

On Edge Bundling and Node Layout for Mutually Connected Directed Graphs

Naoko Toeda^{*1} Rina Nakazawa^{*2} Takayuki Itoh^{*3}

Takafumi Saito^{*4} Daniel W. Archambault^{*5}

Ochanomizu University^{*1*2*3} Tokyo University of Agriculture Technology^{*4} Swansea University^{*5}
{toeda, leena, itot}@itolab.is.ocha.ac.jp^{*1*2*3} txsaito@cc.tuat.ac.jp^{*4} d.w.archambault@swansea.ac.uk^{*5}

Abstract

Directed graphs are used to represent variety of information, including friendship on social networking services (SNS), pathways of genes, and citations of research papers. Graph drawing is useful to intuitively represent such datasets. This paper presents an edge bundling and a node layout technique for tightly and mutually connected directed graphs. Our edge bundling technique includes three features: ordinary bundling of edges connecting common pairs of node clusters, convergence of multiple bundles connecting to the same node cluster, and shape adjustment of two bundles connecting the same pair of node clusters. This paper includes a case study with a directed paper citation graph.

Keywords--- Directed graph, Edge bundling, Node layout, Graph clustering.

1. Introduction

A variety of information in academic and industry fields can be represented as directed graphs. For example, friendships on social networking services (SNS) can be retrieved via Web APIs and constructed as directed graphs. Pathways of genes are accessible on open databases and therefore we can construct directed pathway graphs. Paper citation networks are also typical directed graphs which can be constructed from open information on digital libraries of academic societies.

Graph drawing techniques are useful to intuitively represent the structures of such directed graphs. Many applications exist to visualize friendships on social networking services or visualize gene pathways or gene-gene interactions. In our observation, many of these software packages feature general graph drawing algorithms which can be applied to undirected graphs. Development of directed-graph-specific drawing algorithms is still an interesting problem.

Edge bundling for comprehensive graph drawing has been an active topic for recent years. There have been many techniques [1-4]. However, many of them mainly support undirected graphs and not directed graphs. In contrast to many of edge bundling techniques gather

edges at their centers, it is also effective to converge edges at their end points [5]. It should be more effective if these approaches are appropriately combined.

This paper presents edge bundling and node layout techniques for drawing tightly and mutually connected directed graphs. Our implementation firstly constructs hierarchy of nodes by recursively applying a graph clustering algorithm, and then applies a node layout algorithm to calculate positions of the nodes. Our implementation applies a directed-graph-specific edge bundling algorithm to effectively represent connectivity among node clusters. The algorithm firstly bundles edges connecting pairs of nodes belonging to the same clusters. We may generate two bundles between a pair of node clusters according to the directions of edges. The bundling algorithm then converges multiple bundles departing from or arriving to the same node cluster. Also, it adjusts shapes of two bundles connecting the same pair of node clusters.

We designed node clustering and layout algorithms so that our edge bundling algorithms for directed graphs work effectively. The node clustering algorithm groups nodes which have many edges connected to the same node preferentially, because this strategy bundles more edges. The node layout algorithm calculates weights of connections between a pair of node clusters according to the smaller number of edges of up to two bundles connecting the clusters, so that tightly connected clusters are placed nearby.

This paper presents a case study using a paper citation network dataset retrieved from ACM Digital Library, and discusses the effectiveness of the presented technique.

2. Related Work

Edge bundling has been an active topic that improves the comprehensibility of drawing complex graphs, which integrates constitutively or positionally similar edges and draws them as bundles. Holten [1] presented a hierarchical edge bundling technique, which hierarchically groups edges connecting nodes belonging to same pairs of clusters. This technique mainly supposes undirected graphs, and therefore it does not feature directed-graph-specific processes. Also, this technique

selects edges to be grouped according to the hierarchy of nodes: it does not consider their positions. On the other hand, several other edge bundling techniques, presented by Selassie et al. [2], Holten et al. [3] and Ersoy et al. [4], group positionally and directionally similar edges. Again, most of these techniques mainly assume undirected graphs.

