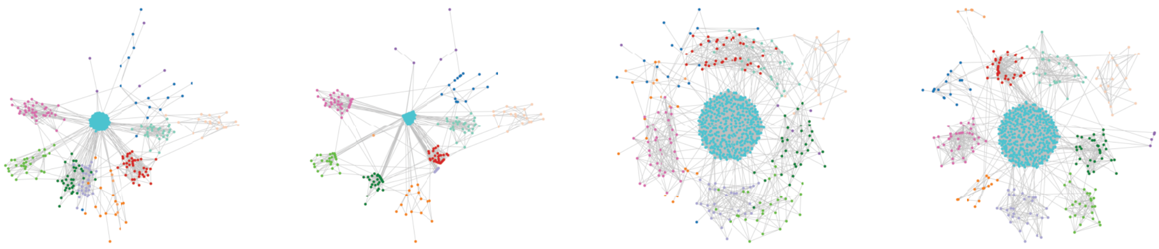
Erschienen in: 2024 IEEE 17th Pacific Visualization Conference, PacificVis 2024, Tokyo, Japan 23-26 April 2024 : Proceedings. Los Alamitos, CA ; u.a.: IEEE, 2024, S. 192-201. ISBN 979-8-3503-9380-4
https://doi.org/10.1109/pacificvis60374.2024.00029

Cluster-Faithful Graph Visualization: New Metrics and Algorithms

Shijun Cai* University of Sydney
Seok-Hee Hong† University of Sydney
Amyra Meidiana‡ University of Sydney
Peter Eades§ University of Sydney
Daniel Keim¶ University of Konstanz



(a) FR (b) `Cluster-kmeans`-FR (c) SM (d) `Cluster-HAC`-SM

Figure 1: Visual comparison on graph *rpg_5_400_12_1*: Our `Cluster-kmeans` and `Cluster-HAC` algorithms compute drawings with perfect cluster faithfulness, with all clusters visually separate, achieving significant improvement over existing graph drawing algorithms.

ABSTRACT

The cluster faithfulness metrics *CQ* measure how faithfully the ground truth clustering of a graph is represented as the geometric clustering in a drawing of the graph. Existing *CQ* metrics use *k*-means clustering, which effectively compute a geometric clustering when the cluster sizes are even, resulting in accurate *CQ* metrics. However, *k*-means clustering tends to compute clusters of even sizes and thus often fails to compute an accurate geometric clustering when the cluster sizes are uneven, leading to inaccurate *CQ* metrics.

In this paper, we present a new cluster faithfulness metric *CQ-HAC*, using HAC (Hierarchical Agglomerative Clustering). HAC can compute a more accurate geometric clustering for uneven cluster sizes than *k*-means clustering. Consequently, *CQ-HAC* can more accurately measure cluster faithfulness, regardless of whether the sizes of clusters are even or uneven. Moreover, we present two algorithms, `Cluster-kmeans` and `Cluster-HAC`, for optimizing cluster faithfulness of graph drawings. Extensive experiments show that in practice, both algorithms always compute perfectly cluster-faithful drawings (i.e., *CQ* = 1) in our experiments using various graphs with both even and uneven cluster sizes, achieving significant improvement over existing graph layouts, including cluster-focused layouts.

Index Terms: Human-centered computing—Visualization—Visualization techniques—Graph drawings

1 INTRODUCTION

Quality metrics (or *aesthetic criteria*) serve an important role in evaluating graph drawings as well as designing new algorithms to optimize the metrics [4]. Traditional aesthetic criteria mainly evaluate the *readability* of a drawing, such as edge crossings, edge bends, total edge length, and angular resolution. These readability metrics measure how well humans can understand data based on the visualization, however they become less effective for evaluating drawings of larger, complex graphs [15].

More recently, *faithfulness* metrics have been introduced for evaluating drawings of large complex graphs, measuring how faithfully a graph drawing *D* displays the ground truth structure of the graph *G* [23]. Faithfulness metrics measure the quality based on how well a visualization method maps the data to a drawing, as opposed to readability metrics which measure how well the human perception maps the drawing to knowledge and understanding.

For example, *stress* [4] is a *distance faithfulness* metric, measuring how proportional the graph theoretic distances in a graph *G* are to the Euclidean distances in a drawing *D* of *G*. Other examples include the *shape-based metrics* [5], measuring the similarity between a graph *G* and the *proximity graph* of a drawing *D*, and the *automorphism faithfulness* metrics [21], measuring how faithfully the ground truth automorphisms of *G* are displayed as *symmetries*.

In particular, the *cluster faithfulness* metrics *CQ* measure how faithfully the geometric clustering of vertices in *D* represents the ground truth clustering of vertices in *G* [20]. Existing *CQ* metrics compute the geometric clustering using *k-means clustering*, which divides a set into *k* subsets by minimizing within-class variance [18]; we denote *CQ* metrics using *k*-means as *CQ-kmeans*.

However, graphs often come with clustering structures of uneven cluster sizes. When the clustering includes clusters that are uneven in size, *k*-means clustering often fails to correctly identify the clusters, leading to less accurate *CQ* metrics.

Additionally, a number of graph drawing algorithms have been presented to display clusters or community structures, such as LinLog [25] and Backbone [26]. Although these algorithms are often able to obtain high cluster faithfulness, they are not specifically designed to optimize the cluster faithfulness metrics and do not always succeed in obtaining perfect cluster faithfulness [20]. Therefore, there is a clear gap in the literature for algorithms specifically designed to optimize cluster faithfulness in graph drawing.

In this paper, we present new cluster faithfulness metrics *CQ-HAC*, using *HAC (Hierarchical Agglomerative Clustering)*, a bottom-up hierarchical clustering algorithm [11]. HAC can accurately compute geometric clustering even when the cluster sizes are uneven,

*e-mail: shijun.cai@sydney.edu.au
†e-mail: seokhee.hong@sydney.edu.au
‡e-mail: amyra.meidiana@sydney.edu.au
§e-mail: peter.eades@sydney.edu.au
¶e-mail: keim@uni-konstanz.de

and as a result, *CQ-HAC* can effectively compute cluster faithfulness for various graphs with even or uneven cluster sizes.

We then present `Cluster-kmeans` and `Cluster-HAC`, algorithms for optimizing cluster faithfulness, which take an input drawing D of a graph G with ground truth clusters C, compute the geometric clustering C' in D, and then iteratively move the position of vertices in D until $C' = C$.

Our experiments show that in practice, both algorithms compute perfectly cluster-faithful drawings (i.e., $CQ = 1$). This is a significant improvement over existing cluster-focused graph layouts (e.g., [16, 25, 26]), which sometimes fail to obtain perfect *CQ* on graphs with uneven cluster sizes [20].

Our main contributions can be summarized as follows:

1. We present a new cluster faithfulness metric, *CQ-HAC*, which utilizes HAC to compute a geometric clustering. *CQ-HAC* more accurately measures the cluster faithfulness when the sizes of clusters are uneven, unlike previous cluster faithfulness metrics using k-means clustering (*CQ-kmeans*).

   Deformation experiments validate *CQ-HAC* can effectively measure cluster faithfulness for graphs with various clustering structures, and can measure cluster faithfulness more accurately than *CQ-kmeans* on graphs with uneven cluster sizes.

2. We present two algorithms for cluster-faithful graph drawing, `Cluster-kmeans` and `Cluster-HAC`, optimizing cluster faithfulness metrics computed using k-means clustering and HAC respectively. Experiments with a variety of graphs both with even and uneven cluster sizes show that in practice `Cluster-kmeans` and `Cluster-HAC` always compute perfectly cluster-faithful drawings (i.e., $CQ = 1$).

   Moreover, `Cluster-kmeans`-FR and `Cluster-kmeans`-SM, i.e., `Cluster-kmeans` combined with FR (Fruchterman-Reingold) [7] and SM (Stress Majorization) [8], as well as `Cluster-HAC`-FR and `Cluster-HAC`-SM, obtain significant improvement over FR and SM: on average 122% and 240% respectively, on *CQ-kmeans*, and 52% and 71% respectively, on *CQ-HAC*.

