Adaptive Disentanglement based on Local Clustering in Small-World Network Visualization

Arlind Nocaj, Mark Ortmann, and Ulrik Brandes

Abstract—Small-world networks have characteristically low pairwise shortest-path distances, causing distance-based layout methods to generate hairball drawings. Recent approaches thus aim at finding a sparser representation of the graph to amplify variations in pairwise distances. Since the effect of sparsification on the layout is difficult to describe analytically, the incorporated filtering parameters of these approaches typically have to be selected manually and individually for each input instance. We here propose the use of graph invariants to determine suitable parameters automatically. This allows us to perform adaptive filtering to obtain drawings in which the cluster structure is most prominent. The approach is based on an empirical relationship between input and output characteristics that is derived from real and synthetic networks. Experimental evaluation shows the effectiveness of our approach and suggests that it can be used by default to increase the robustness of force-directed layout methods.

Index Terms—Automatic Parameter Selection, Adaptive Edge Filtering, Graph Simplification, Force-directed Layout, Small-World Networks, Network Visualization.

1 INTRODUCTION

Social networks are often highly connected and exhibit the so-called small-world property [1]. Networks with this characteristic in general have a small average pairwise shortest-path distance and a high local density. For Facebook friendship graphs, for example, this means that any person is connected by only a few intermediate connections with every other person in the network.

Although visualization approaches such as force-directed layout methods offer high quality results for many graphs, their capabilities are limited for small-world graphs. The resulting drawings are typically cluttered and look like a hairball, in which the intrinsic graph structure is not visible.

This makes one of the most important tasks in social network analysis [2], the discovery and visual exploration of cohesive sub-groups and their relation to each other in a network, very problematic.

The main problem of these layout methods is that they try to directly translate pairwise graph-theoretic distances to Euclidean distances. Yet, if the variation of pairwise distances is low, as for the hairball graphs, the resulting layout has low variation in Euclidean distances too.

A large variety of approaches have been proposed to reduce the clutter in drawings of hairball graphs. While geometry-based visualization techniques, such as edge bundling [3], are not likely to be of benefit when applied to hairball drawings due to the missing variation in the distribution of nodes in the plane, modified layout algorithms seem to be more promising as they allow to compensate for structural effects in a graph. Boitmans et al. [4], for example, propose to modify the pairwise distances in such a way that adjacent nodes of high degree are placed further apart. This allows to enlarge the core of the layout and is especially suited for graphs with an extremely skewed degree distribution. Another proposal that was made is to distort the layout [5] in a manually selected area of interest. A different general approach is to identify clusters in the graph and then use these clusters for the visualization of the network [6], [7]. This approach shifts the problem of visualization to the selection of a clustering or community detection method.

A different way of tackling this problem is graph simplification, where the idea is to reduce the set of edges in a graph to the most important ones. This sparser subgraph, the so-called backbone, can then be used to layout the graph.

Simplification approaches which maintain specific properties of the graph, like spectra [8], cuts/connectivity [9], or shortest-paths [10], [11], tend to maintain the small pairwise distances as well, which means that the resulting backbones are still hairball graphs.

The most promising simplification approach for the discovery and visual exploration of cohesive sub-groups is to filter out edges based on an embeddedness criterion [7], [12], [13], [14], [15], which is determined based on the local density around an edge.

All methods following this approach require as input a threshold parameter according to which the backbone is extracted, which in turn is laid out using standard force-directed methods. Finding the right threshold parameter to retrieve meaningful network visualizations is very costly due to the non-linear influence of the parameter on the final visualization. A small difference in the threshold value might already change the layout completely. Consequently, identifying the optimal threshold, for which the group structure is most prominent in the layout, on a trial and error basis is a very time consuming task, particularly for large networks.
For these methods we here propose a preprocessing technique which adaptively determines the threshold value for which the group structure is most evident in the graph and in the layout (as indicated in our experiments). Our technique quantifies for each possible threshold value the quality of the inherent groups in the resulting backbone using the clustering coefficient.

Our contributions are:

1) A novel approach to quantify the impact of every single threshold on the group structure of the backbone.
2) An efficient dynamic algorithm maintaining the clustering coefficient under edge deletion, running in $O(\alpha(G)m)$ total time, where $m$ is the number of edges in the graph and $\alpha(G)$ the arboricity - minimum number of spanning forests required to cover the edge set of a graph $G$.
3) An extensive evaluation of the effectiveness of our approach on a number of “real-world” as well as synthetic networks.

