Distance Measures for Geometric Graphs

Sushovan Majhi*[1] and Carola Wenk†[2]

[1]School of Information, University of California, Berkeley, USA
[2]Department of Computer Science, Tulane University, New Orleans, USA

Abstract

A geometric graph is a combinatorial graph, endowed with a geometry that is inherited from its embedding in a Euclidean space. Formulation of a meaningful measure of (dis-)similarity in both the combinatorial and geometric structures of two such geometric graphs is a challenging problem in pattern recognition. We study two notions of distance measures for geometric graphs, called the geometric edit distance (GED) and geometric graph distance (GGD). While the former is based on the idea of editing one graph to transform it into the other graph, the latter is inspired by inexact matching of the graphs. For decades, both notions have been lending themselves well as measures of similarity between attributed graphs. If used without any modification, however, they fail to provide a meaningful distance measure for geometric graphs—even cease to be a metric. We have curated their associated cost functions for the context of geometric graphs. Alongside studying the metric properties of GED and GGD, we investigate how the two notions compare. We further our understanding of the computational aspects of GGD by showing that the distance is $\mathcal{NP}$-hard to compute, even if the graphs are planar and arbitrary cost coefficients are allowed.

1 Introduction

Graphs have been a widely accepted object for providing structural representation of patterns involving relational properties. One of the most important aspects of such representation is that the problem of pattern recognition becomes the problem of quantifying (dis-)similarity between a query graph and a model or prototype graph. The problem of defining a relevant distance measure for a class of graphs has been looked into for almost five decades now and has a myriad of applications including chemical structure matching, fingerprint matching, face identification, and symbol recognition. Depending on the class of graphs of interest and the area of application, several methods have been proposed. If the use case requires a perfect matching of two graphs, then the problem of graph isomorphism can be considered [5]; whereas, subgraph isomorphism can by applied for a perfect matching of parts of two graphs. These techniques are not, however, lenient with (sometimes minor) local and structural deformations of the two graphs. To address this issue, several alternative distance measures have been studied. We particularly investigate *edit distance* [6, 8] and *inexact matching distance* [3]. The former is based on partially matching two graphs through an inexact matching relation (Definition 2), while the latter makes use of elementary edit transformations (such as deletion, insertion, relabeling of vertices and edges), and the distance is defined as the minimum cost of transforming one graph to the other. Although these distance measures have been battle-proven for attributed graphs (i.e., combinatorial graphs with finite label sets), the formulations seem inadequate in providing meaningful similarity measures for geometric graphs.

A geometric graph belongs to a special class of attributed graphs having an embedding into a Euclidean space $\mathbb{R}^d$, where the vertex and edge attributes are inferred from the Euclidean locations of the vertices and Euclidean lengths of the edges, respectively. In the last decade, there has been a gain in practical applications involving comparison of geometric graphs. Examples include road-network or map comparison [1], detection of chemical structures using their spatial bonding geometry, etc. In addition, large datasets like [7] are being curated by pattern recognition and machine learning communities.

Despite a rich literature on the matching of attributed graphs and a fair count of algorithms benchmarked by both the database community and the pattern recognition community, most of the frameworks become untenable for matching geometric graphs. They remain oblivious to the spatial geometry such graphs are endowed with, consequently giving rise to very *artificial* measures of similarity for geometric graphs. This is not surprising at all—geometric graphs are a special class of labeled graphs after all! For a geometric graph, the significant differences include:

(i) Edge relabeling is not an independent edit operation, but vertex labels dictate the incident edge labels.

(ii) Vertex relabeling amounts to its translation to a different location in the ambient space, and additionally incurs the cost of relabeling of all its adjacent edges.

*smajhi@berkeley.edu
†cwenk@tulane.edu; partially supported by NSF grant CCF 2107434.

