$\delta$-HYPERBOLICITY IN REAL-WORLD NETWORKS: ALGORITHMIC ANALYSIS AND IMPLICATIONS

A dissertation submitted to
Kent State University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

by
Hend Mohammed Alrasheed
May 2018
# Table of Contents

Table of Contents ................................. iii

List of Figures ................................ vii

List of Tables ................................... ix

1 Introduction ..................................... 1
   1.1 Research contribution ......................... 6
       1.1.1 Publications .......................... 8
   1.2 Structural properties of graphs .................... 9
   1.3 Gromov $\delta$-hyperbolic spaces ................. 11

2 Preliminaries and Notations .................... 15
   2.1 Basic definitions ........................... 15
   2.2 Eccentricity ................................ 21
       2.2.1 Eccentricity layering of a graph ............ 22
   2.3 Graph embedding ............................ 23
   2.4 Graph centrality measures ...................... 25

3 $\delta$-Hyperbolicity in Graphs .................. 28
3.1 Definitions and basic properties ................. 28
3.2 Why do we study $\delta$-hyperbolicity in graphs? .. 37
3.3 Computational aspect of $\delta$-hyperbolicity .......... 41

I $\delta$-Hyperbolicity in Graphs - A Structural Point of View 45

4 Core-Periphery Models for Graphs Based on their $\delta$-Hyperbolicity - an Example Using Biological Networks 47
4.1 Introduction ........................................ 49
4.2 Background and related work ...................... 52
4.2.1 Core-periphery and network centrality in complex networks .... 52
4.2.2 Biological networks and the core-periphery structure ......... 53
4.3 Datasets ............................................ 54
4.4 $\delta$-Hyperbolicity and graph classification ............ 56
4.4.1 Hyperbolicity of biological networks ................. 57
4.4.2 Analysis and discussion .......................... 58
4.5 Core-Periphery models based on $\delta$-hyperbolicity .......... 61
4.5.1 Eccentricity-based bending property of $\delta$-hyperbolic graphs .... 62
4.5.2 Core-periphery identification using the eccentricity-based bending property ...................................... 70
4.6 Concluding remarks .................................. 78

5 Interplay Between $\delta$-Hyperbolicity and the Core-Periphery Structure in Graphs 82
5.1 Introduction ........................................ 84
5.2 Datasets ............................................ 86
5.3 $\delta$-Hyperbolicity in graphs .............................................. 89
  5.3.1 $\delta$-Hyperbolicity and dominated vertices ......................... 89
  5.3.2 $\delta$-Hyperbolicity and restricted path lengths ..................... 94
5.4 $\delta$-Hyperbolicity and the core-periphery structure .................. 97
  5.4.1 The minimum-cover-set core ........................................... 99
  5.4.2 $k$-core ............................................................... 101
5.5 Case studies .............................................................. 103
  5.5.1 Biological network ...................................................... 103
  5.5.2 AS-graph .............................................................. 104
5.6 Concluding remarks ....................................................... 104

II Eccentricity Approximating Trees ........................................... 106

6 Eccentricity Approximating Trees in $\delta$-Hyperbolic Graphs .......... 109
  6.1 Introduction ............................................................. 110
  6.2 Preliminaries ............................................................. 114
  6.3 Fast approximation of eccentricities ................................... 114
    6.3.1 Conditional lower bounds on complexities ......................... 115
    6.3.2 Fast additive approximations .................................... 116
  6.4 Experimentation on some real-world networks .......................... 123
    6.4.1 Estimation of eccentricities ..................................... 125

7 Eccentricity Approximating Trees in $(\alpha_1, \Delta)$-Metric Graphs .. 134
  7.1 Introduction ............................................................. 136
  7.2 Preliminaries ............................................................. 138
  7.3 Eccentricity function on $(\alpha_1, \Delta)$-metric graphs ............... 141
7.4 Eccentricity approximating spanning tree construction . . . . . . . . . 145
  7.4.1 Tree construction for unimodal eccentricity functions . . . . . . . 145
  7.4.2 Tree construction for eccentricity functions that are not unimodal . 147
7.5 Experimental results for some real-world networks . . . . . . . . . . . 150
  7.5.1 Datasets . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 152
  7.5.2 Analysis of vertex localities and centers . . . . . . . . . . . . . . . 154
  7.5.3 Eccentricity approximating tree construction and analysis . . . . . 157
7.6 Concluding remarks . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 162

8 Eccentricity $k$-Approximating Tree Algorithms - Comparison 164
  8.1 The algorithms . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 164
  8.2 Comparison . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 166

9 Conclusions and Future Directions 168

Bibliography 171
## List of Figures

1.1 Graph and vertex eccentricity. .................................................. 3
1.2 A visualization of the hyperbolic plane using the Poincaré disc model . 5
1.3 Parallel lines in (a) Euclidean space and (b) hyperbolic space. .......... 13
1.4 Triangles in (a) Euclidean space and (b) hyperbolic space. ............... 13
1.5 An example of a hyperbolic space ............................................ 14

2.1 Diameter, radius, and center in a graph. .................................... 17
2.2 Some common graph types. ..................................................... 20
2.3 Eccentricity Layering of a graph ............................................. 22
2.4 Graph eccentricity layering example ........................................ 23

3.1 The embedding of a complete graph into a weighted tree. ................. 29
3.2 δ-Hyperbolicity examples. ..................................................... 30
3.3 Geodesic triangles, Gromov product, and tripods. .......................... 32
3.4 Four-point-condition definition for δ-hyperbolicity. ......................... 33
3.5 Slim triangles definition for δ-hyperbolicity. ............................... 34
3.6 Thin triangles definition for δ-hyperbolicity. ............................... 35
3.7 A graph with two bi-connected components. ............................... 43
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Distribution of quadruples over different values of δ-hyperbolicity</td>
<td>57</td>
</tr>
<tr>
<td>4.2 Distribution of quadruples over different values of δ-hyperbolicity (three datasets)</td>
<td>58</td>
</tr>
<tr>
<td>4.3 Classification graphs based on their δ-hyperbolicity</td>
<td>60</td>
</tr>
<tr>
<td>4.4 Graph eccentricity layering</td>
<td>63</td>
</tr>
<tr>
<td>4.5 Distribution of vertices over different layers of the graph’s eccentricity layering</td>
<td>64</td>
</tr>
<tr>
<td>4.6 A case in which shortest paths between a diamteral pair do not bend towards the graph’s center</td>
<td>68</td>
</tr>
<tr>
<td>4.7 (\mu_k) values for each network in the graph datasets</td>
<td>71</td>
</tr>
<tr>
<td>4.8 A simplified illustration of the eccentricity layering of a graph and the Maximum-Peak model</td>
<td>72</td>
</tr>
<tr>
<td>4.9 The percentage of the uncovered vertex pairs after the orderly addition of vertices to the core set</td>
<td>78</td>
</tr>
<tr>
<td>5.1 A graph with dominated vertices</td>
<td>90</td>
</tr>
<tr>
<td>5.2 Illustration of Lemma 9</td>
<td>95</td>
</tr>
<tr>
<td>5.3 (p,\delta)-Hyperbolicity of real-world and artificial networks</td>
<td>97</td>
</tr>
<tr>
<td>5.4 (k)-core example</td>
<td>102</td>
</tr>
<tr>
<td>7.1 The relationships between the different graph types used in this chapter</td>
<td>138</td>
</tr>
<tr>
<td>7.2 Forbidden isometric subgraph (W_6^{++})</td>
<td>139</td>
</tr>
<tr>
<td>7.3 An (\alpha_1)-metric graph</td>
<td>143</td>
</tr>
<tr>
<td>7.4 Distribution of vertices with (loc(\cdot) &gt; 1) over different layers of the eccentricity layering in each graph</td>
<td>156</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Summary of notations. ................................................. 16

4.1 Graph datasets and their parameters. .............................. 55
4.2 The hyperbolicity of each graph in the datasets. ................. 56
4.3 The ratio of the $\delta$-hyperbolicity to logarithm of the size and logarithm of
the diameter for each network in the datasets. ...................... 62
4.4 Distribution of vertices over different layers with respect to the graph’s ec-
centricity layering. .................................................. 65
4.5 The effect of the distance between vertex pairs on the bending property. 67
4.6 The absolute and the effective curvity for our graph datasets. .... 69
4.7 The cores of the graph datasets based on the Maximum-Peak model. 74
4.8 The cores of the graph datasets based on the Minimum Cover Set model. 77
4.9 Summary of the graph datasets’ parameters and cores. ............ 80

5.1 Graph datasets and their parameters. .............................. 87
5.2 Statistics of dominated vertices. ................................. 91
5.3 p-$\delta$-Hyperbolicity. ........................................... 93
5.4 p-$\delta$-hyperbolicity and the center of the graph. ................. 98
5.5 $\delta$-Hyperbolicity and the graph’s minimum-cover-set core. ......... 99
5.6 $\delta$-hyperbolicity and the graph’s $k$-core. ............................. 102

6.1 Graph datasets and their parameters. ................................. 124
6.2 Eccentricity estimation based on Proposition 8. ......................... 126
6.3 Distribution of distortion values listed in Table 6.2 ...................... 127
6.4 Eccentricity estimation based on Proposition 9. ......................... 129
6.5 Distribution of distortion values listed in Table 6.4 ...................... 130
6.6 Comparison of the three proposed eccentricity approximating BFS-trees. . 132

7.1 Graph datasets and their parameters. ................................. 151
7.2 Percent of vertices with localities equal to 1 and larger than 1 in each graph of the datasets. .................................................. 155
7.3 The weak diameters and weak radii of the centers of each graph in the datasets. 157
7.4 A spanning tree $T$ constructed by Algorithm EAST. ...................... 160
7.5 A tree $T$ constructed by Algorithm EAT. ................................. 162

8.1 Comparison of the main characteristics of the three eccentricity $k$-approximating tree algorithms. ........................................ 165
8.2 Graph datasets and their parameters. ..................................... 166
8.3 Comparison of the four eccentricity $k$-approximating algorithms: .......... 167
Acknowledgments

First, I would like to thank my advisor Dr. Feodor F. Dragan for his advice and support. This dissertation would not have been possible without the direction and encouragement he has given me over the last few years I spent at Kent State University. I would also like to thank my dissertation committee members, Dr. Javed Khan, Dr. Hassan Peyravi, and Dr. Dmitry Ryabogin for their advice and comments. I would like to thank the faculty and staff of the Department of Computer Science at Kent State University for their help and support.

In addition, I have been very privileged to get to know and to collaborate with many other great people who became friends over the last several years. I learned a lot from you about life, research, how to tackle new problems and how to develop techniques to solve them. Of course, life would not be the same without the fellow computer scientists and friends.

Last but definitely not least, I wish to thank my wonderful husband Fahad and two daughters Yara and Joud. None of this would ever have been possible without their love, their fantastic patience, and continues support. I also wish to thank my parents, siblings, and friends for endless love and encouragement.
Chapter 1

Introduction

Networks are primarily used to denote relational data among different entities in a specific system. Examples include social and communication networks, transportation networks, biological networks, and the Internet. Networks are modeled mathematically using the notion of graphs, and graph theory provides rich appropriate tools for network analysis. Formally, a graph $G = (V, E)$ consists of a set of vertices (or nodes) $V$ and a set of edges $E$ that connect pairs of interacting vertices. The cardinality of $V$ and $E$ are usually denoted by $n$ and $m$ respectively.

The main interest behind a lot of research has been in understanding the structural properties in the topology of large graphs (connection properties of its vertices and edges). The goal is to exploit any hidden properties to increase the efficiency of existing algorithms, as well as to propose new algorithms that are more “natural” to the structure that a graph exhibits. Moreover, understanding the structure of a graph often reveals the underlying foundations behind its several phenomena and patterns. Knowing that a given graph does not have a completely random structure improves the general understanding of its different aspects. Therefore, identifying any structural properties that a graph may possess could
indeed facilitate analyzing it. For example, many difficult optimization problems on graphs become tractable when restricted to some classes of graphs with well-known properties.

Describing the structural properties of a graph is one aspect of graph analysis that aims at capturing particular features of a graph and translating them into predefined parameters with meaningful numbers or mathematical relationships. Structural properties such as the power-law degree distribution (i.e., whether some vertices have many more incident edges compared to the majority of vertices) and the clustering coefficient (the degree to which vertices tend to form clusters) have been studied widely in the literature for a wide range of network types. Those properties are tied to the structure of vertex neighborhoods, i.e., they provide a local type of network analysis.

Another aspect of structural properties that provides a global type of graph analysis is tied to the metric properties of graphs. Metric graph properties are based on the notion of shortest paths between vertices defined as the path with the smallest cost. For example, in unweighted graphs, a path that connects a pair of vertices with the smallest number of edges is considered a shortest path. Some metric structural properties of graphs include the small-world property, the graph diameter (the longest shortest path between any two vertices in a graph), the graph average path length, and the vertex eccentricities. The eccentricity of a vertex \( v \) in a given connected graph \( G = (V, E) \), denoted as \( \text{ecc}(v) \), is defined as the maximum distance between \( v \) and any other vertex \( u \in V \). The maximum eccentricity value is the graph's diameter, and its minimum value belongs to a special vertex (or vertices) that are known as the center of the graph. The center vertex is the one that minimizes the maximum distance to every other vertex in the graph. See Figure 1.1. By definition, the values of the eccentricities represent the distances among vertices (vertices with smaller eccentricities are closer to other vertices than those with larger eccentricities).

Another graph structural property that has been commonly investigated is the \( \delta- \)
hyperbolicity. The $\delta$-hyperbolicity of a graph is a parameter that measures how close the metric structure of a graph is to the metric structure of a tree (a graph with no cycles). The graph $\delta$-hyperbolicity is related to its underlying hyperbolic geometry. Hyperbolic geometry captures the notion of negative curvature which can be generalized as $\delta$-hyperbolicity in more abstract concepts of metric spaces including graphs. A simple connected graph $G = (V, E)$ naturally defines a metric space $(V, d)$ on its vertex set $V$. The distance $d(x, y)$ is defined as the length (number of edges since we deal with unweighted graphs) of a shortest path connecting vertices $x$ and $y$ in $G$. A tree is the main example of a hyperbolic graph. A tree has a $\delta$-hyperbolicity constant of zero ($\delta = 0$), and a graph with 0-hyperbolicity is precisely the tree metric. The importance of $\delta$-hyperbolicity in graphs emerges from the fact that in many applications, many intractable problems become tractable when the graph has a metric structure that is close to the tree metric [58, 136, 74, 66, 23, 79, 136, 155]. More details on this will be provided in Section 3.2.

Hyperbolic spaces play an important role in geometric group theory and in the geometry of negatively curved spaces. Gromov [110] introduced a generalization of hyperbolic (negatively curved) spaces that is applicable to any metric space. Initially, Gromov’s hy-
perbolicity was applied to the study of automatic groups in the science of computation. According to Gromov’s philosophy, many groups can be regarded as geometric objects when viewed from distance. A very important class of such groups is those whose large-scale behavior is similar to that of negatively curved spaces. Generally, the $\delta$-hyperbolicity reflects large-scale characterization of hyperbolic planes. See Figure 1.2. It turns out that Gromov hyperbolicity is also important in divers applied fields including discrete mathematics, biology [13], phylogenetics [87, 88], and several aspects of computer science such as algorithms [103, 64, 58] and networking [124, 132, 155]. Moreover, it was found that the metric spaces of many types of real-world networks have negative curvature in the large scale [177, 153, 155, 70, 6, 5, 13].

In this dissertation, we are concerned with the theoretical and empirical implications of $\delta$-hyperbolic graphs. Specifically we deal with the following two aspects of the $\delta$-hyperbolicity property in graphs:

- The connection between the $\delta$-hyperbolicity of a graph and its core-periphery structure. The core-periphery structure has been found in many real-world networks. In this structure, one part of a graph constitutes a densely connected core and the rest of the graph forms a sparse periphery. This definition highlights the role of vertices during traffic exchange according to their structural position. Vertices that belong to the core play an important role in passing traffic among peripheral vertex pairs. It has been shown that this pattern of traffic exchange is due to the negative curvature or hyperbolicity in those graphs [23, 155]. In this dissertation, we discuss the interplay between the $\delta$-hyperbolicity and the core-periphery structure in graphs. Furthermore, we propose algorithms to identify core vertices in a graph based on its $\delta$-hyperbolicity.

- The connection between the $\delta$-hyperbolicity of a graph and the distances and the
eccentricities of its vertices. The eccentricity of a vertex $u$ in a graph $G$ is the distance between $u$ and a vertex farthest from $u$ in $G$. Vertex eccentricities provide a measure of their degree of centrality. Vertices with smaller eccentricities are more central in a graph compared to vertices with higher eccentricities.

In many applications, it is desirable to map a graph into another structure that is cheapest (in terms of the amount of stored data) and that maintains some of the properties in the original graph. Trees (and spanning trees) are the cheapest structure that a graph can be mapped to and that maintains its connectivity. A distance $k$-approximating tree is a tree in which the difference between the distance of any vertex pair in the tree and in the original graph is always $\leq k$. An eccentricity $k$-approximating tree is a tree in which the difference between the eccentricity of any vertex in the tree and that of the original graph is always $\leq k$.

It is known that any $\delta$-hyperbolic graph can be embedded into a tree with an additive error $O(\delta \log_2 n)$ [110, 58]. In this dissertation, we investigate the eccentricity approximating tree problems for $\delta$-hyperbolic graphs.
1.1 Research contribution

In this dissertation, we study the metric tree-likeness in graphs using Gromov’s notion of $\delta$-hyperbolicity and we propose theoretical and practical applications that exploit this property. Our work can be partitioned into two parts. The first part (Chapters 4 and 5) is oriented towards investigating the relationship between the $\delta$-hyperbolicity and the core-periphery structure in graphs. The second part of this dissertation (Chapters 6, 7, and 8) presents an interesting application of $\delta$-hyperbolicity in graphs. Mainly, it discusses constructing eccentricity $k$-approximating (spanning) trees for two graph classes: the $\delta$-hyperbolic graphs and the $(\alpha_1, \Delta)$-metric graphs.

In chapter 4, we mainly propose two models to partition a graph into a core and periphery parts by exploiting a property that is intrinsic to $\delta$-hyperbolic graphs - the eccentricity-based bending property of shortest paths. We theoretically analyze the essence of this bending in shortest paths by studying its relationship to the distance between vertex pairs. Moreover, we empirically investigate the correlation between a graph’s $\delta$-hyperbolicity and its other global parameters. Specifically, the relationship between the $\delta$-hyperbolicity of a graph and its diameter and size. Based on this analysis we classify graphs with respect to their $\delta$-hyperbolicity into three categories: strongly-hyperbolic graphs, hyperbolic graphs, and non-hyperbolic graphs. Finally, we analyze the $\delta$-hyperbolicity of several types of real-world biological networks, and shed some light on the relationship between the $\delta$-hyperbolicity of a graph and the conciseness of its core.

In Chapter 5, we characterize more closely the $\delta$-hyperbolicity of graphs and empirically identify a graph’s core as the part that is responsible for maximizing its $\delta$-hyperbolicity parameter. Identifying the area in a graph where the $\delta$-hyperbolicity value is maximized is crucial for reducing the cost of its computation. Computing the exact value of the $\delta$-
hyperbolicity is very expensive ($O(n^4)$) where $n$ is the number of vertices - see Section 3.3 for more details). Therefore, in Chapter 5, we first propose a method that can reduce the size of the input graph to only a subset that maximizes its $\delta$-hyperbolicity by analyzing the local dominance relationship between its vertices. Furthermore, we show that the $\delta$-hyperbolicity of a graph can be found in a set of quadruples that are in close proximity. We show that this set concentrates in the core of a graph. We adopt two core definitions each of which represents a different notion of vertex coreness (a transport-based core definition and a density-based core definition). Our observations have crucial implications on computing the $\delta$-hyperbolicity of large graphs. We also apply our ideas to a set of real-world and artificial networks, and we show their suitability to compute the $\delta$-hyperbolicity value with only a fraction of the original calculations.

In Chapter 6, we show that the vertex eccentricities of a $\delta$-hyperbolic graph $G = (V,E)$ can be computed in linear time with an additive one-sided error of at most $c\delta$, i.e., after a linear time preprocessing, for every vertex $v$ of $G$ one can compute in $O(1)$ time an estimate $\hat{e}(v)$ of its eccentricity $ecc_G(v)$ such that $ecc_G(v) \leq \hat{e}(v) \leq ecc_G(v) + c\delta$ for a small constant $c$. We prove that every $\delta$-hyperbolic graph $G$ has a shortest path tree, constructible in linear time, such that for every vertex $v$ of $G$, $ecc_G(v) \leq ecc_T(v) \leq ecc_G(v) + c\delta$.

Recent empirical studies show that many real-world graphs (including Internet application networks, web networks, collaboration networks, social networks, biological networks, and others) have small hyperbolicity. So, we analyze the performance of our algorithms for approximating centrality and distance matrix on a number of real-world networks. Our experimental results show that the obtained estimates are even better than the theoretical bounds.

Chordal graphs are special graph class with no chord-less cycles of size larger than 3. It was shown that chordal graphs has $\delta$-hyperbolicity $\leq 1$. In Chapter 7, using only metric
properties of graphs, we show that every chordal graph has an eccentricity 2-approximating spanning tree. Furthermore, we extend this result to a much larger family of graphs containing among others chordal graphs, the underlying graphs of 7-systolic complexes and plane triangulations with inner vertices of degree at least 7. We also propose two heuristics for constructing eccentricity $k$-approximating trees with small values of $k$ for general unweighted graphs. We validate those heuristics on a set of real-world networks and demonstrate that all those networks have very good eccentricity approximating trees.

In Chapter 8, we compare the eccentricity $k$-approximating tree algorithms proposed in Chapters 6 and 7.

### 1.1.1 Publications

The different chapters of this dissertation are based on the following peer-reviewed publications:


1.2 Structural properties of graphs

The study of structural properties of graphs, which lies in the heart of the analysis of complex networks, deals with the abstract organization of its vertices and edges. The importance of structural properties in graphs is their ability to reveal and describe multiple graph characteristics such as its layout, connectivity, communities, and vertex centrality. It also seeks identifying (or approximating) any geometric fabric that underlies a graph in order to efficiently solve a particular problem or explain a certain phenomenon. Hence, many efficient algorithms and models have been proposed for graph generation and interpretation. From a graph theory point of view, the structural aspects of graphs representing networks are always fundamental despite the type of the network being analyzed.

Graph structural properties range from very basic ones such as the degree distribution of its vertices to very deeper topological properties. Close analysis of general graphs, which used to be thought of as relatively random and unstructured, shows common structural properties associated with them. For example, in most real-world networks, it was observed that the average distance between vertex pairs is very small and the maximum distance
between any pair of vertices in a graph is also small (compared to the graph’s size). This property is known as the small-world property [68, 12]. Also it was observed that most real-world networks are scale-free, i.e., they have a power-law degree distribution (the majority of vertices have small degrees and only very few vertices have higher degree) [3, 131]. A third property that is common to most real-world networks is the existence of a core-periphery structure. A graph with this structure has a densely connected core that realizes most of the shortest paths between peripheral vertices [37, 27, 75]. Other properties include transitivity and the existence of communities [107, 164].

Structural properties in graphs can be local, global, or intermediate scale. In local structural properties, researchers are interested in analyzing the network from a vertex-centric point of view. It takes into account the vertex, its neighboring vertices, and the edges among them. This simple type of analysis can answer questions about behaviors and properties of particular vertices in the graph. For instance, vertex degree and the clustering coefficient of a vertex. On the other hand, global structural properties deal with the graph as a whole to reveal any hidden properties that can be missed when local analyses are employed. For example, the graph’s diameter and the core-periphery structure. The intermediate scale properties, which has been recently introduced [6], can be considered as a coupling between the local and the global analyses. It deals with the properties that emerge from analyzing some part (a subgraph for example) of the network. Accordingly, any graph global property can be turned into an intermediate scale property.

A general property of many real-world networks is the existence of some structure that is meaningfully geometric. I.e., they possess very similar properties as some other metric space with well-known and algorithmically nice properties. Or they can be embedded (mapped) into another (well-known) metric space with a very low distortion or error (see Section 2.3). Examples of well-known or nice metric spaces include low-dimensional spaces
such as the Euclidean $\ell_2$ space, lines, and trees (graphs with no cycles).

Tree-likeness has recently attracted a considerable attention. Tree-likeness in graphs measures how close the structure of a given graph is to the structure of a tree based on some criteria such as its metric (shortest path distances) or its cut (number of vertices that needed to be removed to significantly reduce the connectivity of a graph). The metric tree-likeness of a graph can be measured using Gromov’s $\delta$-hyperbolicity [110] and the cut or partitioning tree-likeness can be measured by the tree decomposition algorithms [169].

The interest of measuring the closeness of a graph to a tree is due to the fact that trees are a very natural class of simple graph metrics for which many algorithmic problems become tractable. Several algorithms that are considered NP-hard for general graphs have polynomial time solutions in the tree structure [58, 66, 79, 136]. Generally, the closer the structure of a graph to a tree, the easier it is to find efficient algorithms to solve a wide range of problems related to it. Moreover, the tree-likeness property has been shown to be crucial in the understanding of some aspects of network traffic such as traffic flow and network congestion. More details will provided on this in Section 3.2.

It was found that many real-world networks are tree-like when analyzed in the global scale [177, 153, 155, 13, 5, 65] and the intermediate scale [6, 34].

### 1.3 Gromov $\delta$-hyperbolic spaces

A metric space $(X, d)$ is a set of points $X$ for which the distance function $d$ between each pair of points $x, y \in X$ is defined and satisfy the following properties:

- $d(x, y) \geq 0$.
- $d(x, y) = 0 \iff x = y$.
- $d(x, y) = d(y, x)$.  

11
• $d(x, y) \leq d(x, z) + d(z, y)$, where $z$ is any point in $X$.

A metric space is a geodesic metric space if every pair of points is connected by a geodesic (a shortest path), i.e., if for every pair of points $x, y \in X$, all points on the shortest path from $x$ to $y$ are in $X$, where the shortest path can be seen as a continuous curve. The Euclidean space, the hyperbolic space, the tree metric, and some graphs are all examples of geodesic metric spaces. A simple connected unweighted graph $G = (V, E)$ naturally defines a metric space $(V, d)$ on its vertex set $V$. The distance $d(x, y)$ is defined as the length (number of edges) of a shortest path connecting vertices $x$ and $y$ in $G$.

The hyperbolic space is a type of non-Euclidean space with negative curvature. The main difference between the Euclidean and the hyperbolic metric spaces is that in the latter there may exist a line and a point not on the line with at least two parallels to the given line passing through the given point (Figure 1.3), which negates the parallel postulate in the Euclidean geometry. The line segments in a hyperbolic space are usually curved; as a result, the angles of a hyperbolic triangle add up to strictly less than $180^\circ$ (Figure 1.4). This contributes to the reason why the hyperbolic plane is considered wider than the Euclidean plane although the triangles in the Euclidean plane are wider than in the hyperbolic one. A common example of a hyperbolic space is a horse saddle (see Figure 1.5). We refer the interested readers to the following books about hyperbolic geometry [134, 20, 167, 111, 28].

Because of the fundamental differences between the hyperbolic and the Euclidean geometries, representing (or visualizing) hyperbolic spaces in Euclidean spaces has some difficulties. There are several equivalent models for representing hyperbolic spaces each of which emphasizes different specific aspects of hyperbolic geometry. For example, the Poincaré unit disc model (see Figure 1.2), the Klein unit disc model, and the hyperboloid model.
Mikhail Gromov (1987) introduced a notion of abstract hyperbolic spaces that was known as Gromov’s $\delta$-hyperbolicity [110]. Initially, the research was developed in the setting of geometric group theory and was applied mainly to the study of automatic groups [163] in which it played an important role in sciences of the computation. In recent years, Gromov $\delta$-hyperbolicity has been employed in the analysis of general metric spaces [33, 21]. One of the first natural questions to ask is if a given metric space $\delta$-hyperbolic in the sense of Gromov. The notion of $\delta$-hyperbolicity have been studied and developed by many authors. Some important applications of Gromov’s hyperbolicity include secure transmission of information [123, 124], the spread of viruses through a network [124], and the study of DNA data [44].

The concept of Gromov’s $\delta$-hyperbolicity captures the essence of negatively curved spaces including the classic hyperbolic space and of discrete spaces such as trees [110]. A geodesic metric space is a $\delta$-hyperbolic space if its points satisfy certain metric relations (depending on the value of $\delta$). The parameter $\delta$ in Gromov’s $\delta$-hyperbolicity measures the extent to which a geodesic metric space embeds into a tree metric. A geodesic metric space
is exactly the tree metric if and only if $\delta = 0$. Trees allow nearly isometric embeddings into hyperbolic spaces. For comparison, trees do not generally embed into Euclidean spaces because trees need an exponential amount of space to branch out. This is provided in the hyperbolic geometry [137]. In recent years several research efforts have been dedicated towards showing that metrics used in geometric function theory are Gromov hyperbolic. In particular, many negatively curved surfaces are quasi-isometric to a simple graph [173] which makes exploring the hyperbolicity criteria for graphs even more interesting. A connected graph $G = (V, E)$ is $\delta$-hyperbolic if the metric space $(V, d)$ is $\delta$-hyperbolic. The $\delta$-hyperbolicity in graphs is discussed in more details in Chapter 3.

The mathematical properties of Gromov hyperbolic spaces and its applications have been of great interest in graph theory [123, 124, 58, 155, 31, 25]. The concept of hyperbolicity appears also in discrete mathematics and in multiple domains in computer science. For example, diameter and distance estimation [58], distance and routing labeling [66], and traffic flow and congestion minimization [155, 25, 63]. Furthermore, the idea of hyperbolicity has been applied in several routing problems [79, 136]. Also the hyperbolicity constant of geodesic metric spaces can be viewed as a measure of how tree-like the spaces are.
Chapter 2

Preliminaries and Notations

In this chapter, we first review some basic graph definitions and concepts that we will be using throughout the dissertation. The definitions are also summarized in Table 2.1. Then we visit the concepts of vertex eccentricity and centrality in graphs. We also discuss the concept of graph embedding.

