algorithm based on shortest-path graphs to depict set membership of items on a map has been presented by Meulemans et al. [17] Lex et al. [18] introduced a novel visualization technique for the quantitative analysis of sets, their intersections, and aggregates of intersections. Simonetto et al. [6] developed another algorithm to generate Euler-like diagrams. These algorithms were mainly designed to draw diagrams for a

ICEMIS '15, September 24-26, 2015, Istanbul, Turkey
© 2015 ACM. ISBN 978-1-4503-3418-1/15/09□$15.00
DOI: http://dx.doi.org/10.1145/2832987.2833023

very small number of sets with small sizes. Here, in contrast to these approaches, we are considering the problem of visualizing a large number (tens to hundreds) of larger (tens of items) sets and we also consider undrawable cases. In order to handle undrawable instances, we allow errors.

An existing tool, daVinci [7] is used as a user interface for graph layout in many applications. Luo et al. [8] proposed an ambiguity-free edge- bundling technique to improve the visualization of very dense graphs. Many spectral approaches for graph visualization exist, that use the eigenvectors of the graph matrix to produce a mapping of the graph vertices to the Euclidean space [9, 10]. Koren[11] developed an algorithm that uses the eigenvectors of the Laplacian to visualize graphs. Here, we are using a graph-drawing algorithm to visualize overlapping sets. The methods we develop are based on spectral decomposition of the graph representation of sets. They utilize the eigenvectors of the Laplacian of the graph to compute the best Euclidian coordinates of the items of the sets.

MDS methods are used to decrease the dimensionality of data while preserving as much information as possible about these data.

Leeuw and Mair [12] proposed a solution to the multidimensional scaling problem by means of the majorization algorithm. Their method is intended to minimize the stress and functions to majorize stress were elaborated. Agarwal et al. [13] introduced an iterative local improvement method for solving many variants of multidimensional scaling problems. The algorithm starts by choosing a point and moving it so that the cost function is locally optimal and repeats the procedure until convergence is achieved. Chen and Buja [14] proposed a local multidimensional scaling method that constructs a global embedding from local information. The method localizes versions of MDS stress functions by using force paradigm and a tuning parameter.

In this work, in addition to the two-dimensional mapping of the items on the Euclidian plane, we also require optimization of the representation of the sets, so that we can visualize the items and sets together.

3. PROBLEM DEFINITION

Formally, we have n items grouped into m sets where $Si = \{Ij\}$, $1 \leq i \leq m, 1 \leq j \leq n$. We want to plot these items on the Euclidean plane and group them in a way that makes it easy to see the relationship between these items. Without loss of generality, it is required that the co-membership graph produced by the sets to be visualized is connected. If the graph is not connected, we pre-process the data and provide multiple connected graphs and visualize them separately. This is natural because we are interested in visualizing the overlap between sets, so we do not have to visualize disjoint sets in the same drawing.

Example 1: Consider the following group of sets and items. We refer to the *i-th* set as S_i and the *j-th* item as I_j.

$S_1 = \{I_1, I_2, I_3\}$

$S_2 = \{I_1, I_2, I_5\}$

$S_3 = \{I_3, I_4\}$

These sets need to be visualized, so that each item is represented by a point, and each group of items belong to a set is bounded by a circle that represents that set.

4. PROPOSED ALGORITHMS

This section presents our method for visualizing overlapping sets. The algorithm regards the sets as a graph and visualizes them using the Laplacian of the graph. Namely, we use the eigenvectors to compute the optimal position for the graph nodes (items) as well as the circles that represent the sets in the two-dimensional plane. Drawing the sets by randomly assigning values to the x and y coordinates of each item in the sets does not provide useful information about the sets and their relations. Even with a very small number of items the drawing does not represent the dataset correctly.

In this study, we have proposed several algorithms based on the spectral decomposition of the graph representation of sets. We first start with the algorithm, which is based on a co-membership graph, and then we perform some improvements to this algorithm.

4.1 Algorithm based on co-membership graph representation of sets (CMG)

Since many of the existing algorithms aim to visualize graphs, we first represent the sets to be visualized as a graph. For this purpose, we construct the co-membership graph of the sets. Figure 1 shows the co-membership graph of the Example 1 given in Section 3.

