Metric Tree-Like Structures in Real-World Networks: An Empirical Study

Muad Abu-Ata and Feodor F. Dragan
Department of Computer Science, Algorithmic Research Laboratory, Kent State University, Kent, Ohio 44242

Based on solid theoretical foundations, we present strong evidence that a number of real-world networks, taken from different domains (such as Internet measurements, biological data, web graphs, and social and collaboration networks) exhibit tree-like structures from a metric point of view. We investigate a few graph parameters, namely, the tree-distortion and the tree-stretch, the tree-length and the tree-breadth, Gromov's hyperbolicity, the cluster-diameter and the cluster-radius in a layering partition of a graph; such parameters capture and quantify this phenomenon of being metrically close to a tree. By bringing all those parameters together, we provide efficient means for detecting such metric tree-like structures in large-scale networks. We also show how such structures can be used. For example, they are helpful in efficient and compact encoding of approximate distance and almost shortest path information and in quick and accurate estimation of diameters and radii of those networks. Estimating the diameter and estimating the radius of a graph (or distances between arbitrary vertices) are fundamental primitives in many network and graph mining algorithms. © 2015 Wiley Periodicals, Inc. NETWORKS, Vol. 67(1), 49–68 2016

Keywords: large-scale networks; metric tree-like structures; hyperbolicity; tree-stretch; tree-distortion; tree-length; tree-breadth; layering partition; graph algorithms; approximation algorithms

1. INTRODUCTION

Large networks are everywhere. Can we understand their structure and exploit it? For example, understanding key structural properties of large-scale data networks is crucial for analyzing and optimizing their performance, as well as improving their reliability and security [59]. In prior empirical and theoretical studies, researchers have mainly focused on features such as small world phenomenon, power law degree distribution, navigability, and high clustering coefficients (see [8, 9, 12, 27, 40, 53, 54, 57, 70]). Those nice features were observed in many real-world complex networks and graphs arising in Internet applications, in the biological and social sciences, and in chemistry and physics. Although those features are interesting and important, as noted in [59], the impact of intrinsic geometric and topological features of large-scale data networks on performance, reliability, and security is of much greater importance (see also [10, 64]).

Recently, a few papers have explored geometric characteristics of real-world networks, namely the *hyperbolicity* (sometimes called also the *global curvature*) of the network (see, e.g., [4, 21, 29, 51, 59, 67]). It was shown that a number of data networks, including Internet application networks, web networks, collaboration networks, social networks, and others, have small hyperbolicity. It has been suggested [59] that the property, observed in real-world networks, in which traffic between nodes tends to go through a relatively small core of the network, as if the shortest path between them is curved inwards, may be due to global curvature of the network. Furthermore, Kennedy et al. [51] proposes that "hyperbolicity in conjunction with other local characteristics of networks, such as the degree distribution and clustering coefficients, provide a more complete unifying picture of networks, and helps classify in a parsimonious way what is otherwise a bewildering and complex array of features and characteristics specific to each natural and man-made network."

The hyperbolicity of a graph/network can be viewed as a measure of how close a graph is to a tree metrically; the smaller the hyperbolicity of a graph, the closer it is metrically to a tree. Generally, it is known [65] that a graph G is metrically a tree if and only if G is a block graph, that is, each biconnected component of G is a complete graph. As many real-world networks have a highly connected (dense) core with "whiskers" or "tendrils" that are connected by short paths through the core [57, 67] (i.e., those networks can be viewed as dense subgraphs and subtrees glued together), it is natural to analyze how close they are to trees (or, equivalently, to block graphs) metrically. Recent empirical results on hyperbolicity [4, 21, 29, 51, 59, 67] suggest that many real-world complex networks and graphs may possess tree-like structures from a metric point of view.

In this article, we substantiate this claim through analysis of a collection of real data networks. We investigate

Received June 2014; accepted July 2015
Correspondence to: F. F. Dragan; e-mail: dragan@cs.kent.edu
DOI 10.1002/net.21631
Published online 25 August 2015 in Wiley Online Library (wileyonlinelibrary.com).

a few more, recently introduced graph parameters, namely, *tree-distortion* and *tree-stretch, tree-length* and *tree-breadth*, Gromov's *hyperbolicity, cluster-diameter* and *cluster-radius* in a *layering partition* of a graph. All these parameters attempt to capture and quantify this phenomenon of being metrically close to a tree and can be used to measure metric tree-likeness of a real-world network. Recent advances in theory (see appropriate sections for details) allow us to calculate or accurately estimate those parameters for sufficiently large networks. By examining topologies of numerous publicly available networks, we demonstrate the existence of metric tree-like structures in a wide range of large-scale networks, from communication networks to various forms of social and biological networks.

Throughout this article, we discuss these parameters and recently established relationships between them for unweighted and undirected graphs. It turns out that all these parameters are at most constant or logarithmic factors apart from each other. Hence, a constant bound on one of them translates in a constant or almost constant bound on another. We say that a graph with n vertices and m edges *has a tree-like structure from a metric point of view* (equivalently, *is metrically tree-like*) if any one of those parameters is a small constant, it is not larger than $\log_2(n+m)$.

Recently, Adcock et al. [4] pointed out that "although large informatics graphs such as social and information networks are often thought of as having hierarchical or tree-like structure, this assumption is rarely tested, and it has proven difficult to exploit this idea in practice; ... it is not clear whether such structure can be exploited for improved graph mining and machine learning ...."

In this article, by bringing all those parameters together, we provide efficient means for detecting such metric tree-like structures in large-scale networks. We also show how such structures can be used. For example, they are helpful in efficient and compact encoding of approximate distance and almost shortest path information and in quick and accurate estimation of diameters and radii of those networks. Estimating the diameter and estimating the radius of a graph (or distances between arbitrary vertices) are fundamental primitives in many network and graph mining algorithms.

Graphs that are metrically tree-like have many algorithmic advantages. They allow for efficient approximate solutions for a number of optimization problems. For example, they admit a PTAS for the Traveling Salesman Problem [56], have an efficient approximate solution for the problem of covering and packing by balls [26], admit additive sparse spanners [24, 33] and collective additive tree-spanners [36], enjoy efficient and compact approximate distance [24, 42] and routing [24, 32] labeling schemes, and have efficient algorithms for quick and accurate estimations of diameters and radii [23]. We elaborate more on these results in appropriate sections.

For the first time, metric parameters such as tree-length and tree-breadth, tree-distortion and tree-stretch, cluster-diameter and cluster-radius are examined, and the algorithmic advantages of having those parameters bounded by small constants are discussed for such a wide range of large-scale networks.

This article is structured as follows. In Section 2, we give notation and basic notions used in the article. In Section 3, we describe our graph datasets. The next four sections are devoted to analysis of corresponding parameters measuring metric tree-likeness of our graph datasets: cluster-diameter and cluster-radius in Section 4; hyperbolicity in Section 5; tree-distortion in Section 6; tree-breadth, tree-length, and tree-stretch in Section 7. In each section, we first give theoretical background on the parameter(s) and then present our experimental results. Additionally, an overview of implications of those results is provided. In Section 8, we further discuss the algorithmic advantages when a graph is metrically tree-like. Finally, in Section 9, we give some concluding remarks.

2. NOTATION AND BASIC NOTIONS

All graphs in this article are connected, finite, unweighted, undirected, loopless, and without multiple edges. For a graph $G = (V, E)$, we use n and $|V|$ interchangeably to denote the number of vertices in G. Also, we use m and $|E|$ to denote the number of edges. The *length of a path* from a vertex v to a vertex u is the number of edges in the path. The *distance* $d_G(u, v)$ between vertices u and v is the length of the shortest path connecting u and v in G. The *ball* $B_r(s, G)$ of a graph G centered at vertex $s \in V$ and with radius r is the set of all vertices with distance no more than r from s (i.e., $B_r(s, G) = \{v \in V : d_G(v, s) \leq r\}$). We omit the graph name G and write $B_r(s)$ if the context is about only one graph.

The *diameter* $\mathrm{diam}(G)$ of a graph $G = (V, E)$ is the largest distance between a pair of vertices in G, that is, $\mathrm{diam}(G) = \max_{u,v \in V} d_G(u, v)$. The *eccentricity* of a vertex v, denoted by $\mathrm{ecc}(v)$, is the largest distance from that vertex v to any other vertex, that is, $\mathrm{ecc}(v) = \max_{u \in V} d_G(v, u)$. The *radius* $\mathrm{rad}(G)$ of a graph $G = (V, E)$ is the minimum eccentricity of a vertex in G, that is, $\mathrm{rad}(G) = \min_{v \in V} \max_{u \in V} d_G(v, u)$. The *center* $C(G) = \{c \in V : \mathrm{ecc}(c) = \mathrm{rad}(G)\}$ of a graph $G = (V, E)$ is the set of vertices with minimum eccentricity.

Definitions of graph parameters measuring metric tree-likeness of a graph, as well as notions and notation local to a section, are given in appropriate sections.

3. DATASETS

Our datasets come from different domains such as Internet measurements, biological datasets, web graphs, social, and collaboration networks. Table 1 shows basic statistics of our graph datasets. Each graph represents the largest connected component of the original graph as some datasets consist of one large connected component and many very small ones.

3.1. Biological Networks

PPI. This is a protein–protein interaction network involving the yeast *Saccharomyces Cerevisiae* [49]. Each vertex

TABLE 1. Graph datasets and their parameters: number of vertices, number of edges, diameter, radius

$G = (V, E)$	$\|V\|$	$\|E\|$	diam(G)	rad(G)
PPI [49]	1,458	1,948	19	11
Yeast [15]	2,224	6,609	11	6
DutchElite [30]	3,621	4,311	22	12
EPA [1]	4,253	8,953	10	6
EVA [60]	4,475	4,664	18	10
California [52]	5,925	15,770	13	7
Erdös [11]	6,927	11,850	4	2
Routeview [2]	10,515	21,455	10	5
Homo [68]	16,711	115,406	10	5
AS_CAIDA_20071105 [19]	26475	53,381	17	9
Dimes 3/2010 [66]	26,424	90,267	8	4
Aqualab 12/2007–09/2008 [20]	31,845	143,383	9	5
AS_CAIDA_20120601 [17]	41,203	121,309	10	5
itdk0304 [18]	190,914	607,610	26	14
DBLB-coauth [72]	317,080	1,049,866	23	12
Amazon [72]	334,863	925,872	47	24

represents a protein with an edge representing an interaction between two proteins. Self loops have been removed from the original dataset. The dataset has been analyzed and described in [49].

Yeast. This is a protein–protein interaction network involving budding yeast [15]. Each vertex represents a protein with an edge representing an interaction between two proteins. Self loops have been removed from the original dataset. The dataset has been analyzed and described in [15].

Homo. This is a dataset of protein and genetic interactions in *Homo Sapiens* (Humans) [68]. Each vertex represents a protein or a gene. An edge represents an interaction between two proteins/genes. Parallel edges, representing different resources for an interaction, have been removed. The dataset is obtained from BioGRID, a freely accessible database/repositiory of physical and genetic interactions available at http://www.thebiogrid.org. The dataset has been analyzed and described in [68].

3.2. *Social and Collaboration Networks*

DutchElite. This is data on the administrative elite in the Netherlands collected and analyzed by De Volkskrant and Wouter de Nooy [30]. It is a 2-mode network data representing membership in the administrative and organization bodies in the Netherlands in 2006. A vertex represents either a person or an organization. An edge exists between two vertices if the person vertex belongs to the organization vertex.

EVA. This is a network of interconnections between corporations where an edge exists between two companies (vertices) if one of them is the owner of the other company [60].

Erdös. This is a collaboration network of mathematician Paul Erdös [11]. Each vertex represents an author with an edge representing a paper co-authorship between two authors.

DBLB-coauth. This is a co-authorship network within the DBLP Computer Science bibliography [72]. Vertices of the network represent authors with edges connecting two authors if they published at least one paper together.

3.3. *Web Graphs*

EPA. This dataset represents pages linking to www.epa.gov obtained from Jon Kleinberg's web page, http://www.cs.cornell.edu/courses/cs685/2002fa/ [1]. The pages were constructed by expanding a 200-page response set to a search engine query, as in the hub/authority algorithm. These data were collected some time back, so a number of the links may not exist anymore. The vertices of this graph dataset represent web pages with edges representing links. The graph was originally directed. We ignore the direction of edges to get an undirected graph version of the dataset.

California. This graph dataset was also constructed by expanding a 200-page response set to a search engine query "California," as in the hub/authority algorithm [52]. The dataset was obtained from Jon Kleinberg's page, http://www.cs.cornell.edu/courses/cs685/2002fa/. The vertices of this graph dataset represent web pages with edges representing links between them. The graph was originally directed. We ignore the direction of edges to obtain an undirected graph version of the dataset.

3.4. *Internet Measurements Networks*

Routeview. This is an autonomous system (AS) graph obtained by the University of Oregon Route Views project using looking glass data and routing registry [2]. A vertex in the dataset represents an AS with an edge linking two vertices if there is at least one physical link between them.

AS_CAIDA. These are datasets of the Internet AS relationships derived from BGP table snapshots taken at 24-h intervals over a 5-day period by CAIDA [17, 19]. The AS relationships available are customer-provider (and provider-customer, in the opposite direction), peer-to-peer, and sibling-to-sibling.

Dimes 3/2010. This is an AS relationship graph of the Internet obtained from Dimes [66]. The Dimes project performs traceroutes and pings from volunteer agents (about 1,000 agent computers) to infer AS relationships. A weekly AS snapshot is available. The dataset *Dimes 3/2010* represents a snapshot aggregated over the month of March, 2010. It provides the set of AS level vertices and edges that were found in that month and were seen at least twice.

Aqualab. Peer-to-peer clients are used to collect traceroute paths which are used to infer AS interconnections [20]. Probes were made between December 2007 and September 2008 from approximately 992,000 P2P users in 3,700 ASes.

Itdk. This is an Internet router-level graph where each vertex represents a router with an edge between two vertices if there is a link between the corresponding routers [18]. The dataset snapshot is computed from ITDK0304 skitter and iffinder measurements. The dataset is provided by CAIDA for April 2003 (see http://www.caida.org/data/active/internet-topology-data-kit).