   Furthermore, experiments show that `Cluster-kmeans` and `Cluster-HAC` outperform existing cluster-focused graph drawing algorithms, such as LinLog [25], Backbone [26], and tsNET [16] on cluster faithfulness, always obtaining $CQ = 1$ in our experiments.

For example, Figure 1 shows a visual comparison of FR, `Cluster-HAC`-FR, SM and `Cluster-HAC`-SM on graph *rpg_5_400_12_1* with *uneven* cluster sizes. Both `Cluster-HAC`-FR and `Cluster-HAC`-SM produce perfectly cluster-faithful drawings, ensuring clear separation between clusters without any overlap, while FR and SM produce overlaps between the small clusters.

2 Related Work

2.1 Faithfulness Metrics

Traditionally, graph drawing is evaluated using *aesthetics*, which measure the readability of the drawing [4]. *Faithfulness* metrics, meanwhile, measure how faithfully the ground truth information is depicted in the visualization [23]. A visualization is "faithful" if the ground truth information is faithfully displayed in the visualization.

For example, *stress* [4] is a *distance faithful* metric, which measures the difference between the graph theoretic distance of vertices in a graph G and the Euclidean distance in a drawing D of G. Similarly, the *symmetry faithful* metrics [21] measure how the *automorphisms* of G are displayed as symmetries in D. The *neighborhood faithful* metric [2, 24] measures the similarity between the neighbors of vertices in G and their geometric neighbors in D.

The *shape-based metrics* [5, 12, 19] are one of the widely used faithfulness metrics for a drawing D of a large graph G, which measure how faithfully the structure of G is displayed as a similar "shape" in D. Specifically, it measures the similarity between G and a *proximity graph* [29] P, such as the Relative Neighborhood Graph and the Gabriel Graph, computed from D.

2.2 Cluster Faithfulness Metrics

The *cluster faithfulness metrics CQ-kmeans* [20] measure how faithfully the ground truth clustering C of a graph G is displayed in a drawing D of G. The metric computes a geometric clustering C' of vertices in D using k-means clustering, and computes the similarity between C and C' using *clustering comparison metrics* as follows:

- Adjusted Rand Index (ARI), based on the number of pairs of elements classified into the same or different groups in both C and C' [14, 27];
- Adjusted Mutual Information (AMI), based on how much information of a random variable can be gathered from another random variable [3, 30];
- Fowlkes-Mallows Index (FMI) computed using the number of true positives, false positives, and false negatives [6];
- Homogeneity (HOM), the extent to which each cluster in C' only contains members of the same cluster in C [28]; and
- Completeness (CMP), the extent to which all members of a cluster in C are assigned to the same cluster in C' [28].

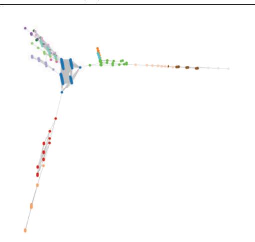
Validation experiments have shown that the *CQ-kmeans* metric is able to effectively measure cluster faithfulness, where ARI and FMI are the most effective clustering comparison metrics [20].

Moreover, *CQ-kmeans* has been used to evaluate the cluster faithfulness of various popular graph drawing algorithms, showing that cluster-focused layouts such as LinLog (LL) [25], Backbone (BB) [26], and tsNET [16] generally obtain higher cluster faithfulness than other layouts. However, these experiments use graphs with relatively well-balanced clusters.

3 CQ-HAC: Cluster Faithfulness Metrics with HAC

CQ-kmeans has limitations on measuring the cluster faithfulness for graphs with uneven cluster sizes, due to the k-means clustering finding clusters with roughly equal sizes. See Table 1, where tsNET produces a more cluster-faithful drawing (i.e., each cluster is well separated) than Pivot MDS (PMDS) [1], however *CQ-kmeans* of tsNET is much lower than that of PMDS.

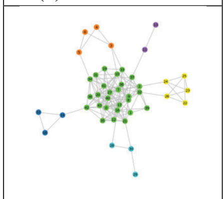
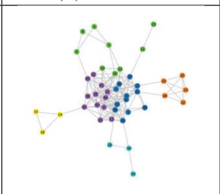
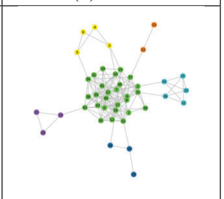
Table 1: An example where $CQ\text{-}kmeans_{ARI}$ fails to measure cluster faithfulness while $CQ\text{-}HAC_{HOM}$ succeeds on graph *rpg_2_500_16_1* with uneven cluster sizes, drawn using PMDS and tsNET.

(a) PMDS	(b) tsNET
	
CQ-kmeans: 0.49 *CQ-HAC*: 0.78	*CQ-kmeans*: 0.22 *CQ-HAC*: 0.92

To overcome the limitation of *CQ-kmeans* for graphs with uneven cluster sizes, we present new cluster faithfulness metrics *CQ-HAC* using HAC [11]. HAC constructs a clustering hierarchy by progressively combining clusters from individual observations in a bottom-up manner. It begins with each observation assigned to its own cluster and gradually merges pairs of clusters as the hierarchy is formed. Specifically, we use *average linkage* as the merging criteria, which minimizes the average distance between points in pairs of clusters. This method can preserve the structures for larger clusters, while effectively addressing the challenges posed by uneven cluster sizes. HAC also can take as a parameter the number of desired clusters, where having the same number of geometric clusters as ground truth clusters is an essential property for cluster faithfulness.

Table 2 shows a comparison between k-means clustering and HAC for a graph with uneven cluster sizes. k-means fails to accurately compute the ground truth clustering, breaking up the large cluster and combining small clusters, while HAC accurately computes the same ground truth clustering.

Table 2: Comparison between k-means and HAC on graph *rpg_2_27_6_1*, drawn using Backbone layout, with vertices color-coded based on (a) the ground truth clustering, (b) geometric clustering computed by k-means, and (c) geometric clustering computed by HAC. k-means fails to accurately compute the ground truth clustering, while HAC successfully computes the ground truth clustering.

(a) Ground truth	(b) k-means	(c) HAC
		

CQ-HAC is computed using the following steps (see Figure 2):

1. Compute a drawing D of a graph G.
2. Compute a geometric clustering C' of vertices in drawing D using HAC.
3. Compute the similarity between the ground truth clustering C of G and C' of D, using clustering comparison metrics.

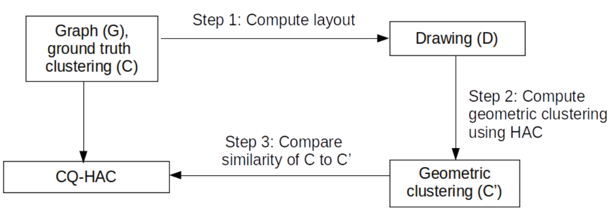


Figure 2: *CQ-HAC* metrics.

CQ-HAC computes scores within the range of [0, 1], where a higher score indicates a more cluster-faithful drawing.

4 Algorithms for Cluster-Faithful Graph Drawings

We now present cluster-faithful graph drawing algorithms, `Cluster-kmeans` and `Cluster-HAC`, for optimizing the cluster faithfulness. Given an initial drawing D of graph G, our algorithms compute a cluster-faithful drawing D', by iteratively improving the position of vertices such that the geometric clustering C' in D' corresponds to the ground truth clustering C of G.

Both algorithms can be used as a post-processing step, taking a graph drawing as an input initial drawing, or can be integrated into a graph drawing algorithm, as an additional optimization step.

