Karlsruhe Reports in Informatics 2011,27
Edited by Karlsruhe Institute of Technology,
Faculty of Informatics
ISSN 2190-4782

Generalizing Geometric Graphs

Edith Brunel, Andreas Gemsa, Marcus Krug,
Ignaz Rutter, Dorothea Wagner

2011

KIT – University of the State of Baden-Wuerttemberg and National
Research Center of the Helmholtz Association

Generalizing Geometric Graphs

Edith Brunel[1], Andreas Gemsa[1], Marcus Krug[1], Ignaz Rutter[1], Dorothea Wagner[1]

Faculty of Informatics, Karlsruhe Institute of Technology (KIT), Germany
firstname.lastname@kit.edu

Abstract. Network visualization is essential for understanding the data obtained from huge real-world networks such as flight-networks, the AS-network or social networks. Although we can compute layouts for these networks reasonably fast, even the most recent display media are not capable of displaying these layouts in an adequate way. Moreover, the human viewer may be overwhelmed by the displayed level of detail. The increasing amount of data therefore requires techniques aiming at a sensible reduction of the visual complexity of huge layouts.
We consider the problem of computing a generalization of a given layout reducing the complexity of the drawing to an amount that can be displayed without clutter and handled by a human viewer. We take a first step at formulating graph generalization within a mathematical model and we consider the resulting problems from an algorithmic point of view. Although these problems are NP-hard in general, we provide efficient approximation algorithms as well as efficient and effective heuristics. At the end of the paper we showcase some sample generalizations. This technical report is an extended version of a paper by the same authors that is to appear in [10].

1 Introduction

As a natural consequence of the increasing amount of available data we are frequently facing large and even huge networks such as road and flight networks, the Internet and social networks with millions of vertices. Visualization of these networks is a key to assessing the inherent graph-based information via human inspection. There are several methods for computing layouts of huge graphs with millions of vertices within a few minutes [28, 31, 25].

But, how do we display such layouts? Modern HD displays feature roughly 2 Mio pixels and a standard A4 page allows roughly 8.7 Mio dots at a resolution of 300 pixels per inch. Although these numbers do sound adequate for large-scale graph visualization at first glance, both media are not at all suited for displaying huge graphs with millions of vertices. Even if we require only a minimal distance of 10 pixels or dots between the vertices of the graph, which yields a distance between vertices of roughly 3 millimeters on the screen and less than 1 millimeter on paper, then we can display only several thousand vertices, and not too many edges. If we additionally seek to display graph structure and keep visual clutter low, the number of vertices we can display degrades even further and may go down as far as less than a hundred for dense graphs.

Even worse, the human perception is not capable of extracting detailed information from huge layouts with millions of vertices. Since, by a simple counting argument, there are incompressible adjacency matrices [32], a graph with only 1 Mio vertices may encode incompressible information of up to 125 Gigabytes. This exceeds by a factor of 3.6 the average daily information consumption of an average American estimated at 34 (highly compressible) Gigabytes of information in the current report on American Consumers [7].

Related Work Known approaches to coping with the huge amount of data by allowing for some kind of abstraction can be categorized into *structural* and *geometric* methods. While structural methods create a new layout for the data, typically using a clustering of the graph, geometric methods are applied to a given layout maintaining the user's mental map [35].

Graph-theoretic clustering methods, which can be used to cluster the graph for visualization are discussed in [21]. Eades and Feng [17] describe a multilevel visualization method for clustered graphs with the aim of visualizing network-based data that has been clustered hierarchically at different levels of abstraction induced by the hierarchy of the clustering. A force-directed layout algorithm based on a hierarchical decomposition of the graph is given by Quigley and Eades [36]. This method allows for visualizing the graph at different levels of abstraction by computing a layout based on a hierarchical grouping of the vertices of the graph. Harel and Koren [27] present a multi-scale algorithm with the purpose of producing nice drawings on large and small scale, respectively. Different levels of abstraction of the graph are obtained by iteratively coarsening the graph. Multi-scale drawing methods are combined with fisheye views by Gansner et al. [22]. Their approach is to compute a layout of a graph whose level of detail deteriorates with increasing distance to the focal node of the layout, that is, they provide a topological version of classical fisheye visualization techniques. Abello el al. [5] discuss graph sketches for very large graphs based on mapping clusters of the graph to certain regions of the screen. Their notion of a sketch is based on a hierchcical clustering of the graph and is mainly focused at exploring the graph via a detail-on-demand strategy without providing a good approximation of the graph's structure and geometry. Rafiei and Curial [37] study the generalization of graphs by sampling.

Classical fisheye visualizations [20, 39], on the other hand, can be directly applied to a given layout and apply a distortion to a given layout to emphasize the structure of the drawing in a certain area of interest. The resolution of the drawing deteriorates towards the boundary of the drawing and parts of the drawing in this area are usually densely cluttered. Abello et al. [4] study the visualization of large graphs with compound-fisheye views and treemaps, employing hierarchical clustering and a treemap representation of this clustering. Edge Bundling techniques [40, 29] aim at reducing the complexity of layouts by bundling similar edges.

Finally, *Generalization* has received considerable attention in cartography [34]. Mackaness and Bear [33] highlight the potential of graph theory for map generalization. Saalfeld states the map generalization problem as a straight-line graph drawing problem [38] and formulates a number of challenges resulting from this perspective. Among others, he asks for a rigorous mathematical model for graph-based generalizations and provable guarantees. We are not aware of any work aiming at assessing this problem to its full extent.

Our Contribution We take a first step towards establishing a mathematical model for the problem of generalizing geometric graphs. A key to assessing the problem of computing a suitable generalization is to find an adequate measure of the quality or appropriateness of a generalization. It is essential to understand the geometric and combinatorial features resulting in the visual complexity of geometric graphs and how they affect human perception. The *geometric features* of the drawing include, among others, the distributions of points and edges as well as the distribution of crossings, the shapes of the faces of the arrangement, especially the outer face, as well as symmetries and peculiarities. The *combinatorial features* include connectivity, structure and length of shortest paths as well as, for instance, planarity. Although we are far from fully understanding the impact of the these features on the human perception we try to incorporate a carefully selected set of these features into our model of a generalization. The generalization should maintain the spirit of the drawing of the graph and

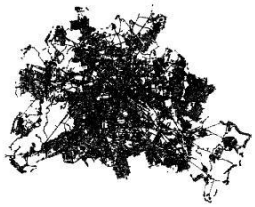


Fig. 1: Vertex-Clutter

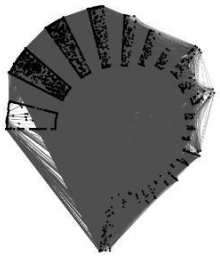


Fig. 2: Edge-Clutter

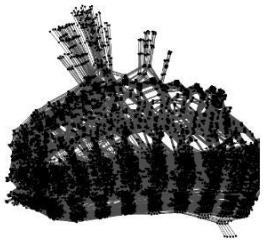


Fig. 3: Vertex-Edge-Clutter

preserve the prominent features while reducing the amount of detailed information to an amount that can be displayed without clutter and handled by a human viewer. Our model is based on the fact that vertices have a fixed size and edges have a fixed width on the screen. *Visual clutter* refers to an agglomeration of overlapping visual features in a limited area that renders these features indistinguishable. Our goal is to either avoid or reduce visual clutter. We identify three types of clutter.

Vertex-Clutter occurs when two or more vertices are too close to each other. It may render the drawing unusable due to hidden edge information; see Fig. 1.

Edge-Clutter occurs when too many edges cross a limited area. Even if vertices are far enough apart, edge clutter may lead to indistinguishable edge information; see Fig. 2.

Vertex-Edge-Clutter occurs when a vertex is too close to an edge. In this case, we are unable to tell, whether the vertex is incident to the edge or not; see Fig. 3.

We devise a framework that allows for assessing all types of clutter in an incremental way by modeling the elimination or reduction of each type of clutter as an optimization problem, which we analyze in terms of complexity. We show that these problems are NP-hard in general and we provide approximation algorithms as well as effective and efficient heuristics that can be applied to huge graphs within reasonable time.

Preliminaries A *geometric graph* is a pair $G = (P, E)$ such that $P \subseteq \mathbb{R}^2$ is a finite set of n points in the plane and E is a set of m straight-line segments with endpoints in P. If not otherwise stated, graph refers to a geometric graph throughout this paper. For $p \in P$ and a non-negative number $r \in \mathbb{R}_0^+$, we denote by $B(p, r)$ the disk with center p and radius r. We model the finite resolution of a screen by assuming that each point p occupies the locus of points whose distance to p is bounded by $s \in \mathbb{R}_0^+$ and, similarly, each edge e occupies the locus of points whose distance to e is bounded by $w \in \mathbb{R}_0^+$.

A *generalization* of G is a pair (H, φ) where $H = (Q, F)$ is a geometric graph with $Q \subseteq P$ such that $\varphi\colon P \to Q$ maps vertices of G to vertices of H and F is a subset of edges resulting from a contraction of G according to φ. Since the subgraph induced by $\varphi^{-1}(q)$ is contracted into a single vertex, we call this subgraph the *cluster* of q, denoted by C_q. Given $Q \subseteq P$, we denote by $\nu\colon P \to Q$ the *Voronoi mapping*; which maps $p \in P$ to its closest neighbor in Q with respect to the Euclidean metric. We call the corresponding clusters *Voronoi clusters*. We especially focus on this mapping since it minimizes $\sum_{p \in P} d(p, \varphi(p))$ and, hence, seems to be a natural mapping. Throughout the paper distance refers to the Euclidean metric.

Organization of the Paper In Section 2, we consider the problem of eliminating vertex-clutter. We discuss our model for the generalization of the vertex set and show NP-hardness of the corresponding optimization problem. We further show that the size of the generalized pointset can be approximated efficiently and we devise an efficient heuristic for further optimization. In Section 3, we study the reduction of edge-clutter. We show that it is in general NP-hard to find a sparse or short subset of the edges maintaining monotone tendencies. When the original graph is complete, however, or if we are not restricted to use edges of the original graph, we can efficiently compute a sparse graph approximately representing monotone tendencies of the edges. In Section 4, we model the problem of reducing vertex-edge clutter and we show how to compute a drawing that allows for unambiguously deciding whether an edge is incident to a vertex or not, thus effectively eliminating vertex-edge clutter. We showcase some sample generalizations and conclude with a short discussion as well as open problems in Section 6.

2 Generalizing the Vertex Set without Vertex-Clutter

In this section we consider the problem of computing a generalization (H, φ) without vertex clutter for a geometric graph $G = (V, E)$, where $H = (Q, F)$. We focus on the case that φ is the Voronoi-mapping assigning each vertex in P to its nearest neighbor in Q. In order to avoid vertex-clutter, we require a minimal distance $r \in \mathbb{R}_0^+$ between the vertices of a generalized geometric graph. Let $\varrho\colon P \to \mathbb{R}_0^+$ be a function that maps a positive real number $\varrho(p) \geq r$ to every point $p \in P$. For each vertex $p \in Q$ in the generalized graph we require that the disk $B(p, \varrho(p))$ does not contain any other point from Q. We call a pointset Q with this property a *ϱ-set of P*. This prerequisite, however, must be balanced with additional quality measures such as the size of the ϱ-set, the clustering induced by φ and the distribution of the points in Q in order to avoid trivial solutions such as a single vertex. Clearly, it is desirable to maximize the size of a ϱ-set in order to retain as many vertices of the original graph as possible. That is, even in the presence of other optimization goals we may assume that the vertex set Q of the generalization constitutes an inclusion-maximal ϱ-set of the original point set P.

Choosing $\varrho \equiv r$ uniformly for all points $p \in P$ may have a severe effect on the distribution of the points when maximizing the size of a ϱ-set since the distances to the nearest neighbors in an inclusion-maximal ϱ-set tend to be uniformly distributed regardless of the original distribution. However, it may be more appropriate to approximate the distribution of the original pointset. In order to approximate this distribution by an inclusion-maximal ϱ-set we can choose ϱ as follows. Let p_0 be the point that maximizes the number of points in $B(p, r) \cap P$ over all $p \in P$ and let $k = |B(p_0, r) \cap P| - 1$ denote the number of points in this disk that are different from p_0. For each $p \in P$ let $d_k(p) \geq r$ denote p's distance to its k-nearest neighbor in P. By choosing $\varrho(p) = d_k(p) \geq r$ any inclusion-maximal pointset will have approximately the same distribution as the original pointset since for each point in the generalized pointset we discarded the same amount of points from the original graph.

Since, in general, it is not clear which behavior is more appropriate, we introduce a parameter $\alpha \in [0, 1]$ and let the user decide by setting $\varrho(p) := \max\{r, \alpha d_k(p)\}$. That is, the user can choose between retaining as many points in areas with low clutter as possible ($\alpha = 0$) and approximating the distribution of the pointset ($\alpha = 1$) as well as interpolations between the two extremes.