3.5. Information Network

Amazon. This is an Amazon product co-purchasing network [72]. The vertices of the network represent products purchased from the Amazon website and the edges link “commonly/frequently” co-purchased products.

4. LAYERING PARTITION, ITS CLUSTER-DIAMETER AND CLUSTER-RADIUS

Layering partition is a graph decomposition procedure introduced in [13, 22] and used in [7, 13, 22, 25] for embedding graph metrics into trees. It provides a central tool in our investigation.

A *layering* of a graph $G = (V, E)$ with respect to a start vertex s is the decomposition of V into the $r + 1$ layers (spheres) $L^i = \{u \in V : d_G(s, u) = i\}, i = 0, 1, \ldots, r$. A *layering partition* $\mathcal{LP}(G, s) = \left\{L^i_1, \ldots, L^i_{p_i} : i = 0, 1, \ldots, r\right\}$ of G is a partition of each layer L^i into clusters $L^i_1, \ldots, L^i_{p_i}$ such that two vertices $u, v \in L^i$ belong to the same cluster L^i_j if and only if they can be connected by a path outside the ball $B_{i-1}(s)$ of radius $i - 1$ centered at s. Here, p_i is the number of clusters in layer i. See Figure 1 for an illustration. A layering partition of a graph can be constructed in $O(n + m)$ time (see [22]).

A *layering tree* $\Gamma(G, s)$ of a graph G with respect to a layering partition $\mathcal{LP}(G, s)$ is the graph whose nodes are the clusters of $\mathcal{LP}(G, s)$ and where two nodes $C = L^i_j$ and $C' = L^{i'}_{j'}$ are adjacent in $\Gamma(G, s)$ if and only if there exist a vertex $u \in C$ and a vertex $v \in C'$ such that $uv \in E$. It was shown in [13] that the graph $\Gamma(G, s)$ is always a tree and, given a start vertex s, can be constructed in $O(n + m)$ time [22]. Note that, for a fixed start vertex $s \in V$, the layering partition $\mathcal{LP}(G, s)$ of G and its tree $\Gamma(G, s)$ are unique.

The cluster-diameter $\Delta_s(G)$ of layering partition $\mathcal{LP}(G, s)$ with respect to vertex s is the largest diameter of a cluster in $\mathcal{LP}(G, s)$, that is, $\mathcal{LP}(G, s)$. The cluster-diameter $\Delta(G)$ of a graph G is the minimum cluster-diameter over all layering partitions of G, that is $\Delta(G) = \min_{s \in V} \Delta_s(G)$.

The cluster-radius $R_s(G)$ of layering partition $\mathcal{LP}(G, s)$ with respect to a vertex s is the smallest number r such that for any cluster $C \in \mathcal{LP}(G, s)$ there is a vertex $v \in V$ with $C \subseteq B_r(v)$. The cluster-radius $R(G)$ of a graph G is the minimum cluster-radius over all layering partitions of G, that is, $R(G) = \min_{s \in V} R_s(G)$.

Clearly, in view of tree $\Gamma(G, s)$ of G, the smaller parameters $\Delta_s(G)$ and $R_s(G)$, the closer graph G is to a tree metrically.

Finding the cluster-diameter $\Delta_s(G)$ and the cluster-radius $R_s(G)$ for a given layering partition $\mathcal{LP}(G, s)$ of a graph G requires $O(nm)$ time,[1] although the construction of the layering partition $\mathcal{LP}(G, s)$ itself, for a given vertex s, takes only $O(n + m)$ time. As the diameter of any set is at least its radius and at most twice its radius, we have the following inequality:

$$R_s(G) \leq \Delta_s(G) \leq 2R_s(G).$$

In Table 2, we show empirical results on layering partitions obtained for datasets described in Section 3. For each graph dataset $G = (V, E)$, we randomly selected a start vertex s and built a layering partition $\mathcal{LP}(G, s)$ of G with respect to s. For each dataset, Table 2 shows the cluster-diameter $\Delta_s(G)$, the number of clusters in a layering partition $\mathcal{LP}(G, s)$ and the average diameter of clusters in $\mathcal{LP}(G, s)$. It turns out that all graph datasets have small average diameter of clusters. Most clusters have diameter 0 or 1, that is, they are essentially cliques (i.e., complete subgraphs) of G. For most datasets, more than 95% of the clusters are singletons or cliques with two or more vertices. Note that, in a graph, a cluster of radius 0 is a single vertex cluster and is an articulation point of the graph.

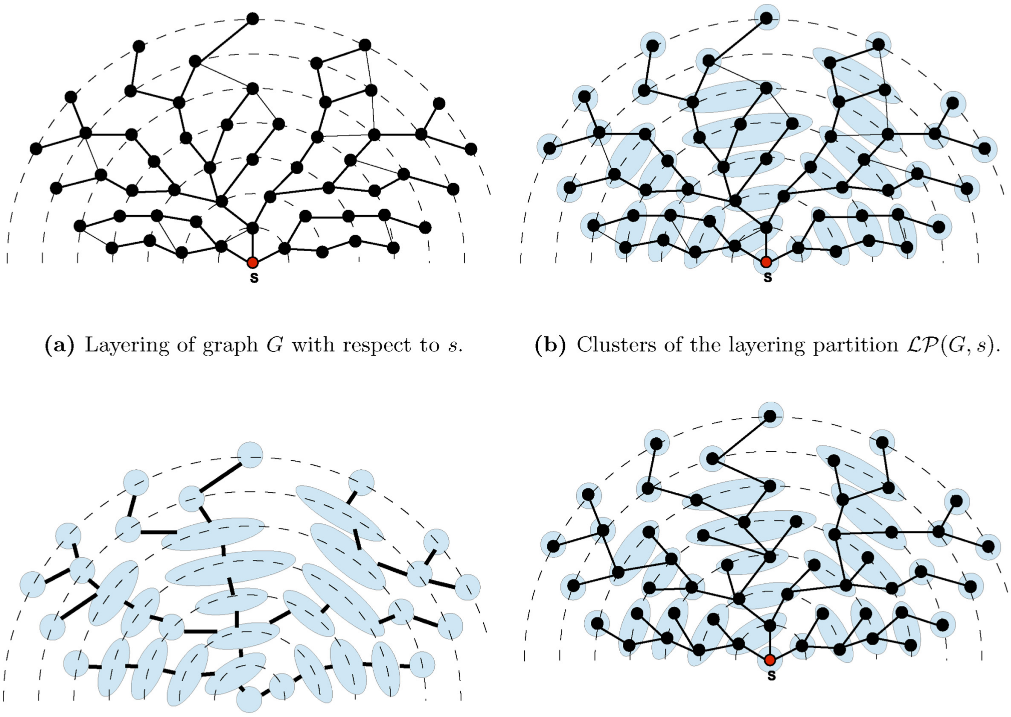
To have a better picture of the overall distribution of diameters of clusters, we show in Table 3 the frequencies of diameters of clusters for three sample datasets: PPI, Yeast, and AS_CAIDA_20071105. It is interesting to note that, in all datasets, the clusters with large diameters induce a connected subtree in the tree $\Gamma(G, s)$. For example, in PPI, the cluster with diameter 8 is adjacent in $\Gamma(G, s)$ to all clusters with diameters 6 and 5. This may indicate that all those clusters are part of the well-connected network core.

Most of the graph parameters discussed in this article could be related to a special tree H introduced in [25] and produced from a layering partition of a graph G.

Canonical tree **H**: A tree $H = (V, F)$ of a graph $G = (V, E)$, called a canonical tree of G, is constructed from a layering partition $\mathcal{LP}(G, s)$ of G by identifying for each cluster $C = L^i_j \in \mathcal{LP}(G, s)$ an arbitrary vertex $x_C \in L_{i-1}$ which has a neighbor in $C = L^i_j$ and by making x_C adjacent in H with all vertices $v \in C$ (see Fig. 1d for an illustration). Vertex x_C is called the support vertex for cluster $C = L^i_j$. It was shown in [25] that the tree H for a graph G can be constructed in $O(n + m)$ time.

The following result [25] relates the cluster-diameter of a layering partition of G to embedability of graph G into the tree H.

[1] The parameters $\Delta(G)$ and $R(G)$ can also be computed in total $O(nm)$ time for any graph G.



(a) Layering of graph G with respect to s.

(b) Clusters of the layering partition $\mathcal{LP}(G,s)$.

(c) Layering tree $\Gamma(G,s)$.

(d) Canonical tree H obtained from the layering partition.

FIG. 1. Layering partition and associated constructs. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

TABLE 2. Layering partitions of the datasets and their parameters. $\Delta_s(G)$ is the largest diameter of a cluster in $\mathcal{LP}(G,s)$, where s is a randomly selected start vertex. For all datasets, the average diameter of a cluster is between 0 and 1. For most datasets, more than 95% of the clusters are singletons or cliques with two or more vertices

$G=(V,E)$	$\|V\|$	diam(G)	# of clusters in $\mathcal{LP}(G,s)$	Cluster-diameter $\Delta_s(G)$	Average diameter of clusters in $\mathcal{LP}(G,s)$	% of clusters having diameter 0 (singletons)	% of clusters having diameter 1 (cliques)
PPI	1,458	19	1,017	8	0.12	94.99%	2.65%
Yeast	2,224	11	1,838	6	0.12	94.60%	1.74%
DutchElite	3,621	22	2,934	10	0.07	98.02%	0%
EPA	4,253	10	2,523	6	0.07	96.12%	2.46%
EVA	4,475	18	4,266	9	0.03	98.57%	0.63%
California	5,925	13	2,939	8	0.09	94.86%	2.28%
Erdös	6,927	4	6,288	4	0.00	99.95%	0.02%
Routeview	10,515	10	6,702	6	0.06	96.08%	2.37%
Homo	16,711	10	6,817	5	0.03	97.64%	1.61%
AS_CAIDA_20071105	26,475	17	17,067	6	0.06	96.44%	2.12%
Dimes 3/2010	26,424	8	16,065	4	0.06	96.27%	2.27%
Aqualab 12/2007–09/2008	31,845	9	16,287	6	0.06	96.25%	2.33%
AS_CAIDA_20120601	41,203	10	26,562	6	0.06	96.58%	2.0%
itdk0304	190,914	26	89,856	11	0.27	85.98%	5.40%
DBLB-coauth	317,080	23	99,828	11	0.45	64.98%	28.0%
Amazon	334,863	47	72,278	21	0.49	75.03%	11.02%

Proposition 1 ([25]). *For every graph $G=(V,E)$ and any vertex s of G,*

$$\forall x,y \in V,\ d_H(x,y)-2 \le d_G(x,y) \le d_H(x,y)+\Delta_s(G).$$

The above proposition shows that the distortion of embedding of a graph G into tree H is additively bounded by $\Delta_s(G)$, the largest diameter of a cluster in a layering partition of G. This result confirms that the smaller the cluster-diameter $\Delta_s(G)$ (cluster-radius $R_s(G)$) of G, the closer graph G is to a tree metric. Note that trees have cluster-diameter and cluster-radius equal to 0. Results similar to Proposition 1 were first used in [13] to embed a chordal graph into a tree with an additive distortion of at most 2 and in [22] to embed a k-chordal graph into a tree with an additive distortion of at most $k/2+2$. In [25], Proposition 1 was used to obtain a 6-approximation

TABLE 3. Frequency of diameters of clusters in layering partition $\mathcal{LP}(G, s)$ (three datasets)

Diameter of a cluster	Frequency	Relative frequency
(a) PPI		
0	966	0.9499
1	21	0.0206
2	14	0.0138
3	5	0.0049
4	5	0.0049
5	1	0.0001
6	4	0.0039
7	0	0
8	1	0.0001
(b) Yeast		
0	981	0.946
1	18	0.0174
2	23	0.0223
3	6	0.0058
4	5	0.0048
5	2	0.0019
6	2	0.0019
(c) AS_CAIDA_20071105		
0	16,459	0.9644
1	361	0.0216
2	174	0.0102
3	46	0.0027
4	21	0.0012
5	4	0.0002
6	2	0.0001

algorithm for the problem of optimal noncontractive embedding of an unweighted graph metric into a weighted tree metric. For every *chordal graph G* (a graph whose largest induced cycles have length 3), $\Delta_s(G) \leq 3$ and $R_s(G) \leq 2$ hold [13]. For every k-chordal graph G (a graph whose largest induced cycles have length k), $\Delta_s(G) \leq k/2 + 2$ holds [22]. For every graph G, embeddable noncontractively into a (weighted) tree with multiplication distortion α, $\Delta_s(G) \leq 3\alpha$ holds [25]. See Section 6 for more on this topic.

5. HYPERBOLICITY

δ-Hyperbolic metric spaces have been defined by Gromov [46] in 1987 via a simple four-point condition: for any four points u, v, w, x, the two larger of the distance sums $d(u, v) + d(w, x), d(u, w) + d(v, x), d(u, x) + d(v, w)$ differ by at most 2δ. They play an important role in geometric group theory and in the geometry of negatively curved spaces, and have recently become of interest in several domains of computer science, including algorithms and networking. For example, (a) it has been shown empirically in [67] (see also [3]) that the Internet topology embeds with better accuracy into a hyperbolic space than into an Euclidean space of comparable dimension, (b) every connected finite graph has an embedding in the hyperbolic plane so that the greedy routing based on the virtual coordinates obtained from this embedding is guaranteed to work (see [55]). A connected graph $G = (V, E)$ equipped with standard graph metric d_G is *δ-hyperbolic* if the metric space (V, d_G) is δ-hyperbolic.