To evaluate our preprocessing technique we use the quadrilateral Simmelian backbone, as the experimental study in [14] with various edge embeddedness metrics suggests that it is very well suited for untangling hairball graphs, while preserving the group structure. Especially for multi-centered small-world networks it enables standard force-directed layout techniques to strongly emphasize the inherent group structure in the final drawing.

We start by giving an overview of the backbone layout approach from [14] in the next section and explain our proposed quantification measure for the adaptive filtering in Section 3. In Section 4 we first evaluate how well this measure approximates the association with known clusters and then investigate its influence on the network layout. Section 5 summarizes our conclusions.

2 Backbone Layout Approach

The overall work flow of the backbone layout approach as suggested in [14] is shown in Fig. 1. In the first step an edge embeddedness metric which only relies on the graph structure is calculated. Based on these edge weights the filtering step removes all edges that are below a given threshold value.

Since the connectedness is indispensable for force-directed layout methods but might be destroyed through the filtering step, the next step reinserts all edges from the union of all maximum spanning trees based on the embeddedness metric ensuring that the backbone stays connected. Note that the union is necessary to avoid arbitrary (non-structure based) preference of an edge over another [11].

The computation of the structural edge embeddedness itself can be divided into two parts. The first part calculates an initial edge weight, the so called quadrilateral weight, which measures how well an edge is integrated in its neighborhood and is defined as

$$Q(u, v) = \frac{q(u, v)}{\sqrt{q(u) \cdot q(v)}}.$$  \hspace{1cm} (1)

with $q(u, v)$ being the number of quadrangles containing edge $(u, v)$ and $q(v) = \sum_{w \in N(v)} q(v, w)$, for $v \in V$ and neighborhood $N(v)$. Thereafter, a reweighting [13] is performed. The motivation for the reweighting step is to account for the fact that a relation $e = (u, v) \in E$ can be of different importance for $u$ and $v$. While $e$ might be the most important relation for $u$, $e$ might be irrelevant from the perspective of $v$, for example because $v$ might have many other more “important” relations and therefore is willing to break tie with $u$.

For an edge $(u, v)$ the reweighting is performed by first ranking the neighbors of $u$ and $v$ according to the initial edge weight. Then $k$ is chosen such that the Jaccard coefficient of the rank $\leq k$ neighbors of $u$ and $v$ is maximized. This maximum Jaccard coefficient can be seen as the agreement of $u$ and $v$ on their ranked neighborhood and is used as the embeddedness metric for the filtering step.

For the computation of the force-directed layout in the final step we use stress minimization [16] initialized by PivotMDS [17] as suggested in [14], [18]. This layout algorithm finds node positions, such that the pairwise Euclidean distances match the graph theoretical distances. To find these positions in each iteration of the algorithm every single node is relocated as a function of all other nodes.

In the next section, we propose a metric that only relies on the graph structure, but is nevertheless an adequate indicator for the final layout quality.
In this section, our goal is to quantify the structural degree of clusterability in networks, which should serve as an indicator of how clear the cluster structure in the network is, but without performing the actual clustering task. This quantification then allows to give visual support for manual selection and a fully automatic parameter extraction.

Instead of performing clustering by choosing among the vast amount of existing methods [19], [20], we measure an often observed side effect of clusters in networks, namely a high average clustering coefficient [21]. The clustering coefficient captures to which extent the neighborhood of a vertex is connected amongst themselves.

In a series of backbones, with varying sparsification parameter, the main assumption is that a backbone with a high clustering coefficient is more likely to contain cohesive groups than a backbone with a low clustering coefficient. If the quantification using the clustering coefficient is effective, its highest value should point us to the sparsification parameter which is very likely to emphasize the grouping information.

Efficient Computation of the Clustering Coefficient

We now investigate how the clustering coefficient can be computed efficiently for every possible sparsification parameter.

The local clustering coefficient is defined as the percentage of closed triples at a vertex $v$:

$$C(v) = \frac{\lambda(v)}{\tau(v)} = \frac{|\{(v_i, v_j) \in E | v_i, v_j \in N_v\}|}{\binom{d(v)}{2}},$$

with $\lambda(v)$ being the number of closed triples (triangles) at $v$ and $\tau(v)$ the number of connected triples at $v$. For $N(v) \leq 1$ we define the clustering coefficient to be zero, which punishes peripheral degree one vertices. The global (or average) clustering coefficient is then

$$C = \frac{1}{|V|} \sum_{v \in V} C(v).$$

Efficient Computation of the Clustering Coefficient

We now investigate how the clustering coefficient can be computed efficiently for every possible sparsification parameter.

