Graph Ordering and Clustering — A Circular Approach

Anna Beer
LMU Munich
Munich, Germany
beer@dbs.ifi.lmu.de

Thomas Seidl
LMU Munich
Munich, Germany
seidl@dbs.ifi.lmu.de

ABSTRACT

As the ordering of data, particularly of graphs, can influence the result of diverse Data Mining tasks performed on it heavily, we introduce the Circle Index, the first internal quality measurement for orderings of graphs. It is based on a circular arrangement of nodes, but takes in contrast to similar arrangements from the field of, e.g., visual analytics, the edge lengths in this arrangement into account. The minimization of the Circle Index leads to an arrangement which not only offers a simple way to cluster the data using a constrained MinCut in only linear time, but is also visually convincing. We developed the clustering algorithm CirClu, which implements this minimization and MinCut, and compared it with several established clustering algorithms achieving very good results. Simultaneously we compared the Circle Index with several internal quality measures for clusterings. We observed a strong coherence between the Circle Index and the matching of achieved clusterings to the respective ground truths in diverse real world datasets.

CCS CONCEPTS

• **General and reference** → **Evaluation**; • **Theory of computation** → **Unsupervised learning and clustering**; • **Human-centered computing** → Graph drawings;

KEYWORDS

Graph Ordering, Clustering, Graphs, Quality measure

ACM Reference Format:
Anna Beer and Thomas Seidl. 2019. Graph Ordering and Clustering — A Circular Approach. In *31st International Conference on Scientific and Statistical Database Management (SSDBM '19), July 23–25, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3335783.3335802

1 INTRODUCTION

Clustering, the art of partitioning data s.t. similar elements belong to the same group, is an established problem for Data Scientists. In contrast, orderings of data are still a neglected subject, even though they do not only serve as useful interim step for clustering algorithms, but also deliver useful information about data, which a pure partitioning cannot provide. The order of nodes in a graph can have severe influence on the clustering algorithm performed on it, and a good ordering enables solving the constrained MinCut problem in linear time, avoiding the exponential complexity of the unconstrained MinCut problem.

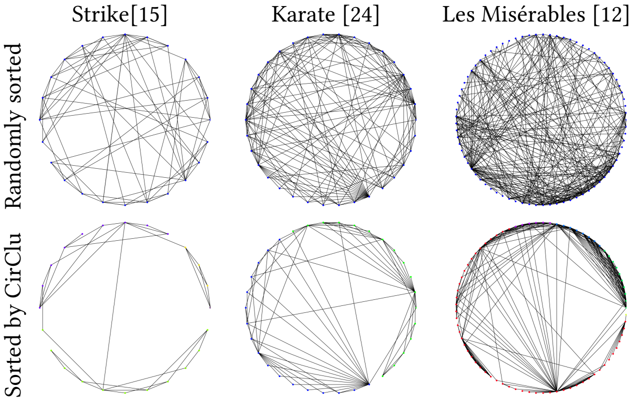

Figure 1: Three of our experiments, sorted randomly vs. sorted by CirClu aiming for a low Circle Index

We introduce a circular embedding of graphs onto the one-dimensional unit-sphere, which minimizes the edge lengths and leads to the Circle Index, our novel internal quality measure for orderings of nodes. Even though circular arrangements of graphs are quite common in visual analytics, none of them regards the edge lengths or uses the circular arrangement for further clustering or mathematical analysis of the respective network. In Fig. 1 we show the main effect of our optimization criterion: the confusing representations of the graphs above are randomly ordered. On the bottom they are ordered and colored according to our simple iterative clustering algorithm, CirClu (Circular Clustering), which we developed to underline the usefulness of our proposed ordering: it minimizes the Circle Index and performs a MinCut on the result, where there are only n possible cuts. Even though the clustering algorithm and also the Circle Index itself are work in progress and preliminary yet, they already deliver surprisingly good results.

We give an overview over related work w.r.t. graph orderings as well as internal quality measures and clustering in Sec. 2. In Sec. 3 we give the mathematical background for our work. In Sec. 4 we define and investigate the Circle Index, which is an internal robust quality measure for orderings of graphs. In Sec. 5 we introduce the clustering algorithm CirClu, which shows the expressiveness of the Circle Index based on several real world experiments presented in Sec. 6. Sec. 7 illustrates a multitude of future work enabled by the circular ordering. Our main contributions are as follows:

- We examine the importance of orderings of graphs
- We introduce an internal quality measure for orderings of graph, the Circle Index. It is, simplified, based on the average length of an edge if all nodes of a graph are arranged on a circle.