1.1 Our Contribution

We study two distance measures, the *geometric edit distance* (GED) and *geometric graph distance* (GGD), in order to provide a meaningful measure of similarity between two geometric graphs. For attributed graphs the corresponding distance measures are equivalent as shown in [2, Proposition 1]. In contrast, we show in Section 2.3 they are not equivalent for geometric graphs.

We mention here the contribution of [4] for introducing GGD as well as discussing different definitions of edit distance in the context of geometric graphs. The authors also prove certain complexity results for GGD, which we improve upon in this paper.

2 Two Distances for Geometric Graphs

A (finite) combinatorial graph $G = (V^G, E^G)$ is called a *geometric graph* of $\mathbb{R}^d$ if the vertex set $V^G \subset \mathbb{R}^d$ and the Euclidean straight-line segments $\left\{\overline{ab} \mid (a,b) \in E^G\right\}$ intersect (possibly) at their endpoints. We denote by $\mathcal{G}(\mathbb{R}^d)$ the set of all geometric graphs of $\mathbb{R}^d$. Two geometric graphs $G = (V^G, E^G)$ and $H = (V^H, E^H)$ are said to be *equal*, written $G = H$, if and only if $V^G = V^H$ and $E^G = E^H$. We make no distinction between a geometric graph $G = (V^G, E^G)$ and its *geometric realization* as a subset of $\mathbb{R}^d$; an edge $(u, v) \in E^G$ can be identified as the line-segment $\overline{uv}$ in $\mathbb{R}^d$, and its length by the Euclidean length $|\overline{uv}|$.

2.1 Geometric Edit Distance (GED)

Given two geometric graphs $G, H \in \mathcal{G}(\mathbb{R}^d)$, we transform G into H by applying a sequence of edit operations. The allowed edit operations and their costs are i) inserting (and deleting) a vertex costs nothing, ii) inserting (and deleting) an edge costs C_E times its length, and iii) translating a vertex costs C_V times the displacement of the vertex plus C_E times the total change in the length of *all* its incident edges. Throughout the paper, we assume that the coefficients C_V and C_E are positive constants. In order to denote a deleted vertex and a deleted edge, we introduce the *dummy vertex* ϵ_V and the *dummy edge* ϵ_E, respectively. While computing edit costs, we follow the convention that $|\epsilon_E| = 0$, $|a \pm \epsilon_V| = 0$ for any $a \in \mathbb{R}^d$, and $(u, v) = \epsilon_E$ if either $u = \epsilon_V$ or $v = \epsilon_V$.

An *edit path* from G to H is a sequence of allowed edit operations to transform G into H. Note that we do not require for an intermediate edit operation to yield a geometric graph. The set of all edits paths between $G, H \in \mathcal{G}(\mathbb{R}^d)$ is denoted as $\mathcal{P}(G, H)$. For any vertex $u \in V^G$ (resp. edge $e \in E^G$), we denote by $P(u)$ (resp. $P(e)$) the end result after its evolution under P. If P deletes the vertex u (resp. edge e), we write $P(v) = \epsilon_V$ (resp. $P(e) = \epsilon_E$). The cost, $\mathrm{Cost}(P)$, of an edit path P is defined to be the total cost of the individual edits. Then $\mathrm{GED}(G, H)$ is defined as cost of the *least* expensive edit path.

Definition 1 (Geometric Edit Distance)**.** *For geometric graphs $G, H \in \mathcal{G}(\mathbb{R}^d)$, their geometric edit distance is defined as*

$$\mathrm{GED}(G, H) \overset{def}{=} \inf_{P \in \mathcal{P}(G,H)} \mathrm{Cost}(P).$$

We prove that GED is, in fact, a metric in the space of geometric graphs without any isolated vertex. As also observed in [4], the following example demonstrates that the distance may not be attained by an edit path, unless an infinite number of edits are allowed: Consider $G, H \in \mathcal{G}(\mathbb{R}^2)$, where G has only one edge (u_1, u_2) and H has only one edge (v_1, v_2) as shown in Figure 1. For