The local clustering coefficient is defined as the percentage of closed triples at a vertex $v$:

$C(v) = \frac{\lambda(v)}{\tau(v)} = \frac{|\{(v_i, v_j) \in E | v_i, v_j \in N_v\}|}{\binom{d(v)}{2}},$

with $\lambda(v)$ being the number of closed triples (triangles) at $v$ and $\tau(v)$ the number of connected triples at $v$. For $N(v) \leq 1$ we define the clustering coefficient to be zero, which punishes peripheral degree one vertices. The global (or average) clustering coefficient is then

$C = \frac{1}{|V|} \sum_{v \in V} C(v).$
clustering coefficient is not clear to us. They give a runtime of \(O(<d>)\) for updating the clustering coefficient, with \(<d>\) being the average degree in the network. Considering the fact that the average degree \(<d>\) can be much smaller than \(n = |V|\), it is not clear to us how the update under deletion of a single edge could be performed in \(O(<d>)\) time while the number of local clustering coefficients having to be updated can be \(n\) in the worst case. Such a case is given when the edge to be deleted is contained in a triangle with each vertex in the graph (for example \(e\) in the figure of Algorithm 1).

**Algorithm 1:** Computation of the Clustering Coefficient for all possible filtering parameters.

**Input:** Graph \(G = (V, E)\) with \(n = |V|, m = |E|\), and edge weights \(w : E \rightarrow \mathbb{R}_{\geq 0}\) with \(k\) distinct values.

**Data:** \(Tr[e]\): set of triangles containing edge \(e\), \(d[v]\): degree of vertex \(v\), \(\lambda[v]\): number of triangles of vertex \(v\), \(\tau(v) := \left(\frac{d[v]}{2}\right)\)

**Output:** Clustering coefficient \(\bar{C}_i\) for backbones

1. \(\lambda[v] \leftarrow \tau[v] \rightarrow \text{triangle listing algorithm} [23]\)
2. for \(v \in V\) do \(C[v] \leftarrow \lambda[v]/\tau(v)\)
3. \(C_0 \leftarrow \bar{C} \leftarrow \frac{1}{n} \sum_{v \in V} C[v]\)
4. partition edges by weight into buckets \(B_1, \ldots, B_k\)
5. sort buckets by decreasing weight
6. for \(i \leftarrow 1\) to \(k\) do
   7.   foreach \(e = (u, v) \in B_i\) do
      8.     // remove contribution of \(e\) from triangle statistics
      9.     \(\bar{C} \leftarrow \bar{C} - (C[u] + C[v])/n\)
     10.    \(\lambda[u] \leftarrow \lambda[u] - \lambda[e]\)
     11.    \(\lambda[v] \leftarrow \lambda[v] - \lambda[e]\)
     12.    \(d[u] \leftarrow d[u] - 1; d[v] \leftarrow d[v] - 1\)
     13.    \(C[u] \leftarrow \lambda[u]/\tau(u)\)
     14.    \(C[v] \leftarrow \lambda[v]/\tau(v)\)
     15.    \(\bar{C} \leftarrow \bar{C} + (C[u] + C[v])/n\)
   16. end
   17. end
18. \(C_i \leftarrow \bar{C}\)

**Iterative Computation**

Let \(\omega : E \rightarrow \mathbb{R}_{\geq 0}\) be the edge weight reflecting a structural edge embeddedness, \(W = \{\omega(e) | e \in E\}\) the set of possible edge weights, and \(G_z, z \in W\) the resulting backbone.

To compute the clustering coefficient for a single graph, we only need to know the number of triangles at each vertex, which takes \(O(o(G)m)\) time [23, 25], where \(o(G)\) is the arboricity, or the minimum number of forests necessary to cover all edges of a graph \(G\). Practically, \(o(G)\) can be considered to be a small constant for social networks [14, 26]. Performing this computation for each backbone individually would take \(O(o(G)m^2)\) time, as the number of possible backbones is in \(O(m)\).

Algorithm 1 describes how this can be done more efficiently by computing the clustering coefficient for the original graph and iteratively updating the triangle statistics for each edge which is being removed.