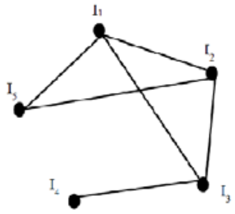

Figure 1. Co-membership graph of the three sets in Example 1

To construct the graph, we first construct the membership matrix M of the group of items. M is an $m \times n$ matrix, where m is the number of sets and n is the number of items. If item I_j is contained by set S_i, the entry M_{ij} is set to one, otherwise, it is set to zero. For example, the membership matrix for the sample instance given in the previous section is the following:

$$M = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \qquad (1)$$

From the membership matrix, we can construct the adjacency matrix U of the co-membership graph G. U is an $n \times n$ matrix, where each entry U_{ij} is set to one if both items i and j appear together in at least one set, or it is set to zero if the two items do not share a set. Besides, all the diagonal entries U_{ii} are set to zero. The adjacency matrix of Example 1 is as follows:

$$U = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \qquad (2)$$

This matrix is called the unweighted adjacency matrix, and it can be computed directly from M using the following equation:

$$U = M^T \otimes M \qquad (3)$$

Here, the operator $\otimes$ specifies a modified matrix multiplication, in which multiplication is replaced by logical

"AND" and the addition is replaced by logical "OR". If both items *i* and *j* appear in the same set more than once (appear together in more than one set), it might be useful to include that information in the co-membership matrix, since the items that appear in many sets together should be located close to each other in the two-dimensional plane. Therefore, we define a weighted co-membership matrix *W*, in which the entry W_{ij} will be equal to the number of sets that contain the two elements together. Once more, all the diagonal entries W_{ii} are set to zero. We can easily compute *W* directly from *M* by the following equation:

$$W = M^T \times M \qquad (4)$$

The weighted adjacency matrix for Example 1 is the following:

$$W = \begin{bmatrix} 0 & 2 & 1 & 0 & 1 \\ 2 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \qquad (5)$$

We then construct the Laplacian matrix L of the co-membership graph. The Laplacian matrix is an $n \times n$ matrix where $L = D - A$ and *D* is the degree matrix, which is an $n \times n$ diagonal matrix where D_{ii} = deg(*i*). The Laplacian L can be computed from the unweighted adjacency matrix *U* as follows:

$$L_{ij}^{(U)} = \begin{cases} \deg(i) & i = j \\ -U_{ij} & i \neq j \end{cases} \quad i,j = 1,2,\ldots,n \qquad (6)$$

Where deg(*i*) in this equation denotes the degree of item *i* (the number of items that appear with item *i* in at least one set). Similarly, the Laplacian can be computed from the weighted adjacency matrix *W* as follow:

$$L_{ij}^{(W)} = \begin{cases} \mathrm{wdeg}(i) & i = j \\ -W_{ij} & i \neq j \end{cases} \quad i,j = 1,2,\ldots,n \qquad (7)$$

Where wdeg(*i*) in this equation denotes the weighted degree of item *i* (the sum of the weights of the edges incident to *i*).

Using the Laplacian is very useful in that we convert the problem to an optimization problem. Consider the problem of mapping the nodes of a graph onto one-dimensional Euclidian space such that the nodes, which are connected with heavier edges are closer to each other in the space. This problem can be formulated as follows:

$$\min_x E(x) \overset{\text{def}}{=} \sum_{\langle i,j \rangle \in E} w_{ij}(x(i) - x(j))^2 \qquad (8)$$

The right-hand-side of the above equation can be written in matrix form as follows:

$$x^T L x = \sum_{i<j} w_{ij}(x_i - x_j)^2 \qquad (9)$$

Therefore, since L is a positive semi-definite matrix, the eigenvector corresponding to the second smallest eigenvalue of the Laplacian provides the optimal solution to this problem. Similarly, the optimal solution to the problem of mapping graph nodes can be computed into two-dimensional Euclidean space by taking the eigenvectors that correspond to the second and third smallest eigenvalues of the Laplacian. In Example 1, if we use the unweighted adjacency matrix for the co-membership graph, the Laplacian will be as follows:

$$L^U = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ -1 & -1 & 0 & 0 & 2 \end{bmatrix} \qquad (10)$$

For the weighted adjacency matrix, the Laplacian is the following:

$$L^W = \begin{bmatrix} 4 & -2 & -1 & 0 & -1 \\ -2 & 4 & -1 & 0 & -1 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ -1 & -1 & 0 & 0 & 2 \end{bmatrix}. \qquad (11)$$

Koren [11] showed that the eigenvectors of the Laplacian are very useful in providing a comprehensible visualization of graphs. Koren [11] also showed that using the degree-normalized eigenvectors of the Laplacian provided more natural visualization than using just the eigenvectors of the Laplacian. Normalization means that we consider the degree when we compute the eigenvectors from the Laplacian. Here we require that

$$x^T D x = 1 \text{ and } y^T D y = 1 \qquad (12)$$

Further details on deriving the eigenvectors from the Laplacian matrix can be found in [11]. In order to comprehensively investigate the use of co-membership graphs in visualizing sets, we use four variants of the co-membership graph *G*: unweighted, weighted, unweighted normalized and weighted normalized.