In contrast to that many edge bundling techniques gathers the edges at their midpoints, it is also effective to converge them at their end points. For example, Luo et al. [5] presented a technique to draw edges as curves to converge them at their end points to avoid bundle ambiguity. However, the technique does not assume a hierarchy. In contrast to these techniques, this paper presents a directed graph visualization technique that cluster nodes, bundle edges at these end points and avoids edge ambiguity.

Graph clustering is a key technique that makes graph drawing more effective and comprehensive. Actually there have been large number of clustering algorithms introduced by several survey papers [6]. Though finding densely connected components is definitely the most popular approach for graph clustering, other approaches are also effective if we want to maximize the number of edges to be bundled. Itoh et al. [7] applied edge-adjacency and node-similarity based graph clustering to make large number of edges bundled and separate key nodes from large clusters. The technique presented in this paper also applies the same clustering algorithm.

Meanwhile, there have been many tree-visualization-like techniques that operate on directed acyclic graphs [8][9], which places nodes based on distances from the starting node. We did not apply this approach since it does not often work well for dense graphs. Instead, we applied general algorithms for graph clustering and node layout, and extended the edge bundling technique for mutually connected directed graphs.

3. Presented Visualization Tool

This section presents the data structure and processing flow of the presented technique.

3.1 Data Structure

This paper defines the data structure of an input graph G consisting of a set of nodes N and edges E as follows:

$$\begin{aligned} G &= \{N, E\} \\ N &= \{n_1, \dots, n_{N_n}\} \\ E &= \{e_1, \dots, e_{N_e}\} \\ n_i &= \{a_{i1}, \dots, a_{iN_a}\} \\ e_i &= \{n_i, n_k\} \end{aligned}$$

Here, n_i denotes the i -th node. We suppose that a node has a multi-dimensional attribute value, where a_{ij} denotes the value of the j -th dimension of the i -th

node. e_i denotes the i -th edges, where the order of two nodes of an edge denotes the direction of the edge. N_n denotes the number of nodes, N_e denotes the number of edges, and N_a denotes the number of dimensions.

3.2 Node Clustering

The technique calculates distances between arbitrary pairs of nodes, and then applies a clustering algorithm to construct the hierarchy of the nodes. This section describes the definition of node distance, and processing flow of the node clustering algorithm.

3.2.1 Node Distance Calculation

The technique calculates two types of distances between pairs of nodes: dissimilarity of feature vectors d_{vec} , and discommonality of the adjacent nodes d_{adj} . It defines the distance between a pair of nodes as

$$d = \alpha d_{vec} + (1.0 - \alpha) d_{adj}$$

Here, α is a user-specified value satisfying $(0.0 \leq \alpha \leq 1.0)$.

Dissimilarity of feature vectors

The technique simply calculates the similarity between the two nodes as the inner product of the feature values. We define the dissimilarity calculated from the inner product as the distance between the two nodes d_{vec} , by the following equation.

$$d_{vec} = 1.0 - inner$$

$$inner = n_i \cdot n_j / |n_i| |n_j|$$

Discommonality of adjacent nodes

We define the discommonality of adjacent nodes simply by the number of commonly connected nodes. To specify the distance d_{adj} between two nodes n_i and n_j , the technique counts the number of nodes which are connected to both n_i and n_j . It simply calculates the distance as follows.

$$d_{adj} = 1.0 / (1 + n_{adj})$$

Here, n_{adj} is the number of nodes connected to both the nodes.

3.2.2 Hierarchical Clustering

Our implementation then constructs hierarchy of nodes based on the above mentioned distances. Currently we generate two-level node clusters by applying an agglomerative clustering algorithm with the furthest neighbor method.

This implementation assumes two thresholds α and β ($\alpha < \beta$). This process firstly groups nodes while satisfying the maximum distance α . This section calls the level of the generated clusters as "cluster A". The process then groups the clusters while satisfying the maximum distance β . This section calls the level of the

larger clusters as “cluster B”.

3.3 Node Layout

Our implementation calculates positions of nodes by the following a bottom-up algorithm. It firstly calculates positions of level A clusters belonging to a level B cluster, and then calculates positions of level B clusters. The second part of this algorithm is similar to the implementation presented in [7].