4.1 Algorithm `Cluster-kmeans`

We first present the algorithm `Cluster-kmeans`, which optimizes *CQ-kmeans*, using the Voronoi Diagram [29], which partitions a

Algorithm 1: Cluster-kmeans

Input: Graph $G = (V, E)$, ground truth clustering of vertices $C = \{c_i, i = 1...k\}$, drawing D of G

while $CQ\text{-}kmeans(C, C') < 1$ **do**
  Compute geometric clustering $C' = \{c'_j, j = 1...k\}$ using k-means Clustering
  Compute the Voronoi diagram based on the cluster centroids of C'
  // Map clusters in C to clusters in C'
  for $c_i \in C$ **do**
    $max(JS) = 0$
    for $c'_j \in C'$ **do**
      $JS(c_i, c'_j) = \frac{|c_i \cap c'_j|}{|c_i \cup c'_j|}$
      if $JS(c_i, c'_j) > max(JS)$ **then**
        $max(JS) = JS(c_i, c'_j)$
      endif
      Map c_i to c'_j which computes $max(JS)$
    end
  end
  // Compute all moving vertices V_{move}
  $V_{move} = \{\}$
  for *each pair of mapped clusters* (c_i, c'_j) **do**
    Add all vertices in $c_i \setminus c'_j$ to V_{move}
  end
  // Update vertex positions in V_{move}
  for $u \in V_{move}$ **do**
    c_i: ground truth cluster containing u
    c'_m: geometric cluster containing u
    c'_j: geometric cluster mapped to c_i
    l_{mj}: line segment between $Centroid(c'_m)$ and $Centroid(c'_j)$
    $Border_{mj}$: border shared by Voronoi cells $Vor(c'_m)$ and $Vor(c'_j)$
    Compute P_{mj}, the intersection of l_{mj} and $Border_{mj}$
    Move u by $(0.1 \times d(P_{mj}, Centroid(c'_j))$ towards $Centroid(c'_j)$
  end
end

space based on a set of points $P = p_1, \ldots, p_k$ where the cell containing a point $p_i \in P$ contains all points closer to p_i than any other point in P. Each cell corresponds to a cluster in a geometric clustering.

Let $C = \{c_i, i = 1, \ldots, k\}$ denote the ground truth clustering of the vertices of a graph G with k clusters. Given a drawing D of G, we first compute the k-means clustering $C' = \{c'_j, j = 1, \ldots, k\}$. For each cluster c'_j in C', we compute the centroid $Centroid(c'_j)$ (i.e., the average of the coordinates of all points belonging to c'_j), and then compute the Voronoi Diagram using the centroids, where we denote the Voronoi cell of $Centroid(c'_j)$ as $Vor(c'_j)$.

We use the Voronoi Diagram for determining the regions for geometric clusters, to improve the quality of the geometric clustering using k-means, i.e., more similar to the ground truth clustering. Note that a brute force method for displaying clusters by assigning k randomly-placed boxes in the drawing area can easily lead to worse quality drawing with possibly many crossings.

Next, we map each cluster $c_i \in C$ to $c'_j \in C'$, by computing the *JS (Jaccard Similarity)* between c_i and c'_j, where $JS(c_i, c'_j) = \frac{|c_i \cap c'_j|}{|c_i \cup c'_j|}$. Specifically, a cluster c_i is mapped to a cluster c'_j, if $JS(c_i, c'_j)$ is highest among other clusters in C'.

For a pair of mapped clusters (c_i, c'_j), we identify "misplaced" vertices $u \in c_i$ and $u \notin c'_j$, and add them to V_{move}, the set of vertices that need to be moved in a drawing to optimize cluster faithfulness.

Finally, for each $u \in V_{move}$, we compute the direction and extent of movement. For a vertex u such that $u \in c_i$ and c_i is mapped to c'_j, however $u \in c'_m \neq c'_j$, we first compute the intersection point P_{mj} between l_{mj} (line segment between $Centroid(c'_m)$ and $Centroid(c'_j)$) and $Border_{mj}$ (border shared by the two Voronoi cells $Vor(c'_m)$ and $Vor(c'_j)$). We then move u inside of $Vor(c'_j)$ such that it can belong to c'_j, when the k-means clustering is computed in the next iteration. Specifically, we move u towards the direction of $Centroid(c'_j)$ by 0.1 times the distance between P_{mj} and $Centroid(c'_j)$. The steps are repeated until $CQ\text{-}kmeans(C, C') = 1$, see Algorithm 1.

The runtime of `Cluster-kmeans` is $O(n^2 + kn)$, where k is the number of clusters and $n = |V|$. Using Lloyd's algorithm [17], k-means clustering can be computed in $O(kn)$ time, and the Voronoi Diagram on k centroids can be computed in $O(k \log k)$ time. Computing the mapping between clusters takes $O(n^2)$, and moving the vertices towards the centroid of the mapped cluster takes $O(n)$ time.

4.2 Algorithm `Cluster-HAC`

We now present `Cluster-HAC`, which optimizes the cluster faithfulness metrics using HAC. For each pair of clusters $(c_i \in C, c'_j \in C')$, any vertex $u \in c_i$ that does not currently belong to c'_j is moved towards the position of the other vertices belonging to c'_j.

Let $C = c_i, i = 1...k$ be the ground truth clustering of a graph G, and l denote the average edge length in D. Given a drawing D of G, we first compute HAC geometric clustering $C' = \{c'_j, j = 1...k\}$ of vertices in D. Then, we map clusters in C to their counterparts in C', and compute the set V_{move} using the same methods described in Section 4.1.

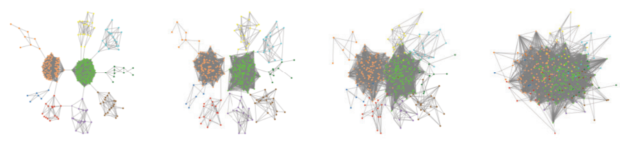
For the final step of moving the vertices $u \in V_{move}$, we first find the vertex $w \in c'_j$ nearest to u. Then we move u towards the direction of w by a magnitude of $d(u, w) + l$.

The above steps are repeated until $CQ\text{-}HAC(C, C') = 1$, see Algorithm 2 for the details. The total runtime of `Cluster-HAC` is $O(n^2)$: HAC using average linkage can be computed in $O(n^2)$ time [22]. Computing the mapping between two clusters takes $O(n^2)$, and moving the vertices towards the correct clusters takes $O(n \log n)$ time.

Algorithm 2: `Cluster-HAC`

Input: Graph $G = (V, E)$, ground truth clustering of vertices $C = \{c_i, i = 1...k\}$, drawing D of G, average edge length l in D

while $CQ\text{-}HAC(C, C') < 1$ **do**
  Compute geometric clustering $C' = \{c'_j, j = 1...k\}$ using HAC
  // Map clusters in C to clusters in C'
  for $c_i \in C$ **do**
    $max(JS) = 0$
    for $c'_j \in C'$ **do**
      $JS(c_i, c'_j) = \frac{|c_i \cap c'_j|}{|c_i \cup c'_j|}$
      if $JS(c_i, c'_j) > max(JS)$ **then**
        $max(JS) = JS(c_i, c'_j)$
      endif
      Map c_i to c'_j which computes $max(JS)$
    end
  end
  // Compute all moving vertices V_{move}
  $V_{move} = \{\}$
  for *each pair of mapped clusters* (c_i, c'_j) **do**
    Add all vertices in $c_i \setminus c'_j$ to V_{move}
  end
  // Update vertex positions in V_{move}
  for $u \in V_{move}$, $u \in c_i$ **do**
    c'_j: geometric cluster mapped to c_i
    w: nearest vertex to u in c'_j
    Move u by $(d(u, w) + l)$ towards w
  end
end



(a) Step 0 (b) Step 2 (c) Step 5 (d) Step 9

Figure 3: Deformation experiment using graph *rpg_6_200_10_2* with uneven cluster sizes, drawn using Backbone layout.

5 CQ-HAC Metrics Validation Experiment

5.1 Experiment Design and Data Sets

To validate the effectiveness of *CQ-HAC*, we conduct deformation experiments. First, we start with highly cluster-faithful graph drawings, which clearly preserve the ground truth clustering (i.e., all clusters are distinguishable without overlaps). We then gradually deform the drawing to be less cluster faithful.