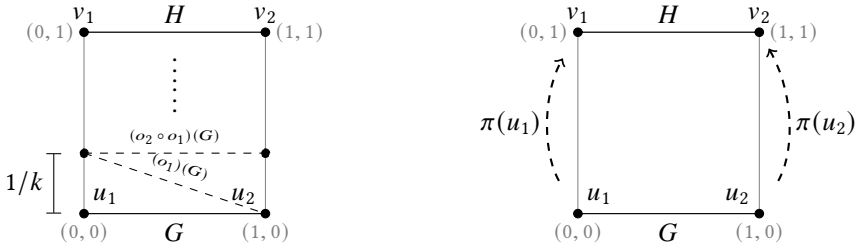

Figure 1: Left: the edit path P_k alternatively moves the left and right vertices of G by distance $1/k$. Consequently, $\mathrm{GED}(G, H) = 2C_V$. Right: The inexact matching π between G and H has been shown to attain the same distance for $\mathrm{GGD}(G, H)$.

any fixed $k \geq 1$, consider the edit path $P_k = \{o_i\}_{i=1}^{2k}$, where o_i translates the left vertex of G up by a distance $1/k$ and then o_{i+1} moves the right vertex by the same distance for any odd i. So, for any i

$$\mathrm{Cost}(o_i) = C_V \frac{1}{k} + C_E \left[\sqrt{(1/k)^2 + 1^2} - 1\right] = C_V \frac{1}{k} + C_E \frac{\frac{1}{k^2}}{\sqrt{\frac{1}{k^2} + 1} + 1}, \text{ and therefore}$$

$$\mathrm{GED}(G, H) \leq \mathrm{Cost}(P_k) = \sum_{i=1}^{2k} \mathrm{Cost}(o_i) = 2C_V + C_E \frac{\frac{2}{k}}{\sqrt{\frac{1}{k^2} + 1} + 1} \xrightarrow{k \to \infty} 2C_V.$$

Therefore, $\mathrm{GED}(G, H) = 2C_V$. However, there is no edit path that attains this cost.

2.2 Geometric Graph Distance (GGD)

The definition of GED is very intuitive but not at all suited for computational purposes. Firstly, there could be infinitely many locations a vertex is allowed to be translated to. Secondly, there are infinitely many edit paths between two graphs—even if the vertices are located on a finite grid. The infinite search space makes the computation of GED illusive. As a feasible alternative we study GGD. The definition is inspired by the concept of inexact matching first proposed in [3], and has been introduced for geometric graphs in [4]. We follow the notation of [3] in order to define it. We first define an *(inexact) matching*.

Definition 2 (Inexact Matching)**.** *Let $G, H \in \mathcal{G}(\mathbb{R}^d)$ be two geometric graphs. A relation $\pi \subseteq (V^G \cup \{\epsilon_V\}) \times (V^H \cup \{\epsilon_V\})$ is called an (inexact) matching if for any $u \in V^G$ (resp. $v \in V^H$) there is exactly one $v \in V^H \cup \{\epsilon_V\}$ (resp. $u \in V^G \cup \{\epsilon_V\}$) such that $(u, v) \in \pi$.*

The set of all matchings between graphs G, H is denoted by $\Pi(G, H)$. Intuitively speaking, a matching π is a relation that covers the vertex sets V^G, V^H exactly once. As a result, when restricted to V_E (resp. V^G), a matching π can be expressed as a map $\pi : V^G \to V^H \cup \{\epsilon_V\}$ (resp. $\pi^{-1} : V^H \to V^G \cup \{\epsilon_V\}$). In other words, when $(u, v) \in \pi$ and $u \neq \epsilon_V$ (resp. $v \neq \epsilon_V$), it is justified to write $\pi(u) = v$ (resp. $\pi^{-1}(v) = u$). It is evident from the definition that the induced map

$$\pi : \{u \in V^G \mid \pi(u) \neq \epsilon_V\} \to \{v \in V^H \mid \pi^{-1}(v) \neq \epsilon_V\}$$

is a bijection. Additionally for edges $e = (u_1, u_2) \in E^G$ and $f = (v_1, v_2) \in E^H$, we introduce the short-hand $\pi(e) := (\pi(u_1), \pi(u_2))$ and $\pi^{-1}(f) := (\pi^{-1}(v_1), \pi^{-1}(v_2))$.