TABLE 4. δ-hyperbolicity of the graph datasets

$G = (V, E)$	$\lvert V\rvert$	$\lvert E\rvert$	$\delta(G)$
PPI	1,458	1,948	3.5
Yeast	2,224	6,609	2.5
DutchElite	3,621	4,311	4
EPA	4,253	8,953	2.5
EVA	4,475	4,664	1
California	5,925	15,770	3
Erdös	6,927	11,850	2
Routeview	10,515	21,455	2.5
Homo	16,711	115,406	2
AS_CAIDA_20071105	26,475	53,381	2.5
Dimes 3/2010	26,424	90,267	2
Aqualab 12/2007- 09/2008	31,845	143,383	2
AS_CAIDA_20120601	41,203	121,309	2

More formally, let G be a graph and let u, v, w, x be four of its vertices. Denote by S_1, S_2, S_3 the three distance sums $d_G(u, v) + d_G(w, x)$, $d_G(u, w) + d_G(v, x)$ and $d_G(u, x) + d_G(v, w)$ sorted in nondecreasing order $S_1 \leq S_2 \leq S_3$. Define the *hyperbolicity of a quadruplet* u, v, w, x as $\delta(u, v, w, x) = \frac{S_3 - S_2}{2}$. Then, the hyperbolicity $\delta(G)$ of a graph G is the maximum hyperbolicity over all possible quadruplets of G, that is,

$$\delta(G) = \max_{u,v,w,x \in V} \delta(u, v, w, x).$$

δ-Hyperbolicity measures the local deviation of a metric from a tree metric; a graph metric is a tree metric if and only if it has hyperbolicity 0. Note that chordal graphs, mentioned in Section 4, have hyperbolicity at most 1 [14], while k-chordal graphs have hyperbolicity at most $k/4$ [71].

In Table 4, we show the hyperbolicities of most of our graph datasets. The computation of hyperbolicities is a costly operation. We were not able to compute it for three very large graph datasets since it would take a very long time to calculate. The best known algorithm [41] to calculate hyperbolicity has time complexity of $O(n^{3.69})$ and involves matrix multiplications. This algorithm still takes a long running time for large graphs and is hard to implement. The authors of [41] also propose a 2-approximation algorithm for calculating hyperbolicity that runs in $O(n^{2.69})$ time and a $2\log_2 n$-approximation algorithm that runs in $O(n^2)$ time. In our computations, we used the naive algorithm which calculates the exact hyperbolicity of a given graph in $O(n^4)$ time via calculating the hyperbolicities of its quadruplets. It is easy to show that the hyperbolicity of a graph is realized on one of its biconnected components. Thus, for very large graphs, we needed to check hyperbolicities only for quadruplets coming from the same biconnected component. Additionally, we used an algorithm by Cohen et al. [28] which has $O(n^4)$ time complexity but performs well in practice as it prunes the search space of quadruplets.

It turns out that most of the quadruplets in our datasets have small δ values (see Table 5). For example, more than 96% of the vertex quadruplets in the EVA and Erdös datasets

TABLE 5. Relative frequency of δ-hyperbolicity of quadruplets in graph datasets that have less than 10,000 vertices

δ graph	PPI	Yeast	DutchElite	EPA	EVA	California	Erdös
0	0.48	0.49	0.54	0.58	0.99	0.49	0.97
0.5	0.36	0.45	0	0.37	0.00	0.41	0.03
1	0.13	0.06	0.42	0.06	0.00	0.09	0.00
1.5	0.02	0.00	0	0.00	–	0.00	0.00
2	0.00	0.00	0.04	0.00	–	0.00	0.00
2.5	0.00	0.00	0	0.00	–	0.00	–
3	0.00	–	0.00	–	–	0.00	–
3.5	0.00	–	0	–	–	–	–
4	–	–	0.00	–	–	–	–
% ≤ 1	98.01	99.82	96.32	99.84	100	99.64	100

have δ values equal to 0. For the remaining graph datasets in Table 5, more than 96% of the quadruplets have $\delta \leq 1$, indicating that all of those graphs are metrically very close to trees.

In the remaining part of this section, we discuss the theoretical relations between the parameters $\delta(G)$ and $\Delta_s(G)$ of a graph. In [23], the following inequality was proven.

Proposition 2 ([23]). *For every n-vertex graph G and any vertex s of G,*

$$\Delta_s(G) \leq 4 + 12\delta(G) + 8\delta(G)\log_2 n.$$

Here, we complement that inequality by showing that the hyperbolicity of a graph is at most $\Delta_s(G)$.

Proposition 3. *For every n-vertex graph G and any vertex s of G,*

$$\delta(G) \leq \Delta_s(G).$$

Proof. Let $\mathcal{LP}(G,s)$ be a layering partition of G and let $\Gamma(G,s)$ be the corresponding layering tree (consult Fig. 1). From construction of $\mathcal{LP}(G,s)$ and $\Gamma(G,s)$, every cluster C of $\mathcal{LP}(G,s)$ separates in G any two vertices belonging to nodes (clusters) of different subtrees of the forest obtained from $\Gamma(G,s)$ by removing node C. Note that every vertex of G belongs to exactly one node (cluster) of the layering tree $\Gamma(G,s)$.

Consider an arbitrary quadruplet x,y,z,w of vertices of G. Let X,Y,Z,W be the nodes in $\Gamma(G,s)$ (i.e., clusters in $\mathcal{LP}(G,s)$) containing vertices x,y,z,w, respectively. Note that nodes X,Y,Z,W are not necessarily all different. In the tree $\Gamma(G,s)$, consider a median[2] node M of nodes X,Y,Z,W, that is, a node M the removal of which from $\Gamma(G,s)$ leaves no connected subtree with more than two nodes from $\{X,Y,Z,W\}$. As a consequence, any connected component of graph $G[V \setminus M]$ (the graph obtained from G by removing vertices of M) cannot have more than 2 vertices out of $\{x,y,z,w\}$. Thus, we may assume, without loss of generality, that for every vertex $s \in \{x,y\}$ and every vertex $t \in \{z,w\}$, either s and t are in different connected components of $G[V \setminus M]$ or $\{s,t\} \cap M \neq \emptyset$, i.e., M intersects all paths of G connecting s and t. See Figure 2 for an illustration.

Let μ_a be the distance from $a \in \{x,y,z,w\}$ to its closest vertex in M. Note that $\mu_a = 0$ is not excluded, that is, $a \in M$ is possible. Let a,b be a pair of vertices from $\{x,y,z,w\}$. If the vertices a,b belong to different components of $G[V \setminus M]$, then M separates a from b and, therefore, $\mu_a + \mu_b \leq d_G(a,b)$. Note that $\mu_a + \mu_b \leq d_G(a,b)$ holds also when one of $\{a,b\}$ or both belong to M. As M intersects all paths of G connecting $s \in \{x,y\}$ and $t \in \{z,w\}$, we get $d_G(x,z) + d_G(y,w) \geq \mu_x + \mu_y + \mu_z + \mu_w$ and $d_G(x,w) + d_G(y,z) \geq \mu_x + \mu_y + \mu_z + \mu_w$. Conversely, all three sums $d_G(x,z) + d_G(y,w)$, $d_G(x,w) + d_G(y,z)$ and $d_G(x,y) + d_G(z,w)$ are less than or equal to $\mu_x + \mu_y + \mu_z + \mu_w + 2\Delta_s(G)$, as by the triangle inequality, $d_G(a,b) \leq \mu_a + \mu_b + \Delta_s(G)$ for every $a,b \in \{x,y,z,w\}$.

Now, as the two larger distance sums are between μ and $\mu + 2\Delta_s(G)$, where $\mu := \mu_x + \mu_y + \mu_z + \mu_w$, we conclude that the difference between the two larger distance sums is at most $2\Delta_s(G)$. Thus, necessarily $\delta(G) \leq \Delta_s(G)$. ■

Combining Proposition 2 with Proposition 1, one obtains also the following interesting result relating the hyperbolicity of a graph G with additive distortion of embedding of G to its canonical tree H.

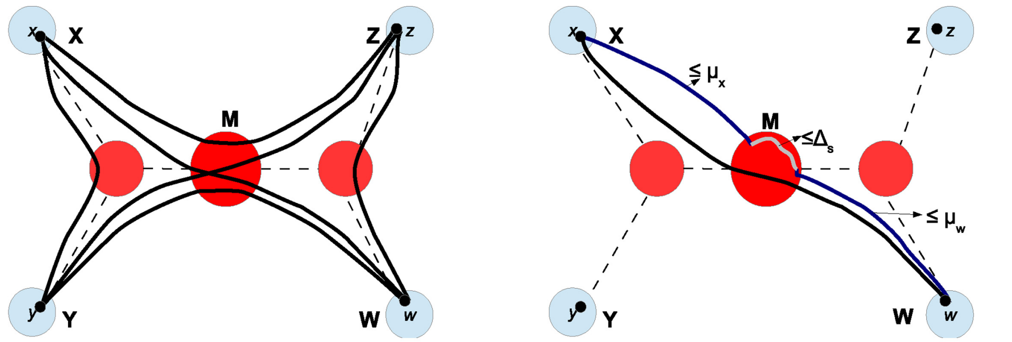
Proposition 4 ([23]). *For any graph $G = (V,E)$ and its canonical tree $H = (V,F)$ the following holds:*

$$\begin{aligned}\forall u,v \in V,\ d_H(u,v) - 2 &\leq d_G(u,v)\\ &\leq d_H(u,v) + O(\delta(G)\log n).\end{aligned}$$

As a canonical tree H is constructible in linear time for a graph G, by Proposition 4, the distances in n-vertex δ-hyperbolic graphs can efficiently be approximated within an additive error of $O(\delta \log n)$ by a tree metric and this approximation is sharp (see [44, 46] and [23, 42]).

Graphs and general geodesic spaces with small hyperbolicities have many other algorithmic advantages. They allow for efficient approximate solutions for a number of optimization problems. For example, Krauthgamer and Lee [56] presented a PTAS for the Traveling Salesman Problem when the set of cities lie in a hyperbolic metric space. Chepoi and Estellon [26] established a relationship between the minimum number of balls of radius $r + 2\delta$ covering a finite subset S of a δ-hyperbolic geodesic space and the size of the maximum r-packing of S and showed how to compute such coverings and packings in polynomial time. Chepoi et al. [23] gave efficient algorithms for quick and accurate estimations of diameters and radii of δ-hyperbolic geodesic spaces and graphs. Additionally, Chepoi et al. [24] showed that every n-vertex δ-hyperbolic graph has an additive $O(\delta \log n)$-spanner with at most $O(\delta n)$ edges and enjoys an $O(\delta \log n)$-additive routing labeling scheme with $O(\delta \log^2 n)$-bit labels and $O(\log \delta)$

[2] It is known that for any set $S \subseteq X$ of nodes of a tree $T = (X,U)$ there is a node v in T, called a median of S, such that any subtree of $T[X \setminus v]$ has at most $|S|/2$ nodes from S. Such a node v can be found in linear time [45].



(a) M is a median node for X, Y, Z, W in $\Gamma(G, s)$.

(b) M separates in G vertices x and y from vertices z and w.

FIG. 2. Illustration for the proof of Proposition 3. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

time routing protocol. We elaborate more on these results in Section 8.

6. TREE-DISTORTION

The problem of approximating a given graph metric by a "simpler" metric is well motivated from several different perspectives. A particularly simple metric of choice, also favored from the algorithmic point of view, is a tree metric, that is, a metric arising from shortest path distance on a tree containing the given points. In recent years, a number of authors considered problems of minimum distortion embeddings of graphs into trees (see [5–7, 25]), most popular among them being a noncontractive embedding with minimum multiplicative distortion.

Let $G = (V, E)$ be a graph. The (multiplicative) *tree-distortion* $\mathrm{td}(G)$ of G is the smallest integer α such that G admits a tree (possibly weighted and with Steiner points) with

$$\forall u, v \in V,\ d_G(u, v) \le d_T(u, v) \le \alpha d_G(u, v).$$

The problem of finding, for a given graph G, a tree $T = (V \cup S, F)$ satisfying $d_G(u, v) \le d_T(u, v) \le \mathrm{td}(G) d_G(u, v)$, for all $u, v \in V$, is known as the *problem of minimum distortion noncontractive embedding of graphs into trees*. In a noncontractive embedding, the distance in the tree must always be larger that or equal to the distance in the graph, that is, the tree distances "dominate" the graph distances.

It is known that this problem is NP-hard, and even more, the hardness result of [5] implies that it is NP-hard to approximate $\mathrm{td}(G)$ better than γ, for some small constant γ. The best known 6-approximation algorithm using a layering partition technique was recently given in [25]. It improves the previously known 100-approximation algorithm from [7] and 27-approximation algorithm from [6]. Below, we will provide a short description of the method of [25].

The following proposition establishes a relationship between the tree-distortion and the cluster-diameter of a graph.

Proposition 5 ([25]). *For every graph G and any its vertex s, $\Delta_s(G)/3 \le td(G) \le 2\Delta_s(G) + 2$.*

Proposition 5 shows that the cluster-diameter $\Delta_s(G)$ of a layering partition of a graph G linearly bounds the tree-distortion $\mathrm{td}(G)$ of G.

Combining Proposition 5 and Proposition 1, the following result is obtained.

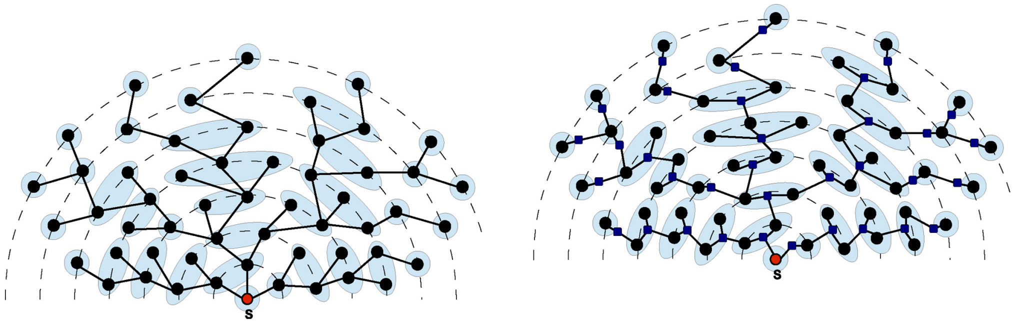
Proposition 6 ([25]). *For any graph $G = (V, E)$ and its canonical tree $H = (V, F)$ the following holds:*

$$\forall u, v \in V,\ d_H(u, v) - 2 \le d_G(u, v) \le d_H(u, v) + 3td(G).$$

Surprisingly, a multiplicative distortion turned into an additive distortion. Furthermore, while a tree $T = (V \cup S, F)$ satisfying $d_G(u, v) \le d_T(u, v) \le \mathrm{td}(G) d_G(u, v)$, for all $u, v \in V$, is NP-hard to find, a canonical tree H of G can be constructed in $O(m)$ time (where $m = |E|$).