For each drawing, we apply nine steps of deformation. In each step, we perturb the coordinates of each vertex by a small value within the range $[0, \delta]$, where δ is a parameter determined by the drawing area. Specifically, δ is set to a value between 0.05 and 0.1, multiplied by the width of the drawing area. For each deformation step, we compute *CQ-HAC* and *CQ-kmeans* for comparison.

Figure 3 shows an example deformation experiment with a graph with uneven cluster sizes, with vertices colored based on the ground truth clustering. At Step 0, all clusters are clearly separated from each other in the drawing, with $CQ\text{-}HAC = 1$. As the positions of vertices are perturbed, vertices from different clusters are progressively mixed together, resulting the clusters gradually overlap their boundaries, leading to lower cluster faithfulness.

We formulate the following hypotheses:

Hypothesis 1 *CQ-HAC scores decrease as drawings are deformed.*

Hypothesis 2 *For graphs with uneven cluster sizes, CQ-HAC can measure the cluster faithfulness more accurately than CQ-kmeans.*

We conduct experiments using two data sets: six graphs with *even* cluster sizes from [20], and six graphs with *uneven* cluster sizes, computed using the random partition graph generator in NetworkX [10]. Specifically, each graph we generated is named in the format [rpg]_[min cluster size]_[max cluster size]_[number of clusters]_[number of very big clusters], where rpg denotes the *random partition graph* method, and number of very big clusters is 0 for graphs with even cluster sizes. See Table 3 for the details.

To compute good initial shape-faithful drawings (i.e., $CQ\text{-}HAC = 1$ or almost 1), we use Backbone (BB) [26] and SM layouts.

5.2 Results

Figure 4 shows the *CQ-HAC* and *CQ-kmeans* scores for each deformation step, averaged per data set. Clearly, *CQ-HAC* scores decrease

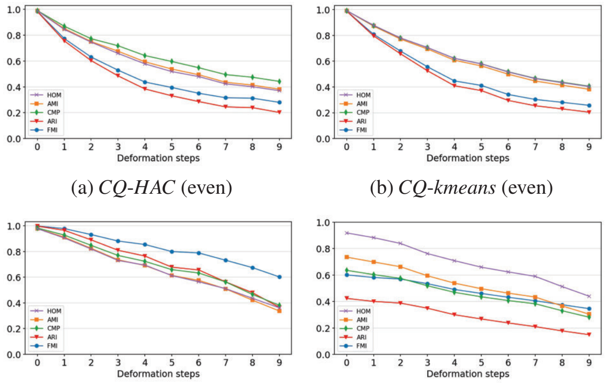
Table 3: Validation data sets for graphs with (a) even, (b) uneven cluster sizes.

(a) Even

Graph	$\lvert V \rvert$	$\lvert E \rvert$	k	*dens*
clust_1000_10_06_03	898	31516	9	35.10
clust_500_10_08_03	439	9552	9	21.76
gnm_30_60_8_2700_30	2685	26098	30	9.72
regular_3_1800_20_055	1773	53103	20	29.95
regular_4_3000_30_08	3045	124727	30	40.96
tree_2000_15_08_02	2082	164180	15	78.86

(b) Uneven

Graph	$\lvert V \rvert$	$\lvert E \rvert$	k	*dens*
rpg_2_200_10_1	310	8133	10	26.24
rpg_2_500_16_1	832	25830	16	31.05
rpg_2_600_13_2	861	47029	13	54.62
rpg_3_200_11_3	531	10592	11	19.95
rpg_6_200_10_2	424	7988	10	18.84
rpg_7_600_8_3	1138	94596	8	83.12



(a) *CQ-HAC* (even) (b) *CQ-kmeans* (even)

(c) *CQ-HAC* (uneven) (d) *CQ-kmeans* (uneven)

Figure 4: Average of *CQ-HAC* (resp., *CQ-kmeans*) for validation experiments on graphs with even (resp., uneven) cluster sizes.

along each deformation step as shown in Figures 4a, 4c , confirming Hypothesis 1.

Note that *CQ-HAC* accurately measures cluster faithfulness for drawings of graphs with uneven cluster sizes, where *CQ-kmeans* fails. Figure 4c shows that *CQ-HAC* correctly computes perfect scores for cluster-faithful drawings at Step 0, while Figure 4d shows that *CQ-kmeans* compute disproportionately low scores for the initial cluster-faithful drawings. While the absolute value of the *CQ-HAC* at the last step is higher than *CQ-kmeans*, for all *CQ-HAC* (except $CQ\text{-}HAC_{FMI}$) the ratio between the lowest and highest values of *CQ-HAC* is greater than that of *CQ-kmeans*, showing a clearer divide between drawings with high and low cluster faithfulness. This supports Hypothesis 2.

5.3 Discussion and Summary

For graphs with even cluster sizes, *ARI* (red) is the most effective for computing *CQ-HAC*, followed by *FMI* (blue), which is consistent with *CQ-kmeans* [20]. See Figures 4a, 4b with the steeper red and blue curves, showing higher sensitivity to changes of cluster faithfulness in drawings.

Meanwhile, for graphs with uneven cluster sizes, *HOM* (purple) and *AMI* (orange) are the most effective for computing *CQ-HAC* with higher sensitivity, see Figure 4c.

6 Cluster-kmeans Experiments

6.1 Experiment Design and Data Sets

We evaluate the effectiveness of `Cluster-kmeans`, as an improvement over the popular FR and SM layouts, using the *CQ-kmeans* metrics. We use the data sets from [20] and additional synthetic sparse (resp., dense) graphs with even (resp., uneven) cluster sizes, computed by the *random partition graph* generator in NetworkX [10], with the same naming scheme as the graphs in Section 5.1. Table 4 shows the details.

For initial layouts, we use *Fruchterman-Reingold (FR)* [7], an example of a *force-directed layout* which is the most popular type of graph layouts, and *Stress Majorization (SM)* [8], an example of stress-based layout which aims to minimize the stress (i.e., distance faithfulness) in a drawing. We denote the algorithm `Cluster-kmeans` using the initial layout FR (resp., SM) as `Cluster-kmeans`-FR (resp., SM). For our experiments, we hypothesize the performance of `Cluster-kmeans` algorithms as follows:

Hypothesis 3 *`Cluster-kmeans`-FR and `Cluster-kmeans`-SM compute better cluster-faithful drawings than FR and SM.*

Hypothesis 4 *`Cluster-kmeans`-FR and `Cluster-kmeans`-SM always obtain CQ-kmeans* $= 1$.

Table 4: Data sets for evaluating `Cluster-kmeans` and `Cluster-HAC`: (a) graphs used in [20], graphs with (b) even cluster sizes, and (c) uneven cluster sizes.

(a) graphs in [20]

Graph	$\lvert V \rvert$	$\lvert E \rvert$	k	*dens*
clusts_tree_30_merge	299	1288	24	4.31
clust_500_10_08_03	439	9552	9	21.76
regular_3_1800_20_055_00005	1773	53103	20	29.95
bp_2000_10_20_05_005_001	1797	49210	30	27.38
clust_bigsmall_2000_20_10_02	2000	54749	30	27.37
wheel_2500_25_04_00005	2485	68844	25	27.70
SS-Butterfly-085	832	12979	10	15.60
email-Eu-core-lcc	986	16064	12	16.29

(b) even

Graph	$\lvert V \rvert$	$\lvert E \rvert$	k	*dens*
rpg_3_7_7_0	40	118	7	2.95
rpg_11_20_13_0	201	987	13	4.91
rpg_22_49_15_0	531	10399	15	19.58
rpg_40_55_14_0	651	17144	14	26.33
rpg_50_69_12_0	715	15361	12	21.48
rpg_65_80_15_0	1086	6375	15	5.87
rpg_76_88_15_0	1259	8615	15	6.84
rpg_101_120_18_0	1988	40615	18	20.43

(c) uneven

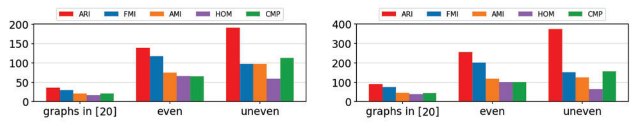
Graph	$\lvert V \rvert$	$\lvert E \rvert$	k	*dens*
rpg_3_200_11_2	390	1872	11	4.80
rpg_7_400_12_1	646	25022	12	38.73
rpg_5_400_12_1	658	16843	12	25.60
rpg_3_400_14_1	681	16873	14	24.78
rpg_4_300_11_3	710	5759	11	8.11
rpg_5_500_15_1	864	26113	15	30.22
rpg_3_500_14_3	965	10792	14	11.18
rpg_2_500_18_4	1027	18186	18	17.71

The number of iterations required for `Cluster-kmeans` is generally low, less than 10 on average. If the initial drawing is relatively cluster faithful, on average 2-3 iterations are sufficient. Otherwise, on average 5-10 iterations are executed.