We consider two measures to assess the quality of a ϱ-set Q. While the size of Q is a measure of the amount of data that is retained, the quality of the clustering induced by φ is a measure for the amount of data that is lost due to the contraction of the vertices. There are several established ways

of assessing the quality of clusterings, such as coverage, performance, conductance [21], and modularity [9]. Since the information contained in the inter-cluster edges is retained in the generalization, we concentrate on assessing the quality of the clusters based on the intra-cluster edges. We consider a measure similar to coverage, which we adapt to our purpose as follows. For each cluster C_q let n_q denote the number of vertices and m_q denote the number of edges in C_q, respectively. We define the *local coverage* of a cluster C_q by $\operatorname{lcov}(C_q) = 2m_q/(n_q(n_q-1))$, i.e., as the amount of intra-cluster coherence that is explained by the intra-cluster edges. The local coverage of the generalization is defined as $\operatorname{lcov}(H,\varphi) = \min_{q\in Q} \operatorname{lcov}(\varphi^{-1}(q))$.

We consider the following multi-objective optimization problem. Given a geometric graph $G = (P,E)$, a non-negative radius $r \in \mathbb{R}_0^+$ and $\alpha \in [0,1]$ the Local Coverage Cluster Packing (LCCP) problem is to compute a ϱ-set $Q \subseteq P$ and a mapping $\varphi\colon P \to Q$ that maximizes both $|Q|$ and $\operatorname{lcov}(H,\varphi)$.

Problem Local Coverage Cluster Packing (LCCP)

Instance: Geometric graph $G = (P,E)$, $r \in \mathbb{R}_0^+$, $\alpha \in [0,1]$
Solution: ϱ-set $Q \subseteq P$, mapping $\varphi\colon P \to Q$
Goal: maximize $\operatorname{lcov}(H,\varphi)$, maximize $|Q|$

First we show that several single-criteria optimization variants of this multi-criteria optimization problem are NP-hard. Then we show how to approximate the size of a ϱ-set efficiently and we devise an efficient heuristic for balancing the size of a ϱ-set with the quality of the induced local coverage.

2.1 Complexity

The problem of computing a ϱ-set of maximum size for $\alpha = 0$ can be reduced to the problem of computing a maximum independent set in the intersection graph of the disks with radius $r/2$ centered at the points in P. Clark et al. [13] prove that this problem is NP-hard in unit-disk graphs, even if the disk representation of the graph is given.

Corollary 1. *Maximizing the size of a ϱ-set is NP-hard for $\alpha = 0$.*

Next, we show that it is also NP-hard to maximize the local coverage in the induced clusters of a ϱ-set as well as the total size of the generalization obtained by choosing a ϱ-set if the clustering is obtained by the Voronoi mapping induced by the points in Q.

Theorem 1. *Maximizing $\operatorname{lcov}(H,\nu)$ of a generalization (H,ν) is NP-hard for $\alpha = 0$.*

Proof. The proof is by reduction from the NP-hard problem Planar Monotone 3-sat [15]. Let $\mathcal{U} = \{x_1,\dots,x_n\}$ be a set of Boolean variables and let $\mathcal{C} = C_1 \wedge C_2 \cdots C_m$ be 3-sat formula. Then $\mathcal{C}$ is called *monotone* if all clauses consist only of positive or only of negative literals. Let $\mathcal{G} = (\mathcal{U} \cup \mathcal{C}, \mathcal{E})$ be the bipartite graph, on the clauses and variables, where $\mathcal{E}$ contains the edge (x_i, C_j) if and only if the literal x_i or its negation is contained in C_j. A *monotone rectilinear representation* of a monotone 3-sat formula is a rectilinear drawing of $\mathcal{G}$ such that the following conditions are met, as illustrated in Figure 4.

(i) The variables and clauses are drawn as axis-aligned boxes such that all variable boxes are on a horizontal line.
(ii) The edges are drawn as vertical line segments connecting the corresponding boxes.
(iii) The drawing does not contain any crossings.

An instance of Planar Monotone 3-sat consists of a *monotone rectilinear representation* of a planar monotone 3-sat instance and we wish to decide, whether the corresponding 3-sat instance is solvable.

A ϱ-set with local coverage 1 is called a *perfect* ϱ-set. A ϱ-set is perfect if and only if the graphs induced by the vertices in each of the Voronoi faces defined by Q are cliques. Given a monotone rectilinear representation of a planar monotone 3-sat formula we will construct a corresponding instance $I = (G = (P, E), \varrho)$ of problem LCCP such that I contains a perfect ϱ-set if and only if the 3-sat formula is satisfiable. For reasons of simplicity our construction is based on a disconnected graph with collinear points, but the construction can be modified in a straightforward way to obtain similar results for connected graphs with vertices in general position. We choose $\varrho \equiv 1.25$ and we construct G from a set of *variable/literal gadgets*, *transmitter/bend gadgets* and *clause gadgets*, which we will describe subsequently. We will use the following trivial observation.

Observation 1 *Every perfect ϱ-set of G contains at least one vertex of each clique of G.*

Variable/Literal Gadget. We distinguish between *basic* and *extended* variable and literal gadgets, respectively. The basic variable gadgets incorporate the functionality needed to correctly represent the variables. These gadgets can be extended to transmit their state along the transmitters. Each basic variable gadget consists of three vertically aligned cliques of size three and four, respectively, as illustrated in Figure 5. Each clique consists of vertically aligned points at distance 1, and the cliques are separated by a vertical gap of 0.5.

Due to Observation 1 a perfect ϱ-set of G must contain at least one vertex in each of the cliques. Note that we cannot choose two vertically subsequent vertices in any of the cliques, since they do not constitute a ϱ-set. Due to the chosen vertical distribution of the vertices, the vertices of any ϱ-set closest to the gaps must be chosen symmetrically to the bisector of the two cliques. Otherwise, the bisector of these vertices will intersect the edges of one of the cliques as illustrated in Figure 5d. Furthermore, we cannot choose the two vertices closest to the gap since they do not constitute a valid ϱ-set. Hence, there are only two valid ϱ-sets of the basic variable gadget, corresponding to the true and false state of the corresponding variable, as illustrated in Figure 5b and 5c, respectively. The variable gadgets can be extended in order to connect them to the transmitter gadgets. In order to extend the variable gadgets, we substitute the cliques of size 3 at the corresponding end by a clique of size 4.

The literal gadgets are composed of basic and extended variable gadgets that are horizontally aligned. The horizontal gap between the gadgets is variable and can be chosen to be 1 or 1.5, as illustrated in Figure 6.

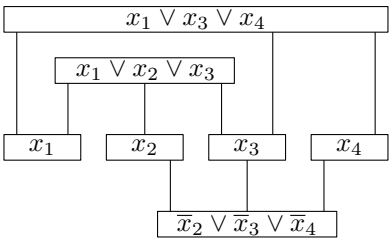
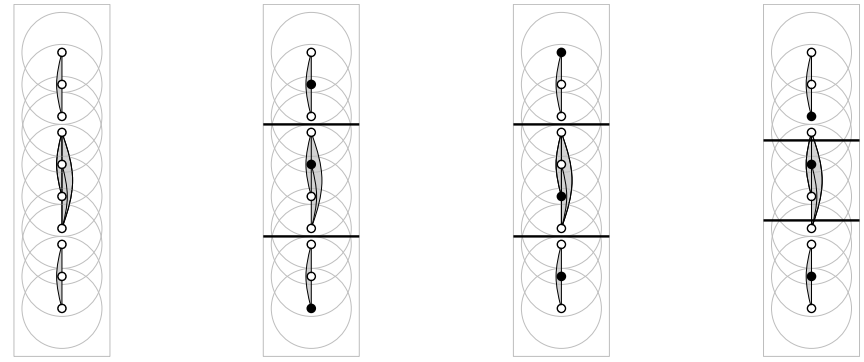
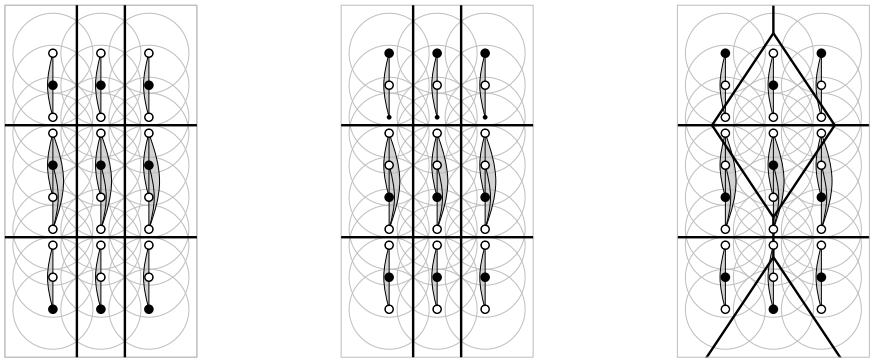


Fig. 4: Monotone rectilinear representation of the 3-sat formula $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4)$.



(a) Graph (b) True (c) False (d) Illegal

Fig. 5: Variable gadget

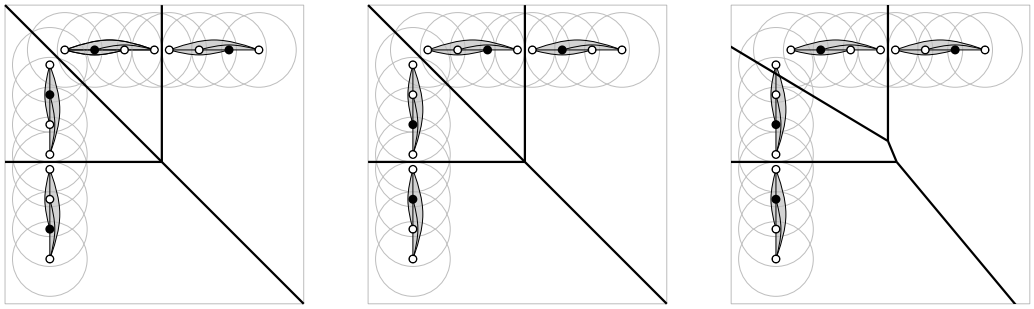


(a) True (b) False (c) Illegal

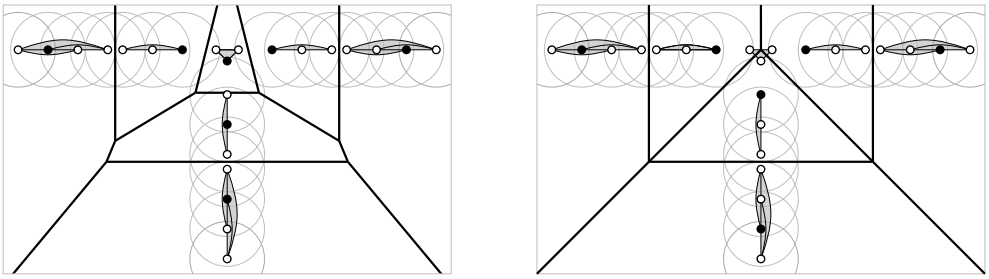
Fig. 6: Literal gadget

Transmitter/Bend Gadget. The transmitter gadgets consist of two vertically aligned cliques of size 4 that are separated by a gap of 0.5 similar to the variable gadgets. When stacked upon the extended variable gadgets with a vertical gap of 0.5 the transmitter gadgets can be in one of two valid states corresponding to the assignment of the variable. The bend gadgets consist of one vertical and one horizontal transmitter gadget as illustrated in Figure 7. Figure 7c illustrates that the state cannot change at the transition between the horizontal and the vertical transmitter segment. Such a change would result in locate coverage strictly smaller than 1.

Clause Gadget. Finally, the clause gadget is constructed as illustrated in Figure 7. It consists of three small gadgets that are arranged in a T-shaped fashion and which are constructed exactly as the topmost two cliques of the basic variable gadget. The functionality of the clause is realized by a small triangle arranged in the middle of the T-shaped figure. If all literals corresponding to the clause are false, then each of the triangle's vertices is contained in the ϱ-ball of one of the vertices contained in the corresponding ϱ-set. Hence, none of the vertices of the triangle may be contained



(a) True (b) False (c) Illegal State



(d) satisfied (e) dissatisfied

Fig. 7: (7a)–(7c) Bend gadget, (7d)–(7e) Clause gadget.

in the ϱ-set and, thus, the vertices of the triangle are mapped to a neighboring point resulting in local coverage strictly less than 1. This is illustrated in Figure 7e. If, on the other hand, at least one of the variables is in a true state, then one of the triangle's vertices can be included in the ϱ-set. The vertices of the triangle are chosen in such a way that that the bisector between any of these vertices and the closest vertex corresponding to a true assignment does not intersect any of the cliques of the clause. As illustrated in Figure 7d, this leads to a Voronoi diagram that does not intersect the edges of G, resulting in a perfect ϱ-set.