By assigning proper weights to edges of a canonical tree H or adding at most $n = |V|$ new Steiner points to H, the authors of [25] achieve a good non-contractive embedding of a graph G into a tree. Recall that a canonical tree $H = (V, F)$ of $G = (V, E)$ is constructed in the following way: identify for each cluster $C = L^i_j \in \mathcal{LP}(G, s)$ of a layering partition $\mathcal{LP}(G, s)$ of G an arbitrary vertex $x_C \in L_{i-1}$ which has a neighbor in $C = L^i_j$ and make x_C adjacent in H to all vertices $v \in C$ (see Fig. 3a). Note that H is an unweighted tree, without any Steiner points, and resembles a BFS-tree of G. Two other trees for G are constructed as follows.

Tree $\mathbf{H}_\ell$: Tree $H_\ell = (V, F, \ell)$ is obtained from H by assigning uniformly the weight $\ell = \max\{d_G(u, v) : uv \text{ is an edge of } H\}$ to all edges of H. So, H_ℓ is a uniformly weighted tree without Steiner points. It turns out that G



(a) Topology of trees H and H_ℓ.

(b) Topology of tree H'_ℓ. Squares denote Steiner points.

FIG. 3. Embedding into trees H, H_ℓ, and H'_ℓ. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

embeds in tree H_ℓ noncontractively. Note that, although the topology of the tree H_ℓ can be determined in $O(m)$ time (H_ℓ is isomorphic to H), computation of the weight ℓ requires $O(nm)$ time. Thus, the tree H_ℓ is constructible in $O(nm)$ total time. See Figure 3a for an illustration.

Tree $\mathbf{H}'_\ell$: Tree $H'_\ell = (V \cup S, F', \ell)$ is obtained from H in the following way. First, introduce one Steiner point p_C for each cluster $C := L^i_j$ and add an edge between each vertex of C and p_C and an edge between p_C and the support vertex x_C for C. Then, assign uniformly the weight $\ell = \frac{1}{2} \max\{\Delta_s(G), \max\{d_G(u,v) : uv \text{ is an edge of } H\}\}$ to all edges of the obtained tree. So, H'_ℓ is a uniformly weighted tree with at most $O(n)$ Steiner points. Again, G embeds into tree H'_ℓ noncontractively and H'_ℓ can be obtained in $O(nm)$ total time. See Figure 3b for an illustration.

Constructed trees have the following distance properties (for comparison, we also include the results for H mentioned earlier).

Proposition 7 ([25]). *Let $G = (V, E)$ be a graph, s be an arbitrary vertex, $\alpha = td(G)$, $\Delta_s = \Delta_s(G)$, and H, H_ℓ, H'_ℓ be trees as described above. Then, for any two vertices x and y of G, the following holds:*

$$d_H(x,y) - 2 \le d_G(x,y) \le d_H(x,y) + \Delta_s,$$

$$d_H(x,y) - 2 \le d_G(x,y) \le d_H(x,y) + 3\alpha,$$

$$d_G(x,y) \le d_{H_\ell}(x,y) \le (\Delta_s + 1)(d_G(x,y) + 2),$$

$$d_G(x,y) \le d_{H_\ell}(x,y) \le \max\{3\alpha - 1, 2\alpha + 1\}\,(d_G(x,y) + 2),$$

$$d_G(x,y) \le d_{H'_\ell}(x,y) \le (\Delta_s + 1)(d_G(x,y) + 1),$$

$$d_G(x,y) \le d_{H'_\ell}(x,y) \le 3\alpha(d_G(x,y) + 1).$$

As pointed out in [25], tree H'_ℓ provides a 6-approximate solution to the problem of minimum distortion noncontractive embedding of graphs into trees.

In our empirical study, we analyze embeddings of our graph datasets into each of these three trees and measure how close these graph datasets resemble a tree from this perspective. We compute the following measures:

- maximum distortion right $:= \max\left\{\frac{d_T(u,v)}{d_G(u,v)} : u, v \in V, d_T(u,v) > d_G(u,v) > 0\right\}$;
- maximum distortion left $:= \max\left\{\frac{d_G(u,v)}{d_T(u,v)} : u, v \in V, d_G(u,v) > d_T(u,v) > 0\right\}$;
- average distortion right $:=$ $\operatorname{avg}\left\{\frac{d_T(u,v)}{d_G(u,v)} : u, v \in V, d_T(u,v) > d_G(u,v) > 0\right\}$;
- average distortion left $:=$ $\operatorname{avg}\left\{\frac{d_G(u,v)}{d_T(u,v)} : u, v \in V, d_G(u,v) > d_T(u,v) > 0\right\}$;
- average relative distortion $:= \operatorname{avg}\left\{\frac{|d_T(u,v) - d_G(u,v)|}{d_G(u,v)} : u, v \in V\right\}$; and
- distance-weighted average distortion $:= \frac{1}{\Sigma_{u,v \in V} d_G(u,v)} \Sigma_{u,v \in V} (d_G(u,v) \cdot \frac{d_T(u,v)}{d_G(u,v)}) = \frac{\Sigma_{u,v \in V} d_T(u,v)}{\Sigma_{u,v \in V} d_G(u,v)}$.

A pair of distinct vertices u, v of $G = (V, E)$ is said to be a *right pair* with respect to tree $H = (V, F)$ if $d_G(u,v) < d_H(u,v)$. If $d_H(u,v) < d_G(u,v)$ then $\{u,v\}$ is called a *left pair*. Note that G has no left pairs with respect to trees H_ℓ and H'_ℓ. Hence, in case of trees H_ℓ and H'_ℓ, we talk only about maximum distortion, average distortion, average relative distortion and distance-weighted average distortion. Distance-weighted average distortion is used in the literature when distortion of distant pairs of vertices is more important than that of close pairs, as it gives larger weight values to distortion of distant pairs (see [50]). Clearly, any tree graph would have maximum distortion, average relative distortion and distance-weighted average distortion equal to 1, 0, and 1, respectively.

Tables 6 and 7 show the results of embedding our graph datasets into trees H, H_ℓ, and H'_ℓ, respectively. It turns out that most of the datasets embed into tree H with average distortion (right or left, right being usually better) between 1 and 1.5. Also, many pairs of vertices enjoy exact embedding into the tree H; they preserve their original graph distances (for example, around 88% of the pairs in the Erdös dataset, 72% of the pairs in Homo, 57% in AS_CAIDA_20120601 preserve their original graph distances). Comparing the results of noncontractive embeddings into trees H_ℓ and H'_ℓ, we observe that maximum distortions are slightly improved in H'_ℓ over

TABLE 6. Distortion results of embedding each dataset into a canonical tree H

Graph	Average distortion left	Max distortion left	% of left pairs	Average distortion right	Max distortion right	% of right pairs	% of pairs $d_T = d_G$	Average relative distortion	Distance-weighted average distortion
PPI	1.50	7	70.50	1.34	3	09.10	20.40	0.25	0.79
Yeast	1.49	5	56.30	1.39	3	12.20	31.50	0.22	0.85
DutchElite	1.54	7	73.00	1.41	3	03.90	23.10	0.25	0.76
EPA	1.50	5	44.66	1.38	3	10.47	44.87	0.18	0.88
EVA	1.30	6	32.31	1.28	3	14.77	52.92	0.11	0.95
California	1.52	5	61.82	1.37	3	07.92	30.25	0.23	0.81
Erdös	1.35	3	02.75	1.41	3	08.91	88.34	0.04	1.02
Routeview	1.41	4	24.39	1.41	3	33.34	42.28	0.21	1.03
Homo	1.53	4	02.83	1.68	3	25.16	72.01	0.18	1.13
AS_CAIDA_20071105	1.48	4	21.43	1.36	3	35.42	43.15	0.19	1.03
Dimes 3/2010	1.54	3	05.74	1.37	3	44.42	49.84	0.18	1.13
Aqualab 12/2007- 09/2008	1.42	4	31.71	1.42	3	35.75	32.54	0.24	1.03
AS_CAIDA_20120601	1.35	4	22.42	1.40	3	20.43	57.15	0.14	1.01
itdk0304	1.60	8	94.85	1.26	3	00.55	04.60	0.33	0.67
DBLB-coauth	1.77	9	95.82	1.25	3	00.59	03.59	0.38	0.62
Amazon	2.48	19	99.17	1.20	3	00.20	00.63	0.54	0.54

TABLE 7. Distortion results of noncontractive embedding of datasets into trees H_ℓ and H'_ℓ

	Tree H_ℓ				Tree H'_ℓ			
Graph	Average distortion	Max distortion	Average relative distortion	Distance-weighted average distortion	Average distortion	Max distortion	Average relative distortion	Distance-weighted average distortion
PPI	5.71	21	4.71	5.53	5.30	16	4.30	5.20
Yeast	4.38	15	3.38	4.25	3.79	12	2.79	3.74
DutchElite	5.45	21	4.45	5.33	6.53	20	5.53	6.46
EPA	4.51	15	3.51	4.39	4.07	12	3.07	3.99
EVA	5.83	18	4.83	5.71	7.78	18	6.78	7.66
California	4.16	15	3.18	4.05	4.99	16	3.99	4.93
Erdös	3.09	9	2.09	3.07	3.07	8	2.07	3.06
Routeview	4.28	12	3.28	4.14	4.80	12	3.80	4.67
Homo	4.65	12	3.65	4.54	3.97	10	2.97	3.95
AS_CAIDA_20071105	4.24	12	3.24	4.12	4.77	12	3.77	4.66
Dimes 3/2010	3.44	9	2.44	3.38	3.36	8	2.36	3.32
Aqualab 12/2007- 09/2008	4.23	12	3.23	4.13	4.54	12	3.54	4.46
AS_CAIDA_20120601	4.11	12	3.11	4.03	4.53	12	3.53	4.49
itdk0304	5.37	24	4.37	5.38	5.71	22	4.71	5.83
DBLB-coauth	5.58	27	4.58	5.54	5.13	22	4.13	5.15
Amazon	8.82	57	7.82	8.78	7.87	42	6.87	7.95

distortions in H_ℓ, but average distortions are very much comparable. Furthermore, distance-weighted average distortions are better in H_ℓ than in H'_ℓ. This confirms Gupta's claim [47] that the Steiner points do not really help.

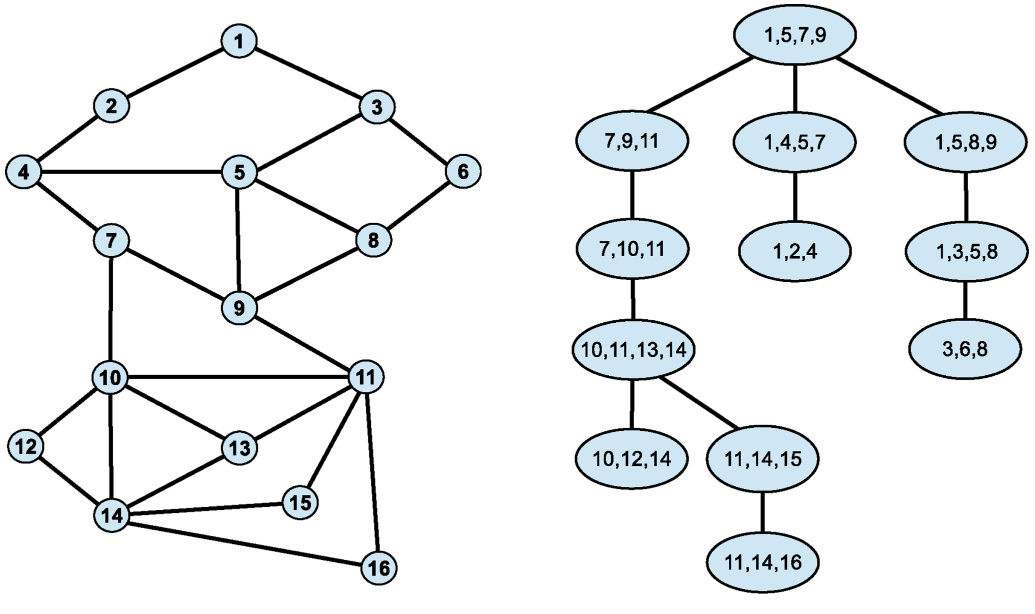
As tree H'_ℓ provides a 6-approximate solution to the problem of minimum distortion noncontractive embedding of graphs into trees, dividing by 6 the maximum distortion values in Table 7 for tree H'_ℓ, we obtain a lower bound on td(G) for each graph dataset G. For example, td(G) is at least 4/3 for Erdös and Dimes 3/2010, at least 5/3 for Homo, at least 2 for Yeast, EPA, Routeview, AS_CAIDA_20071105, Aqualab 12/2007-09/2008 and AS_CAIDA_20120601, at least 8/3 for PPI and California, at least 10/3 for DutchElite, at least 3 for EVA, at least 11/3 for itdk0304 and DBLB-coauth, and at least 7 for Amazon.

7. TREE-BREADTH, TREE-LENGTH, AND TREE-STRETCH

There are two other graph parameters measuring metric tree-likeness of a graph that are based on the notion of tree-decomposition introduced by Robertson and Seymour in their work on graph minors [63].

A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(\{X_i : i \in I\}, T = (I, F))$ where $\{X_i : i \in I\}$ is a collection of subsets of V, called *bags*, and T is a tree. The nodes of T are the bags $\{X_i : i \in I\}$ satisfying the following three conditions (see Fig. 4):

1. $\underset{i \in I}{\cup} X_i = V$;
2. For each edge $uv \in E$, there is a bag X_i such that $u, v \in X_i$;



(a) A graph G.

(b) A tree-decomposition of G.

FIG. 4. A graph and its tree-decomposition of width 3, of length 3, and of breadth 2. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

3. For all $i, j, k \in I$, if j is on the path from i to k in T, then $X_i \cap X_k \subseteq X_j$. Equivalently, this condition could be stated as follows: for all vertices $v \in V$, the set of bags $\{i \in I : v \in X_i\}$ induces a connected subtree T_v of T.

For simplicity, we denote a tree-decomposition $(\{X_i : i \in I\}, T = (I, F))$ of a graph G by $\mathcal{T}(G)$.