6.2 *CQ-kmeans* improvement over FR and SM

We compute all variations of *CQ-kmeans* using five clustering comparison methods, and choose $CQ\text{-}kmeans_{ARI}$ (red), since it is the most effective metric [20]. To measure *CQ-kmeans* improvement by `Cluster-kmeans-FR` over FR, we define $I(CQ{-}kmeans) = \frac{CQ{-}kmeans(\texttt{Cluster-kmeans}{-}FR) - CQ{-}kmeans(FR)}{CQ{-}kmeans(FR)} \times 100$.

Similarly, to measure edge crossing improvement by `Cluster-kmeans-FR` over FR, we define $I(Q_{crossing}) = \frac{Q_{crossing}(FR) - Q_{crossing}(\texttt{Cluster-kmeans}{-}FR)}{Q_{crossing}(FR)} \times 100$.



(a) `Cluster-kmeans-FR` (b) `Cluster-kmeans-SM`

Figure 5: Average *CQ-kmeans* improvement (%) of (a) `Cluster-kmeans-FR` over FR and (b) `Cluster-kmeans-SM` over SM on graphs from [20], and graphs with even and uneven cluster sizes.

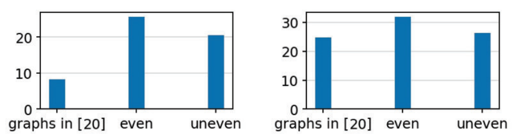
Figure 5 shows significant improvement on *CQ-kmeans* by `Cluster-kmeans-FR` (resp., `Cluster-kmeans-SM`) over FR (resp., SM), where values are averaged per data set, supporting Hypothesis 3.

Over all data sets, `Cluster-kmeans-FR` obtain on average 122% improvement over FR (Figure 5a): for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes), with an average improvement of 36% (resp., 139%, 191%).

Over all data sets, `Cluster-kmeans-SM` obtains on average 240% improvement over SM, see Figure 5b: for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes), with an average improvement of 90% (resp., 255%, 374%).

6.3 Edge Crossing Improvement over FR and SM

Surprisingly, `Cluster-kmeans-FR` (resp., `Cluster-kmeans-SM`) also achieve significant improvement on edge crossings over FR (resp., SM), see Figure 6, where values are averaged per data set.

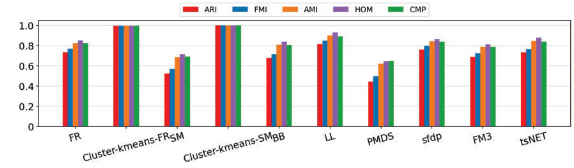


(a) `Cluster-kmeans-FR` (b) `Cluster-kmeans-SM`

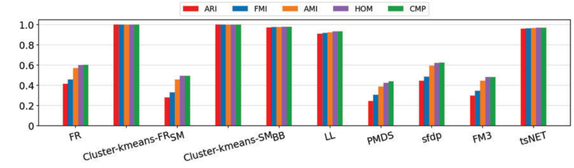
Figure 6: Average edge crossing improvement (%) of (a) `Cluster-kmeans-FR` over FR and (b) `Cluster-kmeans-SM` over SM on graphs used in [20], and graphs with even and uneven cluster sizes.

Over all data sets, on average `Cluster-kmeans-FR` obtains 18% improvements on crossings over FR, see Figure 6a. `Cluster-kmeans-FR` achieves significant improvement over FR on crossings with 26% (resp., 21%) on graphs with even (resp., uneven) cluster sizes, and obtains a considerable improvement over FR on crossings for graphs used in [20] with an average of 8%.

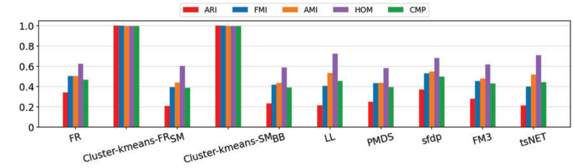
Over all data sets, `Cluster-kmeans-SM` obtains on average 28% improvement on crossing over SM, see Figure 6b. `Cluster-kmeans-SM` obtains significant improvement over SM on crossings for graphs used in [20], achieving an average of 25%,



(a) graphs in [20]



(b) even



(c) uneven

Figure 7: *CQ-kmeans* averaged for (a) graphs used in [20], and graphs with (b) even and (c) uneven cluster sizes.

as well as for graphs with even (resp., uneven) cluster sizes with an average of 32% (resp., 26%).

6.4 *CQ-kmeans* Comparison with Other layouts

Figure 7 shows a comparison on *CQ-kmeans*, averaged per data set in Table 4, with other popular graph layouts, including Backbone (BB) [26], LinLog (LL) [25], PivotMDS (PMDS) [1], tsNET [16], and multi-level graph layouts such as sfdp [13] and FM^3 [9].

For all data sets, we see that in practice `Cluster-kmeans-FR` and `Cluster-kmeans-SM` always obtain perfect $CQ\text{-}kmeans = 1$, supporting Hypothesis 4 and outperforming all other graph layouts.

Figure 7a clearly shows that `Cluster-kmeans-FR` and `Cluster-kmeans-SM` outperform the previously known high performing layouts on *CQ-kmeans* such as tsNET, LL, and BB [20].

Figure 7b shows *CQ-kmeans* for graphs with *even* cluster sizes in Table 4b. Clearly, `Cluster-kmeans-FR` and `Cluster-kmeans-SM` obtain $CQ\text{-}kmeans = 1$. tsNET, LL, and BB also perform well, achieving *CQ-kmeans* very close to 1. The next high performing layouts, sfdp, FM^3, and FR obtain *CQ-kmeans* around 0.7, and PMDS only attain scores below 0.5. This comparison result is consistent with the results in [20].