Clearly, a satisfying assignment of the 3-SAT formula can be transformed into a perfect ϱ-set of G. Conversely, assume that we are given a perfect ϱ-set of G. As argued, the variable gadgets can be in one of two states in this case as illustrated in Figure 5. This state is likewise represented in the adjacent transmitters and will thus be transmitted without error to the clauses. Since the ϱ-set is perfect, one of the central triangle's vertices of each clause gadget must be in the set. Hence, at least one of the adjacent transmitters must be in a true state, corresponding to the assignment of one of the literals. Hence the states of the variables as in Figure 5 correspond to a satisfying assignment of the 3-SAT formula. A sample reduction is illustrated in Figure 8.

Since the reduction is based on deciding whether the given graph contains a perfect ϱ-set whose size is equal to the number of cliques in G it yields that both the maximization of local coverage as well as the maximization of the size of the ϱ-set with given minimum local coverage are NP-hard.

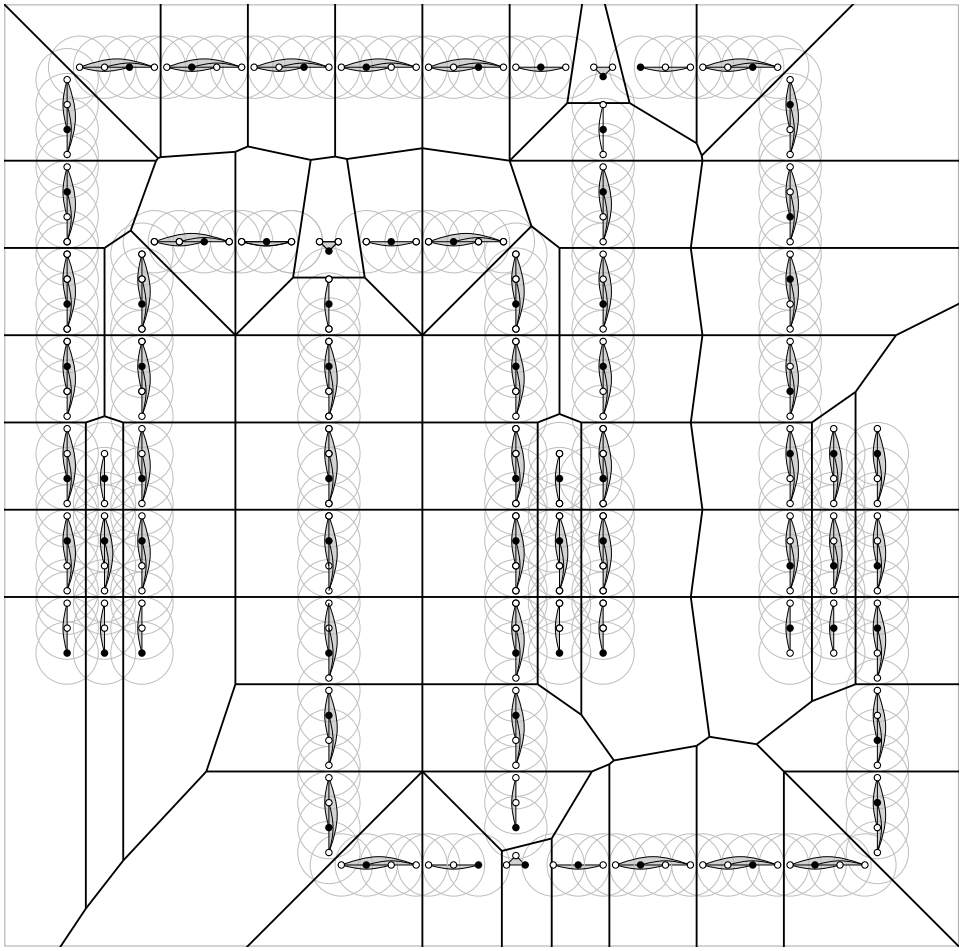


Fig. 8: Sample reduction of the 3-SAT formula $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4)$. The black points constitute a perfect ϱ-set corresponding to the assignment $x_1 = \text{true}, x_2 = \text{true}, x_3 = \text{true}, x_4 = false$.

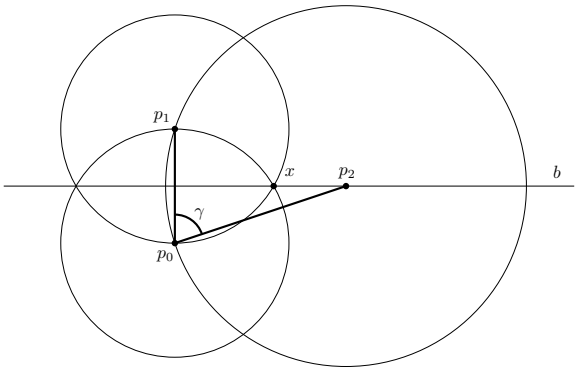


Fig. 9: Illustration for the proof of Lemma 1

□

2.2 Approximating the Maximum Size of a Generalization

Although it is unlikely that we can efficiently compute a ϱ-set with maximum size, we show that we can approximate the size of a maximum ϱ-set.

Theorem 2. *Let G be a geometric graph and let $r \in \mathbb{R}_0^+$ and $\alpha \in [0,1]$ be given. In $O(kn + n\log^5 n(\log\log n)^2)$ time we can compute a generalization $\mathcal{H}$ of G that approximates the maximum number of vertices of a generalization by a factor of $(7k+2)/3$, where $k = \max_{p\in P} |B(p,\varrho(p)) \cap P| - 1$.*

In order to prove Theorem 2 we use the following auxiliary lemma.

Lemma 1. *Let p_0 be a point in the plane and let $k \in \mathbb{N}$. Then there are at most $6k$ points Q such that p_0 is among the k closest points for each of the points $q \in Q$.*

Proof. We first establish the somewhat simpler claim that there are at most 6 points $p_1, \ldots, p_t$ ($t \leq 6$) such that for each p_i no point is closer to p_i than p_0, i.e., p_0 is the closest point to each p_i. Assume without loss of generality that p_1 is closest to p_0 and consider the infinite ray r_1 starting in p_0 in the direction of p_1. Let p_2 be the next point to p_1 in the clockwise cyclic order of $p_1, \ldots, p_t$ around p_0, as illustrated in Figure 9. Without loss of generality we may assume that p_2 is such that the disk D_2 centered at p_2 with radius $d(p_2, p_0)$ touches p_1. If this is not the case, we rotate p_2 around p_0 counter-clockwisely until it touches p_2. By the choice of p_2 there will be no point in the disk around p_2 in its new position. If D_2 touches both p_0 and p_1, then its center must be on the bisector b of p_1 and p_2. Since it must also be outside D_1, the disk centered at p_1 with radius equal to $d(p_0, p_1)$, p_2 cannot be closer to p_1 on b than the intersection x of b and the boundary of D_1. Since p_0, p_1 and x form the corners of a equilateral triangle, the angle between p_0p_1 and p_0p_2 must be at least 60 degrees. By repeating the argument, it is clear that the angle between p_0p_1 and p_0p_i increases by 60 degrees for each $i = 2, \ldots, t$. After at most 6 steps, this angle is at least 360 degrees. Hence there are at most 6 points with the desired property. Since we can put at most $k-1$ additional points in each of the disks, the claim of the lemma holds. □

Now we are ready to prove Theorem 2.

Proof (Proof of Theorem 2). Let H be the graph on the set of points such that pq is a (directed) edge if and only if $q \in B(p, \varrho(p))$. The graph H contains an independent set of size s if and only if G contains a ϱ-set of this size. Each independent set in H corresponds to a ϱ-set in G since each point in H is connected to all points that are closer than $\varrho(p)$ and it is connected to all points q such that p is in the $\varrho(q)$ disk around q. On the other hand, each ϱ-set in G induces an independent set due to this construction.

By choice of ϱ, each vertex has out-degree bounded by $k = \max_{p \in P} |B(p, r) \cap P| - 1$ for any value of α. There is an ingoing edge from q into p if and only if p is among the k closest neighbors of q. By Lemma 1 there are at most $6k$ points such that p is among the closest k points for each of these points. Hence, the in-degree of each vertex is bounded by $6k$. In total, each vertex has degree at most $7k$. Hence, by a result due to Halldórsson and Radhakrishnan [26] we can approximate the maximum size of an independent set by a factor of $(7k + 2)/3$. The algorithm greedily chooses the minimum degree vertex in each step and can be implemented to run in time $O(kn)$, given the graph H.

In order to compute H we locate the points in a closed disk by a circular range query in $O(\log n + k)$ time using $O(n \log^5 n (\log \log n)^2)$ preprocessing time [11]. Hence, the total running time is $O(kn + n \log^5 n (\log \log n)^2)$. □

Based on this approximation, we heuristically compute a ϱ-set Q balancing both the size of Q and the local coverage of the Voronoi clustering induced by Q as follows. For $p \in P$ let $\widetilde{m}(p)$ denote the number of edges whose endpoints are both contained in $B(p, \varrho(p)/2)$ and let $\widetilde{n}(p)$ denote the number of points in $B(p, \varrho(p))$. We can use these values to compute an estimate of the local coverage as summarized in the following lemma.

Lemma 2. *Let Q be an inclusion-maximal ϱ-set and let $\alpha = 0$. Further, let $H = (Q, F)$ be the generalization obtained from $G = (P, E)$ by the Voronoi-mapping ν. Then the value*

$$\min_{q \in Q} \left\{ \frac{2\widetilde{m}(q)}{\widetilde{n}(q)(\widetilde{n}(q) - 1)} \right\}$$

is a lower bound for $\operatorname{lcov}(H, \nu)$.

Proof. For $\alpha = 0$ we have $\varrho \equiv r$. Whenever p is chosen as a cluster center in Q, the points in $B(p, r/2)$ are closer to p than to any other point in Q, since the closest point to p in Q has distance to p at least r. Hence, the edges in $B(p, r/2)$ are intra-cluster edges of C_p. On the other hand, the number of points in each of the clusters is bounded by $\widetilde{n}(p)$ whenever $\alpha = 0$ and Q is an inclusion-maximal ϱ-set. To see this, consider any vertex q that is not contained in $B(p, r)$, but closer to p than to any other cluster center. Then q is contained in none of the disks centered in the cluster centers and, thus, q must be a cluster center itself, since Q is inclusion-maximal. Hence, the claim holds. □

Based on Lemma 2 we propose a heuristic, called Greedy Weight Heuristic, that operates as follows. First we compute an estimate of $2\widetilde{m}(q)/(\widetilde{n}(q)(\widetilde{n}(q) - 1))$ for each $p \in P$ since computing the exact value involves complicated algorithms and data structures. Subsequently, we sort the points according to these estimates in $O(n \log n)$ time and iteratively consider the points in this order. If the current vertex is not covered by the ϱ-disk of a previous vertex, then it is chosen for the ϱ-set, otherwise it is discarded.

Instead of computing $\widetilde{m}(p)$ and $\widetilde{n}(p)$ exactly, we estimate these numbers by counting the number of vertices and edges in the bounding boxes of the disks $B(p, \varrho(p)/2)$. To count the number of edges we use a 4-dimensional range searching query on a data structure containing tuples of points corresponding to edges in E with query time $O(\log^3 m)$ [12]. We use the 2-dimensional counterpart to locate points. Further, we use a data structure for dynamic nearest neighbor queries with $O(\log^2 n)$ query time [6], into which we insert the selected points to decide whether the current point is covered by a previously selected point. The total running time is $O((n+m)\log^3 m + n\log^2 n)$.

3 Minimizing Edge-Clutter

In order to reduce the clutter resulting from an excess of edges in certain areas we must filter out some of the edges without destroying the visual appearance of the graph. The total length of the edges seems to be a good measure for the clutteredness of the graph since it is proportional to the ink used for the drawing. While a minimum spanning tree will minimize this quantity, it is unlikely to preserve the visual appearance of the graph. We therefore require that monotone tendencies of the edges are preserved in order to best maintain the mental map of the adjacencies between vertices of the graph. This also motivated from a recent work by Huang et al. [30], whose controlled user experiments seem to suggest that geodesic paths are more likely to be explored when reading a graph drawing.

Let ℓ be a line in the plane and let $S = (p_1, \ldots, p_k)$ be a sequence of points. We say that S is *ℓ-monotone* if the order of the orthogonal projections of $p_1, \ldots, p_k$ onto ℓ is the same as the order of the points in S. Let $G = (P, E)$ be a geometric graph and let (H, φ) be a generalization of G such that $H = (P, F)$, i.e., $F \subseteq E$. We say that H is a *monotone generalization of G* if for every edge $e \in E$ with endpoints p and q there is a p-q-path π_e in H such that π_e is ℓ_e-monotone, where ℓ_e is the line defined by the endpoints of e. Given $G = (P, E)$ the Shortest Geodesic Subgraph (SGS) problem asks for a monotone generalization H of G minimizing the total length of H.

Problem Shortest Geodesic Subgraph (SGS)

Instance: Geometric graph $G = (P, E)$, $\delta \in \mathbb{R}_0^+$
Solution: Geometric graph $H = (P, F)$ such that $F \subseteq E$ and such that H contains a *monotone path* for each edge $e \in E$
Goal: minimize total length of H

First, we show that Shortest Geodesic Subgraph is NP-hard.