Positions of level A clusters:

1. Generate a graph treating the level A clusters as nodes. Weight the edges according to the number of edges of the input graph connecting pairs of clusters, so that tightly connected clusters are placed closely.
2. Calculate the radii of circles enclosing each of level A clusters based on number of belonging nodes.
3. Apply a spring-force-based node layout algorithm to the above mentioned graph to calculate the positions of level A clusters.
4. Adjust distances between pairs of level A clusters closer to the sum of radii by applying Laplacian smoothing algorithm.
5. Calculate positions of nodes belonging to the level A clusters.

Positions of level B clusters:

1. Generate a graph treating the level B clusters as nodes. Weight the edges according to the number of edges of the input graph connecting pairs of clusters, so that tightly connected clusters are placed closely.
2. Apply the above mentioned node layout algorithm for level A clusters belonging to each of level B clusters.
3. Calculate radii of circles enclosing each of level B clusters based on the node layout results of level A clusters.
4. Adjust distances between pairs of level B clusters closer to the sum of radii by applying Laplacian smoothing algorithm.
5. Calculate positions of level A clusters belonging to the level B clusters.

3.4 Edge Bundling

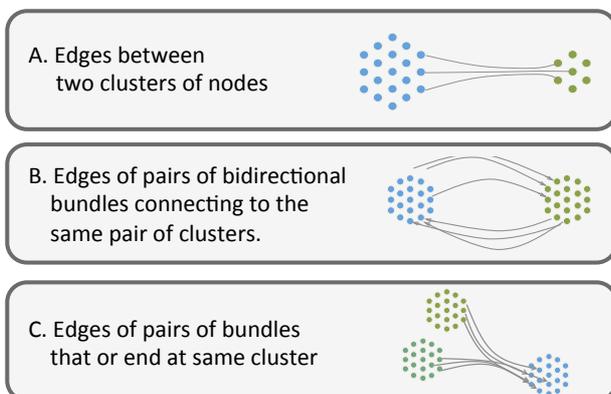


Figure 1. Conditions for edge bundling.

The presented edge bundling technique aims to satisfy the three conditions illustrated in Figure 1. Our technique firstly specifies sets of bundles to be clustered according to Condition C. Then, it calculates shapes and thickness of bundles taking all conditions into account. This section describes the processing flow of bundle selection and edge shape calculation.

3.4.1 Convergence of Bundles

The technique groups adjacent bundles connecting to the same cluster by the following algorithm.

1. Select a cluster.
2. Select a bundle connecting to the cluster in the clockwise order.
3. Calculate the later mentioned score the selected bundle and its immediate left bundle, if they have the same direction.
4. Group the above two bundles if the score is larger than a user-specified threshold.
5. Repeat the similar process to all the clusters.

Here, as illustrated in Figure 2, our implementation calculates the score of two adjacent bundles by the following processing flow:

i. Angle between the two bundles

Let us assume nodes in cluster A are connected to nodes in clusters B and C, and call centers of these clusters A, B, and C. Also, let us call the segments connecting two clusters AB and AC. The technique treats the score as zero if the angle between AB and AC is larger than a user-specified threshold.

ii. Length of convergable parts

The technique generates the bisector of AB and AC, and then generates perpendicular segments from centers of two clusters (B and C) connected to the bisector. Let us call the intersections between the bisector and perpendicular segments D and E. The technique then calculates the lengths of AD and AE. We treat the smaller length as the score of AB and AC, because the distance denotes the convergable part of the two bundles. In Figure 2, AE is the length of convergable parts.

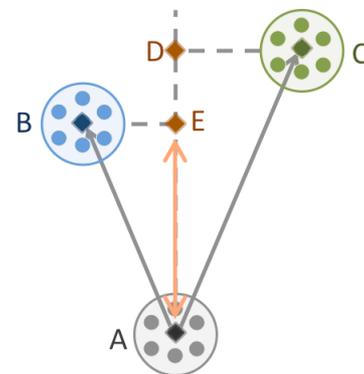


Figure 2. Conditions for convergence of bundles. Orange segment is the convergable part of the two bundles.