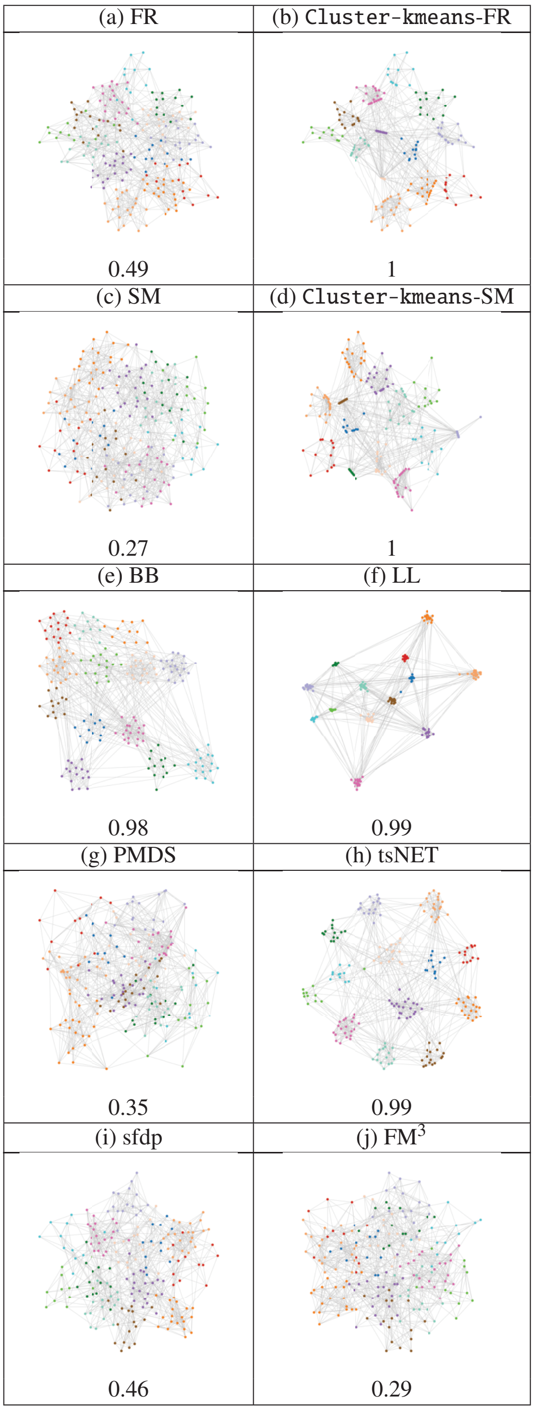
Figure 7c shows *CQ-kmeans* for graphs with *uneven* cluster sizes in Table 4c. As always, `Cluster-kmeans-FR` and `Cluster-kmeans-SM` obtain $CQ\text{-}kmeans = 1$. However, all the other layouts obtain almost the same *CQ-kmeans* values lower than 0.7, which clearly shows the limitations of *CQ-kmeans* in accurately measuring high- and low-performing cluster-faithful layouts.

Specifically, these results demonstrate the limitations of *CQ-kmeans* on graphs with uneven cluster sizes: for the most effective variations $CQ\text{-}kmeans_{ARI}$ (red) and $CQ\text{-}kmeans_{FMI}$ (blue), layouts with high cluster faithfulness (LL, tsNET) obtain almost the same score as layouts with low cluster faithfulness (PMDS), at only about 0.1 difference.

6.5 Visual Comparison and Limitation

Table 5a-d show a visual comparison of FR, `Cluster-kmeans-FR`, SM and `Cluster-kmeans-SM` on graph *rpg_11_20_13_0* with even

Table 5: Visual comparison on *rpg_11_20_13_0* with even cluster sizes and $\mathit{CQ\text{-}kmeans}_{ARI}$ scores for each layout.



cluster sizes. Clearly, `Cluster-kmeans`-FR and `Cluster-kmeans`-SM produce perfectly cluster-faithful drawings, ensuring clear separation between clusters without any overlap, while FR and SM fail to separate clusters with overlaps, supporting Hypothesis 3.

Table 5e-j show a visual comparison with all the other layouts on a graph with even cluster sizes. `Cluster-kmeans`-FR and `Cluster-kmeans`-SM always produce *CQ-kmeans* = 1, and tsNET, LL, and BB also perform quite well.

Table 6: Visual comparison on *rpg_5_400_12_1* (uneven cluster sizes).

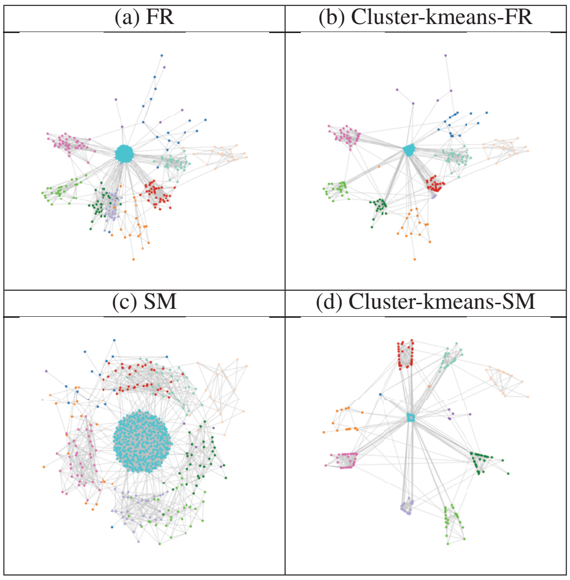


Table 6 shows a visual comparison of FR, `Cluster-kmeans`-FR, SM and `Cluster-kmeans`-SM on graph *rpg_5_400_12_1* with uneven cluster sizes. Although `Cluster-kmeans`-SM obtains *CQ-kmeans* = 1, it collapses the large light blue cluster into a misleadingly small area, while SM with lower *CQ-kmeans* actually more faithfully represents the size of the large light blue cluster.

Table 7 shows a visual comparison of all layouts on a graph with uneven cluster sizes. Note that tsNET has a much lower *CQ-kmeans* than PMDS, despite tsNET clearly showing the clustering structure, while PMDS fails to separate the clusters. Furthermore, FM^3 and PMDS only have a difference of 0.03 on *CQ-kmeans*, despite FM^3 clearly showing the cluster structure better than PMDS.

This limitation is due to the nature of *k*-means clustering, where it fails when dealing with data containing uneven cluster sizes. Specifically, when separating the clusters, *k*-means clustering tends to assign each cluster to regions of equal size. Therefore, `Cluster-kmeans` tends to assign equal size drawing area to all clusters regardless of size, struggling to accurately represent the ground truth structure of graphs with uneven clusters.

6.6 Discussion and Summary

In summary, extensive experiments support Hypotheses 3 and 4, demonstrating the effectiveness of `Cluster-kmeans` for computing perfectly cluster-faithful drawings of graphs with both even and uneven cluster sizes, always obtaining *CQ-kmeans* = 1. Moreover, `Cluster-kmeans`-FR (resp., `Cluster-kmeans`-SM) achieve significant improvement on *CQ-kmeans* of on average 122% over FR (resp., 240% over SM), performing better than existing high performing layouts such as tsNET, LL, and BB, across all data sets.

Surprisingly, `Cluster-kmeans`-FR (resp., `Cluster-kmeans`-SM) obtains better edge crossing metrics over FR (resp., SM). In general, vertices in the same cluster are more likely to have edges than the vertices from different clusters. Since our algorithms draw vertices in the same cluster close together, they could reduce the number of edge crossings by avoiding long intra-cluster edges.

However, `Cluster-kmeans` has limitations for drawing graphs with *uneven* clusters. While `Cluster-kmeans` remains effective in separating all clusters, it may fail to retain the structure of large clusters, due to *k*-means clustering assigning equal area sizes to the geometric clusters. This causes Cluster-kmeans-SM drawings to look very different to SM, as seen in Table 6d, where SM displays the size of the large cluster proportionally, however Cluster-kmeans-SM collapses the drawing area of the large cluster.

7 Cluster-HAC Experiments

7.1 Experiment Design and Data Sets

We evaluate the effectiveness of Cluster-HAC-FR and Cluster-HAC-SM in comparison to FR and SM, as well as other popular graph layouts, including BB, LL, PMDS, sfdp, FM^3 and tsNET. We use the same data sets as used in Section 4.1, see Table 4. For each layout, we compute *CQ-HAC* using HOM (purple) to test similarity between C and C', since it is the most effective for measuring cluster faithfulness using HAC. We also measure edge crossings to compare Cluster-HAC-FR and FR (resp., Cluster-HAC-SM and SM).

In general, the number of iterations taken by Cluster-HAC is very small: when the initial drawing is highly cluster-faithful, on average only 2-3 iterations are performed; for less cluster-faithful initial drawings, on average 4-6 iterations are executed.

We hypothesize the performance of Cluster-HAC algorithms as:

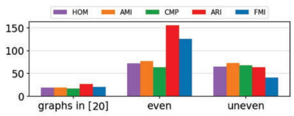
Hypothesis 5 *Cluster-HAC-FR and Cluster-HAC-SM compute significantly better cluster-faithful drawings than FR and SM.*

Hypothesis 6 *Cluster-HAC-FR and Cluster-HAC-SM always obtain CQ-HAC* $= 1$.