Theorem 3. Shortest Geodesic Subgraph *is NP-hard.*

Proof. We reduce from monotone 3-sat, a variant of 3-sat where each clause contains either only positive or only negative literals. Monotone 3-sat is NP-complete [23]. Let φ be an instance of monotone 3-sat with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$. We construct the following instance G_φ of Shortest Geodesic Subgraph. For each variable x we create a kite consisting of vertices ℓ, r, t and f as shown in Figure 10a. Note that the angles at f, ℓ and r are strictly less than 90°, and the angle at t is strictly more than 90°. The two edges incident to the top vertex t are called *top edges*, the edges $f\ell$ and fr are called the left and right *side edges*, respectively. We place the kites so that their foot points lie evenly spaced on the x-axis and the kites are disjoint. Denote by s_ℓ and s_r the slopes of the left and right side edges of a kite, respectively. The region R^+ is the region below the x-axis and to the right of the line through the bottom point of the rightmost kite

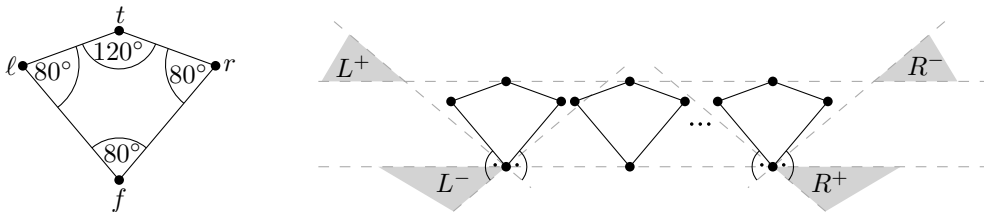


Fig. 10: Overview of the reduction from 3-SAT to Shortest Geodesic Subgraph. A kite with foot point f, top point t and left and right points r and ℓ (a), and the arrangement of the kites in the reduction with the corresponding regions for clause vertices (b).

with slope $-1/s_r$ (i.e., it is perpendicular to the right sides of the kites). Further, we denote by L^+ the region that is above the horizontal line defined by the topmost points of the kites and to the left of the line with slope $-1/s_r$ through the foot point of the leftmost kite. We define R^- and L^- analogously, with s_r replaced by s_ℓ.

It follows immediately from the construction that a path that is monotone in the direction from a point in R^+ to a point in L^+ may not contain any right edge of a kite as this would imply a turn of more than 90°, which is not monotone. Analogously, monotone paths from L^- to R^- may not contain left edges of kites. In our reduction the kites will play the role of variables, and edges from R^+ to L^+ (from L^- to R^-) will play the role of clauses with only positive (only negative) literals.

For each clause C_i consisting of only positive literals, we add a *clause vertex* c_i^1 into R^+ and a clause vertex c_i^2 in L^+. We add *connector edges* that connect c_i^1 to the foot points of all kites that correspond to variables that occur in C_i and that connect c_i^2 to all the left points of kites that correspond to variables that occur in C_i. Finally, we add the *clause edge* $c_i^1c_i^2$. We treat the clauses consisting of only negative literals analogously, except that we place the new vertices in L^- and R^-, respectively, and we connect the new vertices in R^- to the right kite points instead to the left.

This completes our construction, and we claim that an optimal solution of this instance allows us to decide whether the initial formula φ was satisfiable. We will make this more precise in the following. A subset of edges of G_φ is called *tight* if it contains both top edges of each kite, all connector edges, and exactly one of the two side edges of each kite. We now claim the following.

Claim. Any feasible solution contains a tight edge set.

Proof of Claim. First note that the top vertex of each kite is incident to only two edges, hence at least one of them must be in any feasible solution. However, the left edge is not monotone in the direction of the right edge and vice versa. Hence, a feasible solution necessarily contains both of them.

Next, we show that all edges from clause vertices in L^- or R^+ to foot vertices of kites must be contained in every solution. Let c be a vertex in L^- (the case c in R^+ is symmetric) and let f a foot point of a kite that is adjacent to c. We now consider the paths from c to f that avoid the edge cf in our graph. Since G contains no edge connecting two vertices of different kites, any path from c to f that avoids cf must contain at least one vertex $x \neq c$ that is in one of the four regions L^+, L^-, R^-, and R^+. Note that, by construction of the region L^-, the line orthogonal to cf is at least as steep

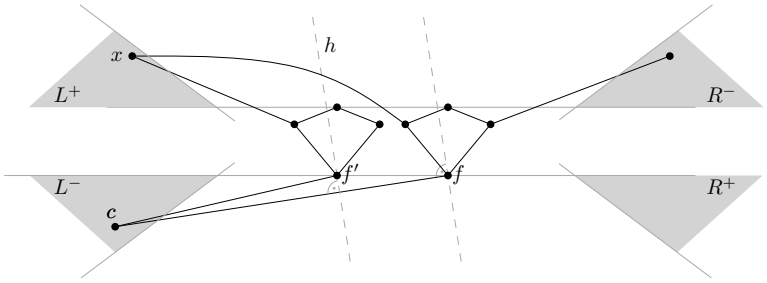


Fig. 11: Illustration for the proof of the first claim. There is no monotone path in G_φ replacing the edge cf since every path avoiding this edge first visits a footpoint f' of a kite before it visits a point $x \in L^+$ whose orthogonal projection onto the line defined by c and f is to the left of the projection of f'.

as the left side of any kite, and hence the points in R^- and in R^+ lie to the right of the line that is orthogonal to cf through f, and thus are not part of any monotone connection from c to f as illustrated in Figure 11. Now assume that x is in L^+ or L^-, and denote by f' the first vertex after c on a cf-path that avoids the edge cf. The regions L^+ and L^- lie to the left of the line h that is orthogonal to cf through the foot point of the leftmost kite. Therefore both the edge cf' and the subpath from x to f must cross this line, and hence project to the same point on the line segment from c to f. This shows that the path is not monotone, and hence cf must be contained in any feasible solution.

Next, consider a vertex c in L^+ and a corresponding edge $c\ell$ to the left vertex of a kite (again the case c in R^- and edge cr where r is the right vertex of a kite is symmetric). As before, every path from c to ℓ avoiding $c\ell$ must contain a vertex $x \neq c$ belonging to one of the four regions. Again, the regions R^- and R^+ are to the right of the line orthogonal to $c\ell$ through ℓ, and can thus not be contained in a monotone $c\ell$-path. Hence, we can assume that x is in L^+ or L^-. Let ℓ' be the first vertex after c on a $c\ell$-path that avoids the edge $c\ell$. If x is in L^-, consider the line orthogonal to $c\ell$ through the foot point of the leftmost kite. By construction this line is at least as steep as the right side of a kite, and hence the region R^+ is to its left. Since any path from c to x must contain a foot vertex of a kite, both the subpath from c to x and the subpath from x to f must cross this line, and thus project to the same point on the edge $c\ell$. Hence the path would not be monotone and we can assume that x is in L^+. Considering the line orthogonal to $c\ell$ through the left point of the leftmost kite as above rules out the existence of such a monotone path.

It remains to show that at least one side edge of each kite must be in any feasible solution. Let f be the foot point of a kite K with left point ℓ, right point r and top point t. We show that G does not contain a monotone $f\ell$-path that avoids both $f\ell$ and fr. First observe that all foot points of kites to the right of K project before f on the line through f and ℓ, directed from f to ℓ, and hence cannot be contained in a monotone $f\ell$-path. Similarly, all non-foot vertices of kites to the left of K project behind ℓ on this line, and hence are also not contained in monotone $f\ell$-paths. The points in R^+ and all points in L^+ can be ruled out similarly. Since a monotone $f\ell$-path needs to contain an edge that connects a vertex whose y-coordinate is at most the y-coordinate of f to a vertex whose y-coordinate is at least the y-coordinate of ℓ, and we cannot use any edge of a kite, the only option is that it uses an edge from a vertex x in L^- to a vertex x' in R^- as illustrated in

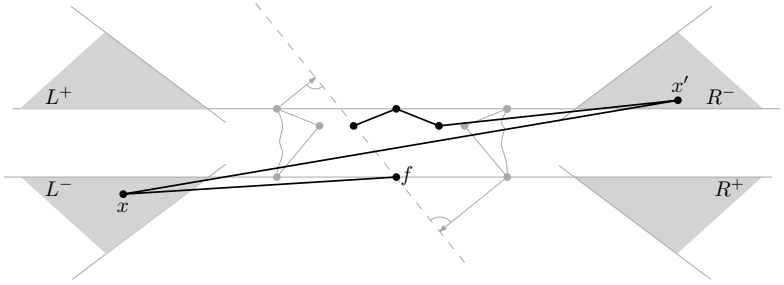


Fig. 12: Illustration for the proof of the first claim. In each of the kites at least one of the side edges must be present. Otherwise, any replacing path for $f\ell$ must use an edge xx' that is not monotone with respect to $f\ell$.

Figure 12. However, the line orthogonal to such an edge is steeper than the edge $f\ell$, and hence xx' is not monotone with respect to $f\ell$. This completes the proof of the claim.

Note that the size, as well as the total length, is the same for all tight edge sets, and hence this size forms a lower bound for the size of a geodesic subgraph. We claim that this bound can be met if and only if φ is satisfiable.

Claim. There exists a tight set that is feasible if and only if φ is satisfiable.

Proof of claim. Note that a tight set is completely specified by giving for each kite the information whether its left or right edge is contained in the set.

Assume that φ is satisfiable and take a satisfying assignment. We construct a tight set E' by taking the left side of a kite if and only if the corresponding variable has the value *true* in the assignment. We now argue that the corresponding set is feasible. The only edges for which we have to check the existence of a monotone replacement path are the clause edges. Let $c_i^1c_i^2$ be a clause edge with c_i^1 in R^+ and c_i^2 in L^+. The edge $c_i^1c_i^2$ by construction corresponds to a clause C_i with only positive literals. Let x_j be a satisfied literal (and thus a satisfied variable) in C_i, and let K_j denote the corresponding kite with foot point f and left point ℓ. By construction E' contains the edges $c_i^1f, f\ell$ and ℓc_i^2, which together form a monotone $c_i^1c_i^2$-path. The argument for a clause edge $c_i^1c_i^2$ with c_i^1 in L^- and c_i^2 in R^-, which corresponds to a clause with only negative literals, is analogous. This proves that the tight set E' is feasible.

Conversely, assume that E' is a feasible tight set. We construct a truth assignment by setting a variable to true if and only if the left edge of the corresponding kite is in E'. Now consider a clause C_i containing the variables x_u, x_v and x_w as positive literals (the case of only negative literals is symmetric). If C_i is not satisfied by our assignment then E' contains none of the left edges of the three kites corresponding to x_u, x_v and x_w. However, by definition the edge set must contain a monotone $c_i^1c_i^2$-path. Such a path may not use any right edge of any kite as this would not be monotone. Hence it necessarily contains a left edge of some kite since E' does not contain any of the clause edges. This implies that any monotone path must first visit the foot point of one of the kites corresponding to x_u, x_v, x_w, then pass on to a vertex $x \neq c_i^1$ in L^- or R^+ and from there to the foot point of another kite. By construction all points in R^+ lie to the right of the line that is orthogonal to $c_i^1c_i^2$ through the foot of the rightmost kite. This excludes the case that x is in R^+. Similarly, the line orthogonal to $c_i^1c_i^2$ through the foot of the leftmost kite separates the points in R^- from the

foot points of all kites. Hence the edges from x to its two incident foot points both cross this line and hence the path is not monotone. This is a contradiction, and hence E' must contain the left edge of at least one of the kites corresponding to x_u, x_v and x_w, thus implying that C_i is satisfied. This proves the claim.

Note that the size L is the same for all tight edge sets. The first claim shows that any geodesic subgraph has length at least L. And thus, the second claim implies that φ is satisfiable if and only if G_φ admits a geodesic subgraph of length at most L. Since the construction can easily be performed in polynomial time this concludes the proof. □

As we have seen, the restriction to edges from the input graph makes it difficult to construct short monotone subgraphs. One possibility is thus to drop this constraint and to allow arbitrary edges. Additionally, we would like to control the distance of the monotone path π_e and the edge it is approximating in terms of monotonicity. This is motivated by the observation that the shortest monotone generalization of a clique, whose vertices are arranged equidistantly on a circle, is given by the convex hull of the pointset. Given a line segment s with length ℓ_s and a point p with distance d_p from s we call the ratio d_p/ℓ_s the *drift of p from s*. The *drift of a path* π_e with endpoints pq is defined as the maximum drift of any point on π_e from the segment pq. Given a geometric graph $G = (P, E)$ and a non-negative real number $\delta \in \mathbb{R}_0^+$ the Sparse Geodesic Network (SGN) problem asks for a geometric graph $H = (P, F)$ with minimum total length such that for each edge e in E there is an ℓ_e-monotone path π_e with drift at most δ, where ℓ_e denotes the line defined by the endpoints of e.