After computing the optimal *x* and *y* coordinates for all the nodes (items) of the graph, Circles are used to represent the sets. Each group of items belonging to one set is bounded by a circle. In order to draw these circles, we need two quantities: the center of each circle and the radii of the circles. For each circle (set), the average of the *x* coordinates of the items that belong to that set is computed and used as the *x* coordinate of the center of the circle representing that set. Similarly, the *y* coordinates for each circle (set) are computed. The radius of the circle is the distance between the center and the furthest item from the center in the set. Computing the radii in such way guarantees that all the items in a set are bounded by the circle representing that set, allowing no false negatives, but allowing some items that are not in the set to be visualized as if they have been in the set.

4.2 Algorithm based on bipartite graph representation of sets (BPG)

In the CMG algorithm, there are some cases, in which two instances of different datasets produce the same co-membership graph. For instance, consider the following group of sets:

$S_1 = \{I_1, I_2, I_3\}$, $S_2 = \{I_1, I_5\}$, $S_3 = \{I_2, I_5\}$, $S_4 = \{I_3, I_4\}$.

This instance produces the same co-membership graph as in Example 1 shown in Figure 1. In other words, different distribution of items to sets might give rise to the same co-membership graph, causing loss of information. To overcome this problem, we proposed an algorithm that is based on the bipartite graph representation of the sets to be visualized, and then compute the Laplacian and eigenvectors. Consequently, the coordinates for the items and the centers of the circles representing the sets can be directly computed. In this algorithm, we use the following equation to compute the adjacency matrix of the bipartite graph:

$$A^T = \begin{bmatrix} 0 & M \\ M^T & 0 \end{bmatrix} \qquad (13)$$

Where M is the membership matrix and the size of A is$m + n \times m + n.$ Note that the zero in the upper left corner of the matrix is an $m \times m$ zero matrix and the zero in the lower right corner is $n \times n$ zero matrix. The adjacency matrix of the bipartite graph in our example is

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \qquad (14)$$

Note that, here, we do not need edge weights since each entry in the adjacency matrix corresponds to exactly one set-member relation. After we construct our adjacency matrix, the Laplacian and the eigenvectors can be computed. The Laplacian for this adjacency matrix becomes:

$$L = \begin{bmatrix} 3 & 0 & 0 & -1 & -1 & -1 & 0 & 0 \\ 0 & 3 & 0 & -1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 2 & 0 & 0 & -1 & -1 & 0 \\ -1 & -1 & 0 & 2 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 2 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (15)$$

After this step, we compute the degree-normalized eigenvectors of the Laplacian. Subsequently, we compute the centers of the circles directly from the resulting eigenvectors of the Laplacian. Namely, we use the first *m* elements of the eigenvector that corresponds to the second smallest eigenvalue as the *x* coordinates of the centers of the sets and the first *m* elements of the eigenvector that corresponds to the third smallest eigenvalue as the *y* coordinates for the set centers. The remaining elements from the second and third low eigenvectors are used to compute the *x* and *y* coordinates for the items respectively, as in the CMG algorithm. We compute the radii of the circles in a way that is similar to CMG; hence, BPG is also guaranteed to have no false negatives.

4.3 Improved bipartite graph based algorithm (IBPG)

When drawing the circles, there might be items that lie far away from the center of the circle that represents a set and most of the other items seem to be clustered around the center. The number of those items that lie far from the center is usually small, and including them in the circle increase the radius of the circle, which subsequently increases the size of the circle allowing for more false positives (items that appear to be in the circle but do not actually belong to the set represented by that circle). As a tradeoff, we try to exclude those items as an effort to keep the number of false positives low. Note that, BPG outperforms CMG, thus we build IBPG on top of BPG. After we compute the *x* and *y* coordinates for the items and circles, for each circle we construct the table shown in Table 1.