The *width* of a tree-decomposition $\mathcal{T}(G) = (\{X_i : i \in I\}, T = (I, F))$ is $\max_{i \in I}|X_i| - 1$. The *tree-width* of a graph G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree-decompositions $\mathcal{T}(G)$ of G [63]. Trees are exactly the graphs with tree-width 1.

The *length* of a tree-decomposition $\mathcal{T}(G)$ of a graph G is $\lambda := \max_{i \in I} \max_{u,v \in X_i} d_G(u, v)$ (i.e., each bag X_i has diameter at most λ in G). The *tree-length* of G, denoted by $\mathrm{tl}(G)$, is the minimum length over all tree-decompositions of G [34]. The chordal graphs are exactly the graphs with tree-length 1. Note that these two graph parameters are not related to each other. For instance, a clique on n vertices has tree-length 1 and tree-width $n - 1$, whereas a cycle on $3n$ vertices has tree-width 2 and tree-length n. Analysis of a few real-world networks (such as Aqualab, AS_CAIDA, Dimes) performed in [29] shows that although those networks have small hyperbolicities, they all have sufficiently large tree-width due to well-connected cores. As we demonstrate below, the tree-length of those graph datasets is relatively small.

The *breadth* of a tree-decomposition $\mathcal{T}(G)$ of a graph G is the minimum integer r such that for every $i \in I$ there is a vertex $v_i \in V$ with $X_i \subseteq B_r(v_i, G)$ (i.e., each bag X_i can be covered by a ball $B_r(v_i, G)$ of radius at most r in G). Note that vertex v_i does not need to belong to X_i. The *tree-breadth* of G, denoted by $\mathrm{tb}(G)$, is the minimum breadth over all tree-decompositions of G [37]. Evidently, for any graph G, $1 \leq \mathrm{tb}(G) \leq \mathrm{tl}(G) \leq 2\mathrm{tb}(G)$ holds. Hence, if one parameter is bounded by a constant for a graph G then the other parameter is bounded for G as well.

Clearly, in view of a tree-decomposition $\mathcal{T}(G)$ of G, the smaller the parameters $\mathrm{tl}(G)$ and $\mathrm{tb}(G)$ of G, the closer graph G is to a tree metrically. Unfortunately, while graphs with tree-length 1 (as they are exactly the chordal graphs) can be recognized in linear time, the problem of determining whether a given graph has tree-length at most λ is NP-complete for every fixed $\lambda > 1$ (see [58]). Judging from this result, it is conceivable that the problem of determining whether a given graph has tree-breadth at most ρ is NP-complete, too.

The following proposition establishes a relationship between the tree-length and the cluster-diameter of a layering partition of a graph.

Proposition 8 ([34]). *For every graph G and any vertex s,* $\Delta_s(G)/3 \leq tl(G) \leq \Delta_s(G) + 1$.

Thus, the cluster-diameter $\Delta_s(G)$ of a layering partition provides easily computable bounds for the hard to compute parameter $\mathrm{tl}(G)$.

One can prove similar inequalities relating the tree-breadth and the cluster-radius of a layering partition of a graph.

Proposition 9. *For every graph G and any vertex s,*

$$\Delta_s(G)/6 \leq R_s(G)/3 \leq tb(G) \leq R_s(G) + 1 \leq \Delta_s(G) + 1.$$

Furthermore, a tree-decomposition of G with breadth at most $3tb(G)$ *can be constructed in* $O(n + m)$ *time.*

Proof.

The proof is similar to the proof from [34] of Proposition 8. First, we show $R_s(G)/3 \leq \text{tb}(G)$. Let $\mathcal{T}(G)$ be a tree-decomposition of G with minimum breadth $\text{tb}(G)$. Let X_1X_2 be an edge of $\mathcal{T}(G)$ and $\mathcal{T}_1, \mathcal{T}_2$ be subtrees of $\mathcal{T}(G)$ after removing the edge X_1X_2. It is known [31] that set $I = X_1 \cap X_2$ separates in G vertices belonging to bags of $\mathcal{T}_1$ but not to I from vertices belonging to bags of $\mathcal{T}_2$ but not to I. Assume that $\mathcal{T}(G)$ is rooted at a bag containing vertex s, the source of the layering partition $\mathcal{LP}(G,s)$. Let C be a cluster from layer L_i (i.e., $C = L_i^j$ for some $j = 1, \ldots, p_i$). Let Z be the nearest common ancestor of all bags of $\mathcal{T}(G)$ containing vertices of C (such Z always exists). Let z be the vertex such that $Z \subseteq B_{\text{tb}(G)}(z,G)$. We will show that $d_G(x,z) \leq 3\text{tb}(G)$ holds for every vertex $x \in C$.

Consider an arbitrary vertex $x \in C$. Necessarily, there is a vertex $y \in C$ and two bags X and Y of $\mathcal{T}(G)$ containing vertices x and y, respectively, such that $Z = \text{NCA}_{\mathcal{T}(G)}(X,Y)$ (i.e., Z is the nearest common ancestor of X and Y in $\mathcal{T}(G)$). Let P be a shortest path of G from s to x. By the separator property above, P intersects Z. Indeed, if neither s nor x is in Z, then for a neighbor Z' of Z on the path of $\mathcal{T}(G)$ from Z to the root, set $I = Z \cap Z'$ separates in G vertex x from vertex s [31]. See Figure 5 for an illustration. Let a be a vertex of $P \cap Z$ closest to s in G. As both x and y belong to C, there exist a path Q from x to y in G using only intermediate vertices w with $d_G(s,w) \geq i$. Path Q also intersects set Z. Let $b \in Q \cap Z$. We have $d_G(s,x) = i = d_G(s,a) + d_G(a,x)$ and $i \leq d_G(s,b) \leq d_G(s,a) + d_G(a,z) + d_G(z,b) \leq d_G(s,a) + 2\text{tb}(G)$. Hence, $d_G(a,x) = i - d_G(s,a) \leq 2\text{tb}(G)$ and, therefore, $d_G(x,z) \leq d_G(x,a) + d_G(a,z) \leq 2\text{tb}(G) + \text{tb}(G) = 3\text{tb}(G)$. Thus, any vertex x of C is at distance at most $3\text{tb}(G)$ from z in G, implying $R_s(G)/3 \leq \text{tb}(G)$.

Note that, for the neighbor x' of x on P, $d(x',z) \leq 3\text{tb}(G) - 1$ must hold, that is, $B_{3\text{tb}(G)}(z,G)$ contains not only all vertices of $C = L_i^j$ but also all neighbors of vertices of C appearing in layer L_{i-1}. This fact will be useful in the second part of this proof.

Now we show that $\text{tb}(G) \leq R_s(G) + 1$. Consider the tree $\Gamma(G,s)$ of the layering partition $\mathcal{LP}(G,s)$ and assume $\Gamma(G,s)$ is rooted at node $\{s\}$. Let $p(C)$ be the parent of node C in $\Gamma(G,s)$. Clearly, $\Gamma(G,s)$ satisfies already conditions 1 and 3 of tree-decompositions and only violates condition 2 as the edges joining vertices in different (neighboring) layers are not yet covered by bags (which are the clusters in this case). We can obtain a tree-decomposition Γ' from $\Gamma(G,s)$ as follows. Γ' will have the same structure as $\Gamma(G,s)$, only the nodes of $\Gamma(G,s)$ will slightly expand to cover additional edges of G and form the bags of Γ'. To each node C of $\Gamma(G,s)$ (assume $C \subseteq L_i$) we add all vertices from its parent $p(C)$ ($p(C) \subseteq L_{i-1}$) which are adjacent to vertices of C in G. This expansion of C results in a bag C^+ of Γ' which, by construction, contains now also each edge uv of G with $u \in C \subseteq L_i$ and $v \in p(C) \subseteq L_{i-1}$. Thus, Γ' satisfies conditions 1 and 2 of tree-decompositions. Also, if $C \subseteq B_r(z,G)$ for some vertex z and integer r, then $C^+ \subseteq B_{r+1}(z,G)$ must hold. Furthermore, each vertex v of G that was in a node C now belongs to bag C^+ and to all bags formed from children of C in $\Gamma(G,s)$ (and only to them). Hence, all bags containing v form a star in Γ'. All these indicate that Γ' is a tree-decomposition of G with breadth at most $R_s(G) + 1$, that is, $\text{tb}(G) \leq R_s(G) + 1$.

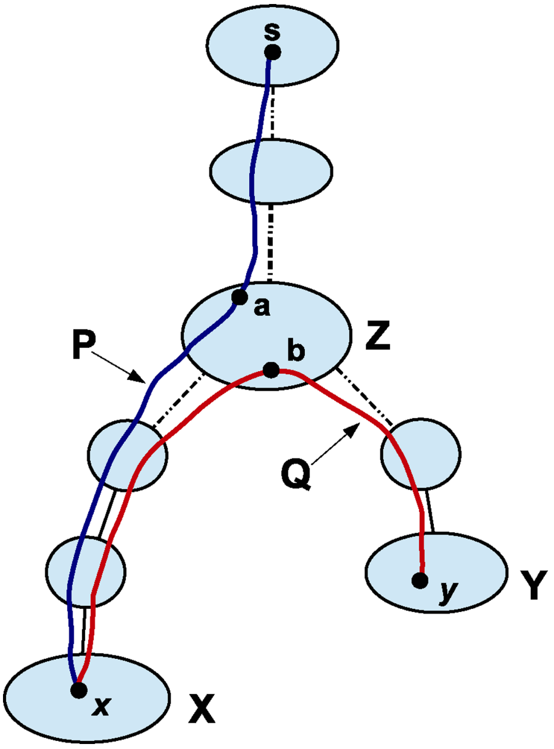


FIG. 5. Illustration for the proof of Proposition 9. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Furthermore, as we indicated in the first part of this proof, for any cluster C there is a vertex z in G such that $C^+ \subseteq B_{3\text{tb}(G)}(z,G)$. The latter implies that the tree Γ' obtained from $\Gamma(G,s)$ has breadth at most $3\text{tb}(G)$. Finally, as Γ' is constructible in linear time and $R_s(G) \leq \Delta_s(G) \leq 2R_s(G)$ holds for every graph G, the proposition follows. ■

Hence, the cluster-radius $R_s(G)$ of a layering partition provides easily computable bounds for the tree-breadth $\text{tb}(G)$ of a graph. In Table 8, we show the corresponding lower and upper bounds on the tree-breadth for some of our datasets. The lower bound is obtained by dividing $R_s(G)$ by 3, the upper bound is obtained by calculating the breadth of the tree-decomposition Γ'.

Reformulating Proposition 1, we obtain the following result.

Proposition 10. *For any graph $G = (V,E)$ and its canonical tree $H = (V,F)$ the following holds:*

$$\forall u,v \in V,\ d_H(u,v) - 2 \leq d_G(u,v) \leq d_H(u,v) + 3tl(G) \leq d_H(u,v) + 6tb(G).$$

Graphs with small tree-length or small tree-breadth have many other nice properties. Every n-vertex graph with tree-length $\text{tl}(G) = \lambda$ has an additive 2λ-spanner with $O(\lambda n +$

TABLE 8. Lower and upper bounds on the tree-breadth of our graph datasets

$G = (V, E)$	$R_s(G)$	Lower bound on tb(G)	Upper bound on tb(G)
PPI	4	2	5
Yeast	4	2	4
DutchElite	6	2	6
EPA	4	2	4
EVA	5	2	5
California	4	2	4
Erdös	2	1	2
Routeview	3	1	4
Homo	3	1	3
AS_CAIDA_20071105	3	1	3
Dimes 3/2010	2	1	2
Aqualab 12/2007- 09/2008	3	1	3
AS_CAIDA_20120601	3	1	3
itdk0304	6	2	6
DBLB-coauth	7	3	7
Amazon	12	4	12

$n \log n$) edges and an additive 4λ-spanner with $O(\lambda n)$ edges, both constructible in polynomial time [33]. Every n-vertex graph G with tb(G) $= \rho$ has a system of at most $\log_2 n$ collective additive tree $(2\rho\log_2 n)$-spanners constructible in polynomial time [36]. Those graphs also enjoy a 6λ-additive routing labeling scheme with $O(\lambda\log^2 n)$-bit labels and $O(\log \lambda)$ time routing protocol [32], and a $(2\rho\log_2 n)$-additive routing labeling scheme with $O(\log^3 n)$-bit labels and $O(1)$ time routing protocol with $O(\log n)$ message initiation time (by combining results of [36] and [38]). See Section 8 for further details.

Here, we elaborate a little bit more on a connection established in [37] between the tree-breadth and the tree-stretch of a graph (and the corresponding tree t-spanner problem).

The *tree-stretch* ts(G) of a graph $G = (V, E)$ is the smallest number t such that G admits a *spanning* tree $T = (V, E')$ with $d_T(u, v) \leq \text{td}_G(u, v)$ for every $u, v \in V$. T is called a tree t-spanner of G and the problem of finding such a tree T for G is known as the tree t-spanner problem. Note that as T is a spanning tree of G, necessarily $d_G(u, v) \leq d_T(u, v)$ and $E' \subseteq E$. The latter makes the tree-stretch parameter different from the tree-distortion where new (not from the graph) edges can be used to build a tree. It is known that the tree t-spanner problem is NP-hard [16]. The best known approximation algorithms have approximation ratio of $O(\log n)$ [37, 39].

The following two results were obtained in [37].

Proposition 11 ([37]). *For every graph G, $tb(G) \leq \lceil ts(G)/2 \rceil$ and $tl(G) \leq ts(G)$.*

Proposition 12 ([37]). *For every n-vertex graph G, $ts(G) \leq 2tb(G)\log_2 n$. Furthermore, a spanning tree T of G with $d_T(u, v) \leq 2tb(G)\log_2 n d_G(u, v)$, for every $u, v \in V$, can be constructed in polynomial time.*

Proposition 12 is obtained by showing that every n-vertex graph G with tb(G) $= \rho$ admits a tree $(2\rho\log_2 n)$-spanner constructible in polynomial time. Together with Proposition 11, this provides a $\log_2 n$-approximate solution for the tree t-spanner problem in general unweighted graphs.

We conclude this section with two other inequalities establishing relations between the tree-stretch, the tree-distortion, and the hyperbolicity of a graph.