Problem Sparse Geodesic Network (SGN)

Instance: Geometric graph $G = (P, E)$, $\delta \in \mathbb{R}_0^+$
Solution: Geometric graph $H = (P, F)$ such that H contains a *geodesic path* for each edge $e \in E$ whose vertices are at distance at most $\delta \cdot |e|$ from the straight line e
Goal: minimize $|F|$

We show the following.

Lemma 3. *Given a (complete) geometric graph $G = (P, E)$, the Delaunay graph $\mathcal{D}(P)$ contains for each edge $e \in E$ an ℓ_e-monotone path π_e with drift at most $1/2$.*

Proof. Let P be a set of points and let $p, q \in P$. Without loss of generality we assume that p and q are on the x-axis such that $x(p) < x(q)$. According to Dobkin et al. [16] we can construct an x-monotone path in the Delaunay graph $\mathcal{D}(P)$ of P as follows. Let $\mathcal{V}(P)$ denote the Voronoi diagram of P and let $p_1, \ldots, p_k$ be the ordered points corresponding to the Voronoi cells that are traversed when following the line from p to q. Then the path $p, p_1, \ldots, p_k, q$ is an x-monotone path in the Delaunay graph. Further, all points p_i are contained within the disk with radius $d(p, q)/2$ centered in the midpoint of the segment pq. Hence, the drift is at most $1/2$. □

Although the Delaunay graph seems to be well suited to represent monotone tendencies, this result also shows the limitations of allowing arbitrary edges. In the following we therefore focus on subgraphs of the original graph and describe a greedy heuristic for computing a monotone generalization with bounded drift δ and short total length, which we call Monotone Drift Heuristic. Given a geometric graph $G = (P, E)$ and a maximal drift δ we sort the edges of G with respect to increasing length in $O(m \log m)$ time. Then we consider the edges $e_1, \ldots, e_m$ in this order and iteratively construct a sequence of graphs $H_0, H_1, \ldots, H_m$, where $H_0 = (P, \emptyset)$. We insert the edge e_i

into H_{i-1} whenever there is no ℓ_{e_i}-monotone path with drift at most δ in H_{i-1}. This can be tested by performing a modified depth-first search exploring only monotone subpaths in $O(n+m)$ time. Hence, the total running time of this approach is $O(nm+m^2)$.

4 Vertex-Edge-Clutter

Vertex-edge-clutter is the most complicated type of clutter since it involves both vertices and edges and the selection of these features cannot be handled independently as in the previous sections. On the other hand, this type of clutter may be considered as the least annoying type of clutter. While vertex-edge clutter is caused by edges that are close to a vertex resulting in the difficulty to determine correct incidences,the human perception is rather good at determining whether a line passes a disk through the center or not. For instance, it is easy to see that the leftmost line in Figure 13 is not incident to the vertex although it crosses the vertex. Additionally, the human perception is also good at determining whether a line has a bend or not, which is illustrated in Figure 13.

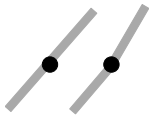


Fig. 13: Line perception

Hence, as long as there is neither vertex-clutter nor edge-clutter and as long as no pair of edges incident to a common vertex form a 180°-angle, we will be able to unambiguously tell whether an edge is incident to a vertex or not. In order to attack vertex-edge clutter we therefore propose the following optimization problem. For a pair of edges incident to a common vertex p we define the *angular straight-line deviation* as the smaller of the two angles that is enclosed by the lines defined by the two edges, respectively. The angular straight-line deviation of p is then defined as the minimum angular straight-line deviation over all pairs of edges incident to p, as illustrated in Figure 14. The angular straight-line deviation of a geometric graph G is the minimum angular straight-line deviation over all vertices of G. Note, that the angular straight-line deviation is maximized if all angles are close to a right angle. Given a geometric graph $G=(P,E)$ and a non-negative value $r \in \mathbb{R}^+$, the Optimal Angle Adjustment problem is to find a new position for each vertex p inside $B(p,r)$ minimizing the angular straight-line deviation of the resulting geometric graph.

Problem Optimal Angle Adjustment

Instance:	Geometric graph $G=(P,E)$, $r \in \mathbb{R}_0^+$
Solution:	Geometric graph $H=(Q,F)$ and a mapping $f: P \to Q$ such that $d(p,f(p)) \leq r$ and such that $f(p)f(q) \in F$ if and only if $pq \in E$
Goal:	maximize the angular straight-line deviation of H

Note that this problem differs considerably from the problem of maximizing the angular resolution of a graph, defined as the minimum angle over all pairs of adjacent edges. The optimal angular resolution of a star-shaped graph with an odd number n of vertices, for instance, will result in zero straight-line deviation, while it is obvious that the optimal straight-line deviation is positive. We tackle the Optimal Angle Adjustment problem by maximizing the vertices' distances from the lines defined by the edges incident to their neighbors. Let $G=(P,E)$ be a geometric graph and let $v \in P$ be a vertex. Let $N(v)$ denote its neighbors in G. Further, let $E(v)$ denote the edges incident to v and let $F(v)$ denote the set of edges incident to the vertices in $N(v)$ but not to v. By moving v we change the angles formed by pairs of edges in $E(v)$ as well as the angles formed by pairs of edges (e,f) such that $e \in E$ and $f \in F$, respectively. Let $L_F(v)$ be the set of lines defined

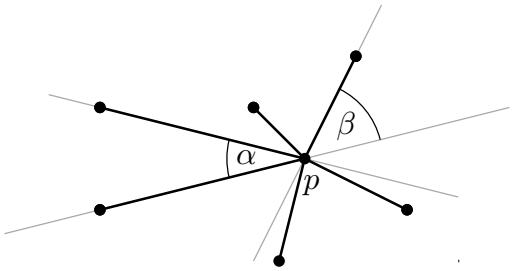


Fig. 14: Angular straight-line deviation of a vertex p. In the drawing the angular straight-line deviation is defined by the angle α.

by the edges in $F(v)$ and let $L_E(v)$ be the set of lines defined by all pairs of vertices in $N(v)$. A vertex v along with the lines defined by the edges in $L_E(v)$ and $L_F(v)$ is illustrated in Figure 15a. Note, that there will be an angle of 180 degrees involving an edge incident to v if and only if v is placed on one of the lines in $L_E(v) \cup L_F(v)$. Given $p \in \mathbb{R}^2$ we denote by $\mu_v(p)$ the minimum distance of p to the lines in $L_E(v) \cup L_F(v)$. We prove the following.

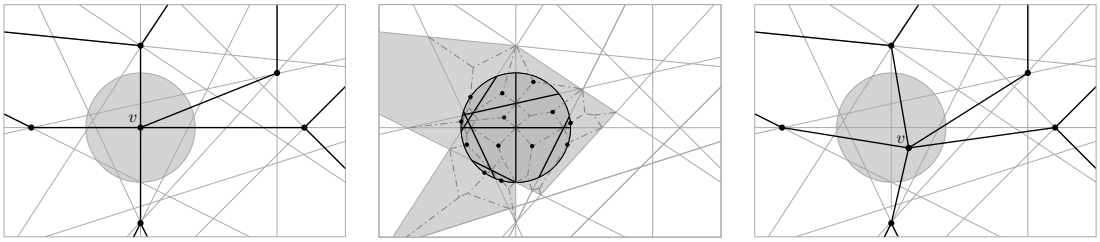
Theorem 4. *Given a graph $G = (P, E)$, a vertex $v \in P$ and a positive radius $r \in \mathbb{R}^+$ we can compute a new position p^* for v in $B(v, r)$ such that $\mu_v(p^*) > 0$ and such that p^* maximizes $\mu_v(p)$ over all $p \in B(v, r)$ in $O(t^3\alpha(t))$ time where $t = \min\{\Delta^2, m\}$, Δ denotes the maximum degree of G and $\alpha(\cdot)$ denotes the inverse Ackermann function.*

Proof. First, we compute the set of edges $L_F(v)'$ incident to v's neighbors, but not to v, that intersect $B(v, r)$ as well as the set of lines $L_E(v)'$ defined by all pairs of v's neighbors intersecting $B(v, r)$. Let $L = L_E(v)' \cup L_F(v)'$. We compute the arrangement of lines in L in $O(|L|^2)$ time. Over each of the resulting faces C we compute the lower envelope of the hyperplanes defining the distance to the boundaries of the faces and project the graph $\mathcal{G}_C$ defined by the resulting 3-dimensional polytope onto the plane. This is illustrated in Figure 15b.

The lower envelope of a set of n hyperplanes can be computed in $O(n^2\alpha(n))$ time where $\alpha(\cdot)$ denotes the inverse of the Ackermann function [18]. Hence the lower envelopes can be computed in time $O(|L|^2\alpha(|L|))$ for each face, resulting in a total complexity of $O(|L|^3\alpha(|L|))$. For each face C we inspect the vertices of $\mathcal{G}_C$ in $B(v, r)$ as well as its intersection with $B(v, c)$ and thus compute the point p^* maximizing μ_v in $B(v, r)$. Then we update the position of v as illustrated in Figure 15c. Since L is bounded by $\max\{\Delta^2, m\}$ we obtain the claimed time complexity. Further, since $r > 0$ and therefore $B(v, r)$ is non-degenerate, there must be a non-degenerate face in the arrangement containing a point p^* in its interior such that $\mu(p^*) > 0$. □

Using Theorem 4 we can incrementally compute a new position for each vertex v such that none of the edges incident to v encloses an angle of 180 degrees with any other edge. Since the angles between pairs of edges that are not incident to v are not affected by this operation, we can iteratively apply Theorem 4 to the vertices one after another to obtain a drawing with strictly positive angular straight-line deviation. At the same time this approach heuristically maximizes this deviation.

Note that we may assume that we apply the angle adjustment to a generalized graph whose complexity tends to be significantly lower than the complexity of the original graph, i.e., both m and Δ should be considerably smaller.



(a) (b) (c)

Fig. 15: Illustration for the proof of Theorem 4. (a) A vertex v and its neighbors as well as the arrangement of lines induced by the respective edges in $L_E(v)$ and $L_F(v)$. (b) Intersection of the circle with projections of the graphs $\mathcal{G}_c$ (dashed) and locally optimal positions (black dots) in the faces. (c) Globally optimal position and resulting new drawing.

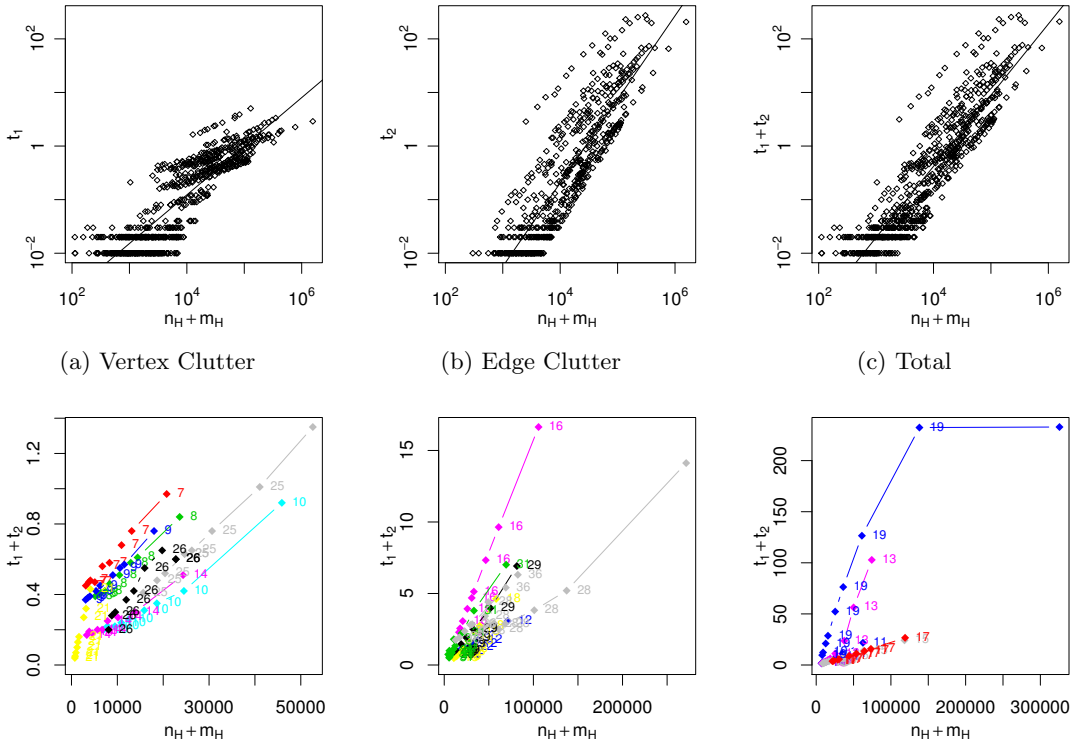
5 Sample Generalizations

In order to evaluate the quality of the described heuristics and in order to obtain estimates for the running time we implemented the described Greedy Weight Heuristic and Monotone Drift Heuristic in C++ using the BOOST library [8] and the CGAL library [1]. All generalizations were computed on a standard Intel Core 2 Duo processor running at 2.00 GHz with 2 GB RAM.