When an edge \(e\) is removed (Line 7), all of its triangles \((Tr[e])\) are destroyed. As already observed by [24], for each vertex in one of these triangles, its local clustering coefficient and thus its contribution to \(\bar{C}\) needs to be updated.

**Correctness of Algorithm 1**

From the definition of the clustering coefficient it follows that \(\bar{C}\) is correct for the initial graph. It remains to show that after each edge deletion \(\bar{C}\) is properly updated. Thus, it suffices to show that each local clustering coefficient \(C[v]\) is correctly updated after an edge \(e = (u, v)\) is removed. The local coefficient changes only for vertices creating a triangle with \(e\). All triangles of \(e \in Tr(e)\) are also triangles at \(u\) and \(v\) (see the figure below Algorithm 1). Since they are all destroyed, we need to reduce \(\lambda[u]\) and \(\lambda[v]\) by \(|Tr[e]|\). For a vertex \(w\) in a triangle with \(e\), exactly one triangle is affected (Line 17 of Algorithm 1). By removing each triangle \(t\) from the triangle sets of the other two edges \((u, w)\) and \((v, w)\), we make sure that this triangle will never be considered again. \(C[w]\) is thus correctly updated. The degrees of \(u\) and \(v\) are also reduced correctly by one, which allows to update \(C[u]\) and \(C[v]\).

**Runtime of Algorithm 1**

The running time of the first part of Algorithm 1 is dominated by the triangle listing \((O(o(G)m))\) and sorting routine \((O(m \log m))\). In the second for-loop, each triangle is processed exactly once, and the required updates need constant time. Since there can be at most \(O(o(G)m)\) triangles, the runtime of the second for-loop is in \(O(o(G)m)\).

4 EVA LUATING THE EFFECTIVENESS

The sparsification of the original graph results in various backbones, based on the chosen sparsification parameter. For each of these backbones, we want to quantify how similar its structure is to a given set of inherent clusters.

Modularity is often used for cluster quality assessment, but we do not use it, due to its counter intuitive behavior: Even for a perfect partitioning of a graph, consisting only of cliques as connected components, modularity can vary and differ a lot from the optimal value of 1. We refer the
Pearson correlation reduces to the phi coefficient, when applied on binary adjacency matrices. The phi coefficient is a variation of Pearson’s correlation coefficient, when applied on binary variables [29]. Freeman also calls it Borgatti’s $\phi$ [28].

The intuitive interpretation is that its value is large, if a graph is similar to a given perfect partitioning and small (towards 0) if the graph is dissimilar to this partitioning.

Since the pairwise shortest-path distances are translated to Euclidean distances by the force-directed layout, we compute the average pairwise shortest-path distance to quantify the expansion of the graph for a specific parameter. In addition to that, we evaluate the influence of the threshold parameter on the final layout using the negated average embeddedness measure. These curves typically have one apex. Figure 4 shows the clustering and phi coefficient value as a proxy for the phi coefficient. Then we evaluate whether this behavior is also reflected in the final layout by looking at the local layout compactness. Thus, a high amount of missing values can bias the phi coefficient as an evaluation criterion.

Note that although empirically many of the social relations correlate with homophilous attribute values, no ground-truth group structure is available for Facebook networks. For this reason, we additionally generate artificial networks with an underlying ground-truth group structure using the planted partition model [31] (PPM) as described in [14] and also the model by Lancichinetti et al. [32] (LFR).

### 4.2 Dataset and Models

For evaluation we use the networks from the Facebook100 dataset [30]. The networks are originally from Facebook and contain social relations of students at 100 higher educational institutes in the US. The network size varies from 762 to 41K vertices and from 16K to 1.6M edges. Additional attributes, such as gender, expected year of graduation, dormitory, etc. are given as vertex attributes. Traud et al. [30] argue that the dormitory has been important for the formation of social relations. Therefore, we use the dormitory attribute as the partitioning $C$ and thus for evaluation with the phi coefficient. While the backbone and clustering coefficient computation consider all vertices of the graph, vertices with a missing dormitory value are ignored when computing the phi coefficient. An example of a similarity between a graph and a perfect partitioning is given in Fig. 3.

### 4.3 Results and Discussion

First, we discuss the usefulness of the clustering coefficient as a proxy for the phi coefficient. Then we evaluate whether this behavior is also reflected in the final layout by looking at the local layout compactness.