Proposition 13 ([35]). *For every graph G, $tl(G) \leq td(G) \leq ts(G) \leq 2td(G)\log_2 n$.*

Proposition 14 ([35]). *For every δ-hyperbolic graph G, $ts(G) \leq O(\delta\log^2 n)$.*

Proposition 13 says that if a graph G is noncontractively embeddable into a tree with distortion td(G) then it is embeddable into a spanning tree with stretch at most $2\text{td}(G)\log_2 n$. Furthermore, a spanning tree with stretch at most $2\text{td}(G)\log_2 n$ can be constructed in polynomial time. Proposition 14 says that every δ-hyperbolic graph G admits a tree $O(\delta\log^2 n)$-spanner. Furthermore, such a spanning tree for a δ-hyperbolic graph can be constructed in polynomial time.

8. USE OF METRIC TREE-LIKENESS

As we have mentioned earlier, metric tree-likeness of a graph is useful in a number of ways. Among other advantages, it allows one to design compact and efficient approximate distance labeling and routing labeling schemes as well as efficient algorithms for quick and accurate estimation of the diameter and the radius of a graph. In this section, we elaborate on these applications. In general, low distortion embedability of a graph G into a tree T allows one to solve approximately many distance related problems on G by first solving them on the tree T and then interpreting that solution on G.

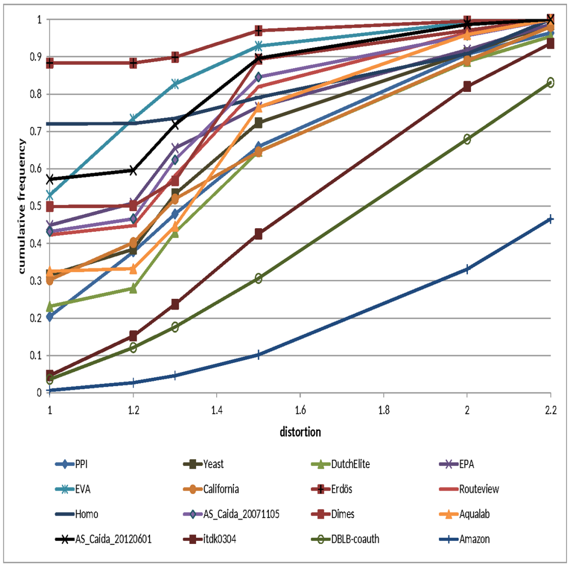
8.1. *Approximate distance queries*

Commonly, when one makes a query concerning a pair of vertices in a graph (adjacency, distance, shortest route, etc.), one needs to make global access to the structure storing that information. A compromise to this approach is to store enough information locally in a label associated with a vertex such that the query can be answered using only the information in the labels of the two vertices in question and nothing else. Motivation of localized data structure in distributed computing is surveyed and widely discussed in [43, 62].

Here, we are mainly interested in the distance and routing labeling schemes, introduced by Peleg (see, e.g., [62]). *Distance labeling schemes* are schemes that label the vertices of a graph with short labels in such a way that the distance between any two vertices u and v can be determined efficiently by merely inspecting the labels of u and v, without using any other information. *Routing labeling schemes* are schemes that label the vertices of a graph with short labels in

Graph	distortion					
	$= 1$	< 1.2	< 1.3	< 1.5	< 2	< 2.2
PPI	20.41	37.68	47.90	65.93	90.68	96.37
Yeast	31.51	38.45	53.22	72.30	91.03	98.55
DutchElite	23.13	27.99	42.97	64.60	88.71	95.44
EPA	44.87	50.83	65.50	76.52	91.82	98.68
EVA	52.92	73.37	82.68	92.83	99.12	99.88
California	30.25	40.21	51.89	64.53	88.97	98.06
Erdös	88.34	88.34	89.84	96.99	99.55	99.98
Routeview	42.28	44.75	58.17	81.94	96.40	99.85
Homo	72.01	72.13	73.48	79.08	90.79	99.97
AS_CAIDA_20071105	43.15	46.60	62.39	84.54	95.68	99.90
Dimes 3/2010	49.84	50.06	56.77	89.30	97.05	99.99
Aqualab 12/2007- 09/2008	32.54	33.23	44.61	76.46	95.93	99.98
AS_CAIDA_20120601	57.15	59.57	71.82	89.58	98.65	99.98
itdk0304	4.60	15.18	23.67	42.54	81.98	93.55
DBLB-coauth	3.59	12.08	17.60	30.64	67.92	83.10
Amazon	0.63	2.67	4.57	10.16	33.10	46.53

(a) Percentage of vertex pairs whose distance was distorted only up to a given value.



(b) Cumulative frequency chart.

FIG. 6. Distortion distribution for embedding of a graph dataset into its canonical tree H. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

such a way that given the label of a source vertex and the label of a destination vertex, it is possible to compute efficiently the port number of the edge from the source that heads in the direction of the destination.

It is known that n-vertex trees enjoy a distance labeling scheme where each vertex is assigned an $O(\log^2 n)$-bit label such that given labels of two vertices the distance between them can be inferred in constant time [61]. We can use for our datasets their canonical trees to compactly and distributively encode their approximate distance information. Given a graph dataset G, we first compute in linear time its canonical tree H. Then, we preprocess H in $O(n \log n)$ time (see [61]) to assign each vertex $v \in V$ an $O(\log^2 n)$-bit distance label. Given two vertices $u, v \in V$, we can compute in $O(1)$ time the distance $d_H(u, v)$ from their labels and output this distance as a good estimate for the distance between u and v in G.

Figure 6 demonstrates how accurately canonical trees represent pairwise distances in our datasets. For a given number $\epsilon \geq 1$, we show how many vertex pairs have a distortion less than ϵ, that is, pairs $u, v \in V$ with $\max\left\{\frac{d_H(u,v)}{d_G(u,v)}, \frac{d_G(u,v)}{d_H(u,v)}\right\} < \epsilon$. We can see that H approximates the distances for most vertex pairs with a high level of accuracy. Exact graph distances were preserved in H for at least 40% of the vertex pairs in 8 datasets (EPA, EVA, Erdös, Routeview, Homo, AS_CAIDA_20071105, Dimes 3/2010, and AS_CAIDA_20120601). At least 50% of the vertex pairs of 6 datasets have distance distortion in H less than 1.2. At least 60% of the vertex pairs for 6 datasets have distance distortion less than 1.3. At least 70% of the vertex pairs of 10 datasets have distance distortion less than 1.5. At least 80% of the vertex pairs of 14 datasets have distance distortion less than 2. At least 90% of the vertex pairs of 14 datasets have distance distortion less than 2.2. For the DBLB-coauth dataset, 80% (90%) of the vertex pairs embed into H with distortion no more than 2.2 (2.4, respectively; not shown on the table). For the Amazon dataset, 80% (90%) of the vertex pairs embed into H with distortion no more than 3.2 (3.8, respectively; not shown on the table).

Hence, using embeddings of our datasets into their canonical trees, we obtain for them a compact and efficient approximate distance labeling scheme. Each vertex of a graph dataset G gets an $O(\log^2 n)$-bit label from the canonical tree and the distance between any two vertices of G can be computed with a good level of accuracy in constant time from their labels only.

8.2. Approximating optimal routes

First, we formally define approximate routing labeling schemes. Let s, r be two real numbers. A family $\Re$ of graphs is said to have an $l(n)$-bit (s, r)-approximate routing labeling scheme if there exist a labeling function L and an efficient algorithm/function f, called the *routing decision* or *routing protocol*. The labeling function L labels the vertices of each n-vertex graph in $\Re$ with distinct labels of up to $l(n)$ bits. Given the label of a current vertex v and the label of the destination vertex (in the header of the packet), the algorithm f decides in time polynomial in the length of the given labels, and using only those two labels, whether this packet has already reached its destination, and if not, to which neighbor of v to forward the packet. Furthermore, the routing path from any source x to any destination y produced by this scheme in a graph G from $\Re$ must have length at most $s \cdot d_G(x, y) + r$. For simplicity, $(1, r)$-approximate labeling schemes (distance or routing) are called r-additive labeling schemes, and $(s, 0)$-approximate labeling schemes are called s-multiplicative labeling schemes.

A very good routing labeling scheme exists for trees [69]. An n-vertex tree can be preprocessed in $O(n \log n)$ time so that each vertex is assigned an $O(\log n)$-bit routing label. Given the label of a source vertex and the label of a destination, it is possible to compute in constant time the port number of the edge from the source that appears on the (shortest) path to the destination.

Unfortunately, a canonical tree H of a graph G is not suitable for approximately routing in G; H may have artificial edges (not coming from G) and, therefore, a path of H from a source to a destination may not be available for routing in G. To reduce the problem of routing in G to routing in a tree T, the tree T needs to be a spanning tree of G. Hence, a spanning tree T of G with minimum stretch (i.e., a tree t-spanner of G with $t = \text{ts}(G)$) would be a perfect choice. Unfortunately, finding a tree t-spanner of a graph with minimum t is an NP-hard problem.

For our graph datasets, one can exploit the facts that they have small tree-breadth/tree-length and/or small hyperbolicity.

If the tree-breadth of an n-vertex graph G is ρ then, by a result from [37], G admits a tree $(2\rho\log_2 n)$-spanner constructible in polynomial time. Hence, G enjoys a $(2\rho\log_2 n)$-multiplicative routing labeling scheme with $O(\log n)$-bit labels and $O(1)$ time routing protocol (routing is essentially done in that tree spanner). Another result for graphs with $\text{tb}(G) = \rho$, useful for designing routing labeling schemes, is presented in [36]. It states that every n-vertex graph G with $\text{tb}(G) = \rho$ has a system of at most $\log_2 n$ collective additive tree $(2\rho\log_2 n)$-spanners, that is, a system $\mathcal{T}$ of at most $\log_2 n$ spanning trees of G such that for any two vertices u, v of G there is a tree T in $\mathcal{T}$ with $d_T(u, v) \leq d_G(u, v) + 2\rho\log_2 n$. Furthermore, such a system $\mathcal{T}$ for G can be constructed in polynomial time [36]. By combining this with a result from [38], we obtain that every n-vertex graph G with $\text{tb}(G) = \rho$ enjoys a $(2\rho\log_2 n)$-additive routing labeling scheme with $O(\log^3 n)$-bit labels and $O(1)$ time routing protocol with $O(\log n)$ message initiation time. The approach of [38] is to assign to each vertex of G a label with $O(\log^3 n)$ bits (distance and routing labels coming from $\log_2 n$ spanning trees) and then, using the label of a source vertex v and the label of a destination vertex u, identify in $O(\log n)$ time a best spanning tree in $\mathcal{T}$ to route from v to u.

If the tree-length of an n-vertex graph G is λ then, by result from [32], G enjoys a 6λ-additive routing labeling scheme with $O(\lambda\log^2 n)$-bit labels and $O(\log \lambda)$ time routing protocol.

If the hyperbolicity of an n-vertex graph G is δ then, by result from [24], G enjoys an $O(\delta \log n)$-additive routing labeling scheme with $O(\delta\log^2 n)$-bit labels and $O(\log \delta)$ time routing protocol. Note that, for any graph G, the hyperbolicity of G is at most its tree-length [23].

Thus, for our graph datasets, there exists a very compact labeling scheme (at most $O(\log^2 n)$ or $O(\log^3 n)$ bits per vertex) that encodes logarithmic length routes between any pair of vertices, that is, routes of length at most $d_G(u, v) + \min\left\{O(\delta \log n), 6\lambda, 2\rho\log_2 n\right\} \leq \text{diam}(G) + O(\log n) \leq O(\log n)$ for each vertex pair u, v of G. The latter implies very good navigability of our graph datasets. Recall that, for our graph datasets, $\text{diam}(G) \leq O(\log n)$ holds.

8.3. Approximating diameter and radius

Recall that the *eccentricity* of a vertex v of a graph G, denoted by $\text{ecc}(v)$, is the maximum distance from v to any other vertex of G, that is, $\text{ecc}(v) := \max_{u \in V} d_G(v, u)$. The *diameter* $\text{diam}(G)$ of G is the largest eccentricity of a vertex in G, that is, $\text{diam}(G) := \max_{v \in V} \text{ecc}(v) = \max_{v,u \in V} d_G(u, v)$. The *radius* $\text{rad}(G)$ of G is the smallest eccentricity of a vertex in G, that is, $\text{rad}(G) := \min_{v \in V} \text{ecc}(v)$. A vertex c of G with $\text{ecc}(v) = \text{rad}(G)$ (i.e., a smallest eccentricity vertex) is called a *central vertex* of G. The *center* $C(G)$ of G is the set of all central vertices of G. Let also $F(v) := \{u \in V : d_G(v, u) = \text{ecc}(v)\}$ be the set of vertices of G furthest from v.

In general (even unweighted) graphs, it is still an open problem whether the diameter and/or the radius of a graph G can be computed faster than the time needed to compute the entire distance matrix of G (which requires $O(nm)$ time for a general unweighted graph). Conversely, it is known that both, the diameter and the radius, of a tree T can be calculated in linear time [48]. That can be done using 2 Breadth-First-Search (BFS) scans as follows. Pick an arbitrary vertex u of T. Run a BFS starting from u to find $v \in F(u)$. Run a second BFS starting from v to find $w \in F(v)$. Then, $d_T(v, w) = \text{diam}(T)$, that is, v, w is a *diametral pair* of T, and $\text{rad}(G) = \lfloor (d_T(v, w) + 1)/2 \rfloor$. To find the center of T it suffices to take one or two adjacent middle vertices of the (v, w)-path of T.