Another perspective of π is discerned when viewed as a matching between portions of G and H, (possibly) after applying some edits on the two graphs. For example, $\pi(u) = \epsilon_V$ (resp. $\pi^{-1}(v) = \epsilon_V$) encodes deletion of the vertex u from G (resp. v from H), whereas $\pi(e) = \epsilon_E$ (resp. $\pi^{-1}(f) = \epsilon_E$) encodes deletion of the edge e from G (resp. f from H). Once the above deletion operations have been performed on the graphs, the resulting subgraphs of G and H become isomorphic, which are finally matched by translating the remaining vertices u to $\pi(u)$. Now, the cost of the matching π is defined as the total cost for all of these operations:

$$\mathrm{Cost}(\pi) = \underbrace{\sum_{\substack{u \in V^G \\ \pi(u) \neq \epsilon_V}} C_V |u - \pi(u)|}_{\text{vertex translations}} + \underbrace{\sum_{\substack{e \in E^G \\ \pi(e) \neq \epsilon_E}} C_E \big||e| - |\pi(e)|\big|}_{\text{edge translations}} + \underbrace{\sum_{\substack{e \in E^G \\ \pi(e) = \epsilon_E}} C_E |e|}_{\text{edge deletions}} + \underbrace{\sum_{\substack{f \in E^H \\ \pi^{-1}(f) = \epsilon_E}} C_E |f|}_{\text{edge deletions}}. \tag{1}$$

Definition 3 (GGD)**.** *For geometric graphs $G, H \in \mathcal{G}(\mathbb{R}^d)$, their geometric graph distance is defined as*

$$\mathrm{GGD}(G, H) \stackrel{def}{=} \min_{\pi \in \Pi(G,H)} \mathrm{Cost}(\pi).$$

2.3 Comparing GED and GGD

As we now have the two notions of distances under our belts, the question of how they compare arises naturally. We have already pointed out that the the analogous notions for attributed graphs yield equivalent distances. To our surprise, they are not generally equal for geometric graphs, as we demonstrate in the following example. Take two graphs $G, H \in \mathcal{G}(\mathbb{R})$, each having three vertices and two edges connecting them as shown in Figure 2. More generally, in Proposition 4 we prove that $\mathrm{GGD}(G, H) \leq \mathrm{GED}(G, H)$.

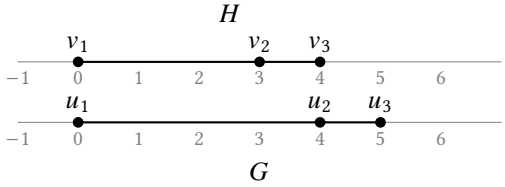

Figure 2: If $C_V = C_E = 1$, then $\mathrm{GGD}(G, H) = 3$ and $\mathrm{GED}(G, H) = 5$. For arbitrary C_E and C_V, the configuration can be easily adjusted so that $\mathrm{GGD}(G, H) < \mathrm{GED}(G, H)$.