The results of the experiments are two curves for each network, similar to those in Fig. 2(a). Going from the left to the right, more and more edges are removed, based on the embeddedness measure. These curves typically have one apex. Figure 4 shows the clustering and phi coefficient value of these apexes against each other.
Due to space constraints we cannot show the results for all Facebook100 networks. Therefore, we picked the 11 Facebook networks labeled with their names in Fig. 4. The selection criterion was coverage of different regions, reflecting various properties of the networks.

Figure 5 shows the curves of the clustering and phi coefficients for the selected networks. Maxima are highlighted with a dashed line.

For the PPM500 network the peak of the clustering coefficient is extremely close to the peak of the phi coefficient, where further filtering of edges would start to only thin out the clusters. If the phi coefficient is high, e.g., for Auburn71, Caltech36, Lehigh96, or Smith60, then the two maxima tend to be close to each other. This means that the maximum clustering coefficient points us to the filtering parameter, where the density is highest for the inherent clusters. Auburn71 is slightly different compared to the other networks, as its phi coefficient is larger than the clustering coefficient on the first half. Looking at Fig. 4, we can see that 80% of its edges have missing values for the dormitory attribute of the corresponding vertices. While the clustering coefficient considers all vertices of the network, the phi coefficient has to ignore the ones with missing values, since no partition is known for them. Knowledge of these missing values might change the shape of the curve.

We can see that for many of the other networks, the phi coefficient does not increase, which might indicate that the dormitories are not an explanatory variable for the inherent clusters. However, there is a clear peak for the clustering coefficient, allowing us to identify an important global aspect of the inherent group structure.

We also experimented with many weighted variants of the clustering coefficient as, e.g., discussed by Opsahl and Panzarasa [33], however we did not see an improvement of accuracy over the unweighted one. The results were comparable. We expect other variants, as the transitive ratio [34], [35], number of triangles in the network divided by the number of triples, to work as well.

Using the graph model of Lancichinetti et al. [32], we generated networks with a ground-truth community structure. We varied the mixing parameter $\mu$ from 0.1 to 0.8 in 0.05 increments, which increases the underlying noise and blurs the group structure increasingly. Our practical experience using this model is that the group structure is hardly existent anymore if the mixing parameter is greater than 0.6. Figure 6 shows that the maximum clustering coefficient suggests...
Fig. 7. Networks generally have a sweet spot at which pairwise distances (gray curve) are increased but where the groups (blue curve) are compact, as indicated by the clustering coefficient (red curve).

Fig. 8. Maximum clustering coefficient and its filtering value (quadrilateral Simmelian) for Facebook100+PPM500. Grouping around three threshold values indicates similar density decay among different inherent communities.

gests nearly the same sparsification ratio as the maximum phi coefficient using the ground-truth information.

This means that the maximum clustering coefficient is a good proxy to identify the sparsification parameter for which the group structure is most prominent in the resulting backbone.

Filtering Value

An interesting observation can be made when looking at the threshold filtering value of the quadrilateral Simmelian backbone for which the clustering coefficient is maximal (Fig. 8). One can clearly see that the Facebook100 networks group around three different threshold values. This suggests that the local community structure is similar in these networks and that the density decay between different communities in a network is consistent (among these groups). It would also be interesting to find out the size of the communities and see if they correlate with these groups. Additional analyses using more information of the institutions, e.g., infrastructural properties, might reveal more explanation for this observed effect.

Layout Quality

For the smaller graphs we computed the force-directed layout for various increasing filter parameters and evaluated the local compactness of that layout (Fig. 7). One can observe that the local compactness curve is very similar to the clustering coefficient. High compactness indicates that the neighbors of a vertex in the graph are very close to this vertex in the layout. Considering the fact that the layout is actually expanding, as denoted by the monotonically increasing average shortest-path curve (Fig. 7 top), this means that the underlying clusters are getting more and more compact in the final layout based on the optimal clustering coefficient parameter.

To see the impact of our adaptive filtering on graphs with a very clear cluster structure, we apply our approach to the PPM500(25%) graph (Fig. 2(d)). Its curve of the clustering coefficient in Fig. 7 shows that the group structure would be destroyed if many edges are removed. This indicates that our approach can be applied even on non-hairball graphs, leaving graphs with a clear group structure mostly as they are.