We performed our experiments on the benchmark set of graphs listed in Table 1 in Appendix A. These graphs have between 1,000 and 100,000 vertices and between 3,000 and 2,000,000 edges, respectively. The table lists, for each graph, an index that is used to identify the graphs in the following figures as well as its size. All but the graphs marked with $\star$ have been taken from the University of Florida sparse matrix collection [14]. The graph `clique-planar` is a planar graph with an implanted clique. The graph `lunar-vis` is a LunarVis [24] layout of a snapshot of the Internet graph at the autonomous systems level that has been taken from the data collected by the University of Oregon Routeviews Project [2]. The graph `email` is a force-based visualization of the graph obtained from the e-mail communication at the faculty of informatics at the Karlsruhe Institute of Technology during a fixed amount of time. The graphs `osm_berlin` and `osm_isleofman` are street networks of Berlin, Germany, and Isle of Man, respectively, that have been extracted from the OpenStreetMap data [3]. The graphs from the University of Florida sparse matrix collection have additionally been layouted using the `sfdp` multi-scale force-based layouter from the graphviz library [19].

For each of the graphs listed in Table 1 as well as for both $\alpha = 0$ and $\alpha = 1$, we performed generalizations with 10 different radii $r_i = \Delta/n + (\Delta/\sqrt{n} - \Delta/N) \cdot i$ in the range $[\Delta/n, \Delta/\sqrt{n}]$ for $i = 0, \ldots, 9$, where $\Delta := x_{\max} - x_{\min}$ is the width of the drawing. For each run, we measured the time t_1 of the Greedy Weight Heuristic and the time t_2 of the Monotone Drift Heuristic. Further, we collected the number of vertices n_H and the the number of edges m_H of the resulting generalized graphs.

Even for the largest input graphs with several thousand vertices and over a million edges, the observed running times were less than 5 minutes. However, most of the running time is caused by the Monotone Drift Heuristic, which has a quadratic worst-case running time. For the Greedy Weight Heuristic, the observed running time was less than 5 seconds for all graphs.



(a) Vertex Clutter (b) Edge Clutter (c) Total

(d) $0.3 \leq t < 3$ (e) $3 \leq t < 20$ (f) > 20

Fig. 16: Running times of the generalization heuristics with respect to the size of the generalization in a log-log-scale plot for $\alpha \in \{0, 1\}$ and drift $= 0.3$ (a)–(c) and running times for the individual graphs in a linear-scale plot for $\alpha = 1$ and drift $= 0.3$ (d)–(f). For reasons of clarity, Figure (a) contains only the graphs with $0.3 \leq t_1 + t_2 < 3$, Figure (b) contains only the graphs with $3 \leq t_1 + t_2 < 20$ and Figure (c) contains only the graphs with $t_1 + t_2 \geq 20$.

Figure 16 shows the running time of the heuristics as a function of the size $n_H + m_H$ of the generalized graphs. Figure 16a shows the running time t_1 of the Greedy Weight Heuristic, Figure 16b shows the running time t_2 of the Monotone Drift Heuristic and Figure 16c shows the resulting combined running time. In order to display all data in one chart, we employed a log-log-scale plot for these figures. We added a line to each chart displaying the results of a linear regression, applied to the log-transformed data. That is, for $x = n_H + m_H$ and for each $y \in \{t_1, t_2, t_1 + t_2\}$ we computed a_y and b_y minimizing the linear least-squares function

$$\min_{a_y, b_y \in \mathbb{R}} \sum_{i=1}^{N} (\log y_i - (a_y \log x_i + b_y))^2$$

where x_i and y_i denote the measured sizes and running times of the single experiments for $i = 1, \ldots, N$, respectively. The results suggest that the running time of the Greedy Weight Heuris-

TIC $t_1 \approx e^{-10.4578} x^{0.9076}$ is approximated by a function that is slightly sub-linear in the size of the generalized graph and the running time of the MONOTONE DRIFT HEURISTIC $t_2 \approx e^{-15.95} x^{1.56}$ is approximated by a super-linear but sub-quadratic function in the size of the generalized graph. The combined running time is approximately $t_1 + t_2 \approx e^{-13.105} x^{1.328}$, which is super-linear.

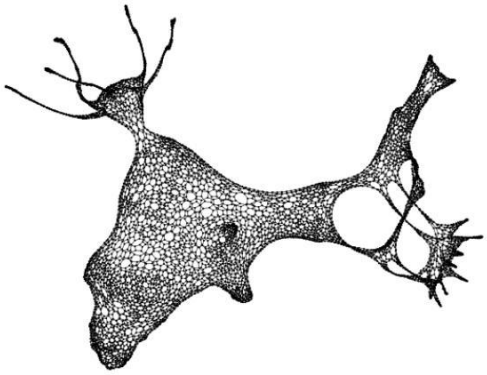
Figures 16d–16f display the running times for $\alpha = 1$ for the individual graphs of our benchmark on a linear scale. Figure 16d contains all graphs for which $t_1 + t_2$ was at most 3 seconds, Figure 16e contains all graphs whose maximum running time was between 3 and 20 seconds and Figure 16f contains all graphs whose running time was more than 20 seconds. With only a few exceptions, such as `ex3sta1`, `TF16` and `conf5_4-8x8-05`, the running times seem to be slightly sub-linear or slightly super-linear in the size of the generalized graph. The results for $\alpha = 0$ are similar.

Next, we shortly discuss the generalized graphs. Figure 17 shows how the parameter α affects the generalization. While the sizes of the generalized graphs for $\alpha = 0$ and $\alpha = 1$ will, in general, differ considerably for a fixed radius r, we chose generalized graphs with roughly the same sizes in order to illustrate the effects of choosing $\alpha = 0$ and $\alpha = 1$, respectively. Figure 17 clearly shows that the generalizations with $\alpha = 1$ are better suited at preserving the distribution of the original point set. However, this is only achieved at the price of a higher resolution of the resulting drawing. On the other hand, the homogeneous distribution of the points resulting from $\alpha = 0$ does not seem to capture the geometric properties of the original very well. Indeed, it seems that setting $\alpha = 0$ is not well suited for most of the graphs we inspected due to this behavior. Therefore, all remaining generalization are performed with $\alpha = 1$ if not otherwise stated.

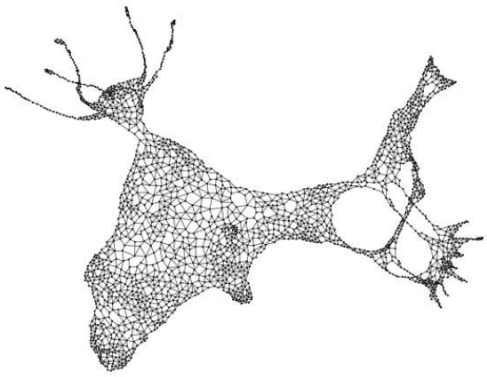
Figure 18 shows how the radius r and the drift δ impact the resulting generalizations for $\alpha = 1$. To illustrate the effects of δ we applied the GREEDY WEIGHT HEURISTIC and MONOTONE DRIFT HEURISTIC for different values of r and δ to a planar graph with an implanted clique whose vertices have been arranged equidistantly on a cycle. While none of the edges of the clique is distinctly perceivable in the original drawing, a higher drift helps remedying this without destroying the impression of a clique even without generalizing the vertex set, as can be seen in the first row of Figure 18. Note that the clique in the middle of the drawing remains a clique when setting $\delta = 0$ for all values of r. With increasing δ, the clique becomes much sparser but is still perceivable as a rather dense subgraph in the generalized graph for all radii.

Finally, Figures 19–26 show some selected sample generalizations. First, we discuss how the heuristics perform on the graphs we used to illustrate the various types of clutter in Figure 3. These graphs as well as the results of the heuristic generalization are displayed in Figures 19–21. Note that the displayed generalized graphs have only 2–5% of the vertices of the respective originals. Clearly, both vertex-clutter and edge-clutter can be significantly reduced without changing the main impression of the graph. However, two drawbacks of our approach are immediately obvious from these illustrations.

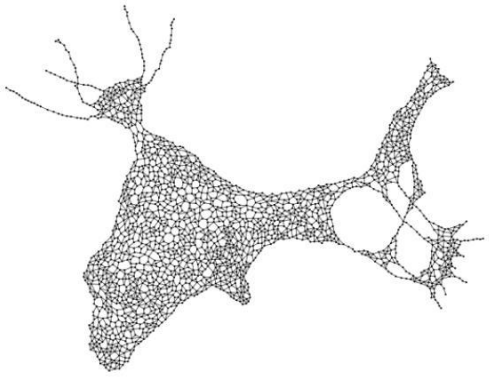
First, consider the graph `oix` and its generalization in Figures 20a and 20b, respectively. In the original, there are many edges between a few vertices on the left and the vertices in the bottom center. Apparently, these edges are mapped to only few monotone paths in the generalization, which changes the impression of the density of the edges. This could be tackled by emphasizing the edges in the generalization according to the number of edges that were mapped to it. As another approach to this problem we could try to approximate the geometric edge distribution. That is, similar to our approximation of the point-set distribution we could try to remove more edges from regions containing few edges and removing fewer edges in regions with many edges. In contrast to approximating the point-set distribution, however, it is not clear how to achieve this



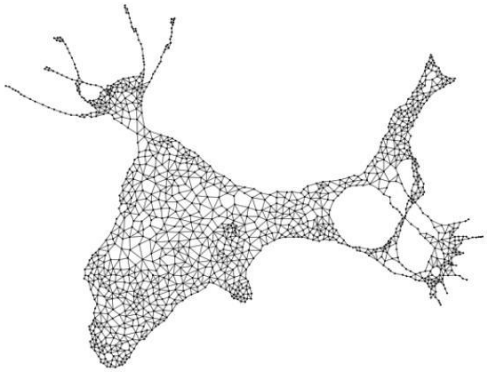
(a) original (n=7920, m=31680)



(b) generalization (n=2104, m=8585)



(d) generalization (n=2098, m=8188)



(c) generalization (n=1305, m=5636)

(e) generalization (n=1358, m=5678)

Fig. 17: Effects of parameter α on the `commanche_dual` graph from the University of Florida sparse matrix collection [14]. (a) Original, (b), (c) Generalization with $\alpha = 1$, (d), (e) Generalization with $\alpha = 0$.