SSDBM '19, July 23–25, 2019, Santa Cruz, CA, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6216-0/19/07…$15.00
https://doi.org/10.1145/3335783.3335802

- We suggest CirClu, a clustering algorithm minimizing this measure iteratively and performing a MinCut on the result, which is then of only linear complexity.

2 RELATED WORK

Even though circular arrangement is already used in several works, they are not comparable to ours: [17] does not work on graphs, [8] uses several small circles, [3] and [14] focus on the aesthetic of the layout with the goal to present graphs neatly or with as few intersecting edges as possible. The ways to achieve such a layout are also manifold: [8] uses circular dilation, [3] chooses a combination of greedy and empirical, and [2] uses a force-directed and fuzzy multilevel approach.

The Circle Index is novel and fills a great gap: there are no quality measures for orderings of graphs yet. Existing quality measures are usually for clusterings and regard only the partitioning, but not the distribution of the nodes inside a cluster. Apart from that there are altogether only a few measures for clusterings without groundtruth — which is not available for most real world datasets. [1] compare those common quality metrics and discover that they all have some weaknesses, most of which cannot be true for the Circle Index by definition: They give better results for smaller numbers of clusters (Modularity, Conductance, Coverage) or larger ones (Silhouette Coefficient, Performance). They cannot deal with high numbers of singleton clusters (Silhouette Coefficient, Modularity) or with large networks (Performance). They do not regard internal cluster/edge density (Coverage, Conductance) or are expensive to calculate (Silhouette Coefficient). All of those metrics are based on the number of edges between different clusters and the number of edges inside the clusters, so those biases are founded already in their definitions. By contrast, Circle Index regards both types of edges (between and inside clusters) indirectly using the circular arrangement of nodes.

Comparative Methods. To demonstrate the expressiveness of the Circle Index, we compare CirClu to different clustering algorithms of diverse areas in Sec. 6, where we use the adjacency matrix as input if necessary. We compare with centroid based k-Means [13], two hierarchical agglomerative clustering methods [22] and Agglomod [16], Eigenvector based Spectral Clustering [18, 20, 21], message passing Affinity Propagation [10], and modularity minimizing Community Louvain Algorithm [4].

Internal Quality Measures of Clusterings. We compare the Circle Index which we adapt for clusterings in Sec. 6 with Modularity, Coverage, and Performance. Modularity measures the strength of division into clusters and gives information about the community structure of networks. Coverage is the relation of the summarized weight of intra-cluster edges to the weight of all edges. Performance is the relation between internal edges in a cluster and edges that do not exist between cluster's nodes to other nodes in the graph.

Graph Orderings. There are related algorithms, which order graphs, but, to the best of our knowledge, they are all looking for linear orderings. [7] is based on the spectrum of the Laplacian, using the Fiedler vector [9], the eigenvector corresponding to the second lowest eigenvalue, to either sort a graph or bisect it. Cuthill-McKee [5] orders nodes s.t. the adjacency matrix is a band matrix with small bandwidth. Those sortings may seem similar to CirClu, but since nodes are ordered linearly, there is a first and a last node which is not naturally for all graphs but trees. Thus, especially cyclic and strongly connected graphs can not be ordered well, as Sec. 6 shows. In a topological ordering [11] for every directed edge (u, v) in a graph, u comes before v. That is only possible if the graph has no directed cycles, i.e., the graph is a tree, and it does not lead to a clustering which finds the characteristically highly interconnected groups. Newer algorithms like [23] look for graph orderings to speedup CPU computing to enhance efficiency of graph algorithms, but do not group similar or highly interconnected nodes together, but such which are frequently accessed together.

3 PRELIMINARIES

Circular Mean. The mean M of a point set N is the point with the lowest possible sum of distances to all points in N, colored red in Fig. 2 for $N = P, Q, R$. The circular mean CM of a point set where all points lie on a circle is the point on the circle line which is closest to the mean of those points, colored purple in the figure.

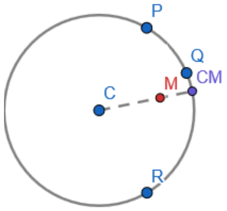

Figure 2: The mean M and the circular mean CM of points P, Q and R.