Proposition 4. *For any geometric graphs $G, H \in \mathcal{G}(\mathbb{R}^d)$, we have*

$$\mathrm{GGD}(G, H) \leq \mathrm{GED}(G, H).$$

Proof. Take an arbitrary edit path $P \in \mathcal{P}(G, H)$. Let us define the matching $\pi \in \Pi(G, H)$ such that

$$\pi \stackrel{\text{def}}{=} \{(u, P(u)) \mid u \in V^G\} \cup \{(P^{-1}(v), v) \mid v \in V^H\}.$$

This definition of π implies that $P(u) = \pi(u)$ for all $u \in V^G$, $P(e) = \pi(e)$ for all $e \in E^G$, and $P^{-1}(f) = \pi^{-1}(f)$ for all $f \in E^H$. Also, it follows from the definitions of the costs that $\mathrm{Cost}(\pi) \leq \mathrm{Cost}(P)$. The definition of $\mathrm{GGD}(G, H)$ then implies that

$$\mathrm{GGD}(G, H) \leq \mathrm{Cost}(\pi) \leq \mathrm{Cost}(P).$$

Since P is chosen arbitrarily, the definition of $\mathrm{GED}(G, H)$ then implies the result. □

3 Computational Complexity

In this section, we discuss the computational aspects of the GGD. We define the problem as follows.

Definition 5 (Problem GGD)**.** *Given geometric graphs $G, H \in \mathcal{G}(\mathbb{R}^d)$ and $\tau \geq 0$, is there a matching $\pi \in \Pi(G, H)$ such that* $\text{Cost}(\pi) \leq \tau$*?*

In [4], the authors show Problem GGD is $\mathcal{NP}$-hard for non-planar graphs, when arbitrary cost coefficients C_V, C_E are allowed. For planar graphs, however, its $\mathcal{NP}$-hardness is proved under the very strict condition that $C_V << C_E$. We prove a stronger result that the problem is $\mathcal{NP}$-hard, even if the graphs are planar and arbitrary C_V, C_E are allowed. Our reduction is from the well-known 3-PARTITION problem. For non-planar graphs, we conjecture that for any $\alpha > 1$, an α-approximation is also $\mathcal{NP}$-hard, i.e, Problem GGD is generally $\mathcal{APX}$-hard.

4 Discussions and Future Work

We have studied two notions for a similarity measure between geometric graphs. In addition to the metric properties of GED and GGD, we also establish that the former is at least as large as the latter. However, it remains unclear if GED can be bounded from above by a constant multiple of GGD. The hardness of GGD for planar graphs, also provokes the question of the hardness of its polynomial-time approximation. Although we conjecture that the distance is even $\mathcal{NP}$-hard to approximate for dimension $d \geq 3$, for planar graphs the possibility of approximating the GGD within a reasonable factor can be further investigated.

Acknowledgments

The authors thank Erfan Hosseini, Erin Chambers, and Elizabeth Munch for fruitful discussions.

References

[1] M. Ahmed, S. Karagiorgou, D. Pfoser, and C. Wenk. *Map Construction Algorithms*. Springer International Publishing, first edition, 2015.

[2] Sébastien Bougleux, Luc Brun, Vincenzo Carletti, Pasquale Foggia, Benoit Gaüzère, and Mario Vento. Graph edit distance as a quadratic assignment problem. *Pattern Recognition Letters*, 87:38–46, 2017.

[3] H Bunke and G Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1(4):245–253, May 1983.

[4] Otfried Cheong, Joachim Gudmundsson, Hyo-Sil Kim, Daria Schymura, and Fabian Stehn. Measuring the Similarity of Geometric Graphs. In Jan Vahrenhold, editor, *Experimental Algorithms*, volume 5526, pages 101–112. Springer, 2009.

[5] D. G. Corneil and C. C. Gotlieb. An efficient algorithm for graph isomorphism. *J. ACM*, 17(1):51–64, 1970.

[6] D. Justice and A. Hero. A binary linear programming formulation of the graph edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1200–1214, August 2006.

[7] Kaspar Riesen and Horst Bunke. IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning. In *Structural, Syntactic, and Statistical Pattern Recognition*, volume 5342, pages 287–297. Springer, 2008.

[8] Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(3):353–362, May 1983.