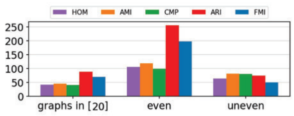
7.2 *CQ-HAC* Improvement over FR and SM

Figure 8 shows significant improvement on *CQ-HAC* by Cluster-HAC-FR (resp., Cluster-HAC-SM) over FR (resp., SM), averaged per data set, supporting Hypothesis 5.

Over all data sets, Cluster-HAC-FR obtains an average 52% improvement on *CQ-HAC* over FR, see Figure 8a: for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes), with an average improvement of 19% (resp., 72%, 65%). Cluster-HAC-SM obtains an average 71% improvement on *CQ-HAC* over SM, see Figure 8b: for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes), with an average improvement of 42% (resp., 106%, 65%).

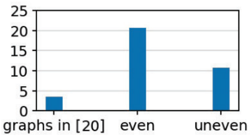


(a) Cluster-HAC-FR

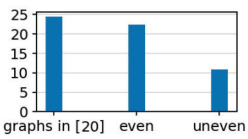


(b) Cluster-HAC-SM

Figure 8: Average *CQ-HAC* improvement (%) of (a) Cluster-HAC-FR over FR and (b) Cluster-HAC-SM over SM on graphs used in [20], and graphs with even and uneven cluster sizes.

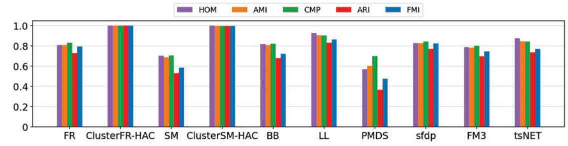


(a) Cluster-HAC-FR

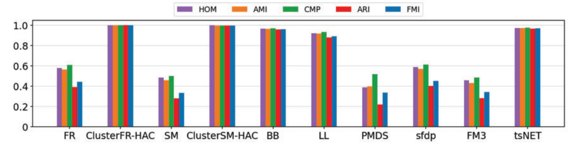


(b) Cluster-HAC-SM

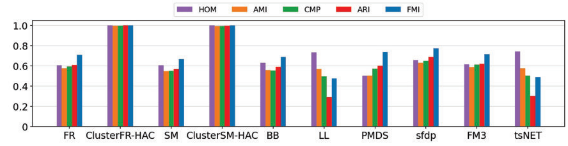
Figure 9: Average edge crossing improvement (%) of (a) Cluster-HAC-FR over FR and (b) Cluster-HAC-SM over SM on graphs used in [20], and graphs with even and uneven cluster sizes.



(a) graphs in [20]



(b) even



(c) uneven

Figure 10: *CQ-HAC* averaged for (a) graphs used in [20], and graphs with (b) even and (c) uneven cluster sizes.

7.3 Edge Crossing Improvement over FR and SM

Surprisingly, Cluster-HAC-FR (resp., Cluster-HAC-SM) also achieve significant improvement on edge crossings over FR (resp., SM), see Figure 9, where values are averaged per data set.

Over all data sets, Cluster-HAC-FR obtains on average 12% improvement on edge crossings over FR (Figure 9a): average improvement of 4% (resp., 21%, 11%) for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes).

Over all data sets, Cluster-HAC-SM obtains on average 19% improvement on edge crossings over SM (Figure 9b): average improvement of 24% (resp., 22%, 11%) for graphs used in [20] (resp., graphs with even cluster sizes, graphs with uneven cluster sizes).

7.4 *CQ-HAC* Comparison with Other Layouts

For all three data sets, we see that in practice Cluster-HAC-FR and Cluster-HAC-SM always achieve perfect *CQ-HAC* $= 1$, supporting Hypothesis 6, and outperform all the other graph layouts. Figure 10 shows a comparison of popular graph layouts on *CQ-HAC*, including Cluster-HAC, FR, SM, BB, LL, PMDS, sfdp, FM^3, and tsNET, averaged for each data set in Table 4.

Figure 10a shows the results on graphs from [20], where Cluster-HAC-FR and Cluster-HAC-SM always achieve *CQ-HAC*=1. LL and tsNET also perform well, achieving *CQ-HAC* scores close to 0.9, followed by sfdp, BB, FM^3 and FR with scores close to 0.8. PMDS obtains the lowest scores, below 0.5.

Figure 10b shows *CQ-HAC* for graphs with *even* cluster sizes listed in Table 4b. As always, Cluster-HAC-FR and Cluster-HAC-SM obtain *CQ-HAC* $= 1$. LL, tsNET and BB also perform quite well, achieving *CQ-HAC* scores close to 0.9. However, all the other layouts obtain *CQ-HAC* values lower than 0.6. PMDS obtains the lowest scores, below 0.4.

Figure 10c shows *CQ-HAC* for graphs with *uneven* cluster sizes in Table 4c, where Cluster-HAC-FR and Cluster-HAC-SM obtain *CQ-HAC* $= 1$. LL and tsNET perform well with *CQ-HAC* scores close to 0.8. All the other layouts obtain scores between 0.5 and 0.7, where PMDS obtains the lowest scores below 0.5.

The results from Figure 10 clearly show that *CQ-HAC* metrics can accurately measure high- and low-performing cluster-faithful layouts for graphs with even or uneven cluster sizes, outperforming

CQ-kmeans metrics as seen in Figure 7, where both high- and low-performing layouts can have similar *CQ-kmeans* scores.

7.5 Visual Comparison

Table 7a-f show a visual comparisons of FR, `Cluster-HAC`-FR, SM and `Cluster-HAC`-SM on graph *rpg_5_400_12_1* with *uneven* cluster sizes. Both `Cluster-HAC`-FR and `Cluster-HAC`-SM produce perfectly cluster-faithful drawings, ensuring clear separation between clusters without any overlap, while FR and SM produce overlaps between the small clusters, supporting Hypothesis 5.

Moreover, `Cluster-HAC`-SM successfully displays the structure of the central big cluster while effectively separating all small clusters without any overlap, achieving significant improvement over `Cluster-kmeans`-SM in Table 6d.

Table 7g-l show a visual comparison on the graph *rpg_5_400_12_1* with other layouts, showing consistent results with Figure 10c. Clearly, `Cluster-HAC`-SM achieves the best result, displaying the structures of both the big and small clusters, compared to LL and tsNET which obtain high *CQ-HAC* but excessively compress the surrounding small clusters into minimal points.

FR, FM^3, and sfdp successfully separate clusters with minimal overlap. However, they compress the big cluster into a small compact area, failing to show the structure of the big cluster. Conversely, SM and BB effectively capture the structure of the large cluster, however with many overlaps between the smaller clusters.

CQ-HAC measures cluster faithfulness more accurately than *CQ-kmeans* for graphs with uneven cluster sizes. Figure 10c shows that there is a clear difference of about 0.25 in $CQ\text{-}HAC_{HOM}$ (purple), between high-performing layouts (LL and tsNET) and low-performing layouts (PMDS). Meanwhile, Figure 7c shows that, even with the best performing variants $CQ\text{-}kmeans_{ARI}$ (red) and $CQ\text{-}kmeans_{FMI}$ (blue), PMDS obtains very similar scores to LL and tsNET.

The visual comparison in Table 7 also shows consistent results. *CQ-HAC* of tsNET is now 0.32 higher than PMDS, consistent with the better quality drawing with well-separated cluster structures, as opposed to *CQ-kmeans* with a higher score for PMDS instead.

7.6 Discussion and Summary

In summary, our experiments support Hypotheses 5 and 6, and demonstrate the effectiveness of `Cluster-HAC` for computing cluster-faithful drawings, always achieving $CQ\text{-}HAC = 1$ in practice.