Table 1. Computing best radius

R	Tp	fp	Fn	T
$\mathbf{r_1}$	tp_1	fp_1	fn_1	t_1
$\mathbf{r_2}$	tp_2	fp_2	fn_2	t_2
⋮	⋮	⋮	⋮	⋮
$\mathbf{r_k}$	tp_k	fp_k	fn_k	t_k

Here, k is the size of the set, *tp* is the number of true positives (i.e., number of the items that belong to a set and are bounded by the circle representing that set), *fp* is the number of false positives (i.e., number of items that do not belong to a set and visualized as they were in that set), *fn* is the number of false negatives (i.e., number of items that belong to a set and visualized as they were not in the set), and r_i represents candidate radii. The way we generate candidate radii is as follows: we start with one item in the set and compute the distance between that item and the center of the circle, and compute all the *tp, fp,* and *fn* associated with that radius. Then, we add the next closest item to the center, in the circle and choose the candidate radius as the distance between the center of the circle and the farthest item so far. Similarly, we compute the entries of the table of the chosen radius, and so on, until we compute all the table entries for all candidate radii. Using this table, we compute the optimal radius of each circle that produces the best value in the following equation:

$$t(i) = \frac{tp(i) - fp(i)}{size(S)} \qquad (16)$$

Obviously, this equation is trying to maximize the number of true positives and minimize the number of false positives, and this is what our algorithm intends to do, (we desire to include the items that belong to the set in the circle and exclude the items that do not belong to the set from the circle). There are some cases, in which various radii have the same *t* value. In such cases, we choose the radius that minimizes the number of false negatives. In other words, we pick the radius that includes more items in the circle without increasing the false positive rate.

4.4 Improving visualization

There are some cases, in which some items might be plotted over each other, and this is natural because if two items occur in the same set of sets, they are mapped to the same point in the Euclidean space. We have isolated those items and plotted them separately using a discretization process, so that they can be easily identified on the drawing. Precisely, we sort the *x* and *y* coordinates of the items and map them to integers from 1 to *n* in increasing order. Subsequently, we plot these integer vectors that are derived from the real-valued vectors.

5. EXPERIMENTAL STUDY

In order to measure the performance of the proposed algorithms, two indicators are used. The first indicator is the total number of errors produced by the algorithm. We compute the membership of the items in the sets indicated by the visualization and compare them with the actual membership of the items in the sets. As a result, we can tell which algorithm represents the true membership of the items more accurately. The second indicator is the total area of the circles we use to represent sets. A smaller area means that the algorithm utilizes the plot area more efficiently. In addition, we focus on the aesthetic aspect of the drawing produced by the methods, as the aim of these algorithms is to produce a drawing that is easy to understand.

To evaluate the performance of the algorithms, we use three classes of randomly generated datasets. We consider three factors when we generate the data, the number of items, the number of sets, and the average set cardinality. For each dataset, we fix two factors at a time, and vary the other factor to see how that factor affects the performance. Besides, we have tested our algorithms on three real datasets. The first one is set of the prime factors of the numbers between 100 and 200. The second dataset is subset of publications list obtained from the web page of Mehmet Koyuturk, from Case Western Reserve University. The third dataset is a group of six biological annotation datasets consisting of genes and their functional annotation according to Gene Ontology (GO). Further details on the datasets, metrics used, and how the experiments were conducted can be found in [15].

6. CONCLUSION

In this paper, we presented spectral algorithms for visualizing overlapping sets using the eigenvectors of the Laplacian of two different graph representations: co-membership graph and bipartite graph. We first introduced an algorithm that is based on the co-membership graph of the sets to be visualized, CMG. We have introduced four different versions of the algorithm. The weighted normalized version of the algorithm performed the other versions. That is because the normalization takes into account the degree of each node in the graph produced from the sets, and for this reason, treats each item equally (the degree of an item means that which items share membership with that particular item). Besides, when weights are considered, we actually are considering the number of times that any two items appear in the same set together, and this affects the relation between those two items. After introducing the CMG algorithm, another algorithm, i.e., the BPG algorithm has been presented. The power of BPG comes from the use of a bipartite graph instead of the co-membership graph, where the set membership information can be directly represented. Furthermore, using this model, the centers of the circles that we use to represent the sets can be directly computed.

IBPG is a further improvement to BPG that optimizes radii of circles that represent sets, and it has demonstrated better performance. However, the use of IBPG depends on whether or not the user wants to allow false negatives. Although IBPG produces some false negatives, it still provides what is required from a drawing algorithm: initial basic information about the structure of the dataset. Besides, the performance of the algorithm can be improved by feeding back subsets of data based on the outputs of the algorithm. Here, the input size would be smaller, allowing the algorithm to produce better looking results. Finally, a discretization procedure is applied to IBPG to help isolating the items that lay over each other, so that they can be easily identified in the drawing.