3.4.2 Edge Drawing

This technique categorizes the edges of the input graph into the following three patterns, and draws them as Bezier curves according to these patterns.

i. Edges of bundles converged to other bundles

The technique places the control points of the Bezier curves on the bisectors of the adjacent two bundles, as illustrated in Figure 3(Left). Position of the control point P_1 is calculated by the following equation.

$$P_1 = A(1 - \alpha) + E\alpha$$

Here, α is a user-specified value satisfying $(0.0 \leq \alpha \leq 1.0)$.

ii. Edges of pairs of bidirectional bundles connecting to the same pair of clusters.

As illustrated in Figure 3(Right), the technique places the control points of the Bezier curves on each side of CD, which is the bisecting perpendicular segment of another segment connecting the centers of clusters A and B. Here, E is the midpoint of side AB and side CD. Positions of the control points P_2 and P_3 are calculated by the following equation.

$$P_2 = E(1 - \beta) + C\beta$$

$$P_3 = E(1 - \beta) + D\beta$$

Here, β is a user-specified value satisfying $(0.0 \leq \beta \leq 1.0)$.

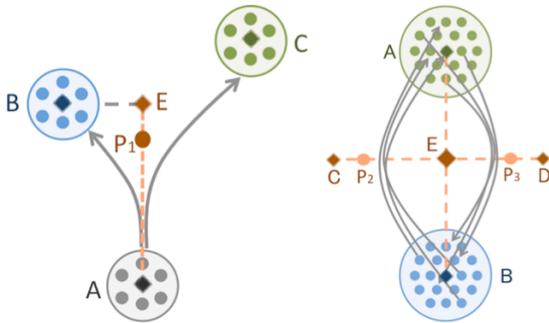


Figure 3. Positions of control points of Bezier curves for edge bundling. (Left) For edges of bundles converged to other bundles. (Right) For edges of pairs of bidirectional bundles connecting to the same pair of bundles.

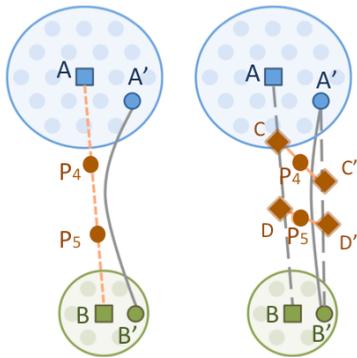


Figure 4. Positions of control points of Bezier curves for edge bundling.

iii. Other Cases

The following processing flow is applied to the edges if the above mentioned two patterns are not applicable. Our implementation places two of the control points on the segment connecting the centers of clusters, as shown in Figure 4(Left), when we would like to tightly bundle the edges. Otherwise, it first calculates the positions which divide the two segments (one connects two nodes, and the other connects centers of the clusters) into three equal-sized parts. It then places the control points on the segments connecting them. Figure 4(Right) illustrates this process.

Position of the control points P_4 and P_5 are calculated by the following equation.

$$\text{if } \gamma > 0.5 \text{ then } r = (\gamma + 0.5) * 2 / 3$$

$$P_4 = Ar + B(1 - r)$$

else $r = \gamma * 2$

$$P_5 = (A * 2 + B) / 3 * r + (A' * 2 + B') / 3 * (1 - r)$$

Here, γ is a user-specified value satisfying $(0.0 \leq \gamma \leq 1.0)$.

4. Case Study with Paper Citation Network

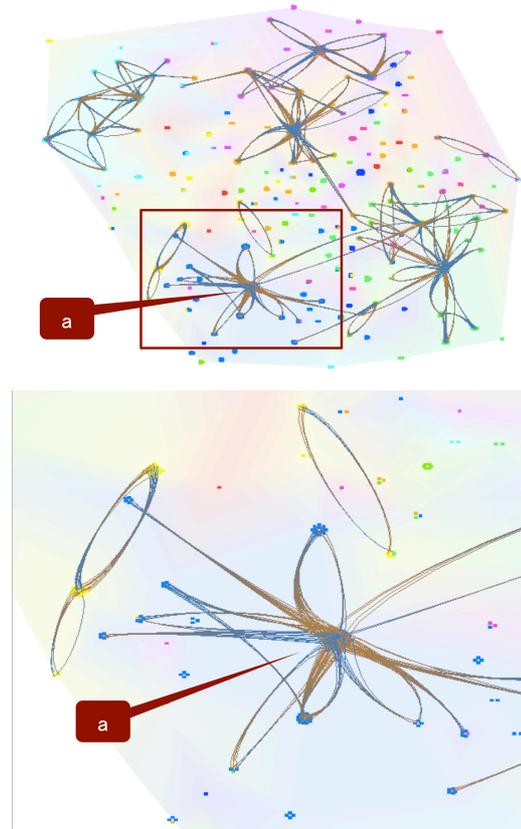


Figure 5. (Upper) Visualization where edges are bundled based on level A clusters. (Lower) Zooming to the cluster (a).