Edge Length. If we arrange all nodes of a graph of size n uniformly in a circle of arbitrary constant radius r we can use basic geometry to calculate the length of an edge, which corresponds to the distance between the according nodes. For the i-th node n_i and the j-th node n_j in the circle we can calculate their distance $d(n_i, n_j)$ as follows:

$$d(n_i, n_j) = 2r * sin\left(\pi \frac{|i-j|}{n}\right) \tag{1}$$

Average Edge Length. Given an ordered graph $G = (V, E)$, where "ordered" means that there exists an injective mapping $f : V \rightarrow \{0, 1, ..., |V|\}$, where the nodes are uniformly arranged in a unit-sphere following the order given by f, we can compute the Average Edge Length: Let $e.n_1$ refer to the source node of edge e and $e.n_2$ refer to its target node. Using Eq. 1, we get for the Average Edge Length $d(G)$ of a graph:

$$d(G) = \frac{1}{|E|}\sum_{e \in E} d(e.n_1, e.n_2) = \frac{2}{|E|}\sum_{e \in E} sin\left(\pi \frac{|f(e.n_1) - f(e.n_2)|}{|V|}\right) \tag{2}$$

Lower Bound for Average Edge Length. A lower bound b_l for the Average Edge Length can be approximated efficiently by regarding all nodes independently and assuming they all lie in the middle of their neighbors, which are located as near as possible to the respective node. This lower bound can only be reached for some special graphs, but captures the structure of the graph enough for our purpose. Let $N(n)$ be the neighbors of node n and div returns the integer quotient of an Euclidean division, then:

$$b_l(G) = \frac{1}{|E|}\sum_{n \in V}\sum_{i=1}^{|N(n)|} sin\left(\pi \frac{i\ div\ 2}{|V|}\right) \tag{3}$$

The 2 in the numerator of Eq. 2 is neutralized as every edge is counted twice. $i\ div\ 2$ origins in the assumed optimal alignment: the minimal edge length from a node to its neighbor is the length to the very next slot on the circle if the node has maximal two neighbors (one left, one right). The third and forth neighbor would then each be 2 slots away and so on, resulting in $i\ div\ 2$.

Minimum Cut. To obtain clusters based on the ordered graph, we partition it similar to the Minimum Cut (MinCut) problem. Where the original MinCut has exponentially many possibilities to set cuts, we have only $|V|$, starting in the center of the circle. The number of edges cut can easily be counted and should be low to produce a good separation of two clusters. To avoid getting clusters of only one or a few rather outlying points we do not minimize the number of edges cut per se, but the ratio R of cut edges to possible cut edges:

$$R = \frac{|E_C \cap (U \times W)|}{|U| \cdot |W|}. \tag{4}$$

U and W are two disjoint sets in which the cut divides the nodes of a cluster C and E_C are all edges lying in the cluster which is split. Edges between nodes which are both not in the cut cluster C are not counted.

4 CIRCLE INDEX

To obtain comparable values for the quality of orderings of a graph we take the average edge length as well as its lower bound into account. Otherwise we would receive lower (meaning better quality) values for larger or sparser graphs. Thus we define the Circle-Index $CI(G)$ of a graph as relation between the average edge length $d(G)$ of the graph and the lower bound $b_l(G)$ for it:

$$CI(G) = \frac{d(G)}{b_l(G)} \tag{5}$$

The Circle Index closes two gaps in the field of quality measures for graphs: While the ordering of nodes in a graph is not only an important preprocessing step in many clustering algorithms, but also highly beneficial for soft and fuzzy clustering, there are no quality measures for it, yet. Note that every graph clustering algorithm needs to use *some* ordering of the nodes for the input of the graph alone. Second, it allows to evaluate orderings without knowing the labeling, which is very often not available. It particularly overcomes the limitations of existing measures for evaluation of clusterings discussed in Sec. 2.

5 CIRCLU

CirClu is a simple iterative algorithm minimizing the Circle Index CI and performing a MinCut afterwards. As points of a graph $G = (V, E)$ lie uniformly distributed in a circle, there are $|V|$ possible slots a node can occupy, which are handled as a linked list. One by one, each node is moved to the slot nearest to the circular mean (see Sec. 3) of its neighbors, which is a greedy minimization of this point's edge lengths. Nodes between the new slot of the moved node and the old slot move up into the direction where less nodes have to move (to close the gap emerged at the old slot and make place at the new slot).

When the CI does not decrease anymore, we reached a local minimum and can now easily compute where to set cuts similar to the MinCut problem to partition the graph: we only have $|V|$ possibilities to set the cut, where there would be $2^{|V|}$ different possibilities to cut arbitrarily through the graph. A cut is performed from the center of the circle to the circle line and the intersected edges are counted for every possible cut. The cuts are ordered by ascending R as defined in Eq. 4 and set one by one until the desired number of clusters is reached. The first partitioning emerges with the second cut, afterwards every cut generates a new cluster by dividing an old.