2.1 Basic definitions

A graph $G = (V, E)$ is an ordered pair $(V, E)$, where $V$ is the set of vertices and $E \subseteq V^2$ is the set of edges. All graphs occurring in this work are (unless explicitly indicated) simple, connected, finite, and with no self-loops. For a graph $G = (V, E)$, we use $|V| = n$ to denote the number of vertices in $G$ and $|E| = m$ to denote the number of edges in $G$. We define the size of the graph denoted as $\text{size}(G)$ as the sum of the number of vertices and the number of edges in $G$, i.e.,

$$\text{size}(G) = |V| + |E|$$
Table 2.1: Summary of notations used frequently throughout this work.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G = (V, E)$</td>
<td>Graph $G$ with vertex set $V$ and edge set $E$.</td>
</tr>
<tr>
<td>$size(G)$</td>
<td>Graph size; $size(G) =</td>
</tr>
<tr>
<td>$\rho(u, v)$</td>
<td>A shortest path connecting two vertices $u$ and $v$.</td>
</tr>
<tr>
<td>$d(u, v)$</td>
<td>Distance between two vertices $u$ and $v$ defined as the length of a shortest path that connects $u$ and $v$.</td>
</tr>
<tr>
<td>$I(u, v)$</td>
<td>Interval between two vertices $u$ and $v$; $I(u, v) = {w \in V : d(u, w) + d(w, v) = d(u, v)}$</td>
</tr>
<tr>
<td>$ecc(u)$</td>
<td>Eccentricity of vertex $u$; $ecc(u) = \max_{v \in V} {d(u, v)}$</td>
</tr>
<tr>
<td>$diam(G)$</td>
<td>Graph diameter; $diam(G) = \max_{u, v \in V} {d(u, v)}$.</td>
</tr>
<tr>
<td>$N(u)$</td>
<td>Neighborhood of vertex $u$; $N(u) = {v \in V : uv \in E}$.</td>
</tr>
<tr>
<td>$\deg_{out}(u)$</td>
<td>Number of directed edges leaving vertex $u$.</td>
</tr>
<tr>
<td>$\deg_{in}(u)$</td>
<td>Number of directed edges pointing at vertex $u$.</td>
</tr>
<tr>
<td>$\deg(u)$</td>
<td>Number of vertices in vertex $u$’s neighborhood (undirected graph); $\deg(u) =</td>
</tr>
<tr>
<td>$N_r(u)$</td>
<td>Neighborhood of distance at most $r$ from vertex $u$; $N_r(u) = {v \in V : d(u, v) \leq r}$.</td>
</tr>
<tr>
<td>$F(u)$</td>
<td>Set of vertices located at maximum distance from vertex $u$; $F(u) = {v \in V : d(u, v) = ecc(u)}$.</td>
</tr>
<tr>
<td>$rad(G)$</td>
<td>Graph radius; $rad(G) = \min_{u \in V} {ecc(u)}$.</td>
</tr>
<tr>
<td>$C(G)$</td>
<td>Graph center; $C(G) = {u \in V : ecc(u) = rad(G)}$.</td>
</tr>
<tr>
<td>$G_W = (W, E_W)$</td>
<td>Subgraph of $G$ induced by vertex set $W$. $W \subseteq V$ and $E_W = {uv \in E : u, v \in W}$</td>
</tr>
</tbody>
</table>
Figure 2.1: A graph $G = (V, E)$. The number next to each vertex is its eccentricity. $G$ has diameter 5, radius 3, and its center is vertex $c$.

A path $P$ of length $k$ from vertex $u$ to vertex $v$ is a sequence of adjacent vertices $u_0, u_1, ..., u_k$, where $u_0 = u$ and $u_k = v$, and a shortest path (also called a geodesic) from $u$ to $v$, denoted by $\rho_G(u,v)$ is a path that minimizes the length of this path (more than one shortest path may exist between any pair of vertices).

We use $d_G(u,v)$ to denote the distance between a pair of vertices $u$ and $v$ in $G$. When $G$ is an unweighted graph, the distance is defined as the number of edges in a shortest path $\rho_G(u,v)$ that connects the two vertices $u$ and $v$. When $G$ is weighted, the distance equals the sum of the weights of all the edges of a shortest path between $u$ and $v$. $d_G(u,v) = \infty$ if there is no path that connects $u$ and $v$. All graphs we deal with in this dissertation are unweighted and we assume all traffic follows shortest paths.

We define the interval $I_G(u,v)$ as the set of vertices on all shortest paths between $u$ and $v$, i.e.,

$$I_G(u,v) = \{ w \in V : d(u,w) + d(w,v) = d(u,v) \}$$

The eccentricity of a vertex $u \in V$ $\text{ecc}_G(u)$ is the distance between $u$ and a vertex farthest from $u$, i.e.,
$e cc_G(u) = \max_{v \in V} \{d(u, v)\}$

We omit the subscript $G$ from $d_G(u, v)$, $\rho_G(u, v)$, $I_G(u, v)$, and $e cc_G(u)$ when the context is only about the graph $G$.

The *diameter* of a graph $diam(G)$ is the length of a longest shortest path between any two vertices $u$ and $v$ in $G$, i.e.,

$$diam(G) = \max_{u, v \in V} \{d(u, v)\}$$

When $d(u, v) = diam(G)$, then we call the vertex pair $(u, v)$ a *diametral pair*. In some graphs, the diameter is much higher than the average path length, which indicates the impact of a few outlier vertex pairs on its value. Therefore, the parameter *effective diameter* was proposed. The effective diameter of a graph represents the maximum distance between a fraction of vertex pairs (we use 90% in this work) of a graph.

The *center* of a graph $C(G)$ constitutes the vertex (or vertices) with the minimum maximum distance to every other vertex in the graph. In other words, the graph’s center constitutes all vertices with minimum eccentricity.

$$C(G) = \{u \in V : ecc(u) = rad(G)\}$$

This minimum maximum distance is known as the *radius* $rad(G)$ of the graph.

$$rad(G) = \min_{u \in V} \{ecc(u)\}$$

Every undirected connected graph has a non-empty center. See Figure 2.1 for an example. In general graphs, when $|C(G)| > 1$, it may or may not be connected. Obviously,
when a graph is disconnected, the values of the radius \( \text{rad}(G) \) and diameter \( \text{diam}(G) \) are undefined.

We use \( N(u) \) to denote the neighborhood of vertex \( u \)

\[
N(u) = \{ v \in V : uv \in E \}
\]

Each vertex \( u \in V \) in an undirected graph \( G = (V, E) \) has a degree that denotes the number of incident vertices \( (\text{degree}(u) = |N(u)|) \). When \( G \) is directed, each vertex \( u \) has an out-degree which represents the number of edges pointing from \( u \) to other vertices and an in-degree which represents the number of edges pointing towards \( u \). We denote the out-degree and the in-degree as \( \text{degree}_\text{out}(u) \) and \( \text{degree}_\text{in}(u) \) respectively.

Generally, for each integer \( r \geq 0 \), let \( N_r(u) \) denotes the neighborhood of distance at most \( r \) centered at \( u \) in an undirected graph \( G \). I.e.,

\[
N_r(u) = \{ v \in V : d(u, v) \leq r \}
\]

Accordingly, \( N_1(u) \) has all vertices that are adjacent to \( u \) \( (N_1(u) = N(u)) \), \( N_2(u) \) includes the set of vertices that are at most two-hops away from \( u \) and so on.

A subgraph \( G_W = (W, E_W) \), where \( W \subseteq V \) and \( E_W = \{ uv \in E : u, v \in W \} \), is called the subgraph of \( G \) induced by \( W \). An induced subgraph \( G_W \) of a graph \( G \) is isometric if the distance between any pair of vertices in \( G_W \) is the same as that in \( G \). The subgraph \( G_W \) is a clique of \( G \) if every pair of its vertices are pairwise adjacent.

Next we present short definitions of some graph classes that frequently appear in this dissertation. A tree is an acyclic graph. The center of a tree consists of at most two adjacent vertices [39]. A spanning tree \( T = (V, E') \) of \( G \) is a tree that covers all vertices in \( G \) and in which \( E' \subseteq E \). Every connected undirected graph has a spanning tree. When
Figure 2.2: Some common graph types.

\[ E' \not\subset E, \text{ then } T \text{ is just a tree of } G. \]

A complete graph is a graph \( G = (V, E) \) in which every pair of vertices is connected by an edge, i.e., if \( G \) is a clique. In a complete graph \( G \), \( diam(G) = rad(G) = 1 \) and \( C(G) = V \). A block graph is a graph in which every biconnected component is a clique. A graph \( G \) is chordal if its largest induced chordless cycle is of length three. A cycle in a graph is a closed chain of vertices, and a cycle graph is a graph that consists of a single cycle. Finally, a two dimensional grid graph is an \( m \times n \) lattice graph. See Figure 2.2
2.2 Eccentricity

For a vertex $u$, we define the set $F(u)$ of vertices at most distance from $u$. The eccentricity $ecc(u)$ is the distance from $u$ to a vertex $v \in F(u)$, i.e., the distance between $u$ and any of its farthest neighbors $v$. If the graph $G$ is disconnected, the eccentricity values of all vertices will be infinite. The minimum and maximum eccentricity values represent the graph’s radius $rad(G)$ and diameter $diam(G)$ respectively. The vertices with minimum eccentricity constitutes the center of the graph $C(G)$. It is a well-known fact that the diameter and the radius of a graph are related through the following inequality

$$rad(G) \leq diam(G) \leq 2rad(G)$$

The importance of vertex eccentricities and graph center emerges in multiple applications. An example is the facility location problem in which the goal is to decide the optimal location for a new facility (a hospital, a school, or a station) according to some defined criteria. When the criteria is to find a location such that the maximum distances traveled from every other vertex to the facility are minimized, the problem can be solved by finding the center of the graph.

The following are two well-known properties of the eccentricity function in graphs.

**Remark 1.** If $u$ and $v$ are two adjacent vertices, then $|ecc(u) - ecc(v)| \leq 1$.

For example, in Figure 2.1, $ecc(v) - ecc(w) = 0$ and $ecc(d) - ecc(u) = 1$. This is different from trees in which the eccentricities of any pair of adjacent non-central vertices is always equal to one.

The following remark can be concluded from Remark 1.

**Remark 2.** Let $w_0, w_1, ..., w_i$ be a path in $G$ and $ecc(w_0) > ecc(w_i)$, and $j$ is an integer
such that $\text{ecc}(w_0) > j > \text{ecc}(w_i)$, then there exists an integer $z$, $0 \leq z \leq i$, such that $\text{ecc}(u_z) = j$.

### 2.2.1 Eccentricity layering of a graph

The eccentricity layering of a graph $G = (V, E)$ denoted as $\mathcal{EL}(G)$ partitions its vertices into concentric circles or layers $\ell_r(G)$, $r = 0, 1, \ldots$. Each layer $r$ is defined as $\ell_r(G) = \{u \in V : \text{ecc}(u) - \text{rad}(G) = r\}$. Here $r$ represents the index of the layer. The inner-most layer (layer 0) encloses the graph’s center $C(G)$, i.e., all vertices with the minimum eccentricity or of eccentricity equal to $\text{rad}(G)$; this layer has index $r = 0$. Then the first layer includes all vertices who have their eccentricities equal to $\text{rad}(G) + 1$, and so on. The vertices in the last layer (outer-most layer) will have eccentricities equal to the diameter of the graph. Figure 2.3 demonstrates an illustration for the layers and Figure 2.4 shows the eccentricity layering of the graph in Figure 2.1.

Any vertex $v \in \ell_r(G)$ has level (or layer) $\text{level}(v) = r$. Throughout this work, we use layer 0 to indicate the central layer, layer 1 to indicate the layer that directly succeeds the central layer, and etc. Therefore, when we say lower layers, we mean layers towards the central layer, and the opposite for higher layers. The vertices in Figure 2.1 can be
partitioned into three layers according to the eccentricity layering as follows. \( \ell_0(G) = \{c\} \), 
\( \ell_1(G) = \{a, b, u, y, x, v, z, w\} \), and \( \ell_2(G) = \{d, e, f, h, i, j\} \).

### 2.3 Graph embedding

Informally, graph embedding involves mapping a graph metric into another host metric space that has “simpler” structural properties with as small error as possible. In other words, it involves mapping a high-dimensional metric space into a smaller-dimensional space. Graph embedding has several practical applications such as high dimensional data analysis, distance estimation, topology construction, and visualization. It has also attracted much attention for its theoretical and mathematical content.

**Definition 1** (Space embedding). A metric space embedding (or simply an embedding) of a metric space \((X, d)\), where \(X\) is the underlying metric space and \(d : X \times X \to \mathbb{R}\) is the metric distance function, into another metric space \((X', d')\) is a map \(f : X \to X'\).

An embedding \(f\) of a metric space \((X, d)\) into a metric space \((X', d')\) has an additive distortion \(k \geq 0\) when the following is true for every vertex pair \(u\) and \(v \in X\).
\[ |d_X(u,v) - d'_{X'}(f(u), f(v))| \leq k \]

Embedding \( f \) has a multiplicative distortion \( k \geq 0 \) when the following is true for every vertex pair \( u \) and \( v \in X \).

\[
\frac{1}{k} d'_{X'}(f(u), f(v)) \leq d_X(u,v) \leq k d'_{X'}(f(u), f(v))
\]

Here we are mainly concerned with embedding the graph metric \((V,d)\) of a graph \( G = (V,E) \) into another metric space \((V',d')\), where \( d \) measures the length of a shortest path between two vertices.

The quality of an embedding is measured by its worst case distortion. An embedding is called distance-preserving or isometric if for all \( u,v \in X \), \( d(u,v) = d'(f(u), f(v)) \), i.e., when the value of the additive distortion \( k = 0 \). In this case, running an algorithm on the host metric space provides an exact solution to the problem. Otherwise, this technique results in an approximation algorithm for the given problem. In general, embedding a graph into another metric space with minimum distortion is NP-hard [8].

A special class of host metric spaces is the tree metric due to its simplicity. Many algorithmic problems become tractable on trees. A metric space \((X,d)\) is a tree metric space if it can be embedded into a weighted tree \( T = (X \cup Y, F) \) that satisfies \( \forall x, y \in X, d_X(x,y) = d_T(x,y) \). Note that \( T \) may have more points that do not exist in \( X \).

Another way for checking the embeddability of a metric space into the tree metric (a hyperbolic space) is by Gromov’s \( \delta \)-hyperbolicity. A metric space (and accordingly a graph) is exactly the tree metric when the value \( \delta = 0 \) according to Gromov’s \( \delta \)-hyperbolicity (in Section 3.1, we discuss the various definitions of \( \delta \)-hyperbolicity). When \( \delta > 0 \), the metric space is a hyperbolic space with distortion \( \delta \) (or it approximates a hyperbolic space). More
details on the importance of embedding metric spaces (graphs) into hyperbolic spaces are provided in Section 3.2.

Some graph types are known to have 0-hyperbolicity ($\delta = 0$) according to Gromov. Those graphs can be embedded into the tree metric with no distortion. Examples include tree graphs, complete graphs and block graphs. Chordal graphs are known to have $\delta \leq 1$ [44]. A feature that those graph types have in common is that they either are cycle-free or have induced cycles of small lengths ($\leq 3$).

On the other hand, an embedding of an $n$-cycle graph or an $n \times n$-grid graph (both have high $\delta$-hyperbolicity) into a hyperbolic-metric space results into a multiplicative distortion of at least $\Omega(n/\log n)$. However, both graph types can be embedded into the Euclidean space with only constant multiplicative distortions [189].

2.4 Graph centrality measures

The main purpose behind the notion of centrality and centrality measures is to rank vertices in a given graph based on their importance. We discuss centrality measures that assign a numerical value to each vertex based on the structure of the graph. A centrality measure $\zeta$ returns the centrality $c_{\zeta}(u)$ of every vertex $u \in V$. We briefly describe the following four classic centrality measures.

Degree centrality. The degree centrality, which is the simplest centrality measure, considers the central vertices of the network as the set of vertices that have the highest number of connections. I.e., the degree centrality ranks vertices according to their degrees.

$$c_D(u) = \text{degree}(u)$$
The degree centrality can be considered as a local measure since it only relies on the number of neighbors of a vertex to determine its centrality [39]. In Figure 2.1, vertex $u$ has the highest degree centrality.

**Shortest-path betweenness centrality.** The shortest-path betweenness centrality (or the betweenness centrality for short) is a widely used concept in social networks. It expresses how much effect each vertex (or actor) has in the communication of the network assuming that all traffic follows shortest paths between every two vertices (taking into account that one or more shortest paths may exist between any given pair). Given a connected finite graph $G = (V, E)$, the shortest-path betweenness of a vertex $u \in V$ measures the total number of shortest paths between every pair of vertices $x$ and $y$ that pass through $u$ such that $u \neq x$ and $u \neq y$ to exclude the cases when $u$ is the source or the destination vertex. Let $\alpha_{xy}(u)$ be the fraction of shortest paths between $x$ and $y$ that pass through $u$, i.e.,

$$\alpha_{xy}(u) = \frac{\sigma_{xy}(u)}{\sigma_{xy}}$$

where $\sigma_{xy}(u)$ is the number of all shortest paths between $x$ and $y$ that pass through $u$ and $\sigma_{xy}$ is the number of all shortest paths between $x$ and $y$. The shortest-path betweenness centrality $c_B(u)$ of a vertex $u$ can be calculated as [39]:

$$c_B(u) = \sum_{x \in V} \sum_{y \in V} \alpha_{xy}(u)$$

A higher value of this index indicates the higher importance of the role that the vertex plays in the data exchange process among distant vertices. In Figure 2.1, the vertex with the highest betweenness centrality is vertex $u$. Vertices $d, e, h, i$, and $j$ are ranked last ac-
cording to betweenness centrality.

**Eccentricity centrality.** The eccentricity centrality suggests that the center of a graph includes the vertex (or vertices) that minimizes the maximum distance to all other vertices (minimum eccentricity). For a given vertex \( u \), the eccentricity centrality of \( u \) is defined as [115]

\[
c_{E}(u) = \frac{1}{\max_{v \in V} \{d(u, v)\}}
\]

A vertex \( u \) is a central vertex when \( ecc(u) = rad(G) \). For the example in Figure 2.1, vertex \( c \) has the highest eccentricity centrality.

**Closeness centrality.** The closeness centrality considers the center as the subset of vertices with the minimum total distance to all other vertices. In other words, the closeness centrality \( c_{C}(u) \) of a vertex \( u \) is [39]

\[
c_{C}(u) = \frac{1}{\sum_{v \in V} d(u, v)}
\]

The vertex with highest closeness centrality is known as the median of the graph. In Figure 2.1, vertex \( a \) is the node with the highest closeness centrality.

Note that centrality measures may or may not agree on ranking vertices in a given graph. In Figure 2.1 for example, the degree centrality and the betweenness centrality measures both rank vertex \( u \) as most central. On the other hand, the eccentricity centrality rank vertex \( c \) as the most central vertex and the closeness centrality rank vertex \( a \) as the most central vertex.
Chapter 3

\(\delta\)-Hyperbolicity in Graphs

On continuous metric spaces, \(\delta\)-hyperbolicity is a measure of negative curvature. The Euclidean plane (with zero curvature) is not hyperbolic. On discrete metric spaces, \(\delta\)-hyperbolicity measures how far is the (shortest path) metric space from a tree metric. A simple undirected graph \(G = (V, E)\) is naturally a discrete metric space \((V, d)\) with distance \(d\) defined as the number of edges in a shortest path between every vertex pair.

In what follows, we focus our discussions on hyperbolic graphs although all the definitions, results, and analyses can be similarly stated for any discrete hyperbolic space. First, we present the several common definitions of \(\delta\)-hyperbolicity and we discuss their basic properties. Second, we discuss the importance of studying and investigating the \(\delta\)-hyperbolicity in graphs. Finally, we briefly present the computational challenges of computing the \(\delta\)-hyperbolicity value.

3.1 Definitions and basic properties

The hyperbolicity in graphs is typically measured by Gromov’s hyperbolicity parameter \(\delta\) which essentially measures the deviation of a metric space from the tree metric [110].
Figure 3.1: The metric of a complete graph (a) is a tree metric. (b) The embedding of $G$ into a weighted tree $T$ in which each edge has a weight of $\frac{1}{2}$.

**Definition 2** (Tree metric). A metric space $(X, d)$ is called a tree metric space if it can be embedded into a weighted tree $T = (X \cup Y, F)$ that satisfies $\forall x, y \in X, \ d(x, y) = d_T(x, y)$.

The value of $\delta$ is bounded by the diameter of the graph [95, 172]. Trees have $\delta = 0$ (also called 0-hyperbolic). Trees, complete graphs, and block graphs are all 0-hyperbolic and accordingly can be (isometrically) embedded into a tree metric and are considered of strong hyperbolicity [195] (See Figure 3.1). On the other hand, a $n$-cycle graph, where $n$ is the number of vertices, is $\frac{n}{4}$-hyperbolic and an $n \times n$-grid graph is $n - 1$-hyperbolic. In this sense, cycles and grid graphs are classic examples of non-hyperbolic graphs. Other graphs with hyperbolicity values slightly higher than zero include chordal graphs with $\delta \leq 1$ [44] and $k$-chordal graphs with $\delta \leq \lfloor \frac{k}{4} \rfloor$, where $k \geq 4$ [195]. See Figure 3.2.

There are several equivalent definitions (up to a multiplicative factor) of Gromov’s hyperbolicity. In this section, we present several definitions of $\delta$-hyperbolicity followed by the asymptotic relationships between them.

The original definition that Gromov introduced for $\delta$-hyperbolicity is based on the notion of geodesic triangles and the Gromov product [110]. Given a graph $G = (V, E)$, a geodesic triangle $\Delta(x, y, z)$ for three arbitrary vertices $x, y, z \in V$ is the union of three
Figure 3.2: The $\delta$-hyperbolicity values of some common graph types.

shortest paths (geodesic segments) connecting $x$, $y$, and $z$. I.e., $\Delta(x, y, z) = \rho(x, y) \cup \rho(y, z) \cup \rho(x, z)$ (Figure 3.3 (a)).

**Definition 3** ($\delta$-Hyperbolicity using Gromov product). A graph $G = (V, E)$ is $\delta$-hyperbolic for some $\delta \geq 0$ if for all $x, y, z, w \in X$ the following holds:

$$
(x\mid z)_w \geq \min((x\mid y)_w, (y\mid z)_w) - \delta
$$

(3.1)

where $(x\mid y)_z$ is the Gromov product of vertices $x, y \in V$ with respect to $z$ and is defined as:

$$
(x\mid y)_z = \frac{1}{2}(d(x, z) + d(z, y) - d(x, y))
$$

The Gromov product measures the failure of the triangle inequality to be an equality.
Geometrically, the value of Gromov product can be interpreted as follows. In a geodesic triangle $\Delta(x, y, z)$, there are three vertices $u \in \rho(x, y)$, $w \in \rho(y, z)$, $v \in \rho(x, z)$ such that $d(x, u) = d(x, v)$, $d(y, u) = d(y, w)$, and $d(z, v) = d(z, w)$ [47]. Let $a_x = d(x, u) = d(x, v)$, $a_y = d(y, u) = d(y, w)$, and $a_z = d(z, v) = d(z, w)$, then

$$a_x = \frac{1}{2}(d(x, y) + d(x, z) - d(y, z)) = (y|z)_x$$

$$a_y = \frac{1}{2}(d(x, y) + d(y, z) - d(x, z)) = (x|z)_y$$

$$a_z = \frac{1}{2}(d(x, z) + d(y, z) - d(x, y)) = (x|y)_z$$

An illustration is provided in Figure 3.3 (b). If the geodesic triangle $\Delta(x, y, z)$ is a tripod, (see Figure 3.3(c)), then $(x|y)_z = (z, w)$. I.e., $(x|y)_z$ is the distance along $\rho(x, z)$ (or $\rho(y, z)$) before the two geodesics separate.

In graphs, $(x|y)_z$ resembles the distance from $z$ to a shortest path connecting the pair $(x, y)$. $(x|y)_z = 0$ only if $z \in \rho(x, y)$ and the values increase as the distance from $z$ to $\rho(x, y)$ increases. If the given graph is a tree, the Gromov product is minus the length of the shared subpath between the shortest paths connecting $z$ and $x$ and $z$ and $y$ respectively.

Generally, in a $\delta$-hyperbolic graph $G$, two geodesics $\rho(x, y)$ and $\rho(x, z)$ of any geodesic triangle $\Delta(x, y, z)$ coming out of the common vertex $x$ run together within distance $\delta$ as long as the distance between them is no more than $(y|z)_x$ and after that they start to diverge. This point of view becomes effective at distances large compared to $\delta$ [47].

The definition of hyperbolicity using Gromov product can be rewritten as the four-point condition definition. In the four-point condition definition, the inequality is written directly in terms of the distances among the vertices instead of their Gromov product.

**Definition 4** ($\delta$-Hyperbolicity using the four-point condition). In a graph $G = (V, E)$,
Given four vertices $x, y, u$, and $v \in V$ with $d(x, y) + d(u, v) \geq d(x, u) + d(y, v) \geq d(x, v) + d(y, u)$, the hyperbolicity of the quadruple $x, y, u, v$ denoted as $hb(x, y, u, v)$ is defined as:

$$hb(x, y, u, v) = \frac{d(x, y) + d(u, v) - (d(x, u) + d(y, v))}{2}.$$  

(3.2)

And the $\delta$-hyperbolicity of the graph is

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

(3.3)

In other words, the hyperbolicity of a graph $G$ is the minimum value of $\delta$ for which $G$ is $\delta$-hyperbolic. Here $\delta = hb(G)$. In trees, the largest two distance sums for any quadruple are always equal. This may not be the case for some quadruples in graphs. See Figure 3.4. In unweighted graphs, the hyperbolicity is valued in multiples of $\frac{1}{2}$ for the four-point condition definition. A graph is exactly the tree metric if the largest two distance sums are equal. This definition of $\delta$-hyperbolicity is mathematically straightforward since it does not require the knowledge of the exact shortest paths. However, it has the disadvantage of being geometrically unclear compared to the slim triangles definition (presented below).
Figure 3.4: Four-point-condition definition for $\delta$-hyperbolicity. Three distance sums defined over four vertices $x, y, u, v$ in (a) Tree metric and (b) Graph metric. In the tree metric, the three distance sums are $d(x, y) + d(u, v) = \mu_x + \mu_y + \mu_u + \mu_v + 2\eta$, $d(x, v) + d(u, y) = \mu_x + \mu_v + \mu_u + \mu_y + 2\eta$, and $d(x, u) + d(y, v) = \mu_x + \mu_u + \mu_y + \mu_v$. In a graph metric, the three distance sums are (in this case) $d(x, y) + d(u, v) = \mu_x + \mu_y + \mu_u + \mu_v + 2\eta$, $d(x, v) + d(u, y) = \mu_x + \mu_v + \mu_u + \mu_y + 2\eta + 2\xi$, and $d(x, u) + d(y, v) = \mu_x + \mu_u + \mu_y + \mu_v + 2\xi$.

**Definition 5** ($\delta$-Hyperbolicity using slim triangles). Given a graph $G = (V, E)$ and three vertices $x, y, z \in V$, a geodesic triangle $\Delta(x, y, z)$ is $\delta$-slim if each of its sides is contained in the $\delta$-neighbourhood of the union of the other two sides. $G$ is said to be $\delta$-hyperbolic if every triangle in $G$ is $\delta$-slim.

For example, the geodesic triangle $\Delta(x, y, z)$ in Figure 3.5 is $\delta$-slim if for any vertex $v \in \rho(x, y)$, the distance from $v$ to $\rho(x, z) \cup \rho(z, y)$ is $\leq \delta$, and similarly for vertices on the other two segments. This definition is credited to Rips.

In trees, each side of a triangle is completely contained in the union of the other two. Also, two geodesics from a common basepoint agree up to a certain point, then diverge completely. The most general triangle in a tree is a tripod, so there is always (at least) one vertex common to all three sides of a triangle. Graphs with 0-slim triangles are exactly the graphs with tree metric. A geodesic triangle is 0-slim if every vertex of it lies on two different shortest-paths.

**Definition 6** ($\delta$-Hyperbolicity using thin triangles). Let $G = (V, E)$ be a graph and
Figure 3.5: Slim triangles definition for \( \delta \)-hyperbolicity.

\( \Delta(x, y, z) \) be a geodesic triangle. Define three points \( m_x, m_y, \) and \( m_z \) as:

- \( m_x \in \rho(y, z) \) such that \( d(y, m_x) = \alpha_y = (x|z)_y \) and \( d(z, m_x) = \alpha_z = (y|x)_z \).
- \( m_y \in \rho(x, z) \) such that \( d(x, m_y) = \alpha_x = (y|z)_x \) and \( d(z, m_y) = \alpha_z = (x|y)_z \).
- \( m_z \in \rho(x, y) \) such that \( d(x, m_z) = \alpha_x = (y|z)_x \) and \( d(y, m_z) = \alpha_y = (x|z)_y \).

There exists a unique isometry \( \varphi \) which maps \( \Delta(x, y, z) \) to a tripod \( T(x, y, z) \) consisting of three solid segments \([x, m_x]\), \([y, m_y]\), and \([z, m_z]\) of lengths \( \alpha_x, \alpha_y, \) and \( \alpha_z \), respectively. This isometry maps the vertices \( x, y, z \) of \( \Delta(x, y, z) \) to the respective leaves of \( T(x, y, z) \) and the points \( m_x, m_y, \) and \( m_z \) to the center \( m \) of this tripod. Any other point of \( T(x, y, z) \) is the image of exactly two points of \( \Delta(x, y, z) \). A geodesic triangle \( \Delta(x, y, z) \) is called \( \delta \)-thin if for all points \( u, v \in \Delta(x, y, z) \), \( \varphi(u) = \varphi(v) \) implies \( d(u, v) \leq \delta \). A graph \( G = (V, E) \) whose all geodesic triangles \( \Delta(u, v, w) \), \( u, v, w \in V \), are \( \delta \)-thin is called a graph with \( \delta \)-thin triangles.

In other words, a triangle is \( \delta \)-thin if the pre-image of every non-leaf tripod point has vertices that are at distance at most \( \delta \) from one another. Note that in the above definition, \( \alpha_y + \alpha_z = d(y, z) \), \( \alpha_x + \alpha_z = d(x, z) \), and \( \alpha_x + \alpha_y = d(x, y) \). See Figure 3.6 for an illustration.
Figure 3.6: Thin triangles definition for $\delta$-hyperbolicity. A geodesic triangle $\Delta(x, y, z)$, the points $m_x, m_y, m_z$, and the tripod $\Upsilon(x, y, z)$

We call a graph $\delta$-hyperbolic (or simply hyperbolic) if it is hyperbolic in any of the above definitions for any $\delta \geq 0$ and we denote the hyperbolicity of a graph $G$ by $hb(G)$. Note that because we deal with finite graphs, the value of the $\delta$-hyperbolicity parameter is always finite. This does not suggest that all finite graphs are tree-like. In fact, the value of $\delta$ (and its relationship to other parameters in a graph) plays an important role in deciding how hyperbolic (or tree-like) a graph is. Trees have $\delta = 0$ (0-hyperbolic) and are considered of strong hyperbolicity. On the other hand, cycle graphs and grid graphs have large hyperbolicity (almost equal to half of their diameters) and accordingly are considered non-hyperbolic. Generally, the smaller the hyperbolicity the closer the graph is to a tree and, as a result, the hyperbolicity property is more evident [65].