We applied a citation network dataset [10] consisting of 1072 full papers presented at ACM SIGGRAPH conferences during 1990 to 1994, and during 2000 to 2010, published by ACM Digital Library. We extracted

the title, publication year, abstract, references, and authors from html files of the papers. This data have 1072 papers (nodes) and 5498 references (edges). Also, we calculated 10 dimensional attribute values to each of the papers from their abstracts by applying generative topic model LDA (Latent Dirichlet Allocation).

4.1 Example of level A clusters

Figure 5 shows a visualization result where edges are bundled based on level A clusters. Many edges connected to nodes in the cluster (a) in Figure 5 are conspicuously bundled. We can assume this cluster has several important papers related to the topic depicted in blue. We checked titles of papers in the cluster (a) and found many papers related to fundamental illumination methods belonged to. Especially, papers related to light transport calculation algorithms, precomputed radiance transfer, and tree structures for shadow calculation are referred by large number of newer papers. The papers in this cluster have mutual connections with several other clusters because these fundamental methods might be inspired by other research topics. We easily found such important paper cluster by just looking at the visualization result.

4.2 Example of level B clusters

Figure 6 shows a visualization result where edges are bundled based on level B clusters. In this figure, many edges connect pairs of nodes painted in the same colors, which depict that many papers in same research topics are referred. On the other hand, nodes painted in yellow-green in the cluster (b) are referred by nodes in the clusters (c), (d), and (e), painted in blue, green, or orange. This is an interesting relationship because papers in the cluster (b) is referred by other research topics.

Cluster (b) includes papers presenting automatic texture generating and retouching techniques. In cluster (c), papers related to 3D rendering techniques refer to papers in cluster (b) because they need textures to map on 3D models for realistic rendering. Furthermore, papers related to image processing belonging to cluster (d), and 3D modeling belonging to cluster (e), refer to papers in cluster (b) for texture mapping with complete images onto complex 3D models. These citation relationship denotes that the topic of cluster (b) is so general and fundamental that the techniques presented in the paper of cluster (b) have been applied to variety of studies.

On the other hand, it was a little bit time-consuming task for us to understand the background of the above mentioned citation relationships, because these clusters, especially cluster (d) and (e), contained variety of research topics. Therefore, we would like to review clustering methods, or definition of multi-dimensional attribute values and node distances.