Interestingly, Chepoi et al. [23] established that this approach of 2 BFS-scans can be adapted to provide quick (in linear time) and accurate approximations of the diameter,

TABLE 9. Estimation of diameters and radii

$G = (V, E)$	diam(G)	rad(G)	# of BFS scans needed to get diam(G)	Estimated radius, that is, ecc$(\cdot)$ of a middle vertex
PPI	19	11	3	12
Yeast	11	6	3	6
DutchElite	22	12	4	13
EPA	10	6	2	7
EVA	18	10	2	10
California	13	7	2	8
Erdös	4	2	2	3
Routeview	10	5	2	5
Homo	10	5	2	6
AS_CAIDA_20071105	17	9	2	9
Dimes 3/2010	8	4	2	5
Aqualab 12/2007- 09/2008	9	5	2	5
AS_CAIDA_20120601	10	5	2	5
itdk0304	26	14	2	15
DBLB-coauth	23	12	2	14
Amazon	47	24	2	26

radius, and center of any finite set S of δ-hyperbolic geodesic spaces and graphs. In particular, for a δ-hyperbolic graph G, it was shown that if $v \in F(u)$ and $w \in F(v)$, then $d_G(v, w) \geq \text{diam}(G) - 2\delta$ and $\text{rad}(G) \leq \lfloor (d_G(v, w) + 1)/2 \rfloor + 3\delta$. Furthermore, the center $C(G)$ of G is contained in the ball of radius $5\delta + 1$ centered at a middle vertex c of any shortest path connecting v and w in G.

As our graph datasets have small hyperbolicities, according to [23], a few (2, 3, 4, ...) BFS-scans, each next starting at a vertex last visited by the previous scan, should provide a pair of vertices x and y such that $d_G(x, y)$ is close to the diameter diam(G) of G. Surprisingly (see Table 9), a few BFS-scans were sufficient to get the exact diameters of all of our graph datasets: for 13 datasets, 2 BFS-scans (just like for trees) were sufficient to find the exact diameter of a graph. Two datasets needed three BFS-scans to find the diameter, and only one dataset required four BFS-scans to get the diameter. We also computed the eccentricity of a middle vertex of a longest shortest path produced by these few BFS-scans and reported this eccentricity as an estimation for the radius. It turned out that the eccentricity of that middle vertex was equal to the exact radius for six datasets, was only one unit larger than the exact radius for eight datasets, and for two datasets was only two units larger than the exact radius.

9. CONCLUSION

Based on solid theoretical foundations, we presented strong evidence that a number of real-world networks, taken from different domains such as Internet measurements, biological datasets, web graphs, social and collaboration networks, exhibit metric tree-like structures. We investigated a few graph parameters, namely, tree-distortion and tree-stretch, tree-length and tree-breadth, Gromov's hyperbolicity, cluster-diameter and cluster-radius in a layering partition of a graph. Such parameters capture and quantify this phenomenon of being metrically close to a tree. Recent advances in theory allowed us to calculate or accurately estimate these parameters for sufficiently large networks. All these parameters are at most constant or (poly)logarithmic factors apart from each other. Specifically, for every n-vertex, m-edge graph G, the graph parameters td(G), tl(G), tb(G), $\Delta_s(G)$, $R_s(G)$ are within small constant factors from each other. The parameters ts(G) and $\delta(G)$ are within a factor of at most $O(\log n)$ from td(G), tl(G), tb(G), $\Delta_s(G)$, $R_s(G)$. The tree-stretch ts(G) is within a factor of at most $O(\log^2 n)$ from the hyperbolicity $\delta(G)$. One can summarize those relationships with the following chains of inequalities:

$$\begin{aligned}
&\delta(G) \leq \Delta_s(G) \leq O(\delta(G)\log n);\\
&\qquad R_s(G) \leq \Delta_s(G) \leq 2R_s(G);\ \text{tb}(G) \leq \text{tl}(G) \leq 2\text{tb}(G);\\
&\delta(G) \leq \text{tl}(G) \leq \text{td}(G) \leq \text{ts}(G)\\
&\qquad \leq 2\text{tb}(G)\log_2 n \leq O(\delta(G)\log^2 n);\\
&\text{tl}(G) - 1 \leq \Delta_s(G) \leq 3\text{tl}(G) \leq 3\text{td}(G) \leq 3(2\Delta_s(G) + 2);\\
&\text{tb}(G) - 1 \leq R_s(G) \leq 3\text{tb}(G) \leq 3\lceil \text{ts}(G)/2 \rceil .
\end{aligned}$$

If one of these parameters or its average version has a small value for a large-scale network (e.g., it is not larger than $\log_2(n + m)$), we say that that network has a metric tree-like structure. Among these parameters, the theoretically smallest ones are $\delta(G)$, $R_s(G)$ and tb(G) (with tb(G) being at most $R_s(G) + 1$). Our experiments showed that average versions of $\Delta_s(G)$ and of td(G) have also very small values for the investigated graph datasets.

In Table 10, we provide a summary of metric tree-likeness measurements calculated for our datasets. Figure 7 shows four important metric tree-likeness measurements (scaled) in comparison. Figure 8 gives pairwise dependencies between those measurements (one as a function of another).

From the experimental results, we observe that in almost all cases the measurements seem to be monotonic with respect to each others. The smaller one measurement for a given dataset, the smaller the other measurements are. There are also a few exceptions. For example, the EVA dataset has relatively large cluster-diameter, $\Delta_s(G) = 9$, but small hyperbolicity, $\delta(G) = 1$. Conversely, the Erdös dataset has $\Delta_s(G) = 4$ while its hyperbolicity $\delta(G)$ is equal to 2 (see Fig. 8a). Yet, the Erdös dataset has better embedability (smaller average distortions) into trees H, H_ℓ, and H'_ℓ than that of EVA, suggesting that the (average) cluster-diameter may have a greater impact on the embedability into trees H, H_ℓ, and H'_ℓ. Comparing the measurements obtained for Erdös with those obtained for Homo, we observe that both have the same hyperbolicity 2; however, Erdös has better embedability (see the average distortion) into trees H, H_ℓ, H'_ℓ. This could be explained by a smaller $\Delta_s(G)$ and average diameter of clusters in the Erdös dataset. Comparing measurements obtained for PPI with those obtained for California (the same holds for AS_CAIDA_20071105 vs. AS_CAIDA_20120601), both

TABLE 10. Summary of tree-likeness measurements

$G = (V, E)$	diam(G)	rad(G)	Cluster-diameter $\Delta_s(G)$	Average diameter of clusters in $\mathcal{LP}(G, s)$	$\delta(G)$	Tree H average distortion*	H_ℓ average distortion	H'_ℓ average distortion	Cluster-radius $R_s(G)$
PPI	19	11	8	0.12	3.5	1.38	5.71	5.30	4
Yeast	11	6	6	0.12	2.5	1.32	4.38	3.79	4
DutchElite	22	12	10	0.07	4	1.41	5.45	6.53	6
EPA	10	6	6	0.07	2.5	1.27	4.51	4.07	4
EVA	18	10	9	0.03	1	1.14	5.83	7.78	5
California	13	7	8	0.09	3	1.35	4.16	4.99	4
Erdös	4	2	4	0.00	2	1.05	3.09	3.07	2
Routeview	10	5	6	0.06	2.5	1.24	4.28	4.80	3
Homo	10	5	5	0.03	2	1.19	4.65	3.97	3
AS_CAIDA_20071105	17	9	6	0.06	2.5	1.23	4.24	4.77	3
Dimes 3/2010	8	4	4	0.06	2	1.20	3.44	3.36	2
Aqualab 12/2007- 09/2008	9	5	6	0.06	2	1.28	4.23	4.54	3
AS_CAIDA_20120601	10	5	6	0.06	2	1.16	4.11	4.53	3
itdk0304	26	14	11	0.27	–	1.57	5.37	5.71	6
DBLB-coauth	23	12	11	0.45	–	1.74	5.58	5.13	7
Amazon	47	24	21	0.49	–	2.47	8.82	7.87	12

$= \frac{(\text{avg. distortion right} \times \#\ \text{right pairs}) + (\text{avg. distortion left} \times \#\ \text{left pairs}) + \#\ \text{undistorted pairs}}{\binom{n}{2}}$

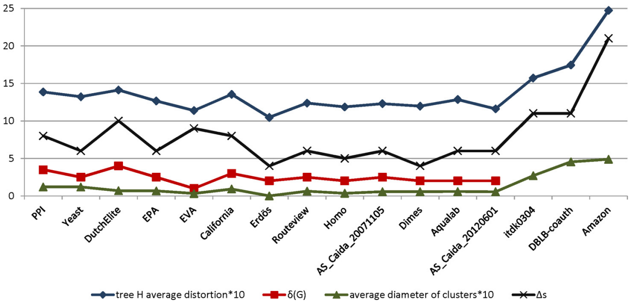


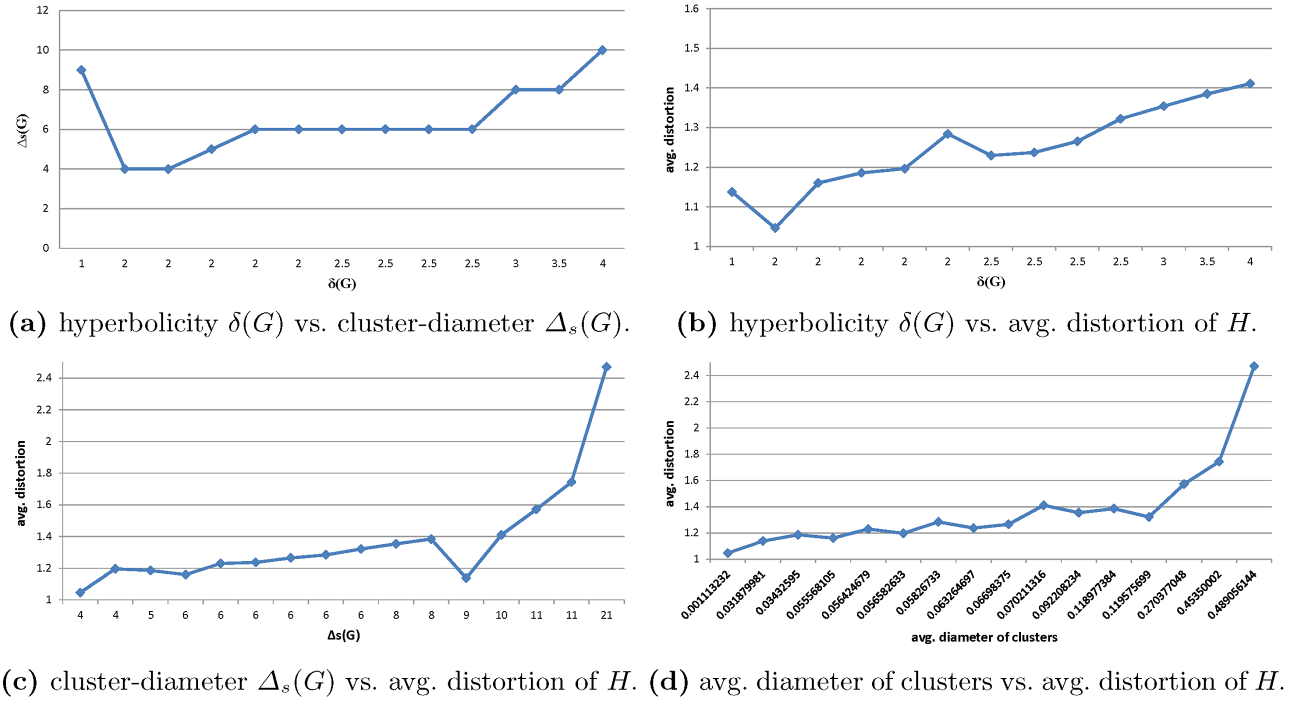
FIG. 7. Four tree-likeness measurements scaled. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

have the same $\Delta_s(G)$ and $R_s(G)$ values; however, California (AS_CAIDA_20120601) has smaller hyperbolicity and average diameter of clusters. We also observe that the datasets Routeview and AS_CAIDA_20071105 have same values of $\Delta_s(G), R_s(G)$, and $\delta(G)$ but AS_CAIDA_20071105 has a relatively smaller average diameter of clusters. This could explain why AS_CAIDA_20071105 has relatively better embedability into H, H_ℓ, and H'_ℓ than Routeview. We can see that the difference in average diameters of clusters was relatively small, resulting in a small difference in embedability.

From these observations, one can suggest that, for classifications of our datasets, all these tree-likeness measurements are important; they collectively capture and explain their metric tree-likeness. We suggest that metric tree-likeness measurements in conjunction with other local characteristics of networks, such as the degree distribution and clustering coefficients, provide a more complete unifying picture of network structures.

We conclude this article with a few open theoretical questions.

1. We observed that many real-world networks do exhibit a metric tree-like structure. Is it just because all those networks are connected, have relatively small number of edges, high clustering coefficient, and Power-Law degree distribution, or is metric tree-likeness is an independent feature of those networks? Generally, can one identify what structural obstacles (perhaps, large isometric cycles, grids, etc.) govern metric tree-unlikeness of a graph?
2. With respect to specific tree-likeness parameters of a graph, we are interested in the following questions. We have listed above all known inequalities between different metric tree-likeness parameters of a graph. Can those bounds be improved? Are they sharp?
3. We have mentioned in Section 6 that tree H'_ℓ provides a 6-approximate solution to the problem of minimum distortion noncontractive embedding of graphs into trees. In Proposition 9, we showed that there is a simple linear-time 3-approximation algorithm for computing the tree-breadth of a graph. Can these approximation bounds be improved? What inapproximability results can be proven for these problems?
4. We know (see Proposition 13) that if a graph G is noncontractively embeddable into a tree with distortion td(G), then it is embeddable into a spanning tree with stretch at most 2td$(G)\log_2 n$, and such an embedding and a corresponding spanning tree can be found in polynomial time. We know also (see Proposition 14) that every δ-hyperbolic graph G admits a tree $O(\delta\log^2 n)$-spanner and such a spanning tree for G can be constructed in polynomial time. Are these results best possible? Can they be improved? In particular, does any δ-hyperbolic graph G admit a tree $O(\delta \log n)$-spanner?
5. In Section 8, we listed a number of problems that can be solved efficiently if tree-likeness parameters of a graph are bounded. What other interesting problems can be solved on metric tree-like graphs more efficiently than on general graphs?



(a) hyperbolicity $\delta(G)$ vs. cluster-diameter $\Delta_s(G)$.

(b) hyperbolicity $\delta(G)$ vs. avg. distortion of H.

(c) cluster-diameter $\Delta_s(G)$ vs. avg. distortion of H.

(d) avg. diameter of clusters vs. avg. distortion of H.

FIG. 8. Tree-likeness measurements: pairwise comparison. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

ACKNOWLEDGMENTS

The authors are very grateful to anonymous referees and to Dr. Douglas Shier for many useful suggestions.