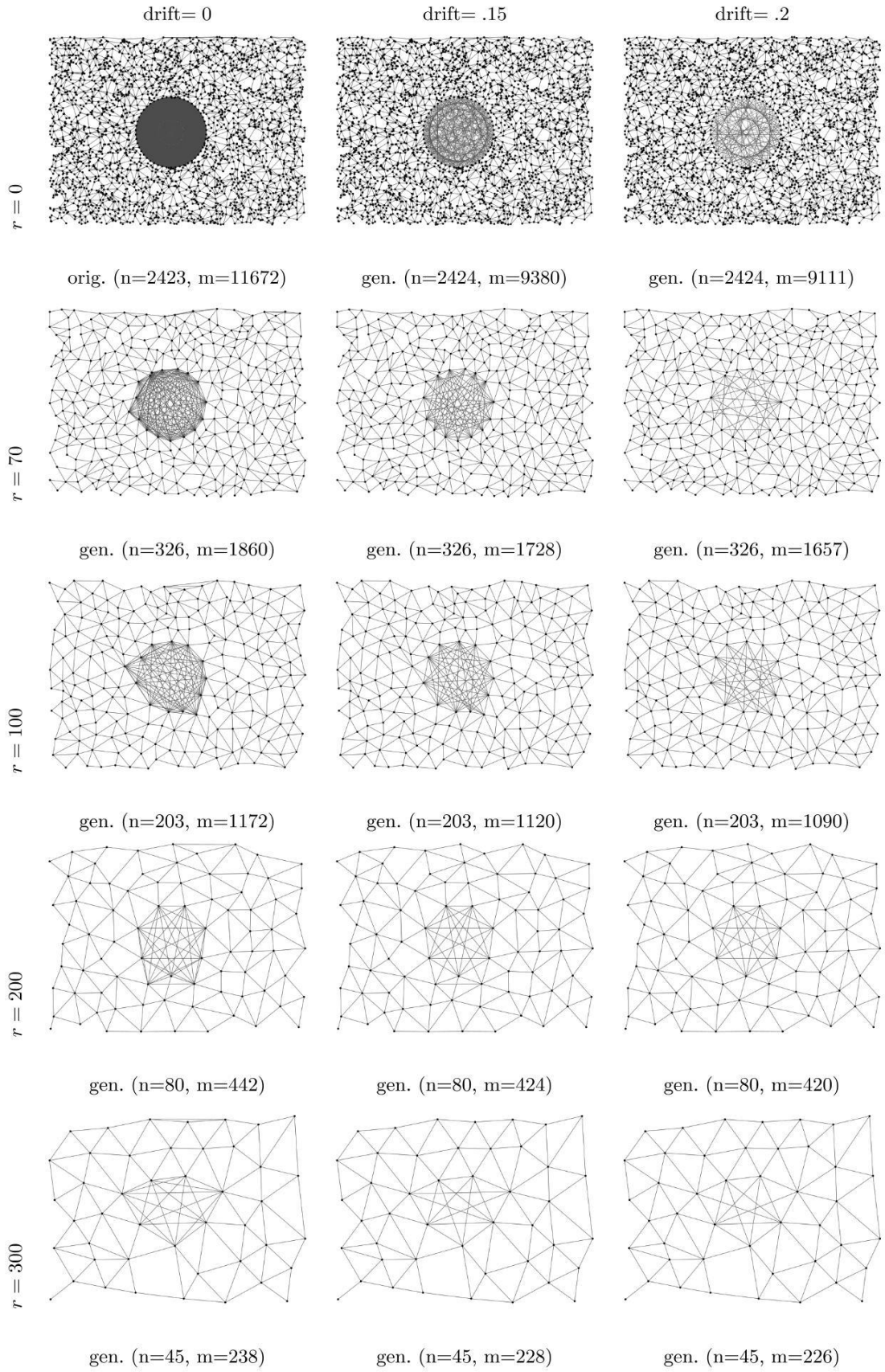
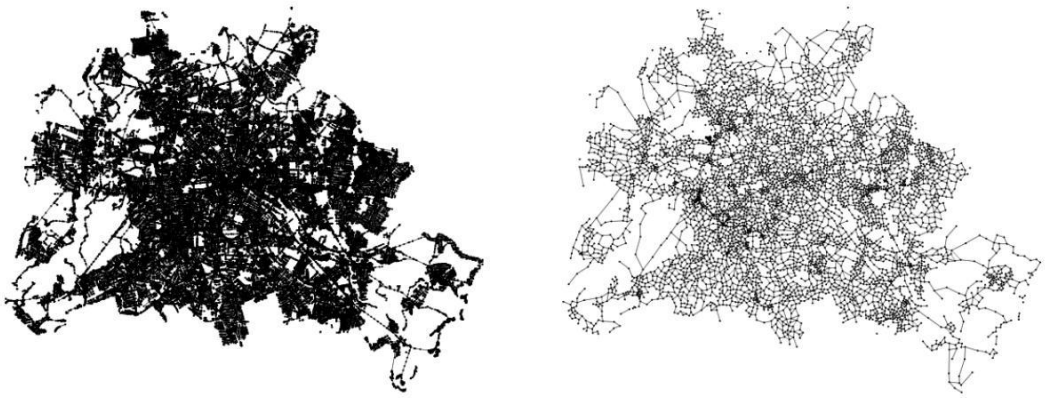


Fig. 18: Effect of radius and drift on the graph `clique-planar`, a planar graph with an implanted clique. All generalizations with $\alpha = 1$.



(a) original (n=106675, m=248390) (b) generalization (n=5649, m=17273)

Fig. 19: Streetmap Data of Berlin [3] (`osm_berlin`)

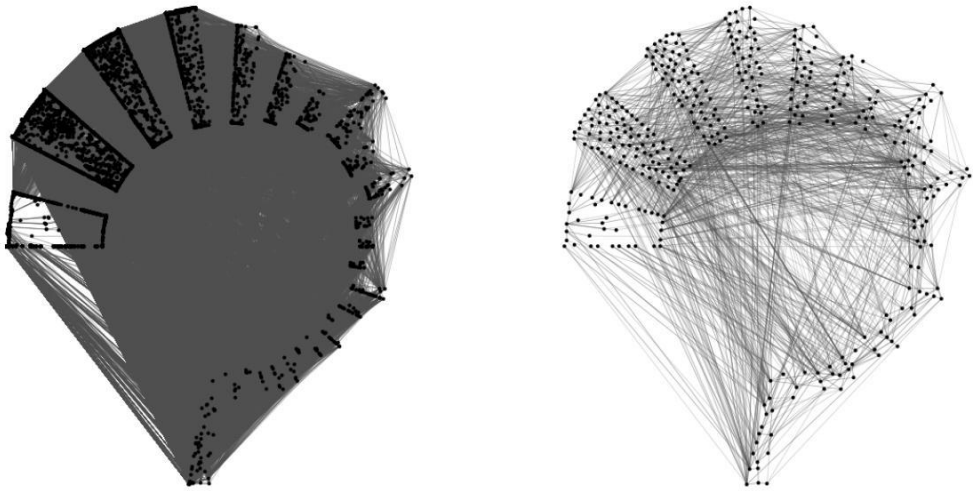
in a straightforward way since a single edge may cross both dense and less dense regions in the drawing.

Second, consider the the graph `PDS10` and its generalization in Figures 21a and 21b, respectively. Clearly, the topmost vertices of the generalization show that our approach may create unwanted adjacencies. These adjacencies are the result of contracting vertices that are close to each other and working on the contracted edge set. While the edges are not false in the sense that each edge in the contraction corresponds to at least one edge of the original, these edges create the wrong visual impression. This problem could be approached, for instance, by trying to approximate the features of the contracted vertex sets. For instance, the average degree of the contracted vertex sets will be roughly two for most of the problematic vertices in these figures, while the resulting degree in the generalization is larger.

The remaining figures serve as a further visual benchmark of the generalization heuristics. While the general (geometric) impression of the graphs are reasonably well maintained, some further issues for future research can be observed.

Consider, for instance the graph `ukerbe1_dual` and its generalization illustrated in Figure 24a and 24b, respectively. While the density of the point set and the size of the faces is maintained quite well, most of the faces are triangulated in the generalization, whereas most of the faces of the original contain four vertices. Further, consider the cube graph illustrated in Figure 25. The topological structure of the generalized graph is rather different from the original. Although the vertices contracted into single clusters are close to each other both geometrically and with respect to graph-distance, the cubic structure is not maintained. Again this may be remedied by approximating the features of the contracted vertices, such as average degree.

While the proposed heuristics do not solve the generalization problem in all its facets, especially with respect to the topological features of the graph, they seem to be well suited at maintaining



(a) original (n=17233, m=74436) (b) generalization (n=397, m=2134)

Fig. 20: LunarVis Layout of the AS-Graph [24] (`lunar-vis`)

the geometric impression of the originals and, thus, form good starting points for future research on this problem. In order to overcome the current difficulties, however, we must explicitly include topological features of the original graph into the generalization process.

6 Conclusion and Open Problems

We have undertaken a first step at studying the problem of generalizing geometric graphs within a rigorous mathematical model. We formalized the problem by considering an incremental framework modeling the elimination or reduction of different types of clutter as optimization problems, which we analyzed in terms of complexity. Since these problems turned out to be NP-hard in general, we also devised efficient approximation algorithms as well as efficient heuristics. We showed how to heuristically eliminate vertex-clutter in $O((n+m)\log^3 m + n\log^2 n)$ time and how to reduce edge clutter in $O(nm+m^2)$ time considering geometric features such as point distributions and geodesic tendencies. After the elimination of vertex-clutter and edge-clutter we can expect the graph to be much smaller than the original graph. Hence, even larger complexities may scale accordingly. Thus, even the relatively high complexity of our heuristic for reducing vertex-edge clutter may be practical.

Even without this step, however, the resulting generalizations exhibit considerably less clutter and are easier to analyze. We showcased some promising generalizations produced by our heuristics. We conclude by listing some open problems.

- Is it possible to approximate both the local coverage and the size of a ϱ-set in the vertex generalization step?
- What is the complexity of the Local Coverage Cluster Packing problem for different types of mappings?

– Is it possible to approximate the size of a shortest geodesic subgraph, possibly in the presence of a limited drift?
– What is the complexity of the optimal angle adjustment problem?
– How can the generalization problem be adapted to a dynamic scenario, where consistency issues play an additional role.

Acknowledgments. We thank Robert Görke for the helpful discussion and for providing the LunarVis layout.

References

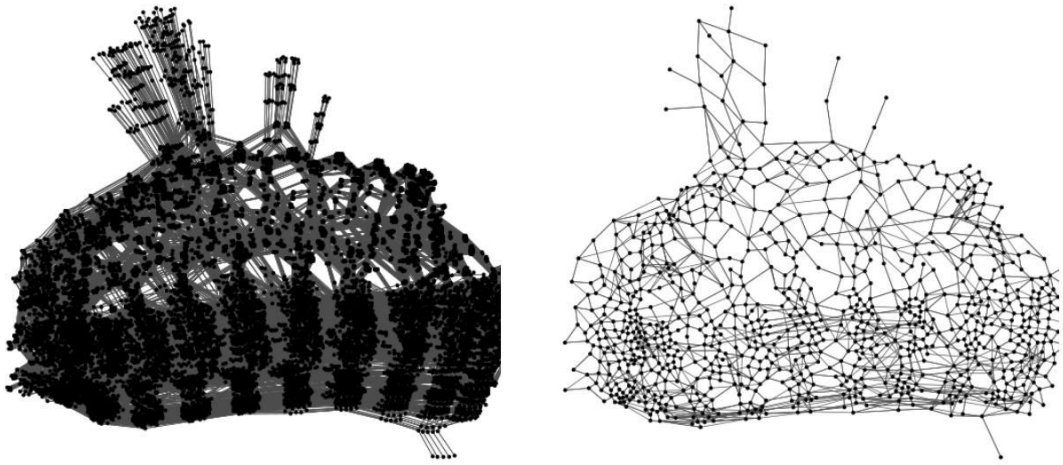
1. CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.
2. University of oregon routeviews project. http://www.routeviews.org/.
3. Openstreetmap database. online, 2011. http://www.openstreetmap.de/.
4. J. Abello, S. Kobourov, and R. Yusufov. Visualizing large graphs with compound-fisheye views and treemaps. In J. Pach, editor, *Graph Drawing*, volume 3383 of *Lect. Not. Comput. Sci.*, pages 431–441. Springer Berlin / Heidelberg, 2005.
5. J. Abello, J. Korn, and I. Finocchi. Graph sketches. In *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, pages 67–. IEEE Computer Society, 2001.
6. J. L. Bentley and J. B. Saxe. Decomposable searching problems I. static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.
7. R. E. Bohn and J. E. Short. How much information? 2009 Report on American consumers. Global Information Industry Center, University of California, San Diego, 2009.
8. Boost C++ libraries, version 1.42. http://www.boost.org.
9. U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Trans. Knowledge and Data Engineering*, 20:172–188, 2008.
10. Edith Brunel, Andreas Gemsa, Marcus Krug, Ignaz Rutter, and Dorothea Wagner. Generalizing Geometric Graphs. In *Proceedings of the 19th International Symposium on Graph Drawing (GD'11)*, Lecture Notes in Computer Science. Springer, 2012. to appear.
11. B. Chazelle, R. Cole, F.P. Preparata, and C. Yap. New upper bounds for neighbor searching. *Information and Control*, 68(1-3):105 – 124, 1986.
12. Bernard Chazelle. Functional approach to data structures and its use in multidimensional searching. *SIAM J. Comput.*, 17:427–462, June 1988.
13. Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165 – 177, 1990.
14. Timothy A. Davis. University of florida sparse matrix collection. *NA Digest*, 92, 1994.
15. M. de Berg and A. Khosravi. Optimal binary space partitions in the plane. In M. Thai and S. Sahni, editors, *Computing and Combinatorics*, volume 6196 of *Lect. Not. Comput. Sci.*, pages 216–225. Springer Berlin / Heidelberg, 2010.
16. David Dobkin, Steven Friedman, and Kenneth Supowit. Delaunay graphs are almost as good as complete graphs. *Discrete & Computational Geometry*, 5:399–407, 1990.
17. Peter Eades and Qing-Wen Feng. Multilevel visualization of clustered graphs. In Stephen North, editor, *Graph Drawing*, volume 1190 of *Lecture Notes in Computer Science*, pages 101–112. Springer Berlin / Heidelberg, 1997.
18. Herbert Edelsbrunner, Leonidas Guibas, and Micha Sharir. The upper envelope of piecewise linear functions: Algorithms and applications. *Discrete & Computational Geometry*, 4:311–336, 1989.
19. J. Ellson, E.R. Gansner, E. Koutsofios, S.C. North, and G. Woodhull. Graphviz and dynagraph – static and dynamic graph drawing tools. In M. Junger and P. Mutzel, editors, *Graph Drawing Software*, pages 127–148. Springer-Verlag, 2003.