Weighted graphs. To handle weighted graphs we can modify equations 2 and 3 by multiplying the length of an edge e with its weight $w(e)$. With that nodes with higher weighted edges are moved closer together than those with lower weighted ones.

Runtime Efficiency. In every iteration step the calculation of the optimal position for a node takes $O(|E|)$ since every node has at most $|E|$ neighbors. This calculation is done for every node, so the runtime until the graph is completely ordered is $O(i * |E| * |V|)$, for i the number of iterations until convergence. In the second step there are $|V|$ possibilities to make a cut between two clusters. For every possibility the ratio R from Equation 4 is calculated which needs $O(|E|)$ each time, so altogether $O(|E| * |V|)$. With those two steps we obtain a runtime of $O(i * |E| * |V|)$ for the complete algorithm.

6 EXPERIMENTAL EVALUATION

We compare the Circle Index (CI) with internal quality measures for clusters, Modularity, Coverage, Performance, and Conductance. To evaluate the results w.r.t. real world data, we regarded only non-synthetic data with a given ground truth. To be independent of flaws in diverse external quality measures aligning clustering results with labels, we use the average of Normalized Mutual Information, Adjusted Rand Index, V-Measure, and Adjusted Mutual Information, so that their biases compensate for each other, and abbreviate this average with EM (for external measures). Simultaneously, we compare k-Means (KM) [13], Agglomerative Clustering (AC) [22], Spectral Clustering (SC) [18, 20, 21], Affinity Propagation (AP) [10], the Community Louvain Algorithm (CL) [4] and Agglomod (A) [16] with CirClu in regards of both, the internal as well as the external measurements. To obtain the Circle Index for non-ordered, but clustered data, the nodes within each cluster are sorted by their degree. We used public python implementations on a 32 GB RAM, 3.4GHz machine for all experiments.[1]

We conducted experiments on the social networks "Zacharys Karate Club" [24] and "Strike" [6, 15], on the trading network "Worldtrade" [6, 19], and the co-occurrence network "Les Misérables" [12]. Results are shown in Fig. 3, where normCI is the normalized CI. The CI was normalized separately for every dataset to values between 0 and 1, corresponding to the worst resp. best result, by a linear normalization and subsequent subtraction from 1[2], to allow a simple visual analysis. We note, that in all cases, the best CI implies the best results w.r.t. the ground truth, i.e., the highest EM. As there are no appropriate ordering algorithms yet, and we could only compare with clustering algorithms, the inversion does not hold:

[1] Our code is online available under: https://github.com/p4nna/CirClu

[2] In detail, for all CIs, we subtracted the minimal occurring CI, divided this value by the range of occurring values for CI, and subtracted the result from 1.

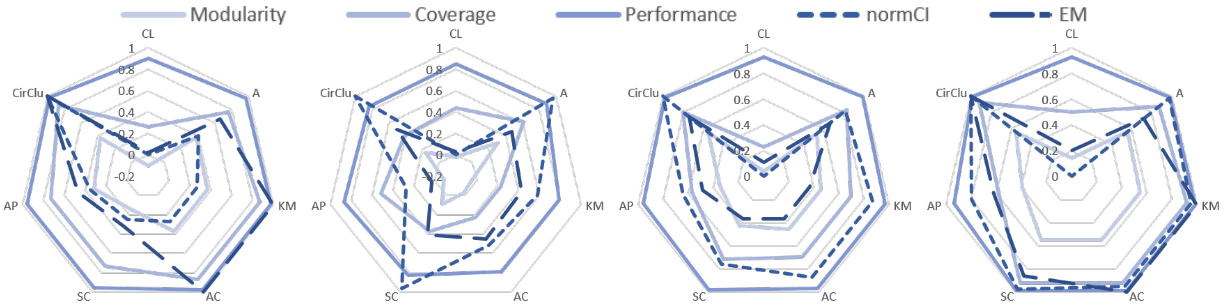

Figure 3: Results for datasets (left to right): Zacharys Karate Club, Worldtrade, Les Misérables, and Strike for algorithms Community Louvain (CL), Agglomod (A), k-Means (KM), Agglomerative Clustering (AC), Spectral Clustering (SC), Affinity Propagation (AP), and CirClu. EM is the average of the External Measures, and normCI the normalized Circle Index

a good clustering does not imply a good ordering as there are too many possible permutations of nodes inside each cluster. CirClu achieved one of the best results for all those datasets, even though they heavily discern in their form: Zacharys Karate Club and Strike have only two clusters, Worldtrade is highly interconnected with rather blurry clusters than clique-like ones, and Les Misérables has several hubs, the protagonists. We also note, that the trend of normCI and EM are, even though stretched for some algorithms, more similar to each other than to the other internal measurements, resulting in similar shapes on the radar charts in Fig. 3.