REFERENCES

[1] Pages linking to www.epa.gov. Obtained from Jon Kleinberg's web page, webpage. Available at: http://www.cs.cornell.edu/courses/cs685/2002fa/.

[2] University of Oregon Route Views Project, webpage. Available at: http://www.routeviews.org/.

[3] I. Abraham, M. Balakrishnan, F. Kuhn, D. Malkhi, V. Ramasubramanian, and K. Talwar, Reconstructing approximate tree metrics, Proc Twenty-Sixth Ann ACM Symp Principles Distrib Comput (PODC '07), Portland, Oregon, USA, 2007, pp. 43–52.

[4] A.B. Adcock, B.D. Sullivan, and M.W. Mahoney, Tree-like structure in large social and information networks, 13th Int Conference Data Mining (ICDM), IEEE, Dallas, Texas, USA, 2013, pp. 1–10.

[5] R. Agarwala, V. Bafna, M. Farach, M. Paterson, and M. Thorup, On the approximability of numerical taxonomy (fitting distances by tree metrics), SIAM J Comput 28 (1999), 1073–1085.

[6] M. Badoiu, E.D. Demaine, M. Hajiaghayi, A. Sidiropoulos, and M. Zadimoghaddam, Ordinal embedding: Approximation algorithms and dimensionality reduction, Approximation, Randomization, Combinatorial Optimization. Algorithms, Techniques, APPROX-RANDOM, Boston, MA, USA, 2008, pp. 21–34.

[7] M. Badoiu, P. Indyk, and A. Sidiropoulos, Approximation algorithms for embedding general metrics into trees, Proc Eighteenth Ann ACM-SIAM Symp Discr Algorithms (SODA '07), SIAM, New Orleans, Louisiana, USA, 2007, pp. 512–521.

[8] A.L. Barabasi and R. Albert, Emergence of scaling in random networks, Science 286 (1999), 509–512.

[9] A.L. Barabási, R. Albert, and H. Jeong, Scale-free characteristics of random networks: The topology of the world-wide web, Physica A 281 (2000), 69–77.

[10] Y. Baryshnikov and G. Tucci, Asymptotic traffic flow in a hyperbolic network, 5th Int Symp Commun Control Signal Process (ISCCSP), Rome, Italy, 2012, pp. 1–4.

[11] V. Batagelj and A. Mrvar, Some analyses of Erdös collaboration graph, Soc Netw 22 (2000), 173–186, Available at: http://vlado.fmf.uni-lj.si/pub/networks/data/Erdos/Erdos02.net.

[12] M. Boguñá, D. Krioukov, and K.C. Claffy, Navigability of complex networks, Nat Phys 5 (2009), 74–80.

[13] A. Brandstädt, V. Chepoi, and F.F. Dragan, Distance approximating trees for chordal and dually chordal graphs, J Algorithms 30 (1999), 166–184.

[14] G. Brinkmann, J. Koolen, and V. Moulton, On the hyperbolicity of chordal graphs, Ann Comb 5 (2001), 61–69.

[15] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, G. Li, and R. Chen, Topological structure analysis of the protein–protein interaction network in budding yeast, Nucleic Acids Res 31 (2003), 2443–2450, Available at: http://vlado.fmf.uni-lj.si/pub/networks/data/bio/Yeast/Yeast.htm.

[16] L. Cai and D.G. Corneil, Tree spanners, SIAM J Discr Math 8 (1995), 359–387.

[17] CAIDA, The CAIDA AS relationships dataset, webpage. Available at: http://www.caida.org/data/active/as-relationships.

[18] CAIDA, The internet topology data kit #0304, webpage. Available at: http://www.caida.org/data/active/internet-topology-data-kit.

[19] CAIDA, The CAIDA AS relationships dataset, webpage. Available at: http://www.caida.org/data/active/as-relationships.

[20] K. Chen, D.R. Choffnes, R. Potharaju, Y. Chen, F.E. Bustamante, D. Pei, and Y. Zhao, Where the sidewalk ends: Extending the internet as graph using traceroutes from P2P users, Proc 5th Int Conf Emerg Netw Exp Technol (CoNEXT '09), ACM, Rome, Italy, 2009, pp. 217–228, Available at: http://www.aqualab.cs.northwestern.edu/projects.

[21] W. Chen, W. Fang, G. Hu, and M.W. Mahoney, On the hyperbolicity of small-world and tree-like random graphs, 23rd Int Symp Algorithms Comput (ISAAC 2012), Vol. 7676 of *Lecture Notes in Computer Science*, Springer, Springer-Verlag, Berlin, Germany 2012, pp. 278–288.

[22] V. Chepoi and F.F. Dragan, A note on distance approximating trees in graphs, Eur J Comb 21 (2000), 761–766.

[23] V. Chepoi, F.F. Dragan, B. Estellon, M. Habib, and Y. Vaxès, Diameters, centers, and approximating trees of delta-hyperbolic geodesic spaces and graphs, Symp Comput Geometry, ACM, College Park, Maryland, USA, 2008, pp. 59–68.

[24] V. Chepoi, F.F. Dragan, B. Estellon, M. Habib, Y. Vaxès, and Y. Xiang, Additive spanners and distance and routing labeling schemes for hyperbolic graphs, Algorithmica 62 (2012), 713–732.

[25] V. Chepoi, F.F. Dragan, I. Newman, Y. Rabinovich, and Y. Vaxès, Constant approximation algorithms for embedding graph metrics into trees and outerplanar graphs, Discr Comput Geom 47 (2012), 187–214.

[26] V. Chepoi and B. Estellon, Packing and covering δ-hyperbolic spaces by balls, Approximation, Randomization, Combinatorial Optimization. Algorithms, Techniques, APPROX-RANDOM, Princeton University, NJ, USA, 2007, pp. 59–73.

[27] F.R.K. Chung and L. Lu, The average distance in a random graph with given expected degrees, Internet Math 1 (2003), 91–113.

[28] N. Cohen, D. Coudert, and A. Lancin, Exact and approximate algorithms for computing the hyperbolicity of large-scale graphs, Rapport de recherche RR-8074, INRIA, Research Centre Sophia Antipolis – Méditerranée, September 2012.

[29] F. de Montgolfier, M. Soto, and L. Viennot, Treewidth and hyperbolicity of the Internet, 10th IEEE Int Symp Netw Comput Appl (NCA), IEEE, Cambridge, Massachusetts, USA, 2011, pp. 25–32.

[30] W. de Nooy, The network data on the administrative elite in The Netherlands in April-June 2006, webpage. Available at: http://vlado.fmf.uni-lj.si/pub/networks/data/2mode/DutchElite.htm. Accessed on November 14, 2013.

[31] R. Diestel, Graph theory, 4th edition, Vol. 173 of *Graduate texts in mathematics*, Springer, Springer-Verlag, Heidelberg, Germany, 2012.

[32] Y. Dourisboure, Compact routing schemes for generalised chordal graphs, J Graph Algorithms Appl 9 (2005), 277–297.

[33] Y. Dourisboure, F.F. Dragan, C. Gavoille, and C. Yan, Spanners for bounded tree-length graphs, Theor Comput Sci 383 (2007), 34–44.

[34] Y. Dourisboure and C. Gavoille, Tree-decompositions with bags of small diameter, Discr Math 307 (2007), 2008–2029.

[35] F.F. Dragan, Tree-like structures in graphs: A metric point of view, 39th Int Workshop Graph-Theoretic Concepts in Comput Sci (WG 2013), Lübeck, Germany, 2013, pp. 1–4.

[36] F.F. Dragan and M. Abu-Ata, Collective additive tree spanners of bounded tree-breadth graphs with generalizations and consequences, Theor Comput Sci 547 (2014), 1–17.

[37] F.F. Dragan and E. Köhler, An approximation algorithm for the tree t-spanner problem on unweighted graphs via generalized chordal graphs, Algorithmica 69 (2014), 884–905.

[38] F.F. Dragan, C. Yan, and D.G. Corneil, Collective tree spanners and routing in AT-free related graphs, J Graph Algorithms Appl 10 (2006), 97–122.

[39] Y. Emek and D. Peleg, Approximating minimum max-stretch spanning trees on unweighted graphs, SIAM J Comput 38 (2008), 1761–1781.

[40] M. Faloutsos, P. Faloutsos, and C. Faloutsos, On power-law relationships of the Internet topology, Proc Conf Appl, Technologies, Architectures, Protocols Comput Commun (SIGCOMM '99), Cambridge, Massachusetts, USA, 1999, pp. 251–262.

[41] H. Fournier, A. Ismail, and A. Vigneron, Computing the Gromov hyperbolicity of a discrete metric space, Inf Process Lett 115 (2015), 576–579.

[42] C. Gavoille and O. Ly, Distance labeling in hyperbolic graphs, 16th Int Symp Algorithms Comput (ISAAC 2005), Sanya, Hainan, China, 2005, pp. 1071–1079.

[43] C. Gavoille and D. Peleg, Compact and localized distributed data structures, Distrib Comput 16 (2003), 111–120.

[44] E. Ghys and P. de la Harpe eds., Sur les groupes hyperboliques d'après M. Gromov, Vol. 83, Progress in Math, Birkhäuser Boston, Inc., Boston, MA, USA, 1990.

[45] A. Goldman, Optimal center location in simple networks, Transportation Sci 5 (1971), 212–221.

[46] M. Gromov, Hyperbolic groups: Essays in group theory, MSRI Publ 8 (1987), 75–263.

[47] A. Gupta, Steiner points in tree metrics don't (really) help, Proc Twelfth Ann ACM-SIAM Symp Discr Algorithms (SODA '01), ACM/SIAM, Washington, DC, USA, 2001, pp. 220–227.

[48] G.Y. Handler, Minimax location of a facility in an undirected tree graph, Transportation Sci 7 (1973), 287–293.

[49] H. Jeong, S.P. Mason, A.L. Barabsi, and Z.N. Oltvai, Lethality and centrality in protein networks, Nature 411 (2001), 41–42, Available at: http://www3.nd.edu/networks/resources.htm.

[50] M.J. Kao, D.T. Lee, and D. Wagner, Approximating metrics by tree metrics of small distance-weighted average stretch, (2013). arXiv:1301.3252.

[51] W.S. Kennedy, O. Narayan, and I. Saniee, On the hyperbolicity of large-scale networks, (2013). ArXiv:1307.0031.

[52] J.M. Kleinberg, Authoritative sources in a hyperlinked environment, J ACM 46 (1999), 604–632, Available at: http://www.cs.cornell.edu/courses/cs685/2002fa/.

[53] J.M. Kleinberg, The small-world phenomenon: An algorithm perspective, Proc Thirty-Second Ann ACM Symp Theory Comput (STOC '00), Portland, OR, USA, 2000, pp. 163–170.

[54] J.M. Kleinberg, Small-world phenomena and the dynamics of information, Neural Informat Process Syst (NIPS 2001), Vancouver, British Columbia, Canada, 2001, pp. 431–438.

[55] R. Kleinberg, Geographic routing using hyperbolic space, 26th IEEE Int Conf Comput Commun (INFOCOM 2007), IEEE, Anchorage, Alaska, USA, 2007, pp. 1902–1909.

[56] R. Krauthgamer and J.R. Lee, Algorithms on negatively curved spaces, 47th Ann IEEE Conf Foundations Comput Sci (FOCS 2006), Berkeley, CA, USA, 2006, pp. 119–132.

[57] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney, Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters, Internet Math 6 (2009), 29–123.

[58] D. Lokshtanov, On the complexity of computing treelength, Discr Appl Math 158 (2010), 820–827.

[59] O. Narayan and I. Saniee, Large-scale curvature of networks, Phys Rev E 84 (2011), 066108.

[60] K. Norlen, G. Lucas, M. Gebbie, and J. Chuang, EVA: Extraction, visualization and analysis of the telecommunications and media ownership network, Proc Int Telecommunications Soc 14th Biennial Conference (ITS2002), Seoul, Korea, 2002. Available at: http://vlado.fmf.uni-lj.si/pub/networks/data/econ/Eva/Eva.htm.

[61] D. Peleg, Proximity-preserving labeling schemes and their applications, 25th Int Workshop Graph-Theoretic Concepts in Comput Sci (WG 1999), Vol. 1665 of *Lecture Notes in Computer Science*, Springer, Ascona, Switzerland, 1999, pp. 30–41.

[62] D. Peleg, Distributed computing: A locality-sensitive approach, SIAM Monographs on Discrete Math. Appl., SIAM, Philadelphia, 2000.

[63] N. Robertson and P.D. Seymour, Graph minors II: Algorithmic aspects of tree-width, J Algorithms 7 (1986), 309–322.

[64] I. Saniee and G. Tucci, Scaling of congestion in small world networks, (2012). arXiv:1201.4291.

[65] C. Semple and M. Steel, Phylogenetics, Oxford University Press, New York, USA, 2003.

[66] Y. Shavitt and E. Shir, DIMES: Let the internet measure itself, Comput Commun Rev 35 (2005), 71–74, Available at: http://www.netdimes.org.

[67] Y. Shavitt and T. Tankel, Hyperbolic embedding of internet graph for distance estimation and overlay construction, IEEE/ACM Trans Netw 16 (2008), 25–36.

[68] C. Stark, B.J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, BioGRID: A general repository for interaction datasets, Nucleic Acids Res 34 (2006), 535–539, Available at: http://thebiogrid.org/, release 3.2.99.

[69] M. Thorup and U. Zwick, Compact routing schemes, Proc Thirteenth Ann ACM Symp Parallel Algorithms Architectures (SPAA '01), Crete Island, Greece, 2001, pp. 1–10.

[70] D. Watts and S. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (1998), 440–442.

[71] Y. Wu and C. Zhang, Hyperbolicity and chordality of a graph, Electron J Comb 18 (2011), Paper #P43, 22 pages.

[72] J. Yang and J. Leskovec, Defining and evaluating network communities based on ground-truth, 12th Int Conference Data Mining (ICDM 2012), IEEE, IEEE Computer Society, Brussels, Belgium, 2012, pp. 745–754, Available at: http://snap.stanford.edu/data/com-Amazon.html, http://snap.stanford.edu/data/com-DBLP.html.