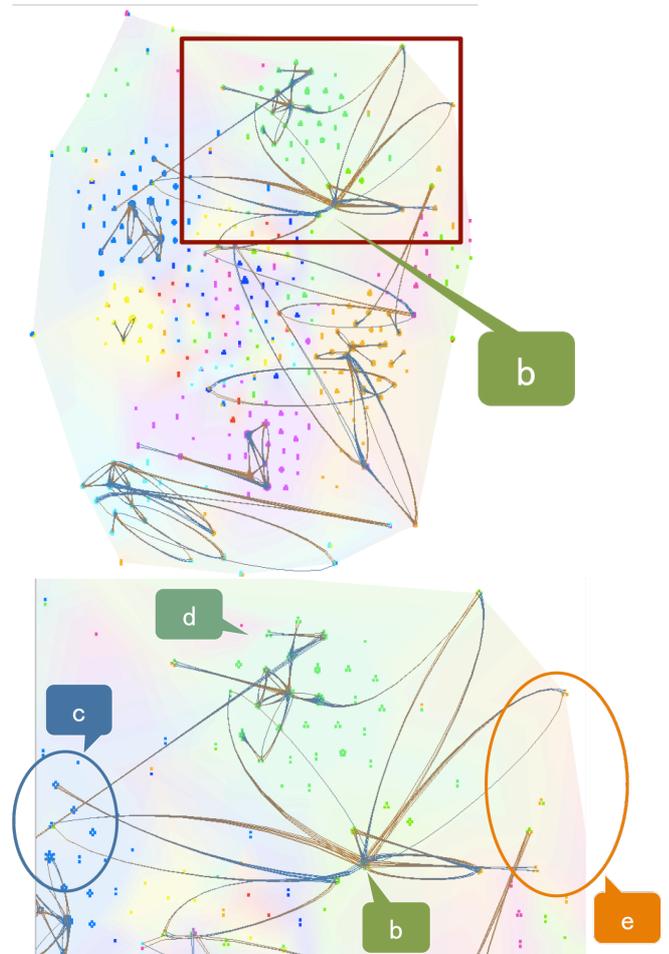


Figure 6. (Upper) Visualization where edges are bundled based on level B clusters. (Lower) Zooming to the cluster (b).

5. Conclusions

This paper presented an edge bundling technique for directed graphs. This technique deforms edges based on the following three points: 1) bundling edges connecting two nodes belonging to same pairs of clusters as ordinary edge bundling techniques, 2) preserving proper distances between bidirectional bundles between the same pairs of clusters, and 3) converging adjacent bundles connecting to the same cluster. This technique is especially effective for directed graphs which have many mutually connected edges. This paper presented a case study with a paper citation network dataset, and discussed what kinds of knowledge can be visualized.

We have many future issues on this study. Regarding edge bundling, we would like to firstly extend the algorithm to calculate positions of control points of Bezier curves. In addition to the conditions described in Section 3.4, we would like to consider avoidance of overlap between bundles and node clusters while calculating positions of control points. Another problem is selection of adjacent bundles to be converged. Our current implementation is based on a greedy algorithm,

and therefore it does not always optimize the selection result. We would like to develop another algorithm which really finds optimal selection results.

Node layout can be also improved. We would like to develop rotation of node clusters [11] so that the total lengths of edges is minimized. It is also an interesting problem how to optimize cluster layouts with the edge bundling presented in this paper.

After improving the technique as above mentioned, we would like to have more case studies with real-world directed graph datasets, and experimental user evaluations.

References

- [1] D. Holten, "Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 12, No. 5, pp. 741-748, 2006.
- [2] D. Selassie, B. Heller, J. Heer, "Divided Edge Bundling for Directional Network Data", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 17, No. 12, pp. 2354-2363, 2011.
- [3] D. Holten, J. J. van Wijk, "Force-Directed Edge Bundling for Graph Visualization", *Computer Graphics Forum*, Vol. 28, No. 3, 2009.
- [4] O. Ersoy, C. Hurter, F. V. Paulovich, G. Cantareira, A. Telea, "Skeleton-based edge bundling for graph visualization", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 17, No. 12, pp. 2364-2373, 2011.
- [5] S.-J. Luo, C.-L. Liu, B.-Y. Chen, K.-L. Ma, "Ambiguity-Free Edge-Bundling for Interactive Graph Visualization", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 18, No. 5, pp. 810-821, 2011.
- [6] S. E. Schaeffer, "Graph Clustering", *Computer Science Review*, Vol. 1, No. 1, pp. 27-64, 2007.
- [7] T. Itoh, K. Klein, "Key-node-Separated Graph Clustering and Layout for Human Relationship Graph Visualization", *IEEE Computer Graphics and Applications*, Vol. 35, No. 6, pp. 30-40, 2015.
- [8] K. Sugiyama, S. Tagawa, M. Toda, "Method for Visual Understanding of Hierarchical System Structures", *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 11, No. 2, pp. 109-125, 1981.
- [9] T. Dwyer, Y. Koren, Dig-CoLa: Directed Graph Layout through Constrained Energy Minimization", *IEEE Symposium on Information Visualization*, pp. 65-72, 2005.
- [10] R. Nakazawa, T. Itoh, T. Saito, "A Visualization of Research Papers Based on the Topics and Citation Network", *18th International Conference on Information Visualisation (IV2015)*, pp. 283-289, 2015.
- [11] D. Archambault, T. Munzner, D. Auber, "TopoLayout: Multilevel Graph Layout by Topological Features", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 13, No. 2, pp. 305-317, 2007.