7 CONCLUSION

We introduced the Circle Index, an internal measure for the quality of an ordering of a graph, where there is no comparable measure yet. Sec. 6 showed, that the Circle Index is also applicable for clusterings without an underlying order inside the clusters. In comparison to established internal measures like Modularity, Coverage, or Performance it is often a better prediction for the quality of a clustering w.r.t. the ground truth. Our new clustering algorithm CirClu orders the nodes of a graph by minimizing the Circle Index and then partitions the graph by finding the minimum cuts inside that ordering. It works well for networks with almost all types of clusters.The embedding onto the unit sphere is natural and does not force the graph to have a first node and a last node (which makes only sense for loop free graphs which are rather rare). Every graph clustering algorithm needs to be given the graph in *any* order, so every clustering algorithm has to deal with it implicitly, even though most of them do not mention this at all. Our new concepts are extensible and build a broad basis to build on. We did not cover directed graphs yet, and different normalizations of the Circle Index should be discussed. Also we did not investigate the usage for Big Data yet, which we plan for future work, since we wanted to focus on the meaningfulness of found clusters. Besides it would be interesting to know how a higher dimensional sphere would affect the algorithm.

ACKNOWLEDGMENTS

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibilities for its content.

REFERENCES

[1] Hélio Almeida, Dorgival Guedes, Wagner Meira, and Mohammed J Zaki. 2011. Is there a best quality metric for graph clusters?. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 44–59.

[2] Mohammadreza Ashouri, Ali Golshani, Dara Moazzmi, and Mandana Ghasemi. 2016. Graphs Drawing through Fuzzy Clustering. *arXiv preprint arXiv:1603.07011* (2016).

[3] Michael Baur and Ulrik Brandes. 2004. Crossing reduction in circular layouts. *WG* 3353 (2004), 332–343.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[5] Elizabeth Cuthill and James McKee. 1969. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th national conference*. ACM, 157–172.

[6] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj. 2011. *Exploratory social network analysis with Pajek*. Vol. 27. Cambridge University Press.

[7] Chris Ding, Xiaofeng He, Hongyuan Zha, Ming Gu, and Horst Simon. 2001. Spectral min-max cut for graph partitioning and data clustering. (2001).

[8] Uğur Doğrusöz, Brendan Madden, and Patrick Madden. 1996. Circular layout in the graph layout toolkit. In *International Symposium on Graph Drawing*. Springer, 92–100.

[9] Miroslav Fiedler. 1973. Algebraic connectivity of graphs. *Czechoslovak mathematical journal* 23, 2 (1973), 298–305.

[10] Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science* 315, 5814 (2007), 972–976.

[11] Arthur B Kahn. 1962. Topological sorting of large networks. *Commun. ACM* 5, 11 (1962), 558–562.

[12] Donald Ervin Knuth. 1993. *The Stanford GraphBase: a platform for combinatorial computing*. Vol. 37. Addison-Wesley Reading.

[13] Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

[14] Erkki Mäkinen. 1988. On circular layouts. *International Journal of Computer Mathematics* 24, 1 (1988), 29–37.

[15] Judd H Michael. 1997. Labor dispute reconciliation in a forest products manufacturing facility. *Forest products journal* 47, 11/12 (1997), 41.

[16] Mark EJ Newman. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69, 6 (2004), 066133.

[17] Gonzalo E Paredes and Luis S Vargas. 2012. Circle-Clustering: A new heuristic partitioning method for the clustering problem. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 1–8.

[18] Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22, 8 (2000), 888–905.

[19] David A Smith and Douglas R White. 1992. Structure and dynamics of the global economy: network analysis of international trade 1965–1980. *Social forces* 70, 4 (1992), 857–893.

[20] X Yu Stella and Jianbo Shi. 2003. Multiclass spectral clustering. In *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE, 313.

[21] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 4 (2007), 395–416.

[22] Joe H Ward Jr. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association* 58, 301 (1963), 236–244.

[23] Hao Wei, Jeffrey Xu Yu, Can Lu, and Xuemin Lin. 2016. Speedup graph processing by graph ordering. In *Proceedings of the 2016 International Conference on Management of Data*. ACM, 1813–1828.

[24] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.