Next we briefly discuss the relationship among the definitions presented above. As stated earlier, all the above definitions of hyperbolicity are asymptotically equivalent. The best known relations between them are given in the following results.

**Proposition 1** ([110, 15]). Geodesic triangles of a $\delta$-hyperbolic graph are $3\delta$-slim.

**Proposition 2** ([11]). An unweighted graph with $\delta$-slim triangles is $(2\delta + \frac{1}{2})$-hyperbolic.

**Proposition 3** ([15, 43, 105, 110]). Geodesic triangles of geodesic $\delta$-hyperbolic spaces or graphs are $4\delta$-thin. Conversely, geodesic spaces or graphs with $\delta$-thin triangles are
\(\delta\)-hyperbolic.

Generally, the \(\delta\)-hyperbolicity of a graph is a global property that does not directly correlate with its other structural properties such as size, diameter, or degree distribution. However, it was shown that the diameter of a graph represents an upper bound for its \(\delta\)-hyperbolicity value.

**Lemma 1** ([95, 172]). For any graph \(G\) with diameter \(diam(G)\) and hyperbolicity \(hb(G)\), \(hb(G) \leq \frac{diam(G)}{2}\).

The following are several properties that are common among \(\delta\)-hyperbolic graphs.

- The existence of highly congested cores where all or most shortest paths among vertex pairs pass [124, 25, 155, 126, 63].

- Most shortest paths between a subset of vertices of a \(\delta\)-hyperbolic graph are contained within a bounded neighborhood (in terms of \(\delta\)) from the median vertex (the vertex with the minimum average distance to all other vertices) [63].

- Multiple shortest paths with common end vertices stay within a relatively small distance [13].

- Traffic can efficiently navigate (i.e., routing on a relatively short path compared to the graph’s diameter) from a vertex to some destination vertex using only local information [32, 95].

We call a space hyperbolic if it is \(\delta\)-hyperbolic in any of the above definitions for any \(\delta \geq 0\). However, note that there is no distinguished value of \(\delta\) other than zero: the property of being hyperbolic is well-defined, but the value of \(\delta\) can change between definitions. I.e., if a graph \(G\) has its \(\delta\)-hyperbolicity value equal to 4, then deciding the hyperbolicity of \(G\)
solely based on $\delta$ is insufficient. In this case, other graph properties need to be included in the $\delta$-hyperbolicity analysis.

3.2 Why do we study $\delta$-hyperbolicity in graphs?

The presence of hyperbolic networks in a big variety of applications attracted many researchers to investigate the negative curvature of different types of graphs. It turns out that many real-world networks are hyperbolic [177, 153, 155, 70, 6, 5, 13]. The motivation behind investigating the $\delta$-hyperbolicity in graphs is multiple fold.

The first is related to the notion of $\delta$-hyperbolicity in graphs as a tree-likeness measure. The tree-metric has well-studied structural properties that allow the design of efficient exact and approximation algorithms. Accordingly, graphs with metrics that are exactly or close to the tree metric have multiple algorithmic advantages. This closeness (between the metric of a graph to that of a tree) is measured by the value $\delta$. For example, tree-like graphs have efficient approximate solutions for several optimization problems. Consider a graph $G = (V, E)$ with $\delta$-hyperbolicity $\delta$ and with $n$ vertices.

- The diameter of $G$ can be approximated with an additive error $2\delta$ in linear time [58].
- The radius and center of a smallest enclosing ball for $G$ can be approximated with an additive error $3\delta$ in linear time [58].
- A distance approximating tree of $G$ can be constructed with an additive error $\delta \log_2(n)$ in linear time [58, 66].
- $G$ supports a $\delta \log(n)$-additive approximate distance labeling scheme\(^1\) with $O(\log^2(n))$ bit labels and constant time distance decoder [103, 66].

---

\(^1\)The distance labeling problem deals with labeling the vertices of a graph in order to compute the distance between any pair of vertices $u$ and $v$ using only the information stored in the labels of $u$ and $v$ (without any other information).
• $G$ admits an $O(\delta \log(n))$-additive routing labeling scheme with $O(\delta \log_2(n))$ bit labels and $O(\log_2(4\delta))$ time routing protocol [66].

• A good covering\(^2\) of a finite subset $V'$ of vertices of $V$ with balls of radius at most $R + \delta$ and a set of the same size of pairwise disjoint balls of radius $R$ centered at points of $V'$ can be found in polynomial time [64].

Recall that the smaller the value of the hyperbolicity parameter $\delta$, the closer the graph is to a tree.

The second fold revolves around the notion of graph embedding and using the hyperbolic space as a host metric space. Recall that graph embedding involves representing a graph’s distance matrix by mapping its vertices to another metric space (introduced in Section 2.2). The general idea in metric space embedding is to embed a high dimensional metric space into a much smaller dimensional space without creating large distortion in pairwise distances. Embedding graphs into hyperbolic metric spaces has several applications including efficient and secure routing. To enable simple and efficient routing in large complex networks, relying on the structure and properties of the network (as oppose to using routing tables) to reduce the amount of information needed for routing is favorable. By embedding a network into the Euclidean plane, each vertex can be assigned a location (using coordinates) and using greedy routing (in which a message progresses to the neighbor physically closer to destination) provides a simple way to reach a destination. Despite its simplicity, this type of greedy routing may not always work successfully. On the other hand, it was proven in [132] that every finite connected undirected graph has an embedding in the hyperbolic space which allows greedy routing for all source-destination pairs. Graphs with small hyperbolicity can be efficiently embedded in hyperbolic spaces.

\(^2\)The problem of covering subsets of $\delta$-hyperbolic graphs by balls is as follows. Given a subset $V'$ of a $\delta$-hyperbolic graph $G$ and a positive number $R$, let $\gamma(V', R)$ be the minimum number of balls of radius $R$ covering $V'$. 

38
It was shown empirically that the Internet topology embeds with better accuracy into a hyperbolic space than into a Euclidean space of comparable dimension [4, 178]. Moreover, in [138], it was observed that social networks embed well into hyperbolic spaces.

Third, hyperbolic graphs enjoy multiple characteristics that makes designing networks with a hyperbolic nature very attractive. The following quote from Jonckheere and Lohsoonthorns paper [123] illustrates this point.

“One of the many security concerns in modern data networks is eavesdropping, that is, unauthorized packet interception along a link with the potential of reconstructing the full message - if all packets are sent along the same optimum path from source to destination, as TCP does under normal conditions. One of the proposed patches to such a security breach is to send packets in a randomized fashion along different, nonoptimal routes [116]. Since the many routes have different delays, out-of-order packet arrival at the destination could create drops if the arrival sequence is altered by more than 3 slots. Unless some robustified TCP protocol is implemented, there is a need to restrict the paths to have costs bounded away from the optimum cost. On a graph, or on a surface or manifold for that matter, these near optimum paths may or may not remain within an identifiable neighborhood of the optimum path. In fact, in classical Riemannian geometry, this behavior is encapsulated in the concept of curvature: In a negative curvature space, the near optimal paths - referred to as quasi-geodesics - remain in a neighborhood of the optimal path, while in positive curvature space the near optimal paths could potentially spread across the whole manifold. For these facts to be applicable to graphs, there is a need to define a curvature concept for non-differentiable structures, a curvature concept upon which the features of Riemannian geometry can be extended to other
structures. In what has been referred to as the most significant development in geometry over the past 20 years, the concept of negative curvature has been redefined in terms of such a more primitive concept as distance and has hence become applicable to graphs. Here we restrict ourselves to hyperbolic (negative curvature) graphs, the chief reason being that Monte Carlo simulation has indicated that the popular “growth, preferential attachment” model of Internet build up promotes negative curvature (see [123], Section 3.6). Furthermore, from the point of view of network architecture, it appears desirable to design it hyperbolic, for the near optimal paths do not have to be sought across the whole network, but can be narrowed down to an identifiable neighborhood of the optimal path.”

Finally, the nature of $\delta$-hyperbolic graphs allows solving other graph related problems. Here we discuss several problems that are related to the core-periphery structure in graphs which is the focus of Part I of this dissertation. It was proven in [24] that traffic tends to pass a finite set of highly congested set of vertices in hyperbolic graphs. In [25], the authors show that the existence of a core in a $\delta$-hyperbolic graph is related to the distances between source-target vertex pairs. Longer distances (global traffic) results in a more concise and congested core. In contrast, shorter distances (local traffic) implies a non congested core.

The core congestion in $\delta$-hyperbolic networks was characterized in [155, 126, 63]. In [155], the authors show that $\delta$-hyperbolicity implies that the load at the graph’s core scales as $n^2$, where $n$ is the number of vertices, compared to scaling as $n^{1.5}$ in Euclidean graphs. The core here is defined based on the notion of the betweenness centrality, i.e., the set of vertices through which most traffic in the network passes. In [63], the authors show that for every subset $U$ of vertices of a $\delta$-hyperbolic graph $G = (V,E)$ there exists a vertex $u \in V$ such that the disk $D(u, 4\delta)$ of radius $4\delta$ centered at $u$ intercepts at least one half of
the total flow between all vertices of $U$. They also show that vertex $u$ is a vertex close to
the median of $U$.

In [16, 17] and in Section 4, we propose two empirical algorithms that exploit the
$\delta$-hyperbolicity property and separate the vertices of a graph into a core and periphery
parts. The separation models are designed based on the behavior of the shortest paths in
hyperbolic graphs and their concentration into one part of the network (or as characterized
above - bending of shortest paths towards the graph’s core).

Another interesting line of research suggests an interpretation of what is measured
by hyperbolicity and provides applications of this quantity. In [34], the authors consider
hyperbolicity as a measure of how democratic a network is (larger hyperbolicity values
indicate a more democratic network).

### 3.3 Computational aspect of $\delta$-hyperbolicity

It is clear from the $\delta$-hyperbolicity definitions listed in Section 3.1 that a direct application
of any definition leads to a brute force algorithm of time $\Omega(n^4)$, where $n$ is the number of
vertices. This is impractical for medium sized networks and even unachievable for large
networks with millions of vertices.

The four-point condition definition has the advantage that it only requires knowledge
of the distances between vertex pairs (as oppose to the slim-triangle definition that re-
quires knowledge of the actual paths). This makes the four-point condition definition very
attractive from a computer science point of view.

The four-point condition definition requires $O(n^4)$ time by simply checking the value of
hyperbolicity at each quadruple, and returning the maximum one. This is computationally
very expensive; for instance, the algorithm has to iterate over $625 \times 10^{12}$ quadruples to com-
pute the hyperbolicity of a graph with 5000 vertices. This requires hours (or even several
days) of computation. In fact, computing the hyperbolicity is costly even when distributed computing techniques are employed [7]. For example, computing the hyperbolicity of a graph with 8104 vertices needed about 7.5 hours on a parallel computer equipped with 512 CPUs and about 4 hours on a parallel computer equipped with 1015 CPUs. Therefore, a lot of work is directed towards computing the hyperbolicity of smaller networks (networks with up to few thousands of vertices) [155, 70, 6, 13, 5, 34]. Thus, a number of exact and approximation algorithms to compute the hyperbolicity value in graphs were proposed.

The best known exact algorithm for calculating the hyperbolicity is proposed in [101] and requires $O(n^{3.69})$ time using the (max, min) matrix multiplication [90] and the tree-metric embedding [110]. The authors also propose a $O(n^{2.69})$ 2-factor approximation algorithm by using a fixed base-point. Those bounds were slightly improved in [89] to an $\tilde{O}(\epsilon^{-1}n^{1+\omega}) (1+\epsilon)$-approximation algorithm, where $O(n^\omega) = O(n^{2.373})$ is the time complexity of matrix multiplications. Also a $(2+\epsilon)$-approximation algorithm with time complexity $\tilde{O}(\epsilon^{-1}n^{\omega})$ was proposed in [89].

A different line of work investigates and proposes heuristics that simplify and accelerate the hyperbolicity computation. For example, it was shown that the hyperbolicity of a graph equals the maximum hyperbolicity over its bi-connected components [153, 70] (Figure [7]). Also, it was shown in [95] that removing all vertices with degree one from a graph does not affect its hyperbolicity. The following result was proposed in [69] using graph decomposition. The goal is to reduce the computation of the hyperbolicity of a graph to the computation of the hyperbolicity of its connected subgraphs.

**Theorem 1** ([69]). Let $G = (V, E)$ be a connected graph, let $X$ be a clique-separator of $G$, and let $C_1, C_2, ..., C_r$ be the connected components of $G \setminus X$. Then $\max\{hb(G_1), hb(G_2), ..., hb(G_r)\} \leq hb(G) \leq \max\{\frac{1}{2}, hb(G_1), hb(G_2), ..., hb(G_r)\} + \frac{1}{2}$, where $G_i = G[C_i \cup X]$.

Several exact and approximation algorithms for computing the hyperbolicity offer low-
Figure 3.7: A graph with two bi-connected components $X$ and $Y$. $hb(G) = \max\{hb(X), hb(Y)\}$

Aging the overall computation time by restricting the number of considered quadruples to those ones that have the potential to maximize the $\delta$-hyperbolicity value. For example, the main idea in [70, 71, 35] is that the $\delta$-hyperbolicity of a quadruple tends to increase with the distances between its vertices. Therefore, all vertex pairs are first sorted according to their distances in a decreasing order. Then the hyperbolicity of quadruples (consisting of two of the sorted vertex pairs) will be analyzed in order, and whenever the hyperbolicity stops increasing, the algorithm terminates and returns the maximum hyperbolicity value that was found.

Even though those algorithms perform well in practice, they have the limitation of an $O(n^4)$ worst case time complexity and they require $O(n^2)$ memory space. To avoid the space limitation, the authors in [71] propose a heuristic that chooses (at each step) distant pairs using the 2-sweep heuristic proposed in [149] (instead of sorting vertex pairs according to their distances). This makes it scalable for very large graph. The algorithm runs in $O(k^2(n + m))$ time and has $O(n)$ memory requirement, where $n$ is the number of vertices, $m$ is the number of edges, and $k$ is a parameter of the algorithm.

Computing the hyperbolicity with sampling and the statistical aspects of curvature plots were discussed in [130]. The authors investigate the number of quadruples needed to
be sampled in order to achieve an interpretation of the curvature plot with a high degree of confidence. They sample quadruples uniformly at random until the standard error of the mean of $\delta'$ gets asymptotically small ($\delta'$ is an estimated value for the hyperbolicity of a graph). The same authors also use statistical sampling of triangles in networks in [155].

Using parameterization, the authors in [98] propose a number of linear FPT (Fixed Parameter Tractable) algorithms to compute graph hyperbolicity. For example, they propose two $O(k^4(n+m))$ running time algorithms. In the first algorithm, $k$ is the number of cover paths (defined as the minimum number of paths that cover all vertices and in which only the endpoints have degree greater than two). In the second algorithm, $k$ represents the feedback edge number (defined as the minimum number of edges needed to be deleted to obtain a forest).
Part I

$\delta$-Hyperbolicity in Graphs - A Structural Point of View
Even though there is a lot of research about the $\delta$-hyperbolicity in graphs in the literature, there are very few practical applications of this quantity. In this part of the dissertation, we provide a direct implication of $\delta$-hyperbolicity in graphs represented by the core-periphery structure.

The core-periphery structure has been found in many real-world networks. In this structure, one part of a graph constitutes a densely connected core and the rest of the graph forms a sparse periphery. This definition highlights the role of vertices during traffic exchange according to their structural position in the graph. Vertices that belong to the core play an important role in passing traffic among peripheral vertex pairs. It has been shown that this pattern of traffic exchange is due to the negative curvature or hyperbolicity in those graphs [23, 155].

In this part of the dissertation, we investigate the connection between the $\delta$-hyperbolicity of a graph and its core-periphery structure and propose algorithms to identify core vertices in a graph based on its $\delta$-hyperbolicity (Chapter 4). Furthermore, we discuss the interplay between the $\delta$-hyperbolicity and the core-periphery structure in graphs and empirically identify a graph’s core as the part that is responsible for maximizing its $\delta$-hyperbolicity parameter (Chapter 5).
Chapter 4

Core-Periphery Models for Graphs Based on their $\delta$-Hyperbolicity - an Example Using Biological Networks

This chapter presents a novel approach to separate a given graph into a core (a set of vertices in which most traffic concentrates) and a periphery parts. The core-periphery structure was identified in many types of real-world networks. Here we use the underlaying geometry of a graph (using its $\delta$-hyperbolicity) and introduce the eccentricity-based bending property. Then we exploit this property to identify the core vertices of a graph by proposing two models: the Maximum-Peak model and the Minimum Cover Set model.

We also investigate the hyperbolicity of several types of biological networks not only by considering Gromov’s notion of $\delta$-hyperbolicity but also by analyzing its relationship to other graph’s parameters. This new perspective allows us to classify graphs with respect
to their hyperbolicity, and to show that many biological networks are hyperbolic. This chapter is based on:


4.1 Introduction

Using graph-theoretical tools for analyzing complex networks to characterize their structures has been the subject of much research. It aids identifying multiple key properties as well as explaining essential behaviors of those systems. A common structure that has been widely recognized in social networks as well as other network disciplines is the core-periphery structure. Multiple coefficients have been proposed to examine the existence of the core-periphery organization in a graph [117, 75]; also, various methods have been introduced to identify the core of a graph [176, 37, 175].

The core-periphery structure suggests partitioning the graph into two parts: the core which is dense and cohesive and the periphery which is sparse and disconnected. Vertices in the periphery part interact through a series of intermediate core vertices. This pattern of communication (where traffic tends to concentrate on a specific subset of the vertices) has been observed in trees where distant nodes communicate via the central node (or nodes) in the tree. δ-Hyperbolicity, which is a measure that shows how close a graph is to a tree, suggests that any shortest path (geodesic) between any pair of vertices bends (to some extent) towards the core of the graph. This phenomenon has been justified by the global curvature of the network which (in case of graphs) can be measured using hyperbolicity (sometimes called also the negative curvature) [155].

There are multiple equivalent definitions for Gromov’s hyperbolicity. Let $G = (V, E)$ be a graph with a distance metric $d$ on $V$ such that the distance between two vertices $x$ and $y$ is the length of a shortest path between $x$ and $y$. A geodesic triangle $\Delta(x, y, z)$ for three arbitrary vertices $x, y, z \in V$ is the union of three shortest paths (geodesic segments) connecting $x, y,$ and $z$. In hyperbolic spaces, any vertex in any side of a geodesic triangle is contained in the $\delta$-neighborhood of the union of the two other sides. This forces the
sides of the triangle to be curved towards its center as its size increases [6].

Multiple complex networks such as the Internet [177, 153], data networks at the IP layer [155], and social and biological networks [13, 5] show low δ-hyperbolicity (low hyperbolicity suggests a structure that is close to a tree structure [65]). Also, it has been observed that networks with this property have highly connected cores [155]. Generally, the core of a graph is specified according to one or more centrality measures. For example, the degree centrality where the core of the graph is the set of vertices that have the highest number of connections, the betweenness centrality which considers the vertices that have the highest number of shortest paths passing through them as the core, and the eccentricity centrality in which the core has the subset of vertices that have the shortest distance to every other vertex.

The δ-hyperbolicity of graphs embeds multiple properties that facilitate solving several problems that found to be difficult in the general graph form. For example, diameter estimation [58], distance and routing labeling [66], and several routing problems [79, 136]. In this chapter, we investigate implications of the δ-hyperbolicity of a network and exploit them for the purpose of partitioning the graph into core and periphery parts.

Our main contributions in this chapter can be summarized as follows:

- We study the hyperbolicity of several real-world biological networks and show that the hyperbolicity of almost all the networks in our datasets is small. This confirms the results in [13]. However, unlike previous efforts, we analyze the relationship between the hyperbolicity and other global parameters of the graph. We find in most of our networks that the hyperbolicity $hb(G)$ is bounded by the logarithm of the diameter ($hb(G) \leq \log_2(diam(G))$) and the logarithm of the size of the graph in terms of the number of vertices and the number of edges ($hb(G) \leq \log_2(size(G))$). Based on this analysis we classify graphs with respect to their hyperbolicity into three categories:

- We formalize the notion of the eccentricity layering of a graph and employ it to introduce a new property that we find to be intrinsic to hyperbolic graphs: the eccentricity-based bending property. Unlike previous work, we investigate the essence of this bending in shortest paths by studying its relationship to the distance between vertex pairs.

- We exploit the eccentricity-based bending property, and based on it, propose two core-periphery separation models: the Maximum-Peak model and the Minimum Cover Set model.

- We apply both models to our biological graph datasets. In contrast to what have been observed in [117], we find that biological networks exhibit a clear-cut core-periphery structure. Then we investigate the relationship between the hyperbolicity of a graph and the conciseness of its core.

This chapter is organized as follows. In Section 4.2, we present some related work on the core-periphery structure and graph centrality measures in networks in general and in biological networks in particular. Section 4.3 describes the kinds of biological networks used in this chapter and presents a summary of their parameters. In Section 4.4, we measure, analyze, and classify the $\delta$-hyperbolicity of the networks in our datasets. Our classification is based on how the hyperbolicity is evident in those graphs. We also discuss how other graph parameters factor in to this classification. Finally in Section 4.5, we propose our eccentricity-based bending property followed by two core-periphery separation models as one of this property’s implications.
4.2 Background and related work

4.2.1 Core-periphery and network centrality in complex networks

The notion of the core-periphery structure has a long history in social network analysis. It deals with identifying the part (or parts) of the network that represents the central part in terms of the network distance, the most congested part in terms of the network traffic, the highly connected part in terms of vertex degrees, or any combination of the three. In [37], Borgatti and Everett formalize the core-periphery structure by developing two families of core-periphery models: the discrete model where vertices belong to one of two classes (core and periphery) and the continuous model which includes three classes (or more) of vertices (core, semiperiphery, and periphery). Then they propose algorithms for detecting each model by finding the partition which maximizes the correlation between the data matrix and the pattern matrix.

Seidman in [176] proposes the $k$-core decomposition as a tool to study the structural properties of large networks focusing on subsets of increasing degree centrality. It partitions the graph into subsets each of which is identified by removing vertices of degree smaller than $k$. Holme in [117] introduces a coefficient that measures if a network has a clear core-periphery structure based on the closeness centrality and the basic definition of clusters. He also shows that the core's neighborhood (for increasing radius) is highly dense (with respect to the number of edges). Unlike our result, he concludes that biological networks do not have a strong core-periphery organization. In [68], the authors show that for some families of random graphs with expected degrees there is a core to which almost all vertices are at distance less than or equal to $O(\log \log n)$. Leskovec et.al. [142] study community structures in large networks by analyzing a big range of different real-world networks. They identify the existence of multiple (smaller) communities that are attached to the core of
the network with very few connections (whiskers). They also observe that some graphs have cores with a nested core-periphery structure.

In the study of communication networks, the core is usually identified by the small dense part of the network that carries out most of the traffic under shortest path routing [25, 155, 174]. Narayan and Iraj [155] show that the load scales as $O(n^2)$ with the number of vertices $n$ at the core of the network. The asymptotic traffic flow in hyperbolic graphs has been studied in [25]. It shows that a vertex $v$ belongs to the core if there exists a finite radius $r$ such that the amount of the traffic that passes through the ball centered at $v$ and with radius $r$ behaves asymptotically as $\theta(n^2)$ as the number of vertices $n$ grows to infinity. The existence of the core in large networks such as the Internet motivates researchers to embed the Internet distance metric in a hyperbolic space for distance estimation [177, 178].

It is quite natural to associate the concepts of the network’s core and the network’s center. Borgatti and Everett [37] argue that each central vertex is a core vertex but the opposite is not true; consequently, all coreness measures are centrality measures.

4.2.2 Biological networks and the core-periphery structure

It has been found in several fields that looking at the overall system reveals more about the functionality of its components as opposed to inspecting its individual elements. Therefore, various types of large-scale biological networks have been constructed to capture the different kinds of interactions between their components. For instance, protein-protein interaction networks have been used to identify the function of individual proteins as well as the purpose behind some unknown interactions [166]. A lot of research efforts were directed to discover some topological properties of the biological networks. It has been shown that some protein structures and protein-protein interaction (PPI) networks [109, 181] and a number of metabolic networks [191, 97, 106] exhibit the small-world prop-
erty (a graph is small-world when its diameter is bounded by the logarithm of its size \( \text{diam}(G) \leq \log_2(\text{size}(G)) \)). A lot of work has focused on analyzing the degree distribution of different biological networks. Power-law degree distribution was caught in protein structure networks [109], PPI networks [181, 106], and metabolic networks [122].

Structure analyses of some biological networks have detected the presence of hierarchal, modular, and core-periphery organization structures. The core-periphery model of several types of biological networks has been studied thoroughly in the literature. [75] proposes a parameter that detects the existence of a core-periphery structure in a metabolic network based on the closeness centrality of metabolites and the network connectivity. In [128], the yeast protein interaction network was analyzed based on the betweenness centrality. They conclude that the high betweenness low connectivity proteins may be working as connectors between separate modules. Using mathematical tools that have been used to analyze sociological networks, [97] studies recognizing the central metabolites in a metabolic pathway network. In [145], the authors demonstrate a systematic exploration of the core-periphery model in protein interaction networks that depends on the connectivity of the vertices. They also classify the peripheral vertices based on their structural relationship with the core. In [146], the authors identify the central metabolites using degree centrality and closeness centrality. They also show the relationship between the average path length and the closeness centralization index of metabolic network.

### 4.3 Datasets

There are various forms of biological networks that have different characteristics according to their origins and to their construction methodologies. Generally, the vertices in a biological network represent biomolecules such as proteins, genes, or metabolites, and the edges represent a chemical, physical, or functional interaction between the connected vertices.
Each of the networks used in this chapter belongs to one of the following types:

- **Protein Interaction (PI) Networks**: Generally, in a PI network, the vertices represent different proteins and the edges represent the connections between the interacting proteins. PI networks have been described as small-world and scale-free networks [128].

- **Neural Networks**: They contain neurons (vertices) which are connected together through synapsis (edges). Neurons have a high tendency to form clusters based on their spatial location. Neural networks are small-world networks [128].

- **Metabolic Networks**: Metabolic networks are represented by metabolites (vertices) and biochemical reactions (directed edges). Usually, metabolites are small molecules (for example amino acids); however, they also can be macromolecules. Metabolic networks show the small-world property and they have a high clustering coefficient. Also, they follow the power-law degree distribution [128].

- **Transcription Networks**: Networks in which vertices are genes and edges represent different interactions (interrelationships) between genes. A transcription network is

Table 4.1: Graph datasets and their parameters: number of vertices $|V|$; number of edges $|E|$; graph’s size $\text{size}(G) = |V| + |E|$; average degree $\bar{d}$; diameter $\text{diam}(G)$; radius $\text{rad}(G)$.

| Network Category   | Network          | $|V|$ | $|E|$ | $\log_2(\text{size}(G))$ | $\bar{d}$ | $\text{diam}(G)$ | $\text{rad}(G)$ |
|--------------------|------------------|------|------|---------------------------|-----------|-----------------|-----------------|
| PI Networks        | B-YEAST-PI       | 1465 | 5839 | 12.8                      | 7.97      | 8               | 5               |
|                    | E-COLI-PI        | 126  | 581  | 9.5                       | 9.2       | 5               | 3               |
|                    | YEAST-PI         | 1728 | 11003| 13.6                      | 12.7      | 12              | 7               |
|                    | S-CEREVISIAE-PI  | 537  | 1002 | 10.5                      | 3.7       | 11              | 7               |
|                    | H-PYLORI-PI      | 72   | 112  | 7.5                       | 3.1       | 7               | 5               |
| Neural Networks    | MACAQUE-BRAIN-1  | 45   | 463  | 9                         | 11.3      | 4               | 2               |
|                    | MACAQUE-BRAIN-2  | 350  | 5198 | 12.4                      | 29.7      | 4               | 3               |
| Metabolic Networks | E-COLI-METABOLIC | 242  | 376  | 9.3                       | 3.1       | 16              | 9               |
|                    | C-ELEGANS-METABOLIC | 453  | 4596 | 12.3                      | 8.9       | 7               | 4               |
| Transcription Networks | YEAST-TRANScription | 321  | 711  | 10                         | 4.4       | 9               | 5               |
Table 4.2: The hyperbolicity of each graph in the datasets. $|V|$; number of edges $|E|$; diameter $diam(G)$; radius $rad(G)$; hyperbolicity $hb(G)$; and the average hyperbolicity $hb'(G)$.

| Network              | $|V|$ | $|E|$ | $diam(G)$ | $rad(G)$ | $hb(G)$ | $hb'(G)$ |
|----------------------|------|------|-----------|----------|---------|---------|
| B-yeast-PI           | 1465 | 5839 | 8         | 5        | 2.5     | 0.299   |
| E-coli-PI            | 126  | 581  | 5         | 3        | 2       | 0.251   |
| Yeast-PI             | 1728 | 11003| 12        | 7        | 3.5     | 0.322   |
| S-cerevisiae-PI      | 537  | 1002 | 11        | 7        | 4       | 0.419   |
| H-pylori-PI          | 72   | 112  | 7         | 5        | 3       | 0.368   |
| Macaque-brain-1      | 45   | 463  | 4         | 2        | 1.5     | 0.231   |
| Macaque-brain-2      | 350  | 5198 | 4         | 3        | 1.5     | 0.203   |
| E-coli-metabolic     | 242  | 376  | 16        | 9        | 4       | 0.483   |
| C-elegans-metabolic  | 453  | 4596 | 7         | 4        | 1.5     | 0.133   |
| Yeast-transcription  | 321  | 711  | 9         | 5        | 3       | 0.365   |

...one type of the gene regulatory networks [150].

We analyze the protein interaction networks of Budding yeast [45], Escherichia coli [46], Yeast [67], Saccharomyces cerevisiae [121], and Helicobacter Pylori [165]. Also, we analyze two different brain area networks of the Macaque monkey [156] [151]; and the metabolic networks of the Escherichia coli [147] and the Caenorhabditis elegans [91]. Finally, we analyze the Yeast transcription network [150].

All graphs in the datasets are unweighted, and we only consider the largest connected component of each network. The size of this component for each network is presented in Table 4.1. We also ignore the directions of the edges.

### 4.4 $\delta$-Hyperbolicity and graph classification

For the purpose of investigating the hyperbolicity of networks, it seems natural to analyze and classify the graphs based on their hyperbolicity. The classification should reflect how strong (evident) the tree-likeness is in the graph’s structure.
Figure 4.1: Distribution of quadruples over different values of $hb(x, y, u, v)$.

In the following subsections not only we measure the hyperbolicity of each of the networks in our graph datasets, but also we relate this value to other graph’s parameters. Upon this analysis, we provide our hyperbolicity-based classification of the graphs.

### 4.4.1 Hyperbolicity of biological networks

We measure $\delta$-hyperbolicity on each bi-connected component of each network in the datasets presented in Section 4.3 using Gromov’s four-point condition. For each network $G$, we identify a bi-connected component that maximize the value of the hyperbolicity since the hyperbolicity of a graph equals the maximum hyperbolicity of its bi-connected components [195, 153, 31].

Table 4.2 shows that almost all networks in our datasets have small hyperbolicity. Even though the definition of $\delta$-hyperbolicity considers the difference between the largest two distance sums among any quadruple and takes into account only the maximum one, this absolute analysis is deficient. Similar to [153, 6], closer analysis to the distribution of the values of $hb(x, y, u, v)$ (see Figure 4.1) shows that only a very small percent of the quadruples (less than 1%) have the maximum value of $\delta$-hyperbolicity while most
<table>
<thead>
<tr>
<th>$2\delta$ Quadruples</th>
<th>%</th>
<th>$2\delta$ Quadruples</th>
<th>%</th>
<th>$2\delta$ Quadruples</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1276345762</td>
<td>73%</td>
<td>0 5529039</td>
<td>55%</td>
<td>0 529467</td>
<td>52%</td>
</tr>
<tr>
<td>1 448324099</td>
<td>26%</td>
<td>1 3934917</td>
<td>39%</td>
<td>1 273481</td>
<td>27%</td>
</tr>
<tr>
<td>2 6795007</td>
<td>1%</td>
<td>2 540463</td>
<td>5%</td>
<td>2 199186</td>
<td>19%</td>
</tr>
<tr>
<td>3 3057</td>
<td>0.002%</td>
<td>3 4658</td>
<td>0.04%</td>
<td>3 21971</td>
<td>2%</td>
</tr>
<tr>
<td>4 48</td>
<td>0.0005%</td>
<td>4 4651</td>
<td>1%</td>
<td>5 32</td>
<td>0.003%</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>5</td>
<td></td>
<td>6 2</td>
<td>0.0002%</td>
</tr>
</tbody>
</table>

Figure 4.2: Distribution of quadruples over different values of $\delta = hb(x, y, u, v)$ (three datasets): (a) C-elegans-metabolic; (b) E-coli-PI; (c) H-pylori-PI.

quadruples have $hb(x, y, u, v) = 0$ (about 40% - 70%). Figure 4.2(a), (b), and (c) present samples of the distribution of the quadruples over the different values of $hb(x, y, u, v)$.

This observation makes it equally important to calculate the value of the average delta $hb'(G)$ (see Table 4.2). This suggests that the maximum value of $hb(G)$ was not expressive for the graph but was affected by some outliers. The average hyperbolicity is defined as:

$$hb'(G) = \frac{\sum_{x,y,u,v \in V} \delta(x, y, u, v)}{\binom{|V|}{4}}$$

### 4.4.2 Analysis and discussion

Our goal is to categorize graphs with respect to their hyperbolicity into three classes: strongly-hyperbolic graphs, hyperbolic graphs, and non-hyperbolic graphs.

Taking into account that trees are 0-hyperbolic, generally, the smaller the value of $hb(G)$, the closer the graph’s structure is to a tree. However, studying the tree-like structure of graphs based solely on the value of the hyperbolicity may not be sufficient to characterize their closeness to a tree structure for two reasons. First, the hyperbolicity is a relative
measure. For example, for a given graph $G = (V, E)$, a value of $hb(G) = 10$ can be seen as too large when $size(G) \approx 10^2$. In this case, the structure of $G$ can be fairly described as being far from a tree. However, when $size(G) \approx 10^7$, the hyperbolicity $hb(G) = 10$ looks much smaller and indicates a tree-like structure. Second, small graph size and (or) small diameter directly yield low hyperbolicity; our classification should be sensitive to such cases. In other words, small $hb(G)$ does not always suggest a graph with a tree-like structure; other graph attributes that might impact the hyperbolicity must be investigated.

We find that the graph’s size $size(G)$ and the graph’s diameter $diam(G)$ play an important role in deciding how hyperbolic a given graph is.

Since finite graphs will always have a finite value for $hb(G)$ such that the four-point condition is true, it is natural to think that the non-hyperbolic class includes only infinite graphs. However, in this study, we only consider finite graphs; accordingly, a non-hyperbolic graph in our sense is a graph with too large $hb(G)$ with respect to the logarithm of the graph’s size, i.e., when it violates the following condition: $hb(G) \leq \log_2(size(G))$.

Recall that $size(G) = |V| + |E|$. In cases where $hb(G) \leq \log_2(size(G))$, we move on and compare $hb(G)$ with the diameter of the graph. To guarantee that the value of the diameter is not directly impacted by the size of the graph, first we require the diameter to be within the following bound: $diam(G) \leq \log_2(size(G))$. This is especially important for excluding the uninteresting cases where the small diameter is a result of the graph’s small size.

Multiple previous works have analyzed the relationship between the hyperbolicity and the diameter. Recall that the graph’s diameter represents an upper bound for $hb(G)$.

**Lemma 2.** [172, 95]. For any graph $G$ with diameter $diam(G)$ and hyperbolicity $hb(G)$, $hb(G) \leq \frac{diam(G)}{2}$.

Moreover, the authors in [129] (using the slim triangles condition for hyperbolicity)
and in [125] (using the four-point condition) argue that the hyperbolicity of the graph is “actually” present when the value of $hb(G)$ is much smaller than the graph’s diameter. They specify that for a graph to be hyperbolic the value of $hb(G)/diam(G)$ must asymptotically scale to zero. In this section, our goal is to ensure that the low value for the hyperbolicity is not a consequence of the graph’s small diameter.

Interestingly, for most of the networks in our graph datasets, we find that $hb(G) \leq \log_2(diam(G))$. Therefore, we say that a graph is

- **Strongly-hyperbolic** if it exhibits (1) $diam(G) \leq \log_2(size(G))$ and (2) $hb(G) \leq \log_2(diam(G))$. Note that a graph that satisfies (1) is in fact a small-world graph.
- **Hyperbolic** when it violates either (1) or (2).
- **Non-hyperbolic** in all other cases.

Note that in the case when $hb(G)$ is small but greater than $\log_2(size(G))$, $hb(G)$ is an insufficient indication for hyperbolicity. We are not saying that the graph is far from the
tree structure; still we point out that hyperbolicity is not very expressive in this case and other tree-likeness measurements may be used.

As Table 4.1 shows, all networks in the datasets, with the exception of networks S-cerevisiae-PI and E-coli-metabolic, exhibit the small-world property. Also, Table 4.2 shows that $hb(G) \leq \log_2(diam(G))$ in all graphs except for the S-cerevisiae-PI and the H-pylori-PI networks. As a result, those three graphs have been classified as hyperbolic graphs, and their hyperbolicity values are on the larger side ($hb(G)$ for S-cerevisiae-PI is 4, $hb(G)$ for H-pylori-PI is 3, and $hb(G)$ for E-coli-metabolic is 4). Also, the values of the average $hb(G)$ ($hb'(G)$) from Table 4.2 are high compared to other networks (0.419, 0.368, and 0.483 respectively). In Figure 4.3, we show this classification. Note that none of the networks in our graph datasets is non-hyperbolic (it is always the case that $hb(G) \leq \log_2(size(G))$).

Quantifying “small” and “large” for $hb(G)$ is not straightforward simply because it is relative. Therefore, we judge according to the difference between $hb(G)$ and $\log_2(\log_2(size(G)))$. The more substantial this difference is the closer the graph’s structure is to a tree structure. For example in Table 4.3, networks C-elegans-metabolic and Macaque-Brain-2 are metrically closer to trees than networks Yeast-Transcription and Yeast-PI.

We also compared the value of $hb(G)$ to $\log_2(size(G))$ and to $\log_2(diam(G))$, and their average for the networks in the two classes: strongly-hyperbolic and hyperbolic. The values are listed in Table 4.3. Note that the three last columns of Table 4.3 show higher values for the hyperbolic graphs.

4.5 Core-Periphery models based on $\delta$-hyperbolicity

It was suggested in [155, 25] that the highly congested cores in many communication networks can be due to their hyperbolicity or negative curvature. Those cores are represented
Table 4.3: The ratio of $hb(G)$ to logarithm of the size of the graph $\log_2(size(G))$ and logarithm of the diameter $\log_2(diam(G))$ for each network in our datasets. $\epsilon$ is the difference $hb(G) - \log_2(\log_2(size(G)))$. The Average value presents the average $(\frac{hb(G)}{\log_2(size(G))} + \frac{hb(G)}{\log_2(diam(G))})/2$.

<table>
<thead>
<tr>
<th>Network</th>
<th>$\log_2(size(G))$</th>
<th>diam(G)</th>
<th>$hb(G)$</th>
<th>$\epsilon$</th>
<th>log2(size(G))</th>
<th>log2(diam(G))</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly-hyperbolic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-elegans-metabolic</td>
<td>12.3</td>
<td>7</td>
<td>1.5</td>
<td>2.1</td>
<td>0.121</td>
<td>0.536</td>
<td>0.329</td>
</tr>
<tr>
<td>B-yeast-PI</td>
<td>12.8</td>
<td>8</td>
<td>2.5</td>
<td>1.2</td>
<td>0.195</td>
<td>0.833</td>
<td>0.514</td>
</tr>
<tr>
<td>Macaque-brain-2</td>
<td>12.4</td>
<td>4</td>
<td>1.5</td>
<td>2.1</td>
<td>0.120</td>
<td>0.750</td>
<td>0.435</td>
</tr>
<tr>
<td>E-coli-PI</td>
<td>9.5</td>
<td>5</td>
<td>2</td>
<td>1.2</td>
<td>0.210</td>
<td>0.869</td>
<td>0.540</td>
</tr>
<tr>
<td>Yeast-transcription</td>
<td>10</td>
<td>9</td>
<td>3</td>
<td>0.3</td>
<td>0.300</td>
<td>0.937</td>
<td>0.636</td>
</tr>
<tr>
<td>Macaque-brain-1</td>
<td>9</td>
<td>4</td>
<td>1.5</td>
<td>1.5</td>
<td>0.166</td>
<td>0.750</td>
<td>0.458</td>
</tr>
<tr>
<td>Yeast-PI</td>
<td>13.6</td>
<td>12</td>
<td>3.5</td>
<td>0.3</td>
<td>0.275</td>
<td>0.972</td>
<td>0.626</td>
</tr>
<tr>
<td>Hyperbolic Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-cerevisia-PI</td>
<td>10.5</td>
<td>11</td>
<td>4</td>
<td>0.380</td>
<td>1.142</td>
<td>0.761</td>
<td></td>
</tr>
<tr>
<td>H-pylori-PI</td>
<td>7.5</td>
<td>7</td>
<td>3</td>
<td>0.400</td>
<td>1.071</td>
<td>0.736</td>
<td></td>
</tr>
<tr>
<td>E-coli-metabolic</td>
<td>9.3</td>
<td>16</td>
<td>4</td>
<td>0.430</td>
<td>1.000</td>
<td>0.715</td>
<td></td>
</tr>
</tbody>
</table>

by vertices that belong to most shortest paths (shortest-path betweenness centrality) and (or) have minimum distances to all other vertices (eccentricity centrality). It was also observed that the negative curvature causes most of the shortest paths to bend making the peak of the arc formed by a shortest path to pass through a vertex in the core. In this section, we formalize this notion of bending in shortest paths by introducing an important property that is intrinsic to $\delta$-hyperbolic graphs (the eccentricity-based bending property). Then we use the implication of this property to aid the partitioning of a graph into core and periphery parts by proposing two models: the Maximum-Peak model and the Minimum Cover Set model. We further apply our models to the biological networks in our datasets. In contrast to what have been observed in [117], we show that biological networks do exhibit a core-periphery structure.

4.5.1 Eccentricity-based bending property of $\delta$-hyperbolic graphs

The eccentricity layering of a graph $G = (V, E)$ denoted as $E\mathcal{L}(G)$ partitions its vertices into concentric circles or layers $\ell_r(G), r = 0, 1, \ldots$. Each layer $r$ is defined as $\ell_r(G) = \{u \in V \mid$
Figure 4.4: Eccentricity Layering of a graph. Darker vertices belong to lower layers.

\[ V : \text{ecc}(u) - \text{rad}(G) = r \}. \] Here \( r \) represents the index of the layer. The inner-most layer (layer 0) encloses the graph’s center \( C(G) \), i.e., all vertices with the minimum eccentricity or of eccentricity equal to \( \text{rad}(G) \); this layer has index \( r = 0 \). Then the first layer includes all vertices who have their eccentricities equal to \( \text{rad}(G) + 1 \), and so on. The vertices in the last layer (outer-most layer) will have eccentricities equal to the diameter of the graph. Figure 4.4 demonstrates an illustration for the layers.

Any vertex \( v \in \ell_r(G) \) has level (or layer) \( \text{level}(v) = r \). We use layer 0 to indicate the central layer, layer 1 to indicate the layer that directly succeeds the central layer, and etc. Therefore, when we say lower layers, we mean layers towards the central layer, and the opposite for higher layers. Figure 4.5 and Table 4.4 show the distribution of the vertices over different layers of the eccentricity layering of each graph in our datasets. Note that the vertices’ population is denser in middle layers in almost all networks.

Let \( G = (V, E) \) be a \( \delta \)-hyperbolic graph, \( \mathcal{E}L(G) \) be its eccentricity layering, and \( C(G) \) be its center. In [58], the following useful metric property of \( \delta \)-hyperbolic graphs was proven.

**Lemma 3** ([58]). Let \( G \) be a \( \delta \)-hyperbolic graph and \( x, y, v, u \) be its four arbitrary vertices.
If $d(v, u) \geq \max\{d(y, u), d(x, u)\}$, then $d(x, y) \leq \max\{d(v, x), d(v, y)\} + 2hb(G)$.

We use this property to establish the following few interesting statements.

**Proposition 4.** Let $G$ be a $\delta$-hyperbolic graph and $x, y, s$ be arbitrary vertices of $G$. If $d(x, y) > 4hb(G) + 1$, then $d(w, s) < \max\{d(x, s), d(y, s)\}$ for any middle vertex $w$ of any shortest $(x,y)$-path.

**Proof.** Let $w$ be a middle vertex of a shortest $(x, y)$-path and let $d(x, w) = \lfloor d(x, y)/2 \rfloor$. Assume, by way of contradiction, that $d(w, s) \geq \max\{d(x, s), d(y, s)\}$. Then, by Lemma 3, $d(x, y) \leq \max\{d(w, x), d(w, y)\} + 2hb(G) = d(w, y) + 2hb(G)$. Since $d(x, y) = d(x, w) + d(w, y)$, we obtain $d(x, w) \leq 2hb(G)$, giving $d(x, y) \leq 4hb(G) + 1$.  

**Proposition 5.** Let $G$ be a $\delta$-hyperbolic graph and $x, y$ be arbitrary vertices of $G$. If $d(x, y) > 4hb(G) + 1$, then on any shortest $(x, y)$-path there is a vertex $w$ with $\text{ecc}(w) < \max\{\text{ecc}(x), \text{ecc}(y)\}$.

**Proof.** Let $w$ be a middle vertex of a shortest $(x, y)$-path given by Proposition 4 and $s$ be a vertex such that $d(w, s) = \text{ecc}(w)$. Then, by Proposition 4, $\text{ecc}(w) = d(w, s) < \max\{d(x, s), d(y, s)\} \leq \max\{\text{ecc}(x), \text{ecc}(y)\}$. 

Figure 4.5: Distribution of vertices over different layers of the graph’s eccentricity layering.
Table 4.4: Distribution of vertices over different layers with respect to the graph’s eccentricity layering. \( \text{rad}(G) \) is the graph’s radius; \(|V|\) is the number of vertices of each graph.

| Network                        | \( \text{rad}(G) \) | \(|V|\) | Layer  |
|-------------------------------|-----------------------|-------|--------|
|                               |                       |       | 0 1 2 3 4 5 6 7 |
| B-YEAST-PI                    | 5                     | 1465  | 90 902 456 17 |
| E-COLI-PI                     | 3                     | 126   | 6 87 33 |
| YEAST-PI                      | 7                     | 1728  | 53 419 805 393 55 3 |
| S-CEREVISIAE-PI               | 7                     | 537   | 26 198 219 79 15 |
| H-PYLORI-PI                   | 5                     | 72    | 14 41 17 |
| MACAQUE-BRAIN-1               | 2                     | 45    | 1 30 14 |
| MACAQUE-BRAIN-2               | 3                     | 350   | 194 156 |
| E-COLI-METABOLIC              | 9                     | 242   | 5 43 54 50 42 36 10 2 |
| C-ELEGANS-METABOLIC           | 4                     | 453   | 17 353 69 14 |
| YEAST-TRANSCRIPTION           | 5                     | 321   | 3 58 204 49 7 |

Denote by \( D(y,r) := \{ v \in V : d(v,y) \leq r \} \) the closed disk of \( G \) of radius \( r \) and centered at vertex \( y \).

**Proposition 6.** Let \( G \) be a \( \delta \)-hyperbolic graph and \( y \) be its arbitrary vertex. Then, \( C(G) \subseteq D(y,4\text{hb}(G)+1) \) or there is a vertex \( v \in D(y,2\text{hb}(G)+1) \) such that \( \text{ecc}(v) < \text{ecc}(y) \).

**Proof.** Consider an arbitrary vertex \( x \in C(G) \) and any shortest \((x,y)\)-path \( P \). Assume that \( x \notin D(y,4\text{hb}(G)+1) \), i.e., \( P \) has length at least \( 4\text{hb}(G)+2 \). Consider the vertex \( v \) of this path which is at distance \( 2\text{hb}(G)+1 \) from \( y \). We claim that \( \text{ecc}(v) < \text{ecc}(y) \). Let \( s \) be a vertex such that \( \text{ecc}(v) = d(v,s) \) and assume that \( \text{ecc}(v) \geq \text{ecc}(y) \), i.e., \( d(v,s) \geq \text{ecc}(y) \geq d(y,s) \).

We know also that \( d(v,s) \geq \text{ecc}(y) \geq \text{rad}(G) = \text{ecc}(x) \geq d(x,s) \). Then, Lemma 3 applied to \( d(v,s) \geq \max\{d(y,s),d(x,s)\} \), gives \( d(x,y) \leq \max\{d(v,x),d(v,y)\} + 2\text{hb}(G) = d(v,x) + 2\delta \).

But, since \( d(x,v) + d(v,y) = d(x,y) \), we get \( 2\text{hb}(G) \geq d(x,y) - d(x,v) = d(v,y) = 2\text{hb}(G) + 1 \), which is impossible.

We define the bend in shortest paths between two distinct vertices \( u \) and \( v \) with \( d(u,v) \geq \)
2, denoted by \( \text{bend}(u,v) \), as follows:

**Definition 7.** \( \forall u,v \in V \), \( \text{bend}(u,v) = \min \{ \text{level}(z) : z \in V \text{ and } d(u,z) + d(z,v) = d(u,v) \} \). Here \( \text{level}(z) = r \iff z \in \ell_r(G) \).

We say that shortest paths between \( u \) and \( v \) bend if and only if a vertex \( z \) with \( \text{ecc}(z) < \max \{ \text{ecc}(u), \text{ecc}(v) \} \) exists in a shortest path between \( u \) and \( v \). In this case we say also that the pair of vertices \( u \) and \( v \) bends. The parameter \( \text{bend} \) decides the extent (or the level) to which shortest paths between vertices \( u \) and \( v \) curve towards the center of the graph (since we are always looking for a \( z \) that belongs to a smaller layer according to the eccentricity layering). Note that in some cases \( \text{bend}(u,v) \) will be assigned either \( \text{ecc}(u) \) or \( \text{ecc}(v) \), whatever is smaller. For example, see the shortest path \( \rho(u,v) \) in Figure 4.4.

Now we analyze how vertex pairs behave in terms of their bending towards the center of the graph \( C(G) \). Specifically, we investigate the effect of the distance between a pair of vertices on the bend of the shortest paths between them. Our findings in this context are summarized in the following two statements.

(A) Despite their distances, most vertex pairs bend. Moreover, among those bending vertex pairs, the majority is represented by those that are sufficiently far from each other.

(B) There is a direct relation between the distance among vertex pairs and how close to the center a shortest path between them bends.

**Motivation and Empirical Evaluation of (A)**

In light of Proposition 5, we investigate how vertex pairs of various distances act with respect to the eccentricity-based bending property. Given two vertices \( u \) and \( v \), we know
Table 4.5: The effect of the distance $k$ between vertex pairs on the bending property. Out of all vertex pairs with distance at least $k$, we show the percentage of those that bend for three of the networks in our graph datasets.

<table>
<thead>
<tr>
<th>$k$</th>
<th>C-elegans-metabolic (diam$(G) = 7$)</th>
<th>B-yeast-PI (diam$(G) = 8$)</th>
<th>Yeast-transcription (diam$(G) = 9$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>96.99%</td>
<td>93.10%</td>
<td>96.65%</td>
</tr>
<tr>
<td>3</td>
<td>99.89%</td>
<td>94.87%</td>
<td>97.77%</td>
</tr>
<tr>
<td>4</td>
<td>100%</td>
<td>98.43%</td>
<td>99.11%</td>
</tr>
<tr>
<td>5</td>
<td>100%</td>
<td>99.93%</td>
<td>99.88%</td>
</tr>
<tr>
<td>6</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>7</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>8</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>9</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

from Proposition 5 that when $d(u, v) > 4hb(G) + 1$, then $\rho(u, v)$ bends. We now analyze this bending in $\rho(u, v)$ when (possibly) $d(u, v) \leq 4hb(G) + 1$. We are motivated by the fact that the small-world property is observed in many biological networks [155] (also refer to Sections 4.2.2 and 4.3). Accordingly, shortest paths of lengths $4hb(G) + 1$ may not even exist in those networks.

Interestingly, we noticed the bend in the majority of shortest paths with lengths $\leq 4hb(G) + 1$. A quick look at Table 4.5 shows that a big percent of vertex pairs of distance at least two bend. For example, in graph C-elegans-metabolic, about 97% of the vertex pairs with distances at least two bend. Furthermore, even though it is not always true (see the example in Figure 4.6), all graphs in our graph datasets have bends for vertex pairs that are at distance equal to the graphs’ diameters.

To quantify the distances at which the bend in vertex pairs happens in each of the networks in our datasets, we define two parameters: the absolute curvity and the effective curvity.

Let $k$ be the distance between a pair of vertices ($2 \leq k \leq \text{diam}(G)$). The absolute curvity $k^*$ is the minimum $k$ such that all pairs with distance $\geq k$ bend. The effective
Motivation and Empirical Evaluation of (B)

Here we examine the impact of the distance on the level to which vertex pairs bend. Let $k$ be the distance between two vertices such that $2 \leq k \leq \text{diam}(G)$. Consider $\mu_k$ as the lowest layer that all vertex pairs of distance $\geq k$ bend to. We define it as: $\mu_k =$
Table 4.6: The absolute curvity $k^*$ and the effective curvity $\tilde{k}$ for our graph datasets. The absolute curvity $k^*$ is the minimum $k$ such that all pairs with distance $\geq k$ bend. The effective curvity $\tilde{k}$ is the minimum $k$ such that more than 90% of the pairs with distance $\geq k$ bend. $\delta = h_b(G)$

<table>
<thead>
<tr>
<th>Network</th>
<th>$diam(G)$</th>
<th>$\delta$</th>
<th>$k^*$</th>
<th>$\tilde{k}$</th>
<th>$k$</th>
<th>$\tilde{k}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-yeast-PI</td>
<td>8</td>
<td>2.5</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>E-coli-PI</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Yeast-PI</td>
<td>12</td>
<td>3.5</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>S-cerevisiae-PI</td>
<td>11</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>H-pylori-PI</td>
<td>7</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Macaque-brain-1</td>
<td>4</td>
<td>1.5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Macaque-brain-2</td>
<td>4</td>
<td>1.5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>E-coli-metabolic</td>
<td>16</td>
<td>4</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>C-elegans-metabolic</td>
<td>7</td>
<td>1.5</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Yeast-transcription</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

This allows us to look at how the bends of the vertex pairs behave with respect to different distances. The algorithm used to find every $\mu_k$ is shown in Algorithm 1. A summary of the results is shown in Figure 4.7.

As expected, we found a direct relation between the distance of vertex pairs and their bend. For example, Figure 4.7 shows that in network yeast-pi, vertex pairs with distances 3, 6, and 9 from one another bend to layers 4, 3, and 2 respectively.

In fact, this observation is a direct implication of Proposition 4 and Proposition 5. For every pair of vertices $u$ and $v$ that are sufficiently far from each other, there is a vertex $z$ in the middle of $\rho(u, v)$ that has less eccentricity (i.e., has a smaller level according to the eccentricity layering) than $u$ or $v$. Applying this argument recursively to the new vertex pairs $(u, z)$ and $(z, v)$, if the distance between $u$ and $z$ is still sufficiently large, then there is another vertex $w$ in the middle of a shortest path $\rho(u, z)$ such that $ecc(w) < \max\{ecc(u), ecc(z)\}$. The same applies to the vertex pair $(z, v)$.

Also, we know from (A) that the distance among vertex pairs does not need to be very
Algorithm 1. Decide the lowest layer $\mu_k$ that all vertex pairs with distance $\geq k$ bend to. $2 \leq k \leq \text{diam}(G)$.

1. for $k \leftarrow 2$ to $\text{diam}(G)$ do
2. max $\leftarrow -1$
3. for every pair $(x, y)$ do
4. if $(d(x, y) \geq k)$ then
5. if $(\text{bend}(x, y) > \text{max})$ then
6. max $\leftarrow \text{bend}(x, y)$
7. end if
8. end if
9. end for
10. $\mu_k \leftarrow \text{max}$
11. print $\mu_k$
12. end for

large for a vertex pair to bend. For example, most vertex pairs with distance as small as two from each other bend.

4.5.2 Core-periphery identification using the eccentricity-based bending property

A well-defined center of a graph is a good starting point for locating its core. According to the pattern of data exchange discussed earlier (in which a shortest path between distant vertices bends towards graph’s center), we choose to identify the core of the graph using the eccentricity centrality measure (see Section 4.2.1).

Even though the center $C(G)$ contains all vertices that are closer to other vertices, this subset is not sufficient to carry out the communication between every pair of vertices ($C(G) \subseteq \text{core}(G)$). More vertices should be added to the core according to their participation in routing the traffic among other vertices. We decide the participation of each vertex $v$ based on its eccentricity and whether or not $v$ lies on a shortest path between a pair of vertices $x$ and $y$ (this notion combines the classic concept of the shortest path betweenness centrality with the eccentricity centrality). We know that a vertex $z$ lies on a shortest path
between a pair of vertices $u$ and $v$ if and only if $d(u, v) = d(u, z) + d(z, v)$.

Obviously, not all graphs exhibit a core-periphery structure. Also, graphs follow this structure with different extents with respect to the quality of their cores. We identify a good graph’s core as the one that (1) includes a small number of layers with respect to the eccentricity layering; and (2) has size (with respect to the number of vertices) that is small compared to the total number of vertices in the graph. The core should also contain vertices that participate in the majority of interactions among other vertices.

In the following two subsections, we discuss two models that exploit the eccentricity-based bending property to partition the vertices in a graph $G$ into two sets: the core $\text{core}(G)$ and the periphery.

**Model I. The Maximum-Peak Model**

Given a $\delta$-hyperbolic graph $G = (V, E)$ along with its eccentricity layering $\mathcal{EL}(G)$, the Maximum-Peak model identifies a separation layer index $p \geq 0$ and defines the core as the subset of vertices formed by layers $\ell_0(G), \ell_1(G), \ldots, \ell_p(G)$.
Figure 4.8: A simplified illustration of the eccentricity layering of a graph (with four layers) and the Maximum-Peak model. $\ell_r(G)$ represents each layer $r$, $0 \leq r \leq 3$. The peaks of shortest paths $\rho(x, y)$ and $\rho(u, v)$ are vertices $w$ and $z$ respectively. The core contains all vertices of the layers of the peaks, i.e., $core(G) = \ell_0(G) \cup \ell_1(G)$.

In light of the eccentricity-based bending property, each $bend(x, y)$ for a pair of vertices $x$ and $y$ represents a peak for shortest paths connecting $x$ and $y$. In this model, we are locating the index of the lowest layer $p$ over all layers that vertex pairs bend to. Index $p$ represents the separation point where the layers of the graph can be partitioned to a core and a periphery. See Figure 4.8 for an illustration.

After identifying all peaks, the core will include all vertices starting at $\ell_0(G)$ (the center) until $\ell_p(G)$. Then the periphery will include the vertices in the remaining layers ($\ell_{p+1}(G)$ to $\ell_m(G)$ where $m$ is the index of the last layer in the eccentricity layering). The core $core(G)$ becomes: $core(G) = \bigcup_{r=0}^{p} \ell_r(G)$.

Again, to avoid the impact that outlier vertices may impose (as discussed in section 4.5.1), we define two types of the separation index $p$: the absolute separation index $p^*$ and the effective separation index $\tilde{p}$. The absolute separation index $p^*$ is the lowest layer that all vertex pairs bend to; we call the core defined by this index the absolute core set $C_{core}^*$. The effective separation index $\tilde{p}$ is the lowest layer where $90\%$ of the vertex pairs bend to, and the core defined by this index is called the effective core set $\tilde{C}_{core}$.

The algorithm used to decide the effective separation index $\tilde{p}$ is presented in Algorithm 2. Table 4.7 shows the cores for the networks in our graph datasets according to the
Algorithm 2. Decide the effective separation index $\tilde{p}$ to partition the graph into a core and a periphery based on the Maximum-Peak model. $\Gamma$ is the total number of vertex pairs with distance $\geq 2$. $m$ is the index of the last layer.

1. $\text{cnt} \leftarrow 0$
2. for $r=0$ to $m$ do
3. for every not counted pair $(x,y)$ with $d(x,y) \geq 2$ do
4. if $(\text{bend}(x,y) \leq r)$ then
5. $\text{cnt}++$ and mark pair $(x,y)$ as counted
6. if ($\text{cnt} \geq 90\%$ of $\Gamma$) then
7. $\tilde{p} \leftarrow r$
8. break and return $\tilde{p}$
9. end if
10. end if
11. end for
12. end for

Maximum-Peak model.

The results in Table 4.7 show (as expected) a big difference in the sizes (with respect to the number of vertices) of the absolute core and the effective core in the majority of the networks in the datasets. This implies that the identification of the absolute separation index was highly affected by a small percent of vertices. Closer analysis to the effective core set $\tilde{C}_{\text{core}}$ suggests that deciding the core according to this notion generates good cores (number of layers in the core is small and the number of vertices is about 25% of the total number of vertices) for some networks such as the YEAST-TRANSCRIPTION, C-ELEGANS-METABOLIC, and YEAST-PI. Also, networks with core sizes between 25% - 50% can be considered good as well; such as the cores of the S-CEREVISIAE-PI and E-COLI-METABOLIC. On the other hand, networks like B-YEAST-PI, E-COLI-PI, MACAQUE-BRAIN-1, MACAQUE-BRAIN-2, and H-PYLORE-PI have too large core sizes compared to the overall graph size. This model is highly affected by the distribution of vertices over the layers (see Figure 4.5). For example, the core of graph B-YEAST-PI has two layers (out of
Table 4.7: The cores of the graph datasets based on the Maximum-Peak model. \(|V|\) is the number of vertices; \(|[Layers]\)| is the number of layers; \(|C_{core-lyr}|^*\) and \(|C_{core}|^*\) are the number of layers and number of vertices in the absolute core set; \(C_{core-lyr}\) and \(|C_{core}|\) are the number of layers and number of vertices in the effective core set.

| Network               | \(|V|\) | \(|[Layers]\)| | \(|C_{core-lyr}|^*\) | \(|C_{core}|^*\) to \(|V|\) | \(|C_{core-lyr}|\) | \(|C_{core}|\) to \(|V|\) |
|-----------------------|--------|----------------|----------------|----------------|----------------|----------------|
| B-YEAST-PI            | 1465   | 4              | 3              | 1448           | ≈ 99\%         | 2              | 902           | ≈ 62\%         |
| E-COLI-PI             | 126    | 3              | 2              | 93             | ≈ 74\%         | 2              | 93            | ≈ 74\%         |
| YEAST-PI              | 1728   | 6              | 5              | 1725           | ≈ 100\%        | 2              | 472           | ≈ 27\%         |
| S-CEREVISIAE-PI       | 537    | 5              | 2              | 537            | ≈ 100\%        | 2              | 223           | ≈ 42\%         |
| H-PYLOHI-PI           | 45     | 3              | 2              | 56             | ≈ 78\%         | 2              | 56            | ≈ 78\%         |
| MACAQUE-BRAIN-1       | 45     | 3              | 2              | 31             | ≈ 69\%         | 2              | 31            | ≈ 69\%         |
| MACAQUE-BRAIN-2       | 350    | 2              | 2              | 350            | 100\%          | 2              | 350           | 100\%          |
| E-COLI-METABOLIC      | 242    | 8              | 7              | 240            | ≈ 99\%         | 3              | 102           | ≈ 42\%         |
| C-ELEGANS-METABOLIC   | 433    | 4              | 3              | 439            | ≈ 97\%         | 1              | 17            | ≈ 4\%          |
| YEAST-TRANSCRIPTION   | 321    | 5              | 4              | 314            | ≈ 98\%         | 2              | 62            | ≈ 19\%         |

We observe that the size of the core varies significantly across the different datasets. For example, the core of the E-coli-PI network consists of only 2 vertices, while the core of the B-Yeast-PI network consists of 902 vertices. This can be due to the different sizes of the datasets and the way the vertices are arranged in the graph. For instance, the E-coli-PI network has a smaller number of vertices compared to the B-Yeast-PI network, which may explain why the core of the E-coli-PI network is smaller.

This can be considered as a balanced core-periphery separation with respect to the number of layers. However, considering the distribution of the vertices in the four layers, which is 90, 902, 465, and 17, explains the increase in the size of the core.

Recall that the core according to this model is defined by the layers and not by the vertices which may result in the addition of some vertices to the core that do not have real contribution (they were added only for their location in the graph’s eccentricity layering). This issue can be resolved by the identification of the core according to the Minimum Cover Set model presented in the following subsection.

Model II. The Minimum Cover Set Model

Consider a graph \(G = (V, E)\) with the eccentricity layering \(\mathcal{E}L(G)\) and with the center \(C(G)\). The way this model works is to start the core set \(\text{core}(G)\) as an empty set and expand it to include vertices which have smaller eccentricity, are closer to the center \(C(G)\), and participate in the traffic between other vertices. This expansion should be orderly, first incorporating the vertices that are more eligible (or have higher priority) to be a part
of the core, and then moving on to vertices who are less eligible. For each vertex $v \in V$, we define the following three parameters according to which we prioritize the vertices in $G$.

- The eccentricity $ecc(v)$. Vertices with smaller eccentricities have higher priority to be in the graph’s core.

- The distance-to-center for a vertex $v$, denoted as $f(v)$, which expresses the distance between $v$ and its closest vertex from the center $C(G)$, i.e., $f(v) = d(v, C(G))$. Note that $f(v) \geq 0$. Vertices with small $f(v)$ are closer to the center; therefore, they have higher priority of being in the core. For example, in Figure 4.8, vertex $y$ is closer to the center than vertex $u$.

- The betweenness $b(v)$. The definition of the betweenness $b(v)$ of $v$ is close to the definition of the classic shortest-path betweenness centrality. It measures how many pairs of distant vertices $x$ and $y$ have $v$ in one of their shortest paths (versus counting all shortest paths in the classic definition (see Section 4.2.1)). It quantifies the participation of a vertex $v$ in the traffic flow process, and we define it as: $b(v) = \text{number of pairs } x, y \in V \text{ with } v \neq x, v \neq y, d(x, y) \geq 2 \text{ and } d(x, v) + d(v, y) = d(x, y)$. According to the core-periphery organization, the betweenness of a vertex should increase as its eccentricity decreases. This means that a vertex that belongs to the central layer of a graph should have a higher value for its $b(v)$ parameter than a vertex that belongs to any other higher layer in the graph’s eccentricity layering.

Our goal in this model is to identify the smallest subset of vertices that participate in all traffic throughout the network. The algorithm for this model comprises two stages. First, in a priority list $T$ we lexicographically sort the vertices according to the three attributes: $ecc(v)$, $f(v)$, and $b(v)$. $T$ now has the vertices in the order that they should be considered to become part of the core. The goal is to ensure that for each pair of vertices $x, y \in V$
Algorithm 3. Decide the core of each graph based on the Minimum Cover Set model. $T$ is a priority list in which vertices are ordered based on: $ecc(v)$, $f(v)$, and $b(v)$. $C^\ast_{core}$ is the absolute core set.

1. $C^\ast_{core} \leftarrow$ first vertex in $T$
2. while (current core does not cover all vertex pairs) do
3.   $v \leftarrow$ next vertex in $T$
4.   if ($v$ covers a new pair) then
5.     $C^\ast_{core} \leftarrow C^\ast_{core} \cup v$
6.   end if
7. end while
8. return $C^\ast_{core}$

there exists at least one vertex $v \in \text{core}(G)$ such that $v \in \rho(x, y)$. In such case, we say that a shortest path $\rho(x, y)$ from $x$ to $y$ is covered by $v$ (a shortest path from $y$ to $x$ is also covered by the same vertex $v$ since we are dealing with undirected graphs).

The second stage starts with a vertex $v$ at the head of $T$ being removed from $T$ and added to an initially empty set $C^\ast_{core}$ that represents the absolute core set. This vertex must cover at least one shortest path between a pair of vertices according to the definition of the betweenness $b(v)$. After this initial step, the process continues by repeatedly removing the vertex $v$ at the head of $T$ and adding it to $C^\ast_{core}$ if and only if $v$ covers a shortest path between an uncovered yet pair $x$ and $y$ (when there is at least one vertex $v \in C^\ast_{core}$ that covers a shortest path between $x$ and $y$, then the pair becomes covered). This step should run until all pairs are covered. Note that we consider the core set $C^\ast_{core}$ as absolute since all vertex pairs must be covered by a vertex in it.

Now the vertices in set $C^\ast_{core}$ represent the core of the graph ($\text{core}(G)$) while the remaining vertices represent the periphery. Algorithm 3 presents the pseudocode, and the number of vertices in the absolute and the effective core sets of each graph of our datasets is listed in Table 4.8.

Close analysis of Table 4.8 shows that each produced absolute core $C^\ast_{core}$ is of a size
Table 4.8: The cores of the graph datasets based on the Minimum Cover Set model. $|V|$ is the number of vertices; $\delta(G)$ is the hyperbolicity; $|C_{core}^*|$ is the number of vertices in the absolute core set; $|\tilde{C}_{core}|$ is the number of vertices in the effective core set; $C_{MaxLyr}^*$ is the largest index layer found among vertices in $C_{core}^*$; and $\tilde{C}_{MaxLyr}$ is the largest index layer found among vertices in $\tilde{C}_{core}$.

| Network             | $|V|$ | $\delta(G)$ | $|C_{core}^*|$ | $|\tilde{C}_{core}|$ to $|V|$ | $C_{MaxLyr}^*$ | $|\tilde{C}_{core}|$ to $|V|$ | $\tilde{C}_{MaxLyr}$ |
|---------------------|------|-------------|----------------|-----------------|----------------|-----------------|----------------|
| B-yeast-PI          | 1465 | 2.5         | 1117           | 76 %            | 3              | 117             | 8 %            | 1              |
| E-coli-PI           | 126  | 2           | 65             | 52 %            | 2              | 13              | 10 %           | 1              |
| Yeast-PI            | 1728 | 3.5         | 902            | 52 %            | 5              | 318             | 18 %           | 2              |
| S-cerevisiae-PI     | 537  | 4           | 438            | 82 %            | 4              | 114             | 21 %           | 1              |
| H-pylori-PI         | 72   | 3           | 54             | 75 %            | 2              | 15              | 21 %           | 1              |
| Macaque-brain-1     | 45   | 1.5         | 20             | 44 %            | 2              | 7               | 16 %           | 1              |
| Macaque-brain-2     | 350  | 1.5         | 197            | 56 %            | 1              | 31              | 9 %            | 0              |
| E-coli-metabolic    | 242  | 4           | 208            | 86 %            | 7              | 66              | 27 %           | 2              |
| C-elegans-metabolic | 453  | 1.5         | 202            | 45 %            | 2              | 12              | 3 %            | 0              |
| Yeast-transcription | 321  | 3           | 155            | 48 %            | 4              | 40              | 12 %           | 1              |

between 44% to 86% of the original number of vertices in the graph. It is important to note that vertices in the core are expected to have different contributions (some vertices cover more vertex pairs than others). Figure 4.9 shows how many vertex pairs are remained uncovered after the orderly addition of vertices to the absolute core. For example, in the network B-YEAST-PI, 80% of vertex pairs are uncovered after adding the first vertex to the absolute core set $C_{core}^*$. However, after adding 20 vertices to $C_{core}^*$, only 35% of the vertex pairs are uncovered. It is also clear from Figure 4.9 that many of the vertices that have been added later to the absolute core set cover a very small percentage of the vertex pairs.

To keep only vertices that are considered higher contributors (cover a large number of vertex pairs) we define the effective core set $\tilde{C}_{core}$. The effective core set is the subset of the core that is sufficient to cover shortest paths between 90% of the vertex pairs in the graph. To obtain $\tilde{C}_{core}$, we examine the vertices of the core $C_{core}^*$ in the same order in which they were added. A new vertex is added to current $\tilde{C}_{core}$ only if more than 10% of the vertex pairs remain uncovered. The results on the core according to both concepts in
Figure 4.9: The percentage of the uncovered vertex pairs after the orderly addition of vertices to the core set $C^*_{core}$. Number $i$ indicates the cardinality of the current core.

this model are presented in Table 4.8. Note that the index of the layer of the last vertex added to the core in each network has significantly decreased.

Because hyperbolic graphs adhere to the property of having shortest paths that bend to the core of the graph, it was natural to think that hyperbolic graphs with lower $\delta(G)$ should have even smaller number of vertices in the core. A quick comparison between the $\bar{C}_{core}$ of each graph with its $\delta(G)$ supports this hypothesis.

4.6 Concluding remarks

The structure of several biological networks has been often described as a chain-like or tree-like topology in molecular biology [13]. This motivates investigating if those networks also admit tree-like structures based on different aspects such as their distances. In Section 4.4.1, we observed that most biological networks appear to have low hyperbolicity suggesting their closeness to a tree structure with respect to their distances [5].

In the tree structure, the communication among distant vertices is carried out through
the center of the tree which is one or two vertices with the smallest eccentricities. The
center in this case represents the core of the network while the rest of the vertices belong
to the periphery. Since strongly-hyperbolic graphs have a structure that is closer to a tree
structure, this motivates the following hypothesis: does hyperbolicity indicate the existence
of a sharper core-periphery dichotomy? In other words, do strongly-hyperbolic graphs have
more concise cores compared to (weak) hyperbolic graphs?

Before we answer this question we will analyze the networks in our datasets. In Section
4.4, we differentiated between two types of hyperbolicity that appear in our graphs. First,
the low hyperbolicity that is caused by a small diameter or graph size; second, the low
hyperbolicity with the value of $hb(G)$ smaller than or equal to the value of $\log_2(diam(G))$
which in turn is at most $\log_2(\log_2(size(G)))$. Whereas both types are considered hyperbolic
graphs, we called the latter category strongly-hyperbolic to point out the similarity in
their structure to a tree despite their sizes and their diameters. Here, we noticed that
graphs of strong-hyperbolicity are actually small-world networks. We are not saying that
the hyperbolicity is intrinsic to small-world networks even though this notion has been
suggested in [155]. For instance, a tree with a large diameter is not small-world but it is
strongly-hyperbolic.

Using the eccentricity-based bending property introduced in Section 4.5.1, we defined
the core vertices of each graph in our datasets. It is clear from Table 4.9 that hyperbolic
networks have larger cores when compared to strongly-hyperbolic networks (which confirms
our hypothesis). Here we only consider cores according to the Minimum Cover Set model to
eliminate any interference that may happen because of the vertex distribution over different
layers of the graph eccentricity layering. The sizes of the cores in strongly-hyperbolic
graphs are less than 20% of the number of vertices of each network. In [174], the authors
showed that the small-world property is not enough for a graph to have the core-periphery
structure. However, it turns out that our graphs that do not exhibit small-world property also have the core-periphery organization but with larger cores. They also have higher values for $hb(G)$ ($hb(G) \geq 3$). Moreover, we see that the more distant the graph is from having the small-world property, (the larger the difference between its $\log_2(size(G))$ and its $diam(G)$), the larger its core is.

We also observed two patterns in strongly-hyperbolic networks named groups 1 and 2 in Table 4.9. The networks in the first group (the C-elegans-metabolic and the B-yeast-PI) have small hyperbolicity ($hb(G) < 3$) and in the same time $hb(G)$ is sufficiently smaller than the value of half the diameter. The cores for those networks are very small (the numbers of vertices in the cores are 3% and 8% of the total number of vertices). The second group has networks that are either with higher hyperbolicity (Yeast-transcription and YEAST-PI), or low hyperbolicity with value of $hb(G)$ very close to $diam(G)/2$. The cores for the networks in group 2 are larger than for those in group 1, yet they are small (9% - 18%).

A future direction would be to analyze the core sets produced by a set of different core identification algorithms and compare those sets with the core sets produced by our

<table>
<thead>
<tr>
<th>Network</th>
<th>$\log_2(size(G))$</th>
<th>$diam(G)$</th>
<th>$\delta(G)$</th>
<th>$\delta'(G)$</th>
<th>$\tilde{C}_{core}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly-hyperbolic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-elegans-metabolic</td>
<td>12.3</td>
<td>7</td>
<td>1.5</td>
<td>0.133</td>
<td>3%</td>
</tr>
<tr>
<td>B-yeast-PI</td>
<td>12.8</td>
<td>8</td>
<td>2.5</td>
<td>0.299</td>
<td>8%</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACAQUE-brain-2</td>
<td>12.4</td>
<td>4</td>
<td>1.5</td>
<td>0.203</td>
<td>9%</td>
</tr>
<tr>
<td>E-coli-PI</td>
<td>9.5</td>
<td>5</td>
<td>2</td>
<td>0.251</td>
<td>10%</td>
</tr>
<tr>
<td>MACAQUE-brain-1</td>
<td>10</td>
<td>9</td>
<td>3</td>
<td>0.365</td>
<td>12%</td>
</tr>
<tr>
<td>YEAST-PI</td>
<td>9</td>
<td>4</td>
<td>1.5</td>
<td>0.231</td>
<td>16%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hyperbolic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-cerevisiae-PI</td>
<td>10.5</td>
<td>11</td>
<td>4</td>
<td>0.419</td>
<td>21%</td>
</tr>
<tr>
<td>H-pylori-PI</td>
<td>7.5</td>
<td>7</td>
<td>3</td>
<td>0.368</td>
<td>21%</td>
</tr>
<tr>
<td>E-coli-metabolic</td>
<td>9.3</td>
<td>16</td>
<td>4</td>
<td>0.483</td>
<td>27%</td>
</tr>
</tbody>
</table>

Table 4.9: Summary of the graph datasets’ parameters and cores: size of the graph $size(G)$; diameter $diam(G)$; hyperbolicity $\delta(G)$; average hyperbolicity $\delta'(G)$; and the $\tilde{C}_{core}$ is the effective core according to the Minimum Cover Set model.
algorithms. Even though different core identification algorithms have different definitions and will (naturally) produce different core sets, it would be interesting to examine the ways in which those algorithms differ.
Chapter 5

Interplay Between $\delta$-Hyperbolicity and the Core-Periphery Structure in Graphs

From the definitions of $\delta$-hyperbolicity, it is clear that finding its exact value is computationally very expensive. This is problematic especially for large networks. The simplest method for computing the $\delta$-hyperbolicity of a graph with $n$ vertices is to use (naively) the four-point condition definition. This straightforward application requires $O(n^4)$ time, and in every step of the algorithm, the distance among a vertex pair need to be retrieved. Thus, it is reasonable to compute the distance matrix of the given network before applying the four-point condition algorithm. The problem here is that the distance matrix of large graphs cannot be stored in memory, and the alternative is to use the adjacency list. This makes computing the distance between every pair of vertices more complex and the overall algorithm more costly. In this chapter, we propose a method that reduces the size of the input graph to only a subset that is responsible for maximizing its hyperbolicity.
We also show that the hyperbolicity of a graph can be found in a set of quadruples that are in close proximity and we empirically show that this set concentrates in the core of the graph. Our observations have crucial implications on computing the $\delta$-hyperbolicity of large graphs. We apply our ideas to a set of real-world and artificial networks, and we show their suitability to compute the $\delta$-hyperbolicity value with only a fraction of the original calculations. This chapter is based on:


5.1 Introduction

Due to their importance, topological properties of complex networks attract a lot of research efforts. The goal is to exploit any hidden properties to increase the efficiency of existing algorithms, as well as to propose new algorithms that are more natural to the structure that a graph exhibits. Topological properties are either global such as the graph’s diameter or local such as the structure of the neighborhood of a vertex. A property that has been investigated recently is the $\delta$-hyperbolicity (negative curvature) of a graph since it has a major impact on its underlying topology [155]. The $\delta$-hyperbolicity (or simply hyperbolicity) measures how close the metric structure of the graph is to the metric structure of the tree [110]. Generally, the smaller the hyperbolicity the closer the graph is to a tree and, as a result, the hyperbolicity property is more evident. Many real-world networks show a tree-like structure with respect to their hyperbolicity [155, 153, 13, 6, 16, 5]. Trees and cliques are 0-hyperbolic, and accordingly are considered hyperbolic graphs. On the other hand, a cycle with $n$ vertices is approximately $n/4$-hyperbolic and an $n \times n$-grid is $(n − 1)$-hyperbolic.

In hyperbolic graphs, it was observed that the traffic heavily concentrates on a small set of vertices (the core of a graph) [155]. The core is defined using multiple measures such as the betweenness centrality, the eccentricity centrality, the closeness centrality, or any combination of these measures.

Global and local properties of a graph can be very different. For example, many networks that do not show a tree-like structure globally (i.e., using global analysis tools such as the $\delta$-hyperbolicity) turned out to exhibit a tree-like structure when they are analyzed locally. This phenomenon was explained by the presence of a core-periphery structure [6, 155].
Let $G = (V,E)$ be a connected graph with distance function $d$ defined as the number of edges on a shortest path between a pair of vertices. Formally, the $\delta$-hyperbolicity can be defined using the four-point condition definition [110]. Given a graph $G = (V,E)$ and four vertices $x$, $y$, $u$, and $v \in V$ with $d(x,y) + d(u,v) \geq d(x,u) + d(y,v) \geq d(x,v) + d(y,u)$, the hyperbolicity of the quadruple $x$, $y$, $u$, $v$ denoted as $hb(x,y,u,v)$ is defined as $hb(x,y,u,v) = (d(x,y) + d(u,v) - (d(x,u) + d(y,v)))/2$ and the $\delta$-hyperbolicity of the graph is $hb(G) = \max_{x,y,u,v \in V} hb(x,y,u,v)$.

Finding the value of the $\delta$-hyperbolicity is computationally very expensive even when distributed computing techniques are employed [7]. From the four-point definition, it is clear that the obvious algorithm requires $O(n^4)$ time, where $n$ is the number of vertices. The limitation of this algorithm is two fold. First, for large networks, this algorithm is impractical and almost un-achievable. Second, calculating the hyperbolicity in dynamic networks, in which vertices constantly join and leave, is costly even for small to medium size networks.

The hyperbolicity of a graph is highly affected by its topology. Any modification of the graph may dramatically change its topology and accordingly its hyperbolicity. For example, consider the removal of one edge in a cycle graph $G$ (with $hb(G) = n/4$). Upon this modification, the new hyperbolicity becomes $hb(G) = 0$. The best known exact algorithm for calculating the hyperbolicity requires $O(n^{3.69})$ time using the (max,min) matrix multiplication [101]. Multiple algorithms were proposed to reduce the size of the input graph. In [70], the authors propose exact and approximation algorithms that restrict the number of considered quadruples to those ones that may maximize the $\delta$-hyperbolicity value. Moreover, they show that the hyperbolicity of a graph equals the maximum hyperbolicity over its bi-connected components.

Here we propose a method that reduces the size of the input graph to only a subset
that is responsible for maximizing its hyperbolicity by analyzing the local dominance relationship between vertices. Furthermore, we show that the hyperbolicity of a graph can be found in a set of quadruples that are in close proximity. In this chapter, we empirically show that this set concentrates in the core of the graph. This suggests that the \( \delta \)-hyperbolicity is, to some extent, a local property. We adopt two core definitions each of which represents a different notion of vertex coreness [140]. The minimum-cover-set core, which can be identified as a transport-based core-periphery structure and the \( k \)-core, which can be identified as a density-based core-periphery structure. Our observations have crucial implications on computing the \( \delta \)-hyperbolicity of large graphs. We apply our ideas to a set of real-world and artificial networks, and we show their suitability to compute the \( \delta \)-hyperbolicity value with only a fraction of the original calculations.

This chapter is organized as follows. Section 5.2 describes the network datasets used in this chapter and presents a summary of their parameters. In Section 5.3, we present two methods that can reduce the number of vertices and quadruples needed to compute the \( \delta \)-hyperbolicity of graphs: the dominance relationship and the \( p \)-\( \delta \)-hyperbolicity. Then in Section 5.4, we show that the \( \delta \)-hyperbolicity in graphs concentrates in their cores. The proposed methods are also applied to two larger datasets in Section 5.5. The conclusions and future work are discussed in Section 5.6.

5.2 Datasets

Throughout this chapter, we analyze a set of real-world networks belonging to various domains. Because computing the hyperbolicity is computationally expensive, we include a set of relatively small-sized networks. We also analyze several synthetic networks with some known structures of roughly same sizes as the real-world networks. All networks are unweighted and we ignore the directions of the edges. We analyze the largest connected
Table 5.1: Graph datasets and their parameters: \(|V|\): number of vertices, \(|E|\): number of edges; \(|V_C|\): number of vertices in the largest component; \(|E_C|\): number of edges in the largest component; \(rad(G)\): graph’s radius; \(diam(G)\): graph’s diameter; \(\bar{d}\): average degree; \(hb(G)\): hyperbolicity; \(\bar{hb}(G)\): average hyperbolicity.

| Network                  | Description                                   | Ref.  | \(|V|\) | \(|E|\) | \(|V_C|\) | \(|E_C|\) | \(rad(G)\) | \(diam(G)\) | Avg. path length | \(\bar{d}\) | \(hb(G)\) | \(\bar{hb}(G)\) |
|-------------------------|-----------------------------------------------|-------|--------|--------|--------|--------|-----------|------------|-----------------|--------|----------|-------------|
| US-Airways              | Transportation network of airlines in the USA | [26]  | 332    | 2126   | 3      | 6      | 2.7       | 12.8       | 1               | 0.14   |           |             |
| Power-Grid              | Topology of the Western States Power Grid of the USA | [206] | 4941   | 6594   | 23     | 46     | \(\approx 19\) | 2.7         | 10              | 1.9    |           |             |
| Email                   | Email interchanges from the university of Rovira i Virgili | [201] | 1133   | 5451   | 5      | 8      | 3.6       | 9.2         | 2               | 0.27   |           |             |
| Dutch-Elite             | Data on the administrative elite in the Netherland | [26]  | 5741   | 5356   | 12     | 22     | \(\approx 8.6\) | 2.4         | 5               | 0.51   |           |             |
| Facebook                | Ego networks of 10 Facebook users             |       | 4015   | 9824   | 4      | 8      | 3.7       | 43.7        | 1.5             | 0.11   |           |             |
| EVA                     | Corporate ownership information network       | [26]  | 8497   | 6726   | 10     | 18     | 7.5       | 21          | 1.5             | 0.21   |           |             |
| AS-Graph-97             | Snapshot of autonomous systems topology of the Internet | [205] | 4855   | 9276   | 6      | 11     | 3.7       | 3.8         | 3               | 0.14   |           |             |
| AS-Graph-99-April       | Snapshot of autonomous systems topology of the Internet | [205] | 5357   | 9824   | 5      | 9      | 3.7       | 3.8         | 3               | 0.14   |           |             |
| AS-Graph-99-July        | Snapshot of autonomous systems topology of the Internet | [205] | 5357   | 9824   | 5      | 9      | 3.7       | 3.8         | 3               | 0.14   |           |             |
| Erdős-Rényi (1.6)      | Erdős–Rényi random graph with probability \(p = 1.6/|V|\) | [26]  | 2500   | 1942   | 18     | 35     | 14.26     | 2.1          | 8               | 1.06   |           |             |
| Erdős-Rényi (2)        | Erdős–Rényi random graph with probability \(p = 2/|V|\) | [26]  | 2500   | 2458   | 14     | 25     | 10.1     | 2.4          | 6.5             | 0.81   |           |             |
| Erdős-Rényi (8)        | Erdős–Rényi random graph with probability \(p = 8/|V|\) | [26]  | 2500   | 10028  | 7      | 8      | \(\approx 4\) | \(\approx 8\) | 3               | 0.33   |           |             |
| Power-Law (2.7)        | Power Law graph with power parameter \(\beta = 2.7\) | [203] | 2500   | 2014   | 18     | 35     | 8.5       | 2.2          | 4.5             | 0.49   |           |             |
| Power-Law (2)          | Power Law graph with power parameter \(\beta = 2\) | [203] | 2500   | 2484   | 14     | 21     | 7.4       | 2.3          | 4               | 0.47   |           |             |
| Power-Law (1.9)        | Power Law graph with power parameter \(\beta = 1.9\) | [203] | 2500   | 3000   | 7      | 8      | 13        | 3.9          | 3               | 0.29   |           |             |
| Power-Law (1.8)        | Power Law graph with power parameter \(\beta = 1.8\) | [203] | 2500   | 4200   | 6      | 11     | 4.5      | 3.7          | 2.5             | 0.27   |           |             |
| Planar-Grid(50 × 50)   | Two dimensional planar grid                   |       | 2500   | 4900   | 50     | 98     | 33.3      | 3.9          | 49              | 3.7    |           |             |
| Planar-Grid(1250 × 2)  | Two dimensional planar grid                   |       | 2500   | 3748   | 626    | 1250    | 417.3     | \(\approx 3\) | 1               | 0.25   |           |             |

In each graph, a vertex represents an autonomous system, and component of each network. See Table 5.1 for a summary.

**Social networks.** We have examined the following four social networks: The Email network [201] that represents the email interchanges between members of the university of Rovira i Virgili, Tarragona. The DUTCH-ELITE network [26] which is a network data on the administrative elite in the Netherland. In the DUTCH-ELITE network, vertices represent persons and organizations that are most important to the Dutch government (2-mode network). An edge connects two vertices if the person vertex belongs to the organization vertex. The Facebook network [204] represents the ego networks (the network of friendship between a user’s friends) of 10 people. Two vertices (users) are connected if they are Facebook friends. The EVA network [26] presents corporate ownership information as a social network. Two vertices are connected with an edge if one is the owner of the other.

**Internet networks.** Each of those graphs represents the Autonomous Systems (AS) topology of the Internet. In each graph, a vertex represents an autonomous system, and
two vertices are connected if the two autonomous systems share at least one physical connection. In this chapter, we examine three AS graphs: AS-Graph-97, AS-Graph-99-April, and AS-Graph-99-July [205] for which the data was collected during November 1997, April 1999, and July 1999 respectively.

**Erdős Rényi random graphs.** In an Erdős Rényi graph with \( n \) vertices, denoted by Erdős-Rényi(\( p \)), every two vertices are independently connected with a fixed probability \( p \). Smaller values for \( p \) (\( 1/n < p < \log(n)/n \)) result in very sparse graphs. In contrast, larger \( p \) values yield dense graphs with very small diameters. Sparser Erdős Rényi graphs exhibit a clear core-periphery structure compared to dense Erdős Rényi graphs [6].

Since we are looking for graphs with large diameters to clearly see the potential of our method in calculating the hyperbolicity of a graph, we choose very small values for \( p \). In our datasets, we include three Erdős Rényi graphs with equal number of vertices \( (n = 2500) \) and with \( p \) of \( 1.6/n \), \( 2/n \), and \( 8/n \) respectively.

**Power-law random graphs.** In a power-law graph, the degrees of the vertices follow (or approximate) a power-law distribution. Here we use a set of power-law graphs generated based on a variation of the Aiello-Chung-Lu model [9, 38]. This model produces a power-law random graph whose degree sequence is determined by a power-law with exponent \( \beta \), where \( \beta \) is the power parameter. Smaller \( \beta \) values (\( \beta < 2 \)) generate power-law graphs with cores that are denser and have smaller diameters compared to power-law graphs with higher \( \beta \) values [144].

Each power-law graph in the network datasets Power-Law(\( \beta \)) has 2500 vertices and a value \( \beta \in \{1.8, 1.9, 2, 2.7\} \).

Finally, we analyze multiple graphs that are expected to have different hyperbolic properties: the US-Airways transportation network [26] and the Power-Grid network [206], which represents the western United States power grid. Also we analyze two planar grid
graphs: Planar-Grid(50 × 50) and Planar-Grid(1250 × 2).

In Table 5.1, most real-world and artificial networks have small $\delta$-hyperbolicity values. Note that the absolute value of the $\delta$-hyperbolicity becomes meaningful when it is compared with other parameters of the graph such as its diameter [16]. Recall that half the diameter represents an upper bound for the $\delta$-hyperbolicity.

5.3 $\delta$-Hyperbolicity in graphs

According to the definition of the $\delta$-hyperbolicity of a quadruple, its value is not dependent on the distances among the vertex pairs; rather, it is affected by the topology present among the vertices. Even though the set of quadruples responsible for maximizing the value of the hyperbolicity has not been characterized, in this section, we present methods that can be used to eliminate vertices (and accordingly quadruples) that do not actively participate in increasing the $\delta$-hyperbolicity of a graph [18].

5.3.1 $\delta$-Hyperbolicity and dominated vertices

There are a few existing methods that aim at reducing the size of the graph without affecting its hyperbolicity. Some of those methods are suggested by the following lemmas.

**Lemma 4 ([95]).** Given a graph $G = (V, E)$ and a vertex $x \in V$ with $\text{degree}(x) = 1$, $\text{hb}(G) = \text{hb}(G - \{x\})$.

**Lemma 5.** Let $G = (V, E)$ be a graph, $x, y, w$ be a triangle in $G$, and let $x$ be a vertex with $\text{degree}(x) = 2$. Then $\text{hb}(G) = \text{hb}(G - \{x\})$.

**Proof.** The proof is formally analogous to the proof of Lemma 8 in [70].

These cases can be generalized using the dominance relationship among vertices.

89
Definition 8. Given a graph $G = (V, E)$ and a vertex $x \in V$, $x$ is said to be dominated by a neighboring vertex $y$ if $N(x) \subseteq N(y)$.

Note that a vertex with degree 1 is also dominated by its only neighbor. See Figure 5.1.

Lemma 6 ([54]). Let $x \in V$ be a dominated vertex in a graph $G = (V, E)$. The subgraph $G_{V - x}$ is isometric.

Proof. Let $G = (V, E)$ be a graph and let $x \in V$ be a vertex that is dominated by a neighboring vertex $y$. Consider a shortest path $\rho(u, v)$ between a pair of vertices $u$ and $v$ such that $x \in \rho(u, v)$. I.e., $d(u, v) = d(u, x) + d(x, v)$. Let $x' \in N(x)$ be the vertex closest to $u$, then $d(u, x) = d(u, x') + 1$. Since $N(x) \subseteq N(y)$, then $d(u, y) = d(u, x') + 1 = d(u, x)$. Similarly, $d(y, v) = d(x, v)$. Therefore, $d(u, v) = d(u, y) + d(y, v)$ for any pair $u$ and $v$. This shows that the distance $d(u, v)$ is not affected by the removal of $x$. I.e., $d(u, v)$ in $G$ equals that in $G_{V - x}$. ■

Next we analyze the effect of removing dominated vertices on the upper bound of the hyperbolicity (the diameter of the graph) and the value of the hyperbolicity.

Lemma 7. Let $G = (V, E)$ be a graph, and let $x \in V$ be a vertex dominated by a neighbor vertex $y$. Then either $diam(G_{V - x}) = diam(G)$ or $diam(G_{V - x}) = diam(G) - 1$
Table 5.2: Statistics of dominated vertices. $S$: set of dominated vertices; $S_{deg=1}$ and $S_{deg=2}$: % of dominated vertices with degrees one and two respectively.

| Network            | $|S|/|V|$ (%) | $S_{deg=1}$ | $S_{deg=2}$ | $hb(G)$ | $hb(G - S)$ | % dec in no. of quad |
|--------------------|--------------|-------------|-------------|---------|-------------|----------------------|
| US-Airways         | ≈ 78.3       | 16.6        | 11.4        | 1       | 1           | 99.8                 |
| Power-Grid         | 30.1         | 24.8        | 3.7         | 10      | 10          | ≈ 76                 |
| Email              | 20.5         | 13.3        | 3.3         | 2       | 2           | 60.1                 |
| Dutch-Elite        | 63.4         | 63.4        | 0           | 5       | 5           | 98.2                 |
| Facebook           | 97.4         | 1.9         | 2.4         | 1.5     | 1.5         | 99.9                 |
| EVA                | 87.8         | 86.8        | 0.9         | 3.5     | 3.5         | 99.9                 |
| AS-Graph-97        | 63.3         | 47.8        | 12.7        | 2       | 2           | 98.2                 |
| AS-Graph-99-April  | 58.4         | 38.1        | 16.5        | 3       | 3           | 97.3                 |
| AS-Graph-99-July   | 57.4         | 36.5        | 16.9        | 2       | 2           | 97                   |
| Erdős-Rényi(1.6)   | 33.9         | 33.9        | 0           | 8       | 8           | ≈ 81                 |
| Erdős-Rényi(2)     | ≈ 27         | 26.9        | 0.05        | 6.5     | 6.5         | 71.6                 |
| Erdős-Rényi(8)     | 0.3          | 0.3         | 0           | 3       | 3           | 1.28                 |
| Power-Law-h(2.7)   | 52.5         | 52.5        | 0           | 4.5     | 4.5         | ≈ 95                 |
| Power-Law-h(2)     | 50.8         | 50.5        | 0.2         | 4       | 4           | 94.1                 |
| Power-Law(1.9)     | 53.3         | 52.9        | 0.4         | 3       | 3           | 99.9                 |
| Power-Law(1.8)     | 51.3         | 50.6        | 0.6         | 2.5     | 2.5         | 99.9                 |
| Planar-Grid(50×50) | 0            | 0           | 0           | 49      | 49          | 0                    |
| Planar-Grid(1250×2)| 0            | 0           | 0           | 1       | 1           | 0                    |

Proof. Let $G = (V, E)$ be a graph and let $x \in V$ be a vertex that is dominated by a neighboring vertex $y$. If $x$ is not a part of any diametral pair, then $diam(G_{V-x}) = diam(G)$. Now assume that the pair $(x, x')$ is a diametral pair in $G$, i.e., $d(x, x') = diam(G)$. Let $\rho(x, x') = x_1, x_2, ..., x_k$ where $x_1 = x$, $x_k = x'$, and $k = d(x, x') + 1$. $x_2 \in N(x)$ and $d(x_2, x') = d(x, x') - 1 = diam(G) - 1$.

Lemma 8. Let $G = (V, E)$ be a graph, and let $x \in V$ be a vertex dominated by a neighbor vertex $y$. Then $hb(G) \leq \max\{1, hb(G_{V-x}) + \frac{1}{2}\}$.

Proof. Let $x$ and $y$ be two vertices defined as above and let $G_X$ be the subgraph induced by the set $X = \{x\} \cup N(x)$. Consider a vertex $z \in G_X$, and three vertices $u, v, w \notin G_X$. We show that $hb(G) \leq \max\{1, hb(G_{V-x}) + \frac{1}{2}\}$ holds for any quadruple that involves vertex $x$. We consider the cases when all the other three vertices in a quadruple belong to $G_X$, when all the other three vertices do not belong to $G_X$, when a quadruple consists of $x, y$, and any two vertices $\notin G_X$, and when a quadruple consists of $x, y, a$ vertex in $G_X$, and a
First, $\text{hb}(G_X) \leq 1$ since $\text{diam}(G_X) \leq 2$ (Lemma 1).

Second, $\text{hb}(x, u, v, w) \leq \text{hb}(y, u, v, w) + \frac{1}{2}$ for any three vertices $u, v, w \notin G_X$. Assume $2\text{hb}(y, u, v, w) = d(y, u) + d(v, w) - d(y, v) - d(u, w)$. Let $A = d(x, u) + d(v, w)$, $B = d(x, v) + d(u, w)$, and $C = d(x, w) + d(u, v)$. When $A \geq B \geq C$, we have $2\text{hb}(x, u, v, w) = d(x, u) + d(v, w) - d(x, v) - d(u, w)$. Since $d(y, u) \leq d(x, u) \leq d(y, u) + 1$ and $d(y, v) \leq d(x, v) \leq d(y, v) + 1$, then $\text{hb}(x, u, v, w) \leq (d(y, u) + 1 + d(v, w) - d(y, v) - d(u, w))/2 \leq \text{hb}(y, u, v, w) + \frac{1}{2}$. When $B \geq A \geq C$, $2\text{hb}(x, u, v, w) = d(x, v) + d(u, w) - d(x, u) - d(v, w)$. Also $d(y, u) \leq d(x, u) \leq d(y, u) + 1$ and $d(y, v) \leq d(x, v) \leq d(y, v) + 1$, and by triangle inequality, $d(u, w) \leq d(y, u) + d(y, w)$ and $d(v, w) \leq d(y, v) + d(y, w)$. Then $2\text{hb}(x, u, v, w) = d(y, v) + 1 + d(y, u) + d(y, w) - d(y, u) - d(y, v) - d(y, w)$, and we get $\text{hb}(x, u, v, w) \leq \frac{1}{2}$.

Finally, when $C \geq A \geq B$, $2\text{hb}(x, u, v, w) = d(x, w) + d(u, v) - d(x, u) - d(v, w)$. By triangle inequality, we get $\text{hb}(x, u, v, w) \geq (1 + d(v, w) - d(v, w))/2 = \frac{1}{2}$.

Third, $\text{hb}(x, y, u, v) \leq \frac{1}{2}$ for any two vertices $u, v \notin G_X$. Consider the following three distance sums for the quadruple $(x, y, u, v)$: $A = d(x, y) + d(u, v)$, $B = d(x, u) + d(y, v)$, and $C = d(x, v) + d(y, u)$. When $A \geq B \geq C$, we have $2\text{hb}(x, y, u, v) = d(x, y) + d(u, v) - d(x, u) - d(y, v) \leq 1 + d(y, u) + d(y, v) - d(y, u) - d(y, v)$ since $d(u, v) \leq d(y, u) + d(y, v)$. Therefore, $\text{hb}(x, y, u, v) \leq \frac{1}{2}$. When $B \geq A \geq C$, we have $2\text{hb}(x, y, u, v) = d(x, u) + d(y, v) - d(x, y) - d(u, v) \leq 1 + d(y, u) + d(y, v) - 1 - d(u, v) = 0$. Finally, when $C \geq A \geq B$, we have $2\text{hb}(x, y, u, v) = d(x, v) + d(y, u) - d(x, y) - d(u, v) \leq 1 + d(y, v) + d(y, u) - 1 - d(u, v) = 0$.

Fourth, we obtain similarly that $\text{hb}(x, y, z, u) \leq \frac{1}{2}$ for any vertex $z \in G_X$ and any vertex $u \notin G_X$.

To be able to obtain all quadruples responsible for maximizing the hyperbolicity we do not consider cases in which vertices become dominated after other vertices have been removed. For example, in Figure 5.1, vertex $v$ is dominated by vertex $u$, which is not
dominated by any other vertex. The hyperbolicity of the original graph $G$ is one, and the hyperbolicity of the graph $G_{V-v}$ is also one. However, after removing vertex $v$, vertices $u$, $x$, and $y$ become dominated by vertex $w$, and the hyperbolicity of $G_{V-\{u,x,y\}}$ is zero.

For each graph in the datasets, we report the percent of the dominated vertices. We also differentiate between dominated vertices of degree 1, degree 2, and degree $>2$. The results are listed in Table 5.2. In almost all networks, the dominated vertices have degrees at most two. This suggests that finding those vertices is computationally easier than what is implied by Definition 8. Also, in all networks, the hyperbolicity was preserved after removing all dominated vertices. This result is even better than what is suggested in Lemma 8.

---

Table 5.3: $p$-$\delta$-Hyperbolicity. $p_{\text{max}}$ is the maximum distance $p$ that achieved $hb_p(G) = hb(G)$, % dec is compared to the total number used to compute $hb(G)$.

<table>
<thead>
<tr>
<th>Network</th>
<th>rad($G$)</th>
<th>diam($G$)</th>
<th>$hb(G)$</th>
<th>$p = \lfloor \frac{\text{rad}(G)}{2} \rfloor$</th>
<th>$p = \frac{\text{rad}(G)}{2}$</th>
<th>$%$ dec</th>
<th>$p = p_{\text{max}}$</th>
<th>$p$ $hb_p(G)$</th>
<th>$%$ dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>US-Airways</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>88.8</td>
<td>2</td>
<td>1</td>
<td>88.8</td>
</tr>
<tr>
<td>Power-Grid</td>
<td>23</td>
<td>46</td>
<td>10</td>
<td>12</td>
<td>6</td>
<td>99.1</td>
<td>21</td>
<td>10</td>
<td>67.3</td>
</tr>
<tr>
<td>Email</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>1.5</td>
<td>76.9</td>
<td>4</td>
<td>2</td>
<td>20.1</td>
</tr>
<tr>
<td>Dutch-Elite</td>
<td>12</td>
<td>22</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>95.5</td>
<td>10</td>
<td>5</td>
<td>24.4</td>
</tr>
<tr>
<td>Facebook</td>
<td>4</td>
<td>8</td>
<td>1.5</td>
<td>2</td>
<td>1</td>
<td>99.1</td>
<td>3</td>
<td>1.5</td>
<td>87.9</td>
</tr>
<tr>
<td>EVA</td>
<td>10</td>
<td>18</td>
<td>3.5</td>
<td>5</td>
<td>2.5</td>
<td>$\approx$ 95.7</td>
<td>7</td>
<td>3.5</td>
<td>64.3</td>
</tr>
<tr>
<td>AS-Graph-97</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>1.5</td>
<td>$\approx$ 80</td>
<td>4</td>
<td>2</td>
<td>33.8</td>
</tr>
<tr>
<td>AS-Graph-99-April</td>
<td>6</td>
<td>11</td>
<td>3</td>
<td>3</td>
<td>1.5</td>
<td>$\approx$ 75</td>
<td>6</td>
<td>3</td>
<td>30.3</td>
</tr>
<tr>
<td>AS-Graph-99-July</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>1.5</td>
<td>76.3</td>
<td>4</td>
<td>$\approx$ 32</td>
<td></td>
</tr>
<tr>
<td>Erdős-Rényi(1.6)</td>
<td>18</td>
<td>35</td>
<td>8</td>
<td>9</td>
<td>4.5</td>
<td>99.5</td>
<td>16</td>
<td>8</td>
<td>55.4</td>
</tr>
<tr>
<td>Erdős-Rényi(2)</td>
<td>14</td>
<td>25</td>
<td>6.5</td>
<td>7</td>
<td>3.5</td>
<td>99.1</td>
<td>13</td>
<td>6.5</td>
<td>21.3</td>
</tr>
<tr>
<td>Erdős-Rényi(8)</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>1.5</td>
<td>98.9</td>
<td>6</td>
<td>3</td>
<td>$\approx$ 0.1</td>
</tr>
<tr>
<td>Power-Law(2.7)</td>
<td>12</td>
<td>23</td>
<td>4.5</td>
<td>6</td>
<td>3</td>
<td>95.1</td>
<td>9</td>
<td>4.5</td>
<td>53.1</td>
</tr>
<tr>
<td>Power-Law(2)</td>
<td>11</td>
<td>21</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>87.2</td>
<td>8</td>
<td>4</td>
<td>43.9</td>
</tr>
<tr>
<td>Power-Law(1.9)</td>
<td>7</td>
<td>13</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>78.2</td>
<td>6</td>
<td>3</td>
<td>11.8</td>
</tr>
<tr>
<td>Power-Law(1.8)</td>
<td>6</td>
<td>11</td>
<td>2.5</td>
<td>3</td>
<td>1.5</td>
<td>95.6</td>
<td>5</td>
<td>2.5</td>
<td>21.5</td>
</tr>
<tr>
<td>Planar-Grid(50×50)</td>
<td>50</td>
<td>98</td>
<td>49</td>
<td>25</td>
<td>12</td>
<td>94.7</td>
<td>98</td>
<td>49</td>
<td>0</td>
</tr>
<tr>
<td>Planar-Grid(1250×2)</td>
<td>626</td>
<td>1250</td>
<td>1</td>
<td>313</td>
<td>1</td>
<td>99.9</td>
<td>2</td>
<td>1</td>
<td>99.9</td>
</tr>
</tbody>
</table>

93
5.3.2 $\delta$-Hyperbolicity and restricted path lengths

Hyperbolicity in some sense is related to the uniqueness of shortest paths. In trees, which are $0$-hyperbolic, there is a single shortest path among every vertex pair. While this property is mostly absent in general graphs, the core-periphery property, which has been recognized in many networks, suggests that when two vertices are relatively far from one another (with respect to their distance), all shortest paths that connect them pass the core of the graph. Let $x$ and $y$ be two vertices that are sufficiently far from one another. To some extent, a shortest path between them can be considered unique even though multiple shortest paths may exist between any pair of intermediate vertices $u, v \in I(x, y)$. Applying this idea on sufficiently far vertices in a quadruple, we observe the following (see Figure 5.2).

**Lemma 9.** Let $G = (V, E)$ be a graph and $x, y, u, v \in V$ be four distinct vertices. Consider four vertices $x', y', u', v'$ such that $x \in I(x', y) \cap I(x', u) \cap I(x', v)$, $y \in I(y', x) \cap I(y', u) \cap I(y', v)$, $u \in I(u', x) \cap I(u', y) \cap I(u', v)$, and $v \in I(v', x) \cap I(v', y) \cap I(v', u)$. Then we have $\text{hb}(x', y', u', v') = \text{hb}(x, y, u, v)$.

**Proof.** Assume that $2\text{hb}(x, y, u, v) = d(x, y) + d(u, v) - (d(x, u) + d(y, v))$. Accordingly, $2\text{hb}(x', y', u', v') = d(x', y') + d(u', v') - (d(x', u') + d(y', v'))$. By the assumption above, we obtain $2\text{hb}(x', y', u', v') = d(x, y) + d(x, x') + d(y, y') + d(u, v) + d(u, u') + d(v, v') - d(x, u) - d(x, x') - d(u, u') - d(y, y') - d(v, v') = 2\text{hb}(x, y, u, v)$.

**Remark 3.** Using Lemma 9, we conclude that $\text{hb}(x', y, u, v) = \text{hb}(x, y, u, v)$, $\text{hb}(x, y', u, v) = \text{hb}(x, y, u, v)$, and $\text{hb}(x, y, u', v) = \text{hb}(x, y, u, v)$.

From the lemma and the remark above, it follows that the $\delta$-hyperbolicity of a quadruple may be increased only because some intermediate quadruple has a higher $\delta$-hyperbolicity (this was also observed experimentally in [6]). Accordingly, the $\delta$-hyperbolicity of some
graphs (especially the ones with clear core-periphery dichotomy) may be found in quadruples that are in close proximity, and it is sufficient to consider those quadruples when computing the graph’s hyperbolicity. Thus, we consider a variation of the definition of the $\delta$-hyperbolicity that restricts the set of considered quadruples to those that are in close proximity.

**Definition 9.** Let $G = (V, E)$ be an undirected and unweighted graph, $\text{diam}(G)$ be its diameter, and $x, y, u, v$ be vertices in $V$ with $d(x, y) \leq p$, $d(x, u) \leq p$, and $d(x, v) \leq p$, where $0 \leq p \leq \text{diam}(G)$. Also let $d(x, y) + d(u, v) \geq d(x, u) + d(y, v) \geq d(x, v) + d(y, u)$ be the three distance sums defined over the four vertices $x, y, u, v$ in a non-increasing order. The $p$-$\delta$-hyperbolicity of the quadruple $x, y, u, v$ denoted as $hb_p(x, y, u, v)$ is defined as

$$hb_p(x, y, u, v) = (d(x, y) + d(u, v) - (d(x, u) + d(y, v))) / 2.$$ 

and the $p$-$\delta$-hyperbolicity of the graph is

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

The choice of distance $p$ is critical. When $p = 0$, $hb_p(G) = 0$ since we get a set of singletons. This value can be very far from the value of the hyperbolicity of the graph. When $p = \text{diam}(G)$, $hb_p(G) = hb(G)$ since we include all possible quadruples. Generally,
when $0 < p < \text{diam}(G)$, $h_{bp}(G) \leq h_b(G)$. For some graph types such as an $n \times n$ grid, the value of the hyperbolicity equals the hyperbolicity of the quadruple with vertices at maximum pair-wise distance. Thus restricting the distances among vertex pairs to any $p < \text{diam}(G)$ results in $h_{bp}(G) < h_b(G)$. In contrast, in a $2 \times n$ grid, $h_{bp}(G) = h_b(G)$ when $p = 2$. Examples of both cases are provided in the datasets.

Table 5.3 and Figure 5.3 show the $p$-$\delta$-hyperbolicity of each graph in the datasets. The table lists $h_{bp}(G)$ for $p = \lceil \text{rad}(G)/2 \rceil$ and $p = p_{\text{max}}$, where $p_{\text{max}}$ is the maximum distance $p$ that achieved $h_{bp}(G) = h_b(G)$. Table 5.3 also shows the decrease in the number of quadruples (compared to the total number of quadruples used to compute $h_b(G)$). In almost all graphs, not only $p_{\text{max}}$ is smaller than the diameter of each network but also $p_{\text{max}} \leq \text{rad}(G)$. The distance $p_{\text{max}}$ needed in the network Erdős-Rényi(8) is $6 = \text{rad}(G) + 1$. This is probably due to the lack of a core in this type of graphs (denser random Erdős Rényi graphs) [6]. It is also interesting to observe that $p_{\text{max}} = 2\delta$ in almost all networks. Figure 5.3 shows that the hyperbolicity increases with distance until a certain point ($p_{\text{max}}$) and then remains the same.

To exploit the $p$-$\delta$-hyperbolicity, it is sufficient to consider quadruples within the graphs core, which may not be unique. In [16], it was observed that the shortest path (or paths) between distant vertices tends to include vertices in the center of the graph $C(G)$ ($C(G) = u \in V : \text{ecc}(u) = \text{rad}(G)$).

**Proposition 7** ([16]). Let $G$ be a $\delta$-hyperbolic graph and $x, y$ be arbitrary vertices of $G$. If $d(x, y) > 4h_b(G) + 1$, then on any shortest $(x, y)$-path there is a vertex $w$ with $\text{ecc}(w) < \max \{\text{ecc}(x), \text{ecc}(y)\}$.

Even though a distance of $4h_b(G) + 1$ may exceed the diameter of the graph in most networks (because of the small-world property), it was shown experimentally in [16] that even pairs with small distances include a vertex in the center (or close to the center).
Here we compute the $p$-$\delta$-hyperbolicity considering only vertices within the center for some of our networks and with a distance $p$ that is equal to $p_{\text{max}}$ (see Table 5.3). The results are listed in Table 5.4. The table shows that even though for some networks the $p$-$\delta$-hyperbolicity is not equal to the hyperbolicity of the graph $hb(G)$, it achieves a value that is very close.

5.4 $\delta$-Hyperbolicity and the core-periphery structure

It was observed in [6] throughout a set of real-world and artificial networks that a treelike structure becomes less evident below a certain size scale; specifically, within the core

Figure 5.3: $p$-$\delta$-Hyperbolicity. (a) Real-world networks. (b) Artificial networks.
of the network. I.e., quadruples whose vertices belong to the core part of the network have high hyperbolicity values while quadruples with vertices that belong to the peripheral part do not actively participate in increasing the hyperbolicity value (they affect $hb_{avg}(G)$ but not $hb(G)$). This confirms that quadruples like the ones described in Lemma 9 and Remark 3 exist in many networks due to the core-periphery structure in those networks.

In this section, we exploit this observation for computing the value of the $\delta$-hyperbolicity by considering only quadruples in the core of a graph.

Recently, two core-periphery structure notions have been discussed in the literature. The transport-based core-periphery structure which was developed based on intuition from transportation networks and the density-based core-periphery structure which was developed based on intuition from social networks [140]. A transport-based core is central to the network (in terms of its betweenness), while a density-based core is densely connected and connected to a sparse periphery.

In this section, we use two core definitions: the minimum-cover-set core, which can be classified as a transport-based core and the $k$-core, which can be classified as a density-based core.

Let $core$ be the set of core vertices in a graph $G$. The core of $G$, denoted by $G_{core}$,
Table 5.5: $\delta$-Hyperbolicity and the graph’s minimum-cover-set core $G_{core}^m$. The percent in parentheses represents the percent of vertices in the minimum-cover-set core subgraph to the total number of vertices in the original graph. $|V_G|$: number of vertices in the original network; $|E_G|$: number of edges in the original network; $diam(G)$: diameter of the original network; $hb(G)$: $\delta$-hyperbolicity of the original network. $|V_{G_{core}^m}|$ and $|E_{G_{core}^m}|$ are number of vertices and edges in the minimum-cover-set core subgraph respectively. $diam(G_{core}^m)$: diameter of the minimum-cover-set core; $hb(G_{core}^m)$: $\delta$-hyperbolicity of the minimum-cover-set core.

| Network         | $|V_G|$ | $|E_G|$ | $diam(G)$ | $hb(G)$ | Time (in sec.) | $|V_{G_{core}^m}|$ | $|E_{G_{core}^m}|$ | $diam(G_{core}^m)$ | $hb(G_{core}^m)$ | Time (in sec.) | % dec in no. of quad | % dec in time |
|-----------------|--------|--------|-----------|---------|---------------|-----------------|-----------------|-------------------|-----------------|---------------|---------------------|---------------|
| US-Airways      | 332    | 2126   | 6         | 1       | 4.59          | 58 (17%)        | 604             | 4                 | 1               | 0.03          | 99.9                | 99.4           |
| PowerGrid       | 4941   | 6594   | 46        | 10      | 110220.18     | 3330 (67%)      | 4567            | 44                | 10              | 53119.17      | 99.9                | 51.8           |
| Email           | 1133   | 5451   | 8         | 2       | 920.22        | 821 (73%)       | 4768            | 6                 | 2               | 204.93        | 95.9                | 72             |
| DutchElite      | 3621   | 4310   | 22        | 5       | 45462.10      | 1271 (35%)      | 1907            | 20                | 5               | 1190.46       | 98.5                | 97.4           |
| Facebook        | 4039   | 82344  | 8         | 1.5     | 63563.08      | 102 (37%)       | 1060            | 6                 | 1.5             | 0.07          | 99.9                | 99.9           |
| EVA             | 4755   | 4661   | 18        | 3.5     | 115221.53     | 527 (12%)       | 643             | 16                | 3.5             | 28.80         | 99.9                | 99.9           |
| AS-Graph-97     | 3015   | 5156   | 9         | 2       | 26955.18      | 710 (24%)       | 1817            | 7                 | 2               | 96.52         | 99.7                | 99.6           |
| AS-Graph-99-April | 4885  | 5276   | 11        | 3       | 210220.88     | 1154 (23%)      | 3195            | 9                 | 3               | 696.18        | 99.7                | 99.7           |
| AS-Graph-99-July | 5357  | 10328  | 9         | 2       | 216527.41     | 1125 (25%)      | 3720            | 8                 | 2               | 1014.96       | 99.6                | 99.5           |
| Erdős Rényi(1.6)| 1582   | 1688   | 35        | 8       | 3071.82       | 885 (56%)       | 940             | 34                | 8               | 300.95        | 90.2                | 90.2           |
| Erdős Rényi(2)  | 1993   | 2418   | 25        | 6.5     | 7688.29       | 1375 (69%)      | 1737            | 23                | 7               | 1761.18       | 77.4                | 77.6           |
| Erdős Rényi(3)  | 2498   | 10028  | 7         | 3       | 19670.19      | 2490 (99%)      | 10018           | 7                 | 3               | 17092.13      | 1.3                 | 5.7            |
| Power-Law(2.7)  | 1189   | 1304   | 23        | 4.5     | 995.20        | 569 (48%)       | 672             | 21                | 4.5             | 65.39         | ≈ 95                 | 93.4           |
| Power-Law(2)    | 1761   | 2042   | 21        | 4       | 4623.33       | 854 (48%)       | 1112            | 19                | 4               | 310.26        | 94.5                | 93.3           |
| Power-Law(1.9)  | 2122   | 3400   | 13        | 3       | 33160.93      | 959 (45%)       | 2194            | 11                | 3               | 484           | 95.8                | 98.5           |
| Power-Law(1.8)  | 2236   | 4122   | 11        | 2.5     | 11253.92      | 1026 (46%)      | 2821            | 9                 | 2.5             | 634.80        | 95.6                | 94.4           |

is the subgraph of $G$ induced by the set $core$. We denote the minimum-cover-set core by $G_{core}^m$ and the $k$-core by $G_{core}^k$. We compute the hyperbolicity of the core of each network in the datasets and compare it to the hyperbolicity of the graph. Note that we exclude the two planar grid networks from the analysis in this section because of their lack of a meaningful core.

5.4.1 The minimum-cover-set core

In Section 4.5.2, we show that the traffic tends to concentrate on vertices with small eccentricity (vertices in and close to the graph’s center). Accordingly, we introduce a core identification model (named the minimum-cover-set model) based on the eccentricity and the betweenness of vertices [16].

99
The minimum-cover-set core of a graph $G$, denoted by $G_{\text{core}}^m$, is the smallest set of vertices that is sufficient to circulate the traffic between distant vertices in a graph [16]. This set includes vertices that have small eccentricities, are close to the graph’s center, and have high betweenness. The betweenness of a vertex $x$ is the number of vertex pairs that have $x$ on a shortest path between them. First, in a priority list $T$, vertices are ranked according to three parameters: the eccentricity, the distance to the center of the graph, and the betweenness. Second, a vertex at the top of $T$ will be added to the core if it is in a shortest path between some vertex pair $x, y$. In this case, pair $x, y$ is covered by the core and will not be considered again.

Table 5.5 lists basic statistics about the minimum-cover-set core for each network in the datasets. Table 5.5 shows that in most real-world networks, the core size (number of vertices) does not exceed 35% of the number of vertices in the original graph. The only exception is networks Email and PowerGrid, which is not expected to present a concise core. In the three Erdős Rényi graphs, the conciseness of the cores seems to correlate with the sparsity of the network. The network Erdős Rényi(8), which has the highest density, does not have a well-defined core-periphery structure.

It is clear from Table 5.5 that while the diameter of each minimum-cover-set core is slightly smaller than the diameter of the network, its hyperbolicity ($hb(G_{\text{core}}^m)$) is equal to the hyperbolicity of the original network ($hb(G)$). The exception is networks Erdős Rényi(2) and Erdős Rényi(8). Table 5.5 also shows the decrease in the number of considered quadruples (compared to the number of quadruples needed to compute $hb(G)$) and the decrease in the running time (compared to the running time needed to compute $hb(G)$). For example, in the Facebook network, there is a 99.9 decrease in the number of considered quadruples. The running time needed to compute the hyperbolicity for the original Facebook network was about 18 hours, but it took only few seconds to compute it for the
minimum-cover-set core $hb(G_{\text{core}}^m)$ \(^1\). In the network PowerGrid, there is a 79.4 decrease in the number of quadruples (the time needed to compute $hb(G)$ and $hb(G_{\text{core}})$ is about 31 hours and 15 hours respectively).

### 5.4.2 $k$-core

The $k$-core decomposition [176] provides a way to decompose a graph that allows the identification of interesting structural properties that are not captured by other simple structural measures. Unlike the $\delta$-hyperbolicity, the $k$-core decomposition is not intended to be a tree-like measure, yet in [6], the authors find the $k$-core of a graph to be an important part of its hyperbolic structure.

**Definition 10.** Given a graph $G = (V, E)$ and a set $C \subseteq V$, $G_C$ is the $k$-core of $G$ if and only if $G_C$ is the maximal induced subgraph with the property that for all $v \in C$, $\text{degree}(v) \geq k$.

Here we denote the $k$-core of a given graph $G$ by $G_{\text{core}}^k$ (Figure 5.4). Each subgraph $G_{\text{core}}^k$ of $G$ is unique and can be found by iteratively deleting vertices with degree less than $k$. The maximum $k$ such that $G$ has a $k$-core is the main core of $G$.

A vertex $x$ has core number $k$ if it belongs to the $k$-core, but not the $k + 1$-core. All vertices with core number $k$ form the $k$-shell. Parameter $k$ refers to the depth of the core (higher $k$ values represent deeper cores). The resulting cores are nested, and each core is not necessarily a connected subgraph. The $k$-core decomposition can be implemented in linear time which makes it applicable to very large graphs [27].

In Table 5.6, we list two different core numbers (depths): $k_{\text{max}}$ which is the main core and $k_\delta$ which is the maximum $k$ such that the core subgraph $G_{\text{core}}^k$ achieves a hyperbolicity

---

\(^1\)All experiments in this chapter were performed on a personal computer with an Intel(R) 2.50 GHz CPU and 16 GB Ram without the use of multiprocessors.
value that is equal to $hb(G)$. We also compute $k_{min}$ (not listed in the table) which is the maximum $k$ such that $G_{core}^k = G$. Table 5.6 also lists the size and the diameter of the $k_{max}$-shell and the size, the diameter, and the hyperbolicity of the $k_{delta}$-shell of each network.

Table 5.6 shows that the $G_{core}^{k_{delta}}$ has smaller diameter compared to each original network. In all networks, $k_{min} = 1$, and $k_{delta}$ is always greater than $k_{min}$ which suggests that the quadruples responsible for increasing the value of the $delta$-hyperbolicity concentrate in a deeper core in the network. Note that in some networks, all vertices belong to the same core ($k_{min} \approx k_{max}$). Some networks such as the US-Airways and the Email have $k_{delta} \approx k_{max}$ which indicates a tree-like structure that concentrates within the deep core of the network.
5.5 Case studies

In this section, we apply the idea of calculating the $\delta$-hyperbolicity of the core to two larger real-world networks for which calculating the exact value of the $\delta$-hyperbolicity is computationally expensive on a personal computer and for which the values of the $\delta$-hyperbolicity are known [5]. The first network is a biological network. The second network is an Internet graph.

5.5.1 Biological network

This biological network represents the protein and genetic interactions in human genus [180]. It consists of 16711 vertices (proteins and genes) and 115406 edges linking each pair of interacting proteins or genes. We focus on the largest connected component in the network which includes 16635 vertices and 115364 edges. The network has diameter 10, radius 5, and average path length 2.87. The $\delta$-hyperbolicity of this network is 2 [5].

The minimum-cover-set core of the network has 6546 vertices (39% of the number of vertices of the original graph), 84889 edges, and diameter 8. The size of the core subgraph allows us to calculate its $\delta$-hyperbolicity and compare it with that of the original network. The hyperbolicity of the minimum-cover-set core is 2, which was calculated with 97.6% less quadruples.

For the $k$-core of the network, $k_{min} = 1$, $k_{max} = 45$, and $k_\delta = 14$. The $k$-core (which corresponds to the $k_\delta$-shell) consists of 3053 vertices, which represents only 18% of the number of vertices in the original graph, and 64085 edges with a diameter of 4. The computation of the $\delta$-hyperbolicity for the $k$-core requires 99.8% less quadruples compared to the number of quadruples required to calculate the $\delta$-hyperbolicity for the original network.
5.5.2 AS-graph

This network depicts the Internet Autonomous Systems (AS) relationships collected by the Cooperative Association for the Internet Data Analysis (CAIDA) [198] during June, 2012. The data was derived from BGP table snapshots taken at 8-hour intervals over a period of five days.

The network includes 41203 vertices and 121309 edges (the average degree is 5.9). The diameter and radius of the network are 10 and 5 respectively. Also, the $\delta$-hyperbolicity of the network is 2 [5]. Because of the size of the network, we remove all dominated vertices before calculating the minimum-cover-set core. About 57% of the vertices in the original network are dominated vertices, and 36% of which has a degree of one.

After removing all dominated vertices, the new network has 17760 vertices, 78576 edges, and the diameter is 9. We extract the minimum-cover-set core of this network which consists of 6576 vertices and 45092 edges with a diameter of 8. Compared to the original network, the size of the core is only 16%, yet it has a $\delta$-hyperbolicity of 2.

We also compute the $k_{max}$-core and the $k_{\delta}$-core for this network. The $k_{max}$-shell has 55 vertices and a diameter of 2, which is too small to achieve the hyperbolicity of the original network. The $k_{\delta}$-shell (with a hyperbolicity equal to the hyperbolicity of the original network) has 3873 vertices, 56054 edges, and a diameter of 5 ($k_{\delta} = 5$).

5.6 Concluding remarks

This chapter describes a method of identifying quadruples that maximize the hyperbolicity using the dominance relationship between vertices. Also it demonstrates an interesting property of the $\delta$-hyperbolicity in networks, which is its realization in quadruples with vertices that are within relatively close proximity and that are close to the graph’s core.
Restricting the calculation of the $\delta$-hyperbolicity to some core of the network enables the computation of its value for large networks. Even though the hyperbolicity of the core may not resemble the exact value of the hyperbolicity of the graph, it provides a reasonable approximation. A key issue that needs to be considered when applying the idea of calculating the hyperbolicity within the core is the type of the network. Restricting hyperbolicity calculation within the core offers a tremendous gain in calculation time for networks with a clear cut core-periphery structure (more concise cores) including social and biological networks. This may not be the case for networks that lack a well-defined core such as some transportation networks and peer-to-peer networks. Moreover, it would be interesting to compare the values of the hyperbolicity within other core definitions. For example, the core that results from including vertices with the highest closeness centrality and/or betweenness centrality.

An interesting focus of subsequent research is the development of a local algorithm that calculates the $p$-$\delta$-hyperbolicity of very large graphs, and the estimation of a $p$ value that guarantees a $p$-$\delta$-hyperbolicity that is close (if not equal) to the hyperbolicity of the original graph.
Part II

Eccentricity Approximating Trees
By definition, a tree is a graph with no cycles, which implies the existence of exactly a single path between every vertex pair in the tree. The simplicity of the tree structure makes it very attractive from an algorithmic and practical point of view. Many problems that are hard in the general graph form have tractable solutions when the graph in question is a tree. For an example, consider the following problem. Given a network with \( n \) vertices, common queries concerning a vertex pair in the network are what is the distance between them? Or, what is a shortest path that connects them? Both queries require knowledge about the global structure of the given network. This is not desirable and sometimes impossible for very large networks. An alternative would be storing a compact piece of information (called a label) locally at each vertex such that an answer to a query can be generated using only the labels of two vertices. In this context, the problem of finding the distance between a vertex pair is called the distance labeling problem, and the problem of finding a shortest path between a vertex pair is called the routing labeling problem. General graphs support an exact distance labeling scheme of size \( O(n) \) bits and \( O(\log \log(n)) \) decode time; whereas, trees support \( O(\log(n)) \) size labels [104] and hyperbolic graphs support \( O(\log^2(n)) \) size labels [66, 103]. Moreover, general graphs support an exact routing labeling scheme of size \( O(n \log(n)) \) bits; whereas, trees support a \( O(\log^2(n)) \) size labels [185] and hyperbolic graphs support a \( O(\delta \log^2(n)) \) size labels [66].

There are other applications in which mapping a graph into a tree is desirable. Even though spanning trees are the cheapest way to maintain the connectivity of graphs, spanning trees may dramatically increase the distances among vertex pairs. Mapping a graph into a spanning tree such that distances in the tree do not exceed distances in the graph by more than an additive factor \( k \) is known as the distance \( k \)-approximating tree problem [49, 41, 135] (a multiplicative version of this problem also exists). In a variation of this problem, known as the eccentricity \( k \)-approximating tree problem, it is important that
every vertex (in the tree) can reach every other vertex in the shortest possible distance.

**Definition 11** (Distance \(k\)-approximating trees). Given an undirected graph \(G = (V, E)\) and an integer \(k \geq 0\), a spanning tree \(T = (V, E')\) of \(G\) is an eccentricity \(k\)-approximating tree if \(d_T(u, v) - d_G(u, v) \leq k\) for every vertex pair \(u, v\) of \(G\).

**Definition 12** (Eccentricity \(k\)-approximating trees). Given an undirected graph \(G = (V, E)\) and an integer \(k \geq 0\), a spanning tree \(T = (V, E')\) of \(G\) is an eccentricity \(k\)-approximating tree if \(ecc_T(v) - ecc_G(v) \leq k\) for every vertex \(v\) of \(G\).

It is known that any \(\delta\)-hyperbolic graph can be embedded into a tree with an additive error \(O(\delta \log_2(n))\) [58]. In this part, we investigate the eccentricity \(k\)-approximating tree problems for \(\delta\)-hyperbolic graphs (Chapter 6). We also investigate the eccentricity \(k\)-approximating tree problem for a special class of graphs: the \((\alpha_1, \Delta)\)-metric graphs (Chapter 7). In each chapter, we propose two algorithms for constructing eccentricity \(k\)-approximating trees; then we apply each on a set of real-world networks. In Chapter 8, we compare the qualities of the proposed algorithms.
Chapter 6

Eccentricity Approximating Trees in $\delta$-Hyperbolic Graphs

The eccentricity function of hyperbolic graphs exhibits the following interesting monotonicity property: the closer a vertex is to the center of a graph $G$, the smaller its eccentricity is. We use this property to show that the eccentricities (and thus the centrality indices) of all vertices of a $\delta$-hyperbolic graph can be computed in linear time with an additive one-sided error of at most $c\delta$ for a small constant $c$. We prove that every $\delta$-hyperbolic graph $G$ has a shortest path tree, constructible in linear time, such that for every vertex $v$ of $G$, $\text{ecc}_G(v) \leq \text{ecc}_T(v) \leq \text{ecc}_G(v) + c\delta$. Recent empirical studies show that many real-world graphs have small hyperbolicity. So, we analyze the performance of our algorithms for approximating eccentricity on a number of real-world networks. Our experimental results show that the obtained estimates are even better than the theoretical bounds.

The results in this chapter were obtained in collaboration with Feodor Dragan, Victor Chepoi, Michel Habib, and Yann Vaxès [61].
6.1 Introduction

The diameter and the radius of a graph \( G = (V, E) \) are two fundamental metric parameters that have many important practical applications in real world networks. The problem of finding the center \( C(G) \) of a graph \( G \) is often studied as a facility location problem for networks where one needs to select a single vertex to place a facility so that the maximum distance from any demand vertex in the network is minimized. In the analysis of social networks (e.g., citation networks or recommendation networks), biological systems (e.g., protein interaction networks), computer networks (e.g., the Internet or peer-to-peer networks), transportation networks (e.g., public transportation or road networks), etc., the eccentricity of a vertex \( v \) is used to measure the importance of \( v \) in the network: the centrality index of \( v \) \([39]\) is defined as \( \frac{1}{\text{ecc}(v)} \).

Being able to compute efficiently the diameter, center, radius, and vertex centralities of a given graph has become an increasingly important problem in the analysis of large networks. The algorithmic complexity of the diameter and radius problems is very well-studied. For some special classes of graphs there are efficient algorithms [2, 29, 40, 48, 56, 62, 72, 85, 92, 114, 160]. However, for general graphs, the only known algorithms computing the diameter and the radius exactly compute the distance between every pair of vertices in the graph, thus solving the all-pairs shortest paths problem (APSP) and hence computing all eccentricities. In view of recent negative results [2, 36, 170], this seems to be the best what one can do since even for graphs with \( m = O(n) \) (where \( m \) is the number of edges and \( n \) is the number of vertices) the existence of a subquadratic time (i.e., \( O(n^{2-\epsilon}) \) time for some \( \epsilon > 0 \)) algorithm for the diameter or the radius problem will refute the well known Strong Exponential Time Hypothesis (SETH). Furthermore, recent work [1] shows that if the radius of a possibly dense graph \( (m = O(n^2)) \) can be computed in subcubic
time \((O(n^{3-\epsilon}))\) for some \(\epsilon > 0\), then APSP also admits a subcubic algorithm. Such an algorithm for APSP has long eluded researchers, and it is often conjectured that it does not exist (see, e.g., [171, 188]).

Motivated by these negative results, researches started devoting more attention to development of fast approximation algorithms. In the analysis of large-scale networks, for fast estimations of diameter, center, radius, and centrality indices, linear or almost linear time algorithms are desirable. One hopes also for the all-pairs shortest paths problem to have \(o(nm)\) time small-constant-factor approximation algorithms. In general graphs, both diameter and radius can be 2-approximated by a simple linear time algorithm which picks any node and reports its eccentricity. A 3/2-approximation algorithm for the diameter and the radius which runs in \(\tilde{O}(mn^{2/3})\) time was recently obtained in [51]. For the sparse graphs, this is an \(o(n^2)\) time approximation algorithm. Furthermore, under plausible assumptions, no \(O(n^{2-\epsilon})\) time algorithm can exist that \((3/2-\epsilon')\)-approximates (for \(\epsilon, \epsilon' > 0\)) the diameter [170] and the radius [2] in sparse graphs. Similar results are known also for all eccentricities: a \(5/3\)-approximation to the eccentricities of all vertices can be computed in \(\tilde{O}(m^{3/2})\) time [51] and, under plausible assumptions, no \(O(n^{2-\epsilon})\) time algorithm can exist that \((5/3-\epsilon')\)-approximates (for \(\epsilon, \epsilon' > 0\)) the eccentricities of all vertices in sparse graphs [2]. Better approximation algorithms are known for some special classes of graphs [42, 57, 58, 72, 73, 81, 84, 86, 193].

The need for fast approximation algorithms for estimating diameters, radii, centrality indices, or all pairs shortest paths in large-scale complex networks dictates to look for geometric and topological properties of those networks and utilize them algorithmically. The classical relationships between the diameter, radius, and center of trees and folklore linear time algorithms for their computation is one of the departing points of this research.

\(\tilde{O}\) hides a polylog factor.
A result from 1869 by C. Jordan [127] asserts that the radius of a tree $T$ is roughly equal to half of its diameter and the center is either the middle vertex or the middle edge of any diametral path. The diameter and a diametral pair of $T$ can be computed (in linear time) by a simple but elegant procedure: pick any vertex $x$, find any vertex $y$ furthest from $x$, and find once more a vertex $z$ furthest from $y$; then return $\{y, z\}$ as a diametral pair. One computation of a furthest vertex is called an FP scan; hence the diameter of a tree can be computed via two FP scans. This two FP scans procedure can be extended to exact or approximate computation of the diameter and radius in many classes of tree-like graphs. For example, this approach was used to compute the radius and a central vertex of a chordal graph in linear time [56]. In this case, the center of $G$ is still close to the middle of all $(y, z)$-shortest paths and $d_G(y, z)$ is not the diameter but is still its good approximation: $d(y, z) \geq \text{diam}(G) - 2$. Even better, the diameter of any chordal graph can be approximated in linear time with an additive error 1 [86]. But it turns out that the exact computation of diameters of chordal graphs is as difficult as the general diameter problem: it is even difficult to decide if the diameter of a split graph is 2 or 3.

The experience with chordal graphs shows that one have to abandon the hope of having fast exact algorithms, even for very simple (from metric point of view) graph-classes, and to search for fast algorithms approximating $\text{diam}(G), \text{rad}(G), C(G), \text{ecc}_G(v)$ with a small additive constant depending only of the coarse geometry of the graph. Gromov hyperbolicity or the negative curvature of a graph (and, more generally, of a metric space) is one such constant. A graph $G = (V, E)$ is $\delta$-hyperbolic [15, 105, 43, 110] if for any four vertices $w, v, x, y$ of $G$, the two largest of the three distance sums $d(w, v) + d(x, y), d(w, x) + d(v, y), d(w, y) + d(v, x)$ differ by at most $2\delta \geq 0$. The hyperbolicity $\delta(G)$ of a graph $G$ is the smallest number $\delta$ such that $G$ is $\delta$-hyperbolic. The hyperbolicity can be viewed as a local measure of how close a graph is metrically to a tree: the smaller the hyperbolicity is,
the closer its metric is to a tree-metric (trees are 0-hyperbolic and chordal graphs are 1-hyperbolic). In [58], the authors initiated the investigation of diameter, center, and radius problems for $\delta$-hyperbolic graphs and showed that the existing approach for trees can be extended to this general framework. Namely, it is shown in [58] that if $G$ is a $\delta$-hyperbolic graph and $\{y,z\}$ is the pair returned after two FP scans, then $d(y,z) \geq diam(G) - 2\delta$, $diam(G) \geq 2rad(G) - 4\delta - 1$, $diam(C(G)) \leq 4\delta + 1$, and $C(G)$ is contained in a small ball centered at a middle vertex of any shortest $(y,z)$-path. Consequently, they obtained linear time algorithms for the diameter and radius problems with additive errors linearly depending on the input graph’s hyperbolicity.

In this chapter, we advance this line of research and provide a linear time algorithm for approximate computation of the eccentricities (and thus of centrality indices) of all vertices of a $\delta$-hyperbolic graph $G$, i.e., we compute the approximate values of all eccentricities within the same time bounds as one computes the approximation of the largest or the smallest eccentricity ($diam(G)$ or $rad(G)$). Namely, the algorithm outputs for every vertex $v$ of $G$ an estimate $\hat{e}(v)$ of $ecc_G(v)$ such that $ecc_G(v) \leq \hat{e}(v) \leq ecc_G(v) + c\delta$, where $c > 0$ is a small constant. In fact, we demonstrate that $G$ has a shortest path tree, constructible in linear time, such that for every vertex $v$ of $G$, $ecc_G(v) \leq ecc_T(v) \leq ecc_G(v) + c\delta$. The main ingredient in proving these results is the following interesting dependency between the eccentricities of vertices of $G$ and their distances to the center $C(G)$: up to an additive error linearly depending on $\delta$, $ecc_G(v)$ is equal to $d(v, C(G))$ plus $rad(G)$. To establish these new results, we have to revisit the results of [58] about diameters, radii, and centers, by simplifying their proofs and extending them to all eccentricities. Finally, we analyze the performance of our algorithms for approximating eccentricities on a number of real-world networks. Our experimental results show that the estimates on eccentricities obtained are even better than the theoretical bounds proved.
Recent empirical studies showed that many real-world graphs (including Internet application networks, web networks, collaboration networks, social networks, biological networks, and others) are tree-like from a metric point of view [5, 6, 35] or have small hyperbolicity [130, 155, 178]. It has been suggested in [155], and recently formally proved in [63], 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 paths between them are curved inwards, is due to the hyperbolicity of the network. Small hyperbolicity in real-world graphs provides many algorithmic advantages. Efficient approximate solutions are attainable for a number of optimization problems [58, 59, 63, 64, 76, 93, 190].

6.2 Preliminaries

Let $G = (V, E)$ be a graph. By $[x, y]$ we denote a shortest path connecting vertices $x$ and $y$ in $G$; we call $[x, y]$ a geodesic between $x$ and $y$. A ball $B(s, r)$ of $G$ 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., $B(s, r) := \{v \in V : d_G(v, s) \leq r\}$).

Denote by $F(x) := \{y \in V : d_G(x, y) = ecc_G(x)\}$ the set of all vertices of $G$ that are most distant from $x$. Vertices $x$ and $y$ of $G$ are called mutually distant if $x \in F(y)$ and $y \in F(x)$, i.e., $ecc_G(x) = ecc_G(y) = d_G(x, y)$.

6.3 Fast approximation of eccentricities

In this section, we give linear and almost linear time algorithms for sharp estimation of the diameters, the radii, the centers and the eccentricities of all vertices in graphs with $\delta$-thin triangles. Before presenting those algorithms, we establish some conditional lower bounds on complexities of computing the diameters and the radii in those graphs.
6.3.1 Conditional lower bounds on complexities

Recent work has revealed convincing evidence that solving the diameter problem in subquadratic time might not be possible, even in very special classes of graphs. Roditty and Vassilevska W. [170] showed that an algorithm that can distinguish between diameter 2 and 3 in a sparse graph in subquadratic time refutes the following widely believed conjecture.

The Orthogonal Vectors Conjecture: There is no $\epsilon > 0$ such that for all $c \geq 1$, there is an algorithm that given two lists of $n$ binary vectors $A, B \subseteq \{0, 1\}^d$ where $d = c \log n$ can determine if there is an orthogonal pair $a \in A, b \in B$, in $O(n^{2-\epsilon})$ time.

Williams [194] showed that the Orthogonal Vectors (OV) Conjecture is implied by the well-known Strong Exponential Time Hypothesis (SETH) of Impagliazzo, Paturi, and Zane [119, 118]. Nowadays many papers base the hardness of problems on SETH and the OV conjecture (see, e.g., [2, 36, 187] and papers cited therein).

Since all geodesic triangles of a graph constructed in the reduction in [170] are 2-thin, we can rephrase the result from [170] as follows.

Statement 1. If for some $\epsilon > 0$, there is an algorithm that can determine if a given graph with 2-thin triangles, $n$ vertices and $m = O(n)$ edges has diameter 2 or 3 in $O(n^{2-\epsilon})$ time, then the Orthogonal Vector Conjecture is false.

To prove a similar lower bound result for the radius problem, recently Abboud et al. [2] suggested to use the following natural and plausible variant of the OV conjecture.

The Hitting Set Conjecture: There is no $\epsilon > 0$ such that for all $c \geq 1$, there is an algorithm that given two lists $A, B$ of $n$ subsets of a universe $U$ of size $c \log n$, can decide in $O(n^{2-\epsilon})$ time if there is a set in the first list that intersects every set in the second list, i.e. a hitting set.
Abboud et al. [2] showed that an algorithm that can distinguish between radius 2 and 3 in a sparse graph in subquadratic time refutes the Hitting Set Conjecture. Since all geodesic triangles of a graph constructed in the reduction in [2] are 2-thin, rephrasing that result from [2], we have.

**Statement 2.** *If for some \( \epsilon > 0 \), there is an algorithm that can determine if a given graph with 2-thin triangles, \( n \) vertices, and \( m = O(n) \) edges has radius 2 or 3 in \( O(n^{2-\epsilon}) \) time, then the Hitting Set Conjecture is false.*

### 6.3.2 Fast additive approximations

In this subsection, we show that in a graph \( G \) with \( \delta \)-thin triangles, the eccentricities of all vertices can be computed in total linear time with an additive error depending on \( \delta \). We establish that the eccentricity of a vertex is determined (up-to a small error) by how far the vertex is from the center \( C(G) \) of \( G \). Finally, we show how to construct a spanning tree \( T \) of \( G \) in which the eccentricity of any vertex is its eccentricity in \( G \) up to an additive error depending only on \( \delta \). For these purposes, we revisit and extend several results mentioned in [58] concerning the linear time approximation of diameter, radius, and centers of \( \delta \)-hyperbolic graphs. For these particular cases, we provide simplified proofs, leading to better additive errors due to the use of thinness of triangles instead of the four point condition and to the computation in \( O(\delta|E|) \) time of a pair of mutually distant vertices.

Define the eccentricity layers of a graph \( G \) as follows: for \( k = 0, \ldots, \text{diam}(G) - \text{rad}(G) \) set

\[
C^k(G) := \{ v \in V : \text{ecc}_G(v) = \text{rad}(G) + k \}.
\]

With this notation, the center of a graph is \( C(G) = C^0(G) \). In what follows, it will be convenient to define also the eccentricity of the middle point \( m \) of any edge \( xy \) of \( G \); set \( \text{ecc}_G(m) = \min\{\text{ecc}_G(x), \text{ecc}_G(y)\} + 1/2. \)
We start with a proposition showing that, in a graph $G$ with $\delta$-thin triangles, a middle vertex of any geodesic between two mutually distant vertices has the eccentricity close to $rad(G)$ and is not too far from the center $C(G)$ of $G$.

**Proposition 8.** Let $G$ be a graph with $\delta$-thin triangles, $u, v$ be a pair of mutually distant vertices of $G$.

(a) If $c^*$ is the middle point of any $(u, v)$-geodesic, then $ecc_G(c^*) \leq \frac{d_G(u, v)}{2} + \delta \leq rad(G) + \delta$.

(b) If $c$ is a middle vertex of any $(u, v)$-geodesic, then $ecc_G(c) \leq \lceil \frac{d_G(u, v)}{2} \rceil + \delta \leq rad(G) + \delta$.

(c) $d_G(u, v) \geq 2rad(G) - 2\delta - 1$. In particular, $diam(G) \geq 2rad(G) - 2\delta - 1$.

(d) If $c$ is a middle vertex of any $(u, v)$-geodesic and $x \in C^k(G)$, then $k - \delta \leq d_G(x, c) \leq k + 2\delta + 1$. In particular, $C(G) \subseteq B(c, 2\delta + 1)$.

**Proof.** Let $x$ be an arbitrary vertex of $G$ and $\Delta(u, v, x) := [u, v] \cup [v, x] \cup [x, u]$ be a geodesic triangle, where $[v, x], [x, u]$ are arbitrary geodesics connecting $x$ with $v$ and $u$. Let $m_x$ be a point on $[u, v]$ which is at distance $(x|u)_v = \frac{1}{2}(d(x, v) + d(v, u) - d(x, u))$ from $v$ and hence at distance $(x|v)_u = \frac{1}{2}(d(x, u) + d(v, u) - d(x, v))$ from $u$. Since $u$ and $v$ are mutually distant, we can assume, without loss of generality, that $c^*$ is located on $[u, v]$ between $v$ and $m_x$, i.e., $d(v, c^*) \leq d(v, m_x) = (x|u)_v$, and hence $(x|v)_u \leq (x|u)_v$. Since $d_G(v, x) \leq d_G(v, u)$, we also get $(u|v)_x \leq (x|v)_u$.

(a) By the triangle inequality and since $d_G(u, v) \leq diam(G) \leq 2rad(G)$, we get

$$d_G(x, c^*) \leq e(u|v)_x + \delta + d_G(u, c^*) - (x|v)_u$$
$$\leq d_G(u, c^*) + \delta = \frac{d_G(u, v)}{2} + \delta \leq rad(G) + \delta.$$
(b) Since \( c^* = c \) when \( d_G(u, v) \) is even and \( d_G(c^*, c) = \frac{1}{2} \) when \( d_G(u, v) \) is odd, we have \( \text{ecc}_G(c) \leq \text{ecc}_G(c^*) + \frac{1}{2} \). Additionally to the proof of (a), one needs only to consider the case when \( d_G(u, v) \) is odd. We know that the middle point \( c^* \) sees all vertices of \( G \) within distance at most \( \frac{d_G(u, v)}{2} + \delta \). Hence, both ends of the edge of \((u, v)\)-geodesic, containing the point \( c^* \) in the middle, have eccentricities at most

\[
\frac{d_G(u, v)}{2} + \frac{1}{2} + \delta = \left\lfloor \frac{d_G(u, v)}{2} \right\rfloor + \delta \leq \left\lfloor \frac{2 \text{rad}(G) - 1}{2} \right\rfloor + \delta = \text{rad}(G) + \delta.
\]

(c) Since a middle vertex \( c \) of any \((u, v)\)-geodesic sees all vertices of \( G \) within distance at most \( \left\lfloor \frac{d_G(u, v)}{2} \right\rfloor + \delta \), if \( d_G(u, v) \leq 2 \text{rad}(G) - 2\delta - 2 \), then

\[
\text{ecc}_G(c) \leq \left\lfloor \frac{d_G(u, v)}{2} \right\rfloor + \delta \leq \left\lfloor \frac{2 \text{rad}(G) - 2\delta - 2}{2} \right\rfloor + \delta < \text{rad}(G),
\]

which is impossible.

(d) In the proof of (a), instead of an arbitrary vertex \( x \), consider any vertex \( x \) from \( C^k(G) \). By the triangle inequality and since \( d_G(u, v) \geq 2 \text{rad}(G) - 2\delta - 1 \) and both \( d_G(u, x), d_G(x, v) \) are at most \( \text{rad}(G) + k \), we get

\[
d_G(x, c^*) \leq (u|v)_x + \delta + (x|u)_v - d_G(v, c^*) = d_G(v, x) - d_G(v, c^*) + \delta \\
\leq \text{rad}(G) + k - \frac{d_G(u, v)}{2} + \delta \leq k + 2\delta + \frac{1}{2}.
\]

Consequently, \( d_G(x, c) \leq d_G(x, c^*) + \frac{1}{2} \leq k + 2\delta + 1 \). On the other hand, since \( \text{ecc}_G(x) \leq \text{ecc}_G(c) + d_G(x, c) \) and \( \text{ecc}_G(c) \leq \text{rad}(G) + \delta \), by statement (a), we get

\[
d_G(x, c) \geq \text{ecc}_G(x) - \text{ecc}_G(c) = k + \text{rad}(G) - \text{ecc}_G(c) \\
\geq (k + \text{rad}(G)) - (\text{rad}(G) + \delta) = k - \delta.
\]
As an easy consequence of Proposition 8(d), we get that the eccentricity \( ecc_G(x) \) of any vertex \( x \) is equal, up to an additive one-sided error of at most \( 4\delta + 2 \), to \( d_G(x, C(G)) \) plus \( rad(G) \).

**Corollary 1.** For every vertex \( x \) of a graph \( G \) with \( \delta \)-thin triangles,

\[
d_G(x, C(G)) + rad(G) - 4\delta - 2 \leq ecc_G(x) \leq d_G(x, C(G)) + rad(G).
\]

**Proof.** Consider an arbitrary vertex \( x \) in \( G \) and assume that \( ecc_G(x) = rad(G) + k \). Let \( c_x \) be a vertex from \( C(G) \) closest to \( x \). By Proposition 8(d), \( d_G(c, c_x) \leq 2\delta + 1 \) and \( d_G(x, c) \leq k + 2\delta + 1 = ecc_G(x) - rad(G) + 2\delta + 1 \). Hence,

\[
d_G(x, C(G)) = d_G(x, c_x) \leq d_G(x, c) + d_G(c, c_x) \leq d_G(x, c) + 2\delta + 1
\]

and

\[
ecc_G(x) \geq d_G(x, c) + rad(G) - 2\delta - 1.
\]

Combining both inequalities, we get

\[
ecc_G(x) \geq d_G(x, C(G)) + rad(G) - 4\delta - 2.
\]

Note also that, by the triangle inequality, \( ecc_G(x) \leq d_G(x, c_x) + ecc_G(c_x) = d_G(x, C(G)) + rad(G) \) (i.e., the right-hand inequality holds for all graphs).

It is interesting to note that the equality \( ecc_G(x) = d_G(x, C(G)) + rad(G) \) holds for every vertex of a graph \( G \) if and only if the eccentricity function \( ecc_G(\cdot) \) on \( G \) is unimodal (i.e., every local minimum is a global minimum)[80]. A slightly weaker condition holds for
all chordal graphs [84]: for every vertex \( x \) of a chordal graph \( G \), \( ecc_G(x) \geq d_G(x, C(G)) + \text{rad}(G) - 1 \).

**Proposition 9.** Let \( G \) be a graph with \( \delta \)-thin triangles and \( u, v \) be a pair of vertices of \( G \) such that \( v \in F(u) \).

(a) If \( w \) is a vertex of a \((u,v)\)-geodesic at distance \( \text{rad}(G) \) from \( v \), then \( ecc_G(w) \leq \text{rad}(G) + \delta \).

(b) For every pair of vertices \( x, y \in V \), \( \max\{d_G(v,x), d_G(v,y)\} \geq d_G(x,y) - 2\delta \).

(c) \( ecc_G(v) \geq \text{diam}(G) - 2\delta \geq 2\text{rad}(G) - 4\delta - 1 \).

(d) If \( t \in F(v) \), \( c \) is a vertex of a \((v,t)\)-geodesic at distance \( \lceil \frac{d_G(v,t)}{2} \rceil \) from \( t \) and \( x \in C^k(G) \), then \( ecc_G(c) \leq \text{rad}(G) + 3\delta \) and \( k - 3\delta \leq d_G(x,c) \leq k + 3\delta + 1 \). In particular, \( C(G) \subseteq B(c, 3\delta + 1) \).

**Proposition 10.** For every graph \( G \) with \( \delta \)-thin triangles, \( \text{diam}(C^k(G)) \leq 2k + 2\delta + 1 \). In particular, \( \text{diam}(C(G)) \leq 2\delta + 1 \).

**6.3.2.1 Diameter and radius**

For an arbitrary connected graph \( G = (V, E) \) and a given vertex \( u \in V \), a most distant from \( u \) vertex \( v \in F(u) \) can be found in linear \((O(|E|))\) time by a breadth-first-search \( \text{BFS}(u) \) started at \( u \). A pair of mutually distant vertices of a connected graph \( G = (V, E) \) with \( \delta \)-thin triangles can be computed in \( O(\delta|E|) \) total time as follows. By Proposition 9(c), if \( v \) is a most distant vertex from an arbitrary vertex \( u \) and \( t \) is a most distant vertex from \( v \), then \( d(v,t) \geq \text{diam}(G) - 2\delta \). Hence, using at most \( O(\delta) \) breadth-first-searches, one can generate a sequence of vertices \( v := v_1, t := v_2, v_3, \ldots v_k \) with \( k \leq 2\delta + 2 \) such that each \( v_i \) is most distant from \( v_{i-1} \) (with, \( v_0 = u \)) and \( v_k, v_{k-1} \) are mutually distant vertices (the initial value \( d(v,t) \geq \text{diam}(G) - 2\delta \) can be improved at most \( 2\delta \) times).
Thus, by Proposition 8 and Proposition 9, we get the following additive approximations for the radius and the diameter of a graph with $\delta$-thin triangles.

**Corollary 2.** Let $G = (V, E)$ be a graph with $\delta$-thin triangles.

1. There is a linear ($O(|E|)$) time algorithm which finds in $G$ a vertex $c$ with eccentricity at most $\text{rad}(G) + 3\delta$ and a vertex $v$ with eccentricity at least $\text{diam}(G) - 2\delta$. Furthermore, $C(G) \subseteq B(c, 3\delta + 1)$ holds.

2. There is an almost linear ($O(\delta|E|)$) time algorithm which finds in $G$ a vertex $c$ with eccentricity at most $\text{rad}(G) + \delta$. Furthermore, $C(G) \subseteq B(c, 2\delta + 1)$ holds.

**6.3.2.2 All eccentricities.**

In what follows, we will show that all vertex eccentricities of a graph with $\delta$-thin triangles can be also additively approximated in (almost) linear time.

**Proposition 11.** Let $G$ be a graph with $\delta$-thin triangles.

(a) If $c$ is a middle vertex of any $(u, v)$-geodesic between a pair $u, v$ of mutually distant vertices of $G$ and $T$ is a BFS($c$)-tree of $G$, then, for every vertex $x$ of $G$, $\text{ecc}_G(x) \leq \text{ecc}_T(x) \leq \text{ecc}_G(x) + 3\delta + 1$.

(b) If $v$ is a most distant vertex from an arbitrary vertex $u$, $t$ is a most distant vertex from $v$, $c$ is a vertex of a $(v, t)$-geodesic at distance $\lceil \frac{d_G(v, t)}{2} \rceil$ from $t$ and $T$ is a BFS($c$)-tree of $G$, then $\text{ecc}_G(x) \leq \text{ecc}_T(x) \leq \text{ecc}_G(x) + 6\delta + 1$.

*Proof.* (a) Let $x$ be an arbitrary vertex of $G$ and assume that $\text{ecc}_G(x) = \text{rad}(G) + k$ for some integer $k \geq 0$. We know from Proposition 8(b) that $\text{ecc}_G(c) \leq \text{rad}(G) + \delta$. Furthermore, by Proposition 8(d), $d_G(c, x) \leq k + 2\delta + 1$. Since $T$ is a BFS($c$)-tree, $d_G(x, c) = d_T(x, c)$
and \( ecc_G(c) = ecc_T(c) \). Consider a vertex \( y \) in \( G \) such that \( d_T(x,y) = ecc_T(x) \). We have

\[
ecct(x) = d_T(x,y) \leq d_T(x,c) + d_T(c,y) \\
\leq d_G(x,c) + ecc_T(c) = d_G(x,c) + ecc_G(c) \\
\leq k + 2\delta + 1 + rad(G) + \delta = rad(G) + k + 3\delta + 1 \\
= ecc_G(x) + 3\delta + 1.
\]

As \( T \) is a spanning tree of \( G \), evidently, also \( ecc_G(x) \leq ecc_T(x) \) holds.

(b) The proof is similar to the proof of (a); only, in this case, \( ecc_G(c) \leq rad(G) + 3\delta \) and \( d_G(c,x) \leq k + 3\delta + 1 \) holds for every \( x \in C^k(G) \) (by Proposition 9(d)).

A spanning tree \( T \) of a graph \( G \) is called an eccentricity \( k \)-approximating spanning tree if for every vertex \( v \) of \( G \) \( ecct(v) \leq ecc_G(v) + k \) holds [84, 162]. Thus, by Proposition 11, we get.

Theorem 2. Every graph \( G = (V,E) \) with \( \delta \)-thin triangles admits an eccentricity \( (3\delta + 1) \)-approximating spanning tree constructible in \( O(\delta|E|) \) time and an eccentricity \( (6\delta + 1) \)-approximating spanning tree constructible in \( O(|E|) \) time.

Theorem 2 generalizes recent results from [162] and our result in [84] (see also Chapter 7) that chordal graphs and some of their generalizations admit eccentricity 2-approximating spanning trees.

Note that the eccentricities of all vertices in any tree \( T = (V,U) \) can be computed in \( O(|V|) \) total time. As we noticed already, it is a folklore by now that for trees the following facts are true:

1. The center \( C(T) \) of any tree \( T \) consists of one vertex or two adjacent vertices.
2. The center \( C(T) \) and the radius \( rad(T) \) of any tree \( T \) can be found in linear time.
(3) For every vertex $v \in V$, $ecc_T(v) = d_T(v, C(T)) + rad(T)$.

Hence, using $BFS(C(T))$ on $T$ one can compute $d_T(v, C(T))$ for all $v \in V$ in total $O(|V|)$
time. Adding now $rad(T)$ to $d_T(v, C(T))$, one gets $ecc_T(v)$ for all $v \in V$. Consequently,
by Theorem 2, we get the following additive approximations for the vertex eccentricities in
graphs with $\delta$-thin triangles.

**Theorem 3.** Let $G = (V, E)$ be a graph with $\delta$-thin triangles.

(1) There is an algorithm which in total linear $(O(|E|))$ time outputs for every vertex $v \in V$
an estimate $\hat{e}(v)$ of its eccentricity $ecc_G(v)$ such that $ecc_G(v) \leq \hat{e}(v) \leq ecc_G(v) + 6\delta + 1$.

(2) There is an algorithm which in total almost linear $(O(\delta|E|))$ time outputs for every
vertex $v \in V$ an estimate $\hat{e}(v)$ of its eccentricity $ecc_G(v)$ such that $ecc_G(v) \leq \hat{e}(v) \leq ecc_G(v) + 3\delta + 1$.

### 6.4 Experimentation on some real-world networks

In this section, we analyze the performance of our algorithms for approximating eccentricities
on a number of real-world networks. Our experimental results show that the estimates
on eccentricities and distances obtained are even better than the theoretical bounds de-
scribed in Corollary 2 and Theorem 3.

We apply our algorithms to six social networks, four email communication networks,
four biological networks, six internet graphs, four peer-to-peer networks, three web net-
works, two product-co-purchasing networks, and four infrastructure networks. Most of the
networks listed are part of the Stanford Large Network Dataset Collection (SNAP) and the
Koblenz Network Collection (KONECT), and are available at [204] and [202]. Characteris-
tics of these networks, such as the number of vertices and edges, the average degree, the
Table 6.1: Graph datasets and their parameters: $|V|$ is the number of vertices, $|E|$ is the number of edges; $|C(G)|$ is the number of central vertices; $\bar{\text{deg}}$ is the average degree; $\text{rad}(G)$ is the graph’s radius; $\text{diam}(G)$ is the graph’s diameter; $\text{diam}_G(C(G))$ is the diameter of the graph’s center; “connected?” indicates whether or not the center of the graph is connected; $\delta(G)$ is the graph’s hyperbolicity. Hyperbolicity values marked with asterisks are approximate.
radius and the diameter, are given in Table 6.1. The numbers listed in Table 6.1 are based on the largest connected component of each network, when the entire network is disconnected. We ignore the directions of the edges and remove all self-loops from each network. Additionally, in Table 6.1, for each network we report the size (as the number of vertices) of its center $C(G)$. We also analyze the diameter and the connectivity of the center of each network. The diameter of the center $\text{diam}_G(C(G))$ is defined as the maximum distance between any two central vertices in the graph. In the last column of Table 6.1, we report the Gromov hyperbolicity $\delta$ of majority of networks\(^2\). Computing the hyperbolicity of a graph is computationally expensive; therefore, we provide the exact $\delta$ values for the smaller networks (those with $|V| \leq 30K$) in our datasets (in some cases, the algorithm proposed in [70] was used). For some larger networks, the approximated $\delta$-hyperbolicity values listed in Table 6.1 are as reported in [130]\(^3\). Most networks that we included in our datasets are hyperbolic. However, for comparison reasons, we included also a few infrastructure networks that are known to lack the hyperbolicity property.

### 6.4.1 Estimation of eccentricities

Following Proposition 8, for each graph in our datasets, we found a pair $u, v$ of mutually distant vertices. In column two of Table 6.2, we report on how many $BFS$ sweeps of a graph were needed to locate $u$ and $v$.Interestingly, for almost all graphs (28 out 33) only two sweeps were sufficient. For four other graphs (including ROAD-PA network whose hyperbolicity is large) three sweeps were needed, and only for one graph (POWER-GRID network) we needed four sweeps.

\(^2\)All $\delta$-hyperbolicity values listed in Table 6.1 were computed using Gromov’s four-point condition definition. As mentioned in [105, 110], geodesic triangles of geodesic $\delta$-hyperbolic spaces are $4\delta$-thin.

\(^3\)For WEB-STANFORD and WEB-BERKSTAN, [130] gives 1.5 and 2, respectively, as estimates on the hyperbolicities. However, the sampling method they used seems to be not very accurate. According to [152], the hyperbolicities are at least 7 for both graphs.
Table 6.2: Qualities of a pair of mutually distant vertices $u$ and $v$, of a middle vertex $c$ of a $(u,v)$-geodesic, and of a $BFS(c)$-tree $T_1$ rooted at vertex $c$. “No. of BFS iterations” indicates how many breadth-first-search iterations were needed to obtain a pair of mutually distant vertices $u$ and $v$. For each vertex $x \in V$, $k(x) := ecc_{T_1}(x) - ecc_G(x)$. Also, $k_{\text{max}} := \max_{x \in V} k(x)$ and $k_{\text{avg}} := \frac{1}{n} \sum_{x \in V} k(x)$. 

<table>
<thead>
<tr>
<th>Network</th>
<th>No. of BFS iterations</th>
<th>$d_G(u,v)$</th>
<th>2$rad(G) - d_G(u,v)$</th>
<th>$ecc_G(c)$</th>
<th>$ecc_{T_1}(c) - rad_G(G)$</th>
<th>$d_G(c,C(G))$</th>
<th>$\text{min}_i : B(c,i) \supseteq C(G)$</th>
<th>$k_{\text{max}}$</th>
<th>$k_{\text{avg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUTCH-ELITE</td>
<td>2</td>
<td>22</td>
<td>2</td>
<td>13</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>2.35</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>2</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0.686</td>
</tr>
<tr>
<td>EVA</td>
<td>2</td>
<td>18</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.571</td>
</tr>
<tr>
<td>SLASHDOT</td>
<td>2</td>
<td>11</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1.777</td>
</tr>
<tr>
<td>LOANS</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>2.064</td>
</tr>
<tr>
<td>TWITTER</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2.569</td>
</tr>
<tr>
<td>EMAIL-VIRGILI</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2.729</td>
</tr>
<tr>
<td>EMAIL-ENRON</td>
<td>2</td>
<td>13</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.906</td>
</tr>
<tr>
<td>EMAIL-EU</td>
<td>2</td>
<td>14</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0.002</td>
</tr>
<tr>
<td>WIKITALK-CHINA</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2.076</td>
</tr>
<tr>
<td>CB-METABOLIC</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1.982</td>
</tr>
<tr>
<td>SC-PP1</td>
<td>3</td>
<td>19</td>
<td>3</td>
<td>12</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>0.981</td>
</tr>
<tr>
<td>YEAST-PP1</td>
<td>3</td>
<td>11</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1.827</td>
</tr>
<tr>
<td>HOMO-PI</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.747</td>
</tr>
<tr>
<td>AS-GRAF-1</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1.791</td>
</tr>
<tr>
<td>AS-GRAF-2</td>
<td>3</td>
<td>11</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>1.124</td>
</tr>
<tr>
<td>AS-GRAF-3</td>
<td>2</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.828</td>
</tr>
<tr>
<td>ROUTVIEW</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.329</td>
</tr>
<tr>
<td>AS-CAIDA</td>
<td>2</td>
<td>17</td>
<td>1</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ITDK</td>
<td>2</td>
<td>26</td>
<td>2</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2.108</td>
</tr>
<tr>
<td>GNUTELLA-06</td>
<td>2</td>
<td>10</td>
<td>2</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>2.907</td>
</tr>
<tr>
<td>GNUTELLA-24</td>
<td>2</td>
<td>10</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>2.697</td>
</tr>
<tr>
<td>GNUTELLA-30</td>
<td>2</td>
<td>11</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>3.167</td>
</tr>
<tr>
<td>GNUTELLA-31</td>
<td>2</td>
<td>11</td>
<td>3</td>
<td>8</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>4.176</td>
</tr>
<tr>
<td>WEB-STANFORD</td>
<td>2</td>
<td>164</td>
<td>0</td>
<td>82</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>28</td>
<td>0.006</td>
</tr>
<tr>
<td>WEB-NOTREDAIM</td>
<td>2</td>
<td>46</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.935</td>
</tr>
<tr>
<td>WEB-BERKSTAN</td>
<td>2</td>
<td>208</td>
<td>0</td>
<td>104</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>22</td>
<td>0.002</td>
</tr>
<tr>
<td>AMAZON-I</td>
<td>2</td>
<td>47</td>
<td>1</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>0.991</td>
</tr>
<tr>
<td>AMAZON-2</td>
<td>2</td>
<td>20</td>
<td>2</td>
<td>12</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>3.735</td>
</tr>
<tr>
<td>ROAD-EURO</td>
<td>2</td>
<td>62</td>
<td>0</td>
<td>36</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0.135</td>
</tr>
<tr>
<td>OPENLIGHT</td>
<td>2</td>
<td>13</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1.879</td>
</tr>
<tr>
<td>POWER-GRID</td>
<td>4</td>
<td>46</td>
<td>0</td>
<td>28</td>
<td>5</td>
<td>8</td>
<td>8</td>
<td>13</td>
<td>5.735</td>
</tr>
<tr>
<td>ROAD-PA</td>
<td>3</td>
<td>794</td>
<td>10</td>
<td>415</td>
<td>13</td>
<td>44</td>
<td>45</td>
<td>98</td>
<td>23.339</td>
</tr>
<tr>
<td>Network</td>
<td>$k_{\text{max}}$</td>
<td>$k_{\text{avg}}$</td>
<td>$%$ of vertices with $k(x) = 0$</td>
<td>$%$ of vertices with $k(x) = 1$</td>
<td>$%$ of vertices with $k(x) = 2$</td>
<td>$%$ of vertices with $k(x) = 3$</td>
<td>$%$ of vertices with $k(x) = 4$</td>
<td>$%$ of vertices with $k(x) = 5$ with $k(x) \geq 6$</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>DUTCH-ELITE</td>
<td>6</td>
<td>2.35</td>
<td>0.1</td>
<td>23.4</td>
<td>62.6</td>
<td>14</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>2</td>
<td>0.69</td>
<td>51.9</td>
<td>27.6</td>
<td>19.5</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>EVA</td>
<td>2</td>
<td>0.57</td>
<td>47.6</td>
<td>47.7</td>
<td>4.7</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>SLASHDOT</td>
<td>3</td>
<td>1.77</td>
<td>2.3</td>
<td>24.1</td>
<td>67.1</td>
<td>6.5</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>LOANS</td>
<td>3</td>
<td>2.06</td>
<td>0.1</td>
<td>13.9</td>
<td>66.3</td>
<td>19.7</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>TWITTER</td>
<td>4</td>
<td>2.56</td>
<td>0.1</td>
<td>64.4</td>
<td>21.2</td>
<td>3.4</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>EMAIL-VIRGILI</td>
<td>4</td>
<td>2.72</td>
<td>0.1</td>
<td>2.3</td>
<td>32</td>
<td>55.7</td>
<td>9.9</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>EMAIL-ENRON</td>
<td>2</td>
<td>0.90</td>
<td>23.4</td>
<td>62.6</td>
<td>14</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>EMAIL-EU</td>
<td>2</td>
<td>0.002</td>
<td>99.8</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>WIKITALK-CHINA</td>
<td>3</td>
<td>2.07</td>
<td>0.1</td>
<td>92.4</td>
<td>7.6</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>CE-METABOLIC</td>
<td>3</td>
<td>1.98</td>
<td>0.2</td>
<td>7.5</td>
<td>86.1</td>
<td>6.2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>SC-PPI</td>
<td>3</td>
<td>0.98</td>
<td>32.4</td>
<td>41.6</td>
<td>21.5</td>
<td>4.5</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>YEAST-PPI</td>
<td>3</td>
<td>1.87</td>
<td>2.3</td>
<td>25.4</td>
<td>55.8</td>
<td>16.8</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>HOMO-PPI</td>
<td>2</td>
<td>0.74</td>
<td>34.2</td>
<td>56.9</td>
<td>8.9</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>AS-GRAIN-1</td>
<td>3</td>
<td>1.79</td>
<td>0.5</td>
<td>24.9</td>
<td>69.7</td>
<td>4.9</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>AS-GRAIN-2</td>
<td>3</td>
<td>1.12</td>
<td>9.6</td>
<td>68.5</td>
<td>21.7</td>
<td>0.2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>AS-GRAIN-3</td>
<td>2</td>
<td>0.82</td>
<td>27.8</td>
<td>61.6</td>
<td>10.6</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>ROUTEVIEW</td>
<td>2</td>
<td>0.32</td>
<td>69.7</td>
<td>27.6</td>
<td>2.7</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>AS-CAIDA</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>ITIK</td>
<td>4</td>
<td>2.10</td>
<td>0.3</td>
<td>12</td>
<td>64.5</td>
<td>22.8</td>
<td>0.4</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Gnutella-06</td>
<td>4</td>
<td>2.50</td>
<td>0.3</td>
<td>5.7</td>
<td>41.1</td>
<td>48.8</td>
<td>4.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Gnutella-24</td>
<td>5</td>
<td>2.69</td>
<td>0.2</td>
<td>1.5</td>
<td>37</td>
<td>50.7</td>
<td>10.5</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Gnutella-30</td>
<td>5</td>
<td>3.16</td>
<td>0.1</td>
<td>1.8</td>
<td>13</td>
<td>52.4</td>
<td>31.8</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>Gnutella-31</td>
<td>6</td>
<td>4.17</td>
<td>0.01</td>
<td>9.4</td>
<td>64.1</td>
<td>13.3</td>
<td>51.1</td>
<td>33.4</td>
<td>0.5</td>
</tr>
<tr>
<td>Web-Stanford</td>
<td>28</td>
<td>0.006</td>
<td>99.9</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
</tr>
<tr>
<td>Web-Nortredam</td>
<td>2</td>
<td>0.93</td>
<td>7.1</td>
<td>92.4</td>
<td>0.5</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Web-BerkStan</td>
<td>22</td>
<td>0.002</td>
<td>99.9</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
</tr>
<tr>
<td>Amazon-1</td>
<td>6</td>
<td>0.991</td>
<td>28.1</td>
<td>48</td>
<td>21.5</td>
<td>1.5</td>
<td>0.5</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>Amazon-2</td>
<td>6</td>
<td>3.735</td>
<td>0.1</td>
<td>0.3</td>
<td>3.6</td>
<td>33.9</td>
<td>46.5</td>
<td>15.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Road-Euro</td>
<td>8</td>
<td>0.138</td>
<td>97.4</td>
<td>0.3</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>OpenFlight</td>
<td>3</td>
<td>1.87</td>
<td>0.2</td>
<td>23.9</td>
<td>63.7</td>
<td>12.2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Power-Grid</td>
<td>13</td>
<td>5.735</td>
<td>14.3</td>
<td>13.1</td>
<td>1.6</td>
<td>1.6</td>
<td>3.9</td>
<td>8.7</td>
<td>39.8</td>
</tr>
<tr>
<td>Road-Pa</td>
<td>98</td>
<td>23.339</td>
<td>0.02</td>
<td>1.5</td>
<td>0.1</td>
<td>2.9</td>
<td>0.2</td>
<td>0.2</td>
<td>95</td>
</tr>
</tbody>
</table>

Table 6.3: Distribution of distortion values $k(x) = \text{ecc}_{T_1}(x) - \text{ecc}_G(x)$, $x \in V$. $k_{\text{max}} := \max_{x \in V} k(x)$. $k_{\text{avg}} := \frac{1}{n} \sum_{x \in V} k(x)$. 127
In column four of Table 6.2, we report for each graph $G$ the difference between $2\text{rad}(G)$ and $d_G(u,v)$. Proposition 8(c) says that the difference must be at most $2\delta + 1$, where $\delta$ is the thinness of geodesic triangles in $G$. Actually, for large number (27 out of 33) of graphs in our datasets, the difference is at most two. Five other graphs have the difference equal to 3, and only ROAD-PA network has the difference equal to 10. We have $d_G(u,v) = \text{diam}(G)$ for 27 graphs in our datasets, including ROAD-PA network whose geodesic triangles thinness is at least 196. For remaining six graphs $d_G(u,v) = \text{diam}(G) − 1$ holds.

We also analyzed the quality of a middle vertex $c$ of a randomly picked shortest path between mutually distant vertices $u$ and $v$. Proposition 8 states that $\text{ecc}_G(c)$ is close to $\text{rad}(G)$ and $c$ is not too far from the graph’s center $C(G)$. Table 6.2 lists the properties of the selected middle vertex $c$. In almost all graphs, vertex $c$ belongs to the center $C(G)$ or is at distance one or two from $C(G)$. Even in graphs with $\text{ecc}_G(c)−\text{rad}(G) > 2$ (POWER-GRID and ROAD-PA), the value $\text{ecc}_G(c)−\text{rad}(G)$ is smaller than what is suggested by Proposition 8(b). It is also clear from Table 6.2 that $c$ is not too far from any vertex in $C(G)$ (look at the radius $i$ of the ball $B(c,i)$ required to include $C(G)$). In all graphs, $i$ is much smaller than $2\delta + 1$ (indicated in Proposition 8(d)).

Following Theorem 2, for each graph $G = (V,E)$ in our datasets, we constructed an arbitrary $\text{BFS}(c)$-tree $T_1 = (V,E')$, rooted at vertex $c$, and analyzed how well $T_1$ preserves or approximates the eccentricities of vertices in $G$. By Theorem 2, $\text{ecc}_G(v) \leq \text{ecc}_{T_1}(v) \leq \text{ecc}_G(v) + 3\delta + 1$ holds for every $v \in V$. In our experiments, for each graph $G$ and the constructed for it $\text{BFS}(c)$-tree $T_1$, we computed $k_{\text{max}} := \max_{v \in V}\{\text{ecc}_{T_1}(v) − \text{ecc}_G(v)\}$ (maximum distortion) and $k_{\text{avg}} := \frac{1}{n} \sum_{v \in V}\text{ecc}_{T_1}(v) − \text{ecc}_G(v)$ (average distortion). For most graphs (see Table 6.2), the value of $k_{\text{max}}$ is small: $k_{\text{max}} = 0$ for one graph, $k_{\text{max}} = 2$ for eight graphs, $k_{\text{max}} = 3$ for nine graphs, $k_{\text{max}} = 4$ for four graphs, $k_{\text{max}} = 5$ for two graphs, and $k_{\text{max}} > 5$ for nine graphs. Also, the average distortion $k_{\text{avg}}$ is much smaller.
\[
\text{Network} \quad \begin{array}{cccccccc}
& \text{ecc}_G(v) & 2\text{rad}(G) - \text{ecc}_G(v) & \text{ecc}_G(w) & d_G(w, C(G)) & \min \text{i:} & B(w, i) \geq C(G) & k_{\text{max}} & k_{\text{avg}} \\
\hline
\text{DUTCH-ELITE} & 22 & 2 & 12 & 0 & 4 & 6 & 2.431 \\
\text{FACEBOOK} & 8 & 0 & 5 & 3 & 3 & 3 & 0.704 \\
\text{EVA} & 18 & 2 & 11 & 1 & 3 & 2 & 0.572 \\
\text{SLASHDOT} & 11 & 1 & 7 & 2 & 2 & 3 & 1.88 \\
\text{LOANS} & 7 & 3 & 5 & 0 & 3 & 3 & 2.031 \\
\text{TWITTER} & 8 & 2 & 5 & 0 & 3 & 3 & 1.821 \\
\text{EMAIL-VIRGILI} & 7 & 3 & 5 & 0 & 4 & 4 & 1.932 \\
\text{EMAIL-ENRON} & 13 & 1 & 7 & 0 & 2 & 2 & 0.903 \\
\text{EMAIL-EU} & 14 & 0 & 7 & 0 & 0 & 2 & 0.002 \\
\text{WIKITALK-CHINA} & 8 & 0 & 5 & 1 & 2 & 3 & 1.791 \\
\text{CE-METABOLIC} & 7 & 1 & 4 & 0 & 1 & 1 & 0.349 \\
\text{SC-PI} & 19 & 3 & 12 & 1 & 6 & 7 & 4.196 \\
\text{YEAST-PI} & 11 & 1 & 7 & 1 & 3 & 4 & 2.558 \\
\text{HOMO-PI} & 9 & 1 & 5 & 0 & 2 & 2 & 0.612 \\
\text{AS-GRAPH-1} & 9 & 1 & 5 & 0 & 2 & 2 & 0.887 \\
\text{AS-GRAPH-2} & 11 & 1 & 6 & 0 & 3 & 2 & 0.833 \\
\text{AS-GRAPH-3} & 9 & 1 & 5 & 0 & 2 & 2 & 0.312 \\
\text{ROUTEVIEW} & 10 & 0 & 5 & 0 & 2 & 2 & 0.329 \\
\text{AS-CAIDA} & 17 & 1 & 9 & 0 & 1 & 0 & 0 \\
\text{ITDK} & 26 & 2 & 15 & 1 & 3 & 5 & 2.702 \\
\text{GNUTELLA-06} & 10 & 2 & 7 & 1 & 5 & 5 & 3.543 \\
\text{GNUTELLA-24} & 11 & 1 & 8 & 3 & 3 & 6 & 4.475 \\
\text{GNUTELLA-30} & 11 & 3 & 8 & 1 & 5 & 6 & 4.034 \\
\text{GNUTELLA-31} & 11 & 3 & 8 & 1 & 5 & 6 & 4.251 \\
\text{WEB-STANFORD} & 164 & 0 & 82 & 0 & 0 & 28 & 0.066 \\
\text{WEB-NOTREDAM} & 46 & 0 & 23 & 0 & 2 & 2 & 0.935 \\
\text{WEB-BERKSTAN} & 208 & 0 & 104 & 0 & 0 & 22 & 0.002 \\
\text{AMAZON-1} & 47 & 1 & 24 & 0 & 3 & 7 & 0.919 \\
\text{AMAZON-2} & 20 & 2 & 11 & 0 & 5 & 5 & 2.03 \\
\text{ROAD-EURO} & 62 & 0 & 31 & 0 & 0 & 8 & 0.135 \\
\text{OPENFLIGHT} & 13 & 1 & 7 & 0 & 2 & 2 & 0.641 \\
\text{POWER-GRID} & 46 & 0 & 23 & 0 & 0 & 4 & 1.409 \\
\text{ROAD-PA} & 772 & 32 & 417 & 21 & 22 & 80 & 22.545 \\
\end{array}
\]

Table 6.4: Qualities of a vertex \( v \) most