20. G. W. Furnas. Generalized fisheye views. *SIGCHI Bull.*, 17:16–23, April 1986.
21. M. Gaertler. Clustering. In U. Brandes and T. Erlebach, editors, *Network Analysis*, volume 3418 of *Lect. Not. Comput. Sci.*, pages 178–215. Springer Berlin / Heidelberg, 2005.
22. E.R. Gansner, Y. Koren, and S.C. North. Topological fisheye views for visualizing large graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 11(4):457–468, July-August 2005.
23. Michael R. Garey and David S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.
24. Robert Görke, Marco Gaertler, and Dorothea Wagner. Lunarvis - analytic visualizations of large graphs. In *Proceedings of the 15th international conference on Graph drawing*, GD'07, pages 352–364, Berlin, Heidelberg, 2008. Springer-Verlag.
25. S. Hachul and M. Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In J. Pach, editor, *Graph Drawing*, volume 3383 of *Lect. Not. Comput. Sci.*, pages 285–295. Springer Berlin / Heidelberg, 2005.
26. M. Halldórsson and J. Radhakrishnan. Greed is good: Approximating independent sets in sparse and bounded-degree graphs. *Algorithmica*, 18:145–163, 1997.
27. D. Harel and Y. Koren. A fast multi-scale method for drawing large graphs. *Journal of graph algorithms and applications*, (6):179—-202, 2002.
28. D. Harel and Y. Koren. Graph drawing by high-dimensional embedding. In *Graph Drawing (GD'02)*, Lect. Not. Comput. Sci., pages 207–219. Springer-Verlag, 2002.
29. Danny Holten and Jarke J. van Wijk. Force-directed edge bundling for graph visualization. In *Proceedings of the 11th Eurographics/IEEE-VGTC Symposium on Visualization*, pages 983–990, 2009.
30. Weidong Huang, P. Eades, and Seok-Hee Hong. A graph reading behavior: Geodesic-path tendency. In *Visualization Symposium, 2009. PacificVis '09. IEEE Pacific*, pages 137 –144, april 2009.
31. Y. Koren, L. Carmel, and D. Harel. Drawing huge graphs by algebraic multigrid optimization. *Multiscale Modeling and Simulation*, 1:645–673, 2003.
32. M. Li and P. M.B. Vitnyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 3 edition, 2008.
33. W. A. Mackaness and K. M. Beard. Use of graph theory to support map generalization. *Cartography and Geographic Information Science*, 20:210–221, 1993.
34. W. A. Mackaness, A.nne Ruas, and L. T. Sarjakoski, editors. *Generalisation of Geographic Information*. Cartographic Modelling and Applications. Elsevier B.V., 2007.
35. Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, 6(2):183–210, 1995.
36. A. Quigley and P. Eades. Fade: Graph drawing, clustering, and visual abstraction. In J. Marks, editor, *Graph Drawing*, volume 1984 of *Lect. Not. Comput. Sci.*, pages 77–80. Springer Berlin / Heidelberg, 2001.
37. D. Rafiei and S. Curial. Effectively visualizing large networks through sampling. In *Visualization, 2005. VIS 05. IEEE*, pages 375–382, October 2005.
38. A. Saalfeld. Map generalization as a graph drawing problem. In R. Tamassia and I. Tollis, editors, *Graph Drawing*, volume 894 of *Lect. Not. Comput. Sci.*, pages 444–451. Springer Berlin / Heidelberg, 1995.
39. Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '92, pages 83–91, New York, NY, USA, 1992. ACM.
40. A. Telea and O. Ersoy. Image-based edge bundles: Simplified visualization of large graphs. *Computer Graphics Forum*, 29(3):843–852, 2010.

A Benchmark Graphs

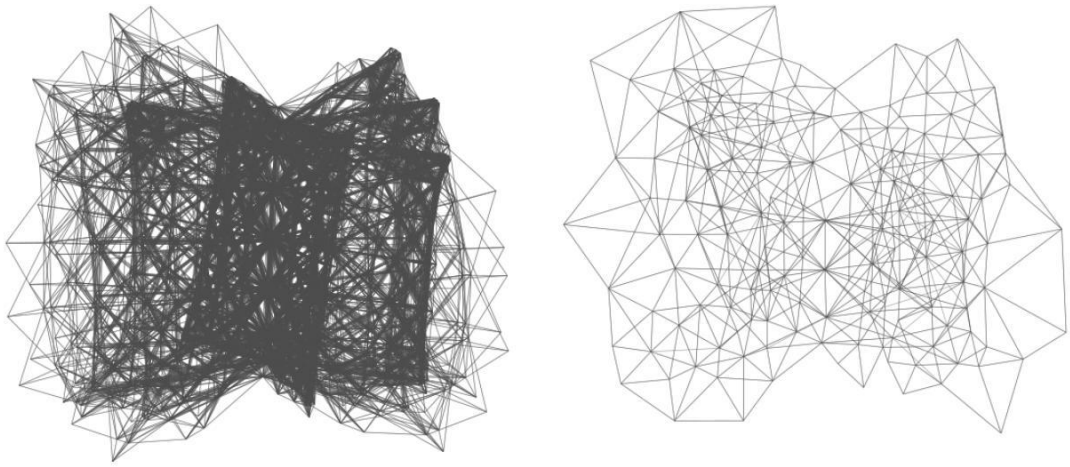
Table 1: Benchmark set of graphs used for our experiments, sorted according to number of vertices. All graphs, except those marked with $\star$, are from the University of Florida sparse matrix collection [14]; `clique-planar` is a planar graph with an implanted clique, `lunar-vis` is a LunarVis layout of the AS-Graph [24], `email` is a force-based layout of the email network of the faculty of informatics and `osm_berlin` and `osm_isleofman` are street networks extracted from the OpenStreetMap data [3].

i	name	n	m
1	bcspwr09	1036	3736
2	bcsstk08	1050	29156
3	bcsstk14	1072	12444
4	bcsstk26	1074	12960
5	can_1072	1133	10902
6	clique-planar*	1141	7465
7	bcsstk35	1242	10426
8	bcsstk36	1723	6511
9	bcsstk37	1733	22189
10	bodyy4	1806	63454
11	c-48	1821	52685
12	cti	1866	7076
13	ex3sta1	1919	32399
14	ford1	1922	30336
15	g7jac060sc	1960	11187
16	jan99jac060	1961	5156
17	jan99jac100sc	2363	7680
18	tandem_vtx	2423	11672
19	TF16	3345	19404
20	dwt_1242	6290	16466
21	email*	7920	31680
22	ex33	16558	149658
23	ex3	16782	678998
24	jagmesh8	16840	96464
25	aug3d	17233	74436
26	cep1	17546	121938
27	commanche_dual	18354	166080
28	conf5_4-8x8-05	18454	253350
29	lpl3	18728	101576
30	nemscem	23052	1143140
31	pds10	24300	69984
32	sstmodel	24494	51256
33	msc01050	25503	1140977
34	netz4504	30237	1450163
35	lunar-vis*	34758	432346
36	osm_berlin*	35460	406632
37	osm_isleofman*	41228	254364
38	plat1919	44514	201050
39	rajat02	49152	2064384
40	stufe	68908	431724
41	ukerbe1_dual	106675	248390



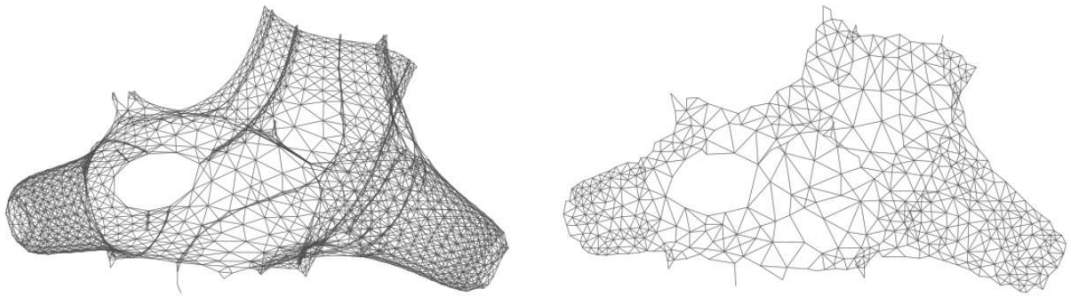
(a) original (n=16558, m=149658) (b) generalization (n=910, m=3520)

Fig. 21: Generalization of the graph `PDS10` from the University of Florida sparse matrix collection [14]



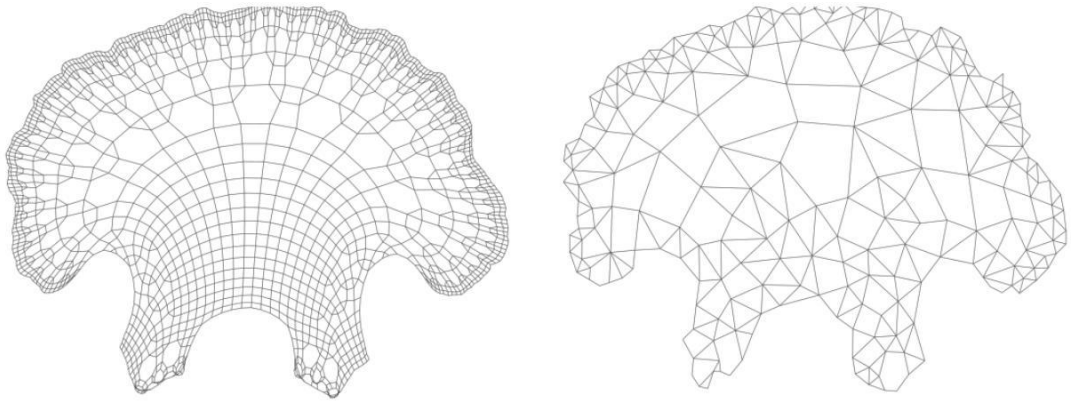
(a) original (n=1050, m=29156) (b) generalization (n=157, m=1110)

Fig. 22: Generalization of the graph `msc01050` from the University of Florida sparse matrix collection [14]



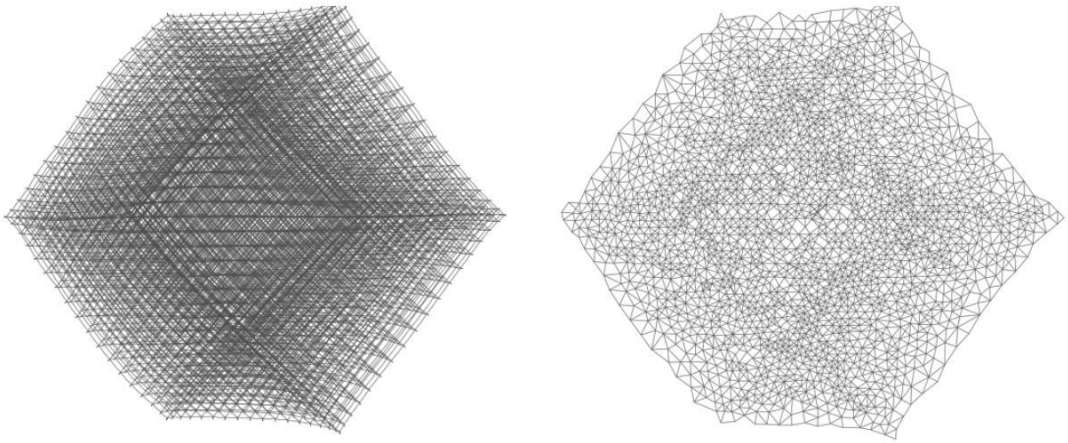
(a) original (n=1242, m=10426) (b) generalization (n=469, m=2608)

Fig. 23: Generalization of the graph `dwt_1242` from the University of Florida sparse matrix collection [14]



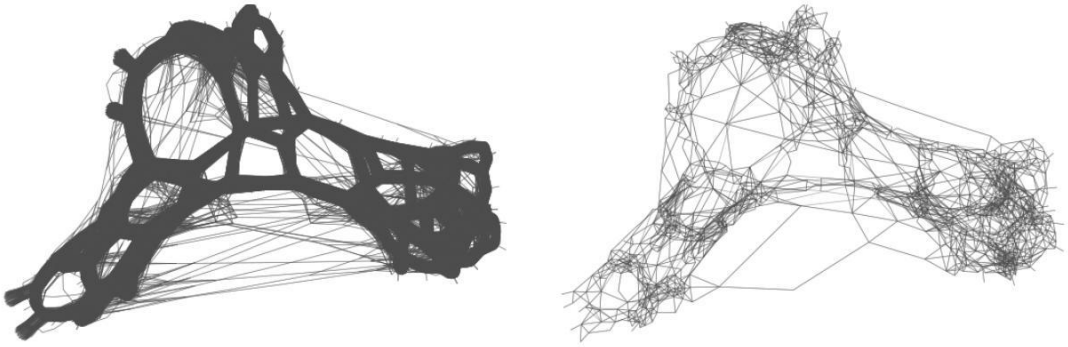
(a) original (n=1866, m=7076) (b) generalization (n=234, m=1062)

Fig. 24: Generalization of the graph `ukerbe1_dual` from the University of Florida sparse matrix collection [14]



(a) original (n=24300, m=69984) (b) generalization (n=2093, m=13546)

Fig. 25: Generalization of the graph `aug3d` from the University of Florida sparse matrix collection [14]



(a) original (n=44514, m=201050) (b) generalization (n=958, m=5763)

Fig. 26: Generalization of the graph `lpl3` from the University of Florida sparse matrix collection [14]