Drawings

Looking at the resulting backbone drawings in Figs. 9 to 11, various clusters can be distinguished for Caltech36, Rice31, Auburn71, and Smith60. Emphasizing the local density in networks with the backbone layout allows us to get insights about the local graph structure in the global context of the graph. The backbone drawings indicate that:

- Many strong communities exist.
- Communities are often highly overlapping.
- Many actors (or vertices) are not well integrated in the strong communities.
A graph clustering (or community detection) method following these suggestions should thus focus on strong, possibly overlapping, communities together with a set of actors, which are there but are not part of these strong communities. Of course these actors could be assigned to their most closest community on demand, e.g., if it is required by the application.

The only exception among the analyzed networks is Harvard1, where no local clusters are visible in the drawing, see Fig. 9(b). Its border position among the Facebook networks in Fig. 8 indicates that it has different structural properties compared to the other selected networks. This might be a consequence of Harvard’s housing policy, which offers students a dorm only in their first year.

Note that the backbone with the maximum clustering coefficient emphasizes the group structure in its global context. It might be necessary to filter out more edges to observe the finer structure of the inherent groups.

4.4 Limitations

Our proposed techniques scale very well to large graphs in terms of runtime. Nevertheless, our quantification of the cluster structure is an aggregation over the whole network and it therefore is not sensitive for single local details. It would be interesting to extend this quantification in a more finely graduated way, e.g., by computing it only for a subset of interactively selected vertices. This would allow more detailed analysis and visualization of particular regions of interests in the graph.

As noticed in [14] the quadrilateral Simmelian backbone is especially useful for multi-centered networks and less for networks with a single center, i.e., core-periphery structure. Our quantification metric can only be as effective as the underlying edge embeddedness metric and it is thus very likely that it has similar drawbacks when applied to single-core networks.

5 Conclusion

We proposed the use of the clustering coefficient to quantify the influence of sparsification ratios on the backbone approach in terms of cluster structure. The experimental evaluation using real-world and synthetic networks confirms its effectiveness when applied to the quadrilateral Simmelian backbone, and the results are likely to extend to other density based backbones as well. In addition, we showed how to efficiently compute the clustering coefficient for every possible threshold parameter.

This is especially useful when exploring and visualizing large networks, where the selection of an appropriate sparsification level on a trial-and-error basis is very cumbersome due to time-intensive layout recomputations.

While both the layout metric and visual inspection of drawings suggest that clusters are indeed clearly pronounced, a detailed user study along the lines of [2] would have to show how significant this visual effect is for specific tasks such as cluster selection.
NOCAJ et al.: ADAPTIVE DISENTANGLEMENT BASED ON LOCAL CLUSTERING IN SMALL-WORLD NETWORK VISUALIZATION

Fig. 10. One of the larger networks from the Facebook100 dataset (Auburn71, |V| = 18k, |E| = 1M). Left: Original force-directed layout, Right: Force-directed layout with our automatic filtering on the quadrilateral Simmelian backbone.

Fig. 11. Smith60, |V| = 3k, |E| = 100k from the Facebook100 dataset. Left: Original force-directed layout, Right: Force-directed layout with our automatic filtering on the quadrilateral Simmelian backbone.

REFERENCES


**Arllind Nocaj** was born in Kosovo. He received a B.Sc. degree with major in computer science in 2009 and an M.Sc. degree in 2011 from the University of Konstanz. He received a prize for the best Master’s degree (university wide). During his PhD he did an internship with Microsoft Research. In 2015 he received his PhD from University of Konstanz. With a background in algorithms, his main interests are in network analysis and visualization. He is currently the lead programmer of visone, a software for analysis and visualization of social networks.

**Mark Ortmann** received a B.Sc. and M.Sc. degree from the University of Konstanz with a major in computer science in 2009 and 2012, respectively. His research interests include network analysis, graph drawing and combinatorial algorithms. He completed an internship under the supervision of Dr. Der Tsi Lee at the National Taichung University (Taiwan) as part of his ongoing PhD studies within the Algorithmics Group at the University of Konstanz.

**Ulrik Brandes** is professor for algorithms at the University of Konstanz since 2003. He received a Diploma degree from RWTH Aachen in 1994 and a PhD from the University of Konstanz in 1999. After his habilitation in 2002 he became associate professor at the University of Passau in the same year. With a background in algorithms, his main interests are in network analysis and visualization, with application to social networks in particular. He is a member of the board of directors of the International Network for Social Network Analysis (INsNA), associate editor of Social Networks, and area editor of Network Science. He was a member of the Graph Drawing Steering Committee from 2007 to 2014, and is on the editorial board of the Journal of Graph Algorithms and Applications.