A major limitation of the proposed methods is the use of circles to represent the sets. A potentially useful improvement in this regard, would be to use the convex hull of the points that represent the items in a set. Since a convex shape would still cause errors, this can be further improved by refining the bounding shape by allowing non-convex shapes as well. However, such an approach would lead to more complicated optimization problems. Moreover, it is more beneficial to compute the center of the circle dynamically as we add items to the circle. The current approach computes the center first and then starts adding items and computes the candidate radii and errors. Since the centers are fixed, there is not much to do to keep the items close to the center. Computing the centers dynamically, though, allow us to cluster the items around the center and produce better drawing.

7. REFERENCES

[1] M. Fisher and H. Gall. 2003. MDS-Views: Visualizing problem report data of large scale software using multidimensional scaling. In *Proceedings of ELISA*, Netherlands.

[2] J. Shi and J. Malik. 2000. Normalized Cuts and Image Segmentation. In *IEEE Transactions on Pattern Analysis and Machine Intell.*, vol. 22 . pp. 888–905.

[3] B. Shneiderman, D. Feldman and A. Rose. 2000. Visualizing digital library search results with categorical and hierarchical axes. In *Proceedings of the ACM Digital Libraries Conference*. New York, ACM Press, pp. 57–66.

[4] J. Flower and J. Howse. 2002. Generating Euler Diagrams, Proc. Diagram, In *Proceedings of International Conference on the Theory and Application of Diagrams*. pp. 61-75.

[5] J. Flower, P. Rodger and P. Mutton. 2003. Layout Metrics for Euler Diagrams. In *Proceedings of 7th International Conference on Information Visualisation*. pp. 272-280.

[6] P. Simonetto, D. Auber, and D. Archambault. 2009. Fully automatic visualisation of overlapping sets. In *Computer Graphics Forum*. vol. 28(3), pp. 967–974.

[7] M. Frohlich and M. Werner. 1997. Demonstration of the interactive graph visualization system. da Vinci', R. Tamassia and I. Tollis (eds.), Proc. In *Symp. Graph Drawing, GD '94*, Lecture Notes in Computer Science, Berlin, Springer-Verlag. vol. 894, pp. 266–269.

[8] S. Clemencon, H. Arazoza, F. Rossi, and V. Tran. 2011. Hierarchical clustering for graph visualization. In *ESANN 2011*. Bruges, Belgium. pp. 227-232..

[9] K. Hall. 1970. An r-dimensional Quadratic Placement Algorithm. In *Management Science*, vol. 17, pp. 219-229.

[10] Y. Koren, L. Carmel and D. Harel. 2002. ACE: A Fast Multiscale Eigenvectors Computation for Drawing Huge Graphs. In *Proceedings of InfoVis'02*, IEEE, pp. 137–144.

[11] Y. Koren. 2003. On spectral graph drawing. In Computing and Combinatorics, In *Proceedings of COCOON 2003*, pp. 496–508.

[12] S. Roweis and L. Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding, *Science*, vol. 290.

[13] A. Agarwal, J. Phillips and S. Venkatasubramanian. 2010. Universal multi-dimensional scaling. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '10). ACM. New York, NY, USA, pp. 1149-1158. DOI=10.1145/1835804.1835948

[14] L. Chen and A. Buja. 2009. Local multidimensional scaling for nonlinear dimension reduction, graph drawing, and proximity analysis. In *Journal of the Americal Statistical Association*, vol. 104, pp. 209–219.

[15] A. Abdalla. 2012. *SAFA: Spectral Approach for Automatic Set Visualizatio*. Master Thesis. Case Western Reserve University, USA.

[16] B. Alsallakh, W. Aigner, S. Miksch and H. Hauser. 2013. Radial Sets: Interactive Visual Analysis of Large

Overlapping Sets. In *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19(12), pp. 2496- 2505.

[17] W. Meulemans, N. Riche, B. Speckmann, B. Alper, T. Dwyer. 2013. KelpFusion: A Hybrid Set Visualization Technique. In *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19(11), pp. 1846-1858.

[18] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot and H. Pfister. 2014. UpSet: Visualization of Intersecting Sets. In *Visualization and Computer Graphics, IEEE Transactions on*, vol. 20(12), pp. 1983-1992.