`Cluster-HAC`-FR (resp., `Cluster-HAC`-SM) always compute perfectly cluster-faithful drawings, achieving significant improvement on *CQ-HAC* of 52% over FR (resp., 71% over SM), and performing better than existing cluster-focused layouts tsNET, LL, and BB across all data sets. The improvement in edge crossing also shows that `Cluster-HAC` obtains perfect cluster faithfulness while also maintaining good quality on a readability metric.

Moreover, `Cluster-HAC` can faithfully display the structure of big clusters as well as small clusters without any overlaps for graphs with uneven cluster sizes, in contrast to `Cluster-kmeans` which often fails to preserve the intra-cluster structure on graphs with uneven cluster sizes. For example, in Table 7f, we see that `Cluster-HAC`-SM maintains a large drawing area proportional to a large cluster, similar to SM, while `Cluster-kmeans`-SM in Table 7e gives a small drawing area disproportionately for the large cluster. `Cluster-HAC` obtaining perfect *CQ-HAC* and better displaying cluster structures on both graphs with even and uneven cluster sizes demonstrate its applicability to a wide range of clustering structures in graphs.

Furthermore, *CQ-HAC* measures cluster faithfulness more accurately than *CQ-kmeans* for graphs with uneven cluster sizes.

8 Conclusion and Future Work

In this paper, we present *CQ-HAC*, a new cluster faithfulness metric that utilizes HAC geometric clustering, to overcome the limitations of existing cluster faithfulness metrics using *k*-means clustering on graphs with uneven cluster sizes.

We also present `Cluster-kmeans` and `Cluster-HAC` algorithms for optimizing cluster faithfulness in graph drawings. Experiments show that both algorithms can compute perfectly cluster-faithful drawings (i.e., $CQ = 1$) for graphs with even and uneven cluster sizes, achieving significant improvement over existing layouts.

Moreover, `Cluster-HAC` perform better than `Cluster-kmeans` for graphs with uneven cluster sizes, successfully preserving both the inter-cluster and intra-cluster structures across all data sets.

Future work includes more comprehensive evaluation of `Cluster-kmeans` and `Cluster-HAC`, using empirical runtime comparison and other metrics such as node displacement, and new algorithms to optimize other faithfulness metrics.

Acknowledgments

This work was supported by ARC (Australian Research Council) grant DP190103301.

References

[1] U. Brandes and C. Pich. Eigensolver methods for progressive multidimensional scaling of large data. In *Graph Drawing*, vol. 4372, pp. 42–53, 2006. 2, 6

[2] S. Cai, A. Meidiana, and S. Hong. DNC: dynamic neighborhood change faithfulness metrics. In *EuroVis*, pp. 49–53, 2022. 2

[3] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991. 2

[4] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999. 1, 2

[5] P. Eades, S. Hong, K. Klein, and A. Nguyen. Shape-based quality metrics for large graph visualization. In *Graph Drawing*, pp. 502–514, 2015. doi: 10.1007/978-3-319-27261-0-41 1, 2

[6] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *JASA*, 78(383):553–569, 1983. 2

[7] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *SPE*, 21(11):1129–1164, 1991. 2, 5

[8] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *Graph Drawing*, pp. 239–250, 2004. 2, 5

[9] S. Hachul and M. Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In *Graph Drawing*, pp. 285–295, 2004. 6

[10] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, LANL, 2008. 4, 5

[11] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer, 2009. 1, 2

[12] S. Hong, A. Meidiana, J. Wood, J. P. Ataides, P. Eades, and K. Park. dgg, drng, DSC: new degree-based shape-based faithfulness metrics for large and complex graph visualization. In *PacificVis*, pp. 51–60, 2022. 2

[13] Y. Hu. Efficient, high-quality force-directed graph drawing. *Mathematica*, 10(1):37–71, 2005. 6

[14] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985. doi: 10.1007/BF01908075 2

[15] S. G. Kobourov, S. Pupyrev, and B. Saket. Are crossings important for drawing large graphs? In *GD*, vol. 8871 of *LNCS*, pp. 234–245. Springer, 2014. 1

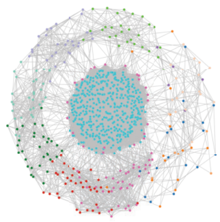
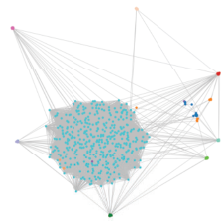
[16] J. F. Kruiger, P. E. Rauber, R. M. Martins, A. Kerren, S. Kobourov, and A. C. Telea. Graph layouts by t-sne. *CGF*, 36(3):283–294, 2017. 2, 6

[17] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. 4

[18] J. MacQueen and et al. Some methods for classification and analysis of multivariate observations. In *Mathematical Statistics and Probability*, vol. 1, pp. 281–297. University of California Press, 1967. 1

[19] A. Meidiana, S. Hong, and P. Eades. Shape-faithful graph drawings. In *Graph Drawing*, vol. 13764, pp. 93–108, 2022. 2

Table 7: Visual comparison on graph *rpg_5_400_12_1* with uneven cluster sizes using $CQ\text{-}kmeans_{ARI}$, $CQ\text{-}HAC_{HOM}$.

(a) FR	(b) `Cluster-kmeans-FR`	(c) `Cluster-HAC-FR`
 *CQ-kmeans*: 0.55 *CQ-HAC*: 0.88	 *CQ-kmeans*: 1 *CQ-HAC*: -	 *CQ-kmeans*: - *CQ-HAC*: 1
(d) SM	(e) `Cluster-kmeans-SM`	(f) `Cluster-HAC-SM`
 *CQ-kmeans*: 0.28 *CQ-HAC*: 0.81	 *CQ-kmeans*: 1 *CQ-HAC*: -	 *CQ-kmeans*: - *CQ-HAC*: 1
(g) BB	(h) LL	(i) PMDS
 *CQ-kmeans*: 0.27 *CQ-HAC*: 0.76	 *CQ-kmeans*: 0.26 *CQ-HAC*: 0.89	 *CQ-kmeans*: 0.35 *CQ-HAC*: 0.63
(j) sfdp	(k) FM^3	(l) tsNET
 *CQ-kmeans*: 0.55 *CQ-HAC*: 0.83	 *CQ-kmeans*: 0.38 *CQ-HAC*: 0.86	 *CQ-kmeans*: 0.26 *CQ-HAC*: 0.95

[20] A. Meidiana, S. Hong, P. Eades, and D. Keim. A quality metric for visualization of clusters in graphs. In *Graph Drawing*, pp. 125–138, 2019. doi: 10.1007/978-3-030-35802-0-10 1, 2, 4, 5, 6, 8

[21] A. Meidiana, S.-H. Hong, P. Eades, and D. Keim. Quality metrics for symmetric graph drawings. In *PacificVis*, pp. 11–15, 2020. 1, 2

[22] F. Murtagh. Complexities of hierarchic clustering algorithms: state of the art. *Computational Statistics Quarterly*, 1(2):101–113, 1984. 4

[23] Q. H. Nguyen, P. Eades, and S.-H. Hong. On the faithfulness of graph visualizations. In *PacificVis*, pp. 209–216, 2013. 1, 2

[24] Q. H. Nguyen, S. Hong, and P. Eades. dnng: Quality metrics and layout for neighbourhood faithfulness. In *PacificVis*, pp. 290–294, 2017. 2

[25] A. Noack. An energy model for visual graph clustering. In *Graph Drawing*, vol. 2912 of *LNCS*, pp. 425–436. Springer, 2003. 1, 2, 6

[26] A. Nocaj, M. Ortmann, and U. Brandes. Untangling the hairballs of multi-centered, small-world online social media networks. *Journal of Graph Algorithms and Applications*, 19(2):595–618, 2015. 1, 2, 4, 6

[27] W. M. Rand. Objective criteria for the evaluation of clustering methods. *JASA*, 66(336):846–850, 1971. 2

[28] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, pp. 410–420, 2007. 2

[29] G. T. Toussaint. *Computational morphology: a computational geometric approach to the analysis of form*. Elsevier, 2014. 2, 3

[30] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *JMLR*, 11(Oct):2837–2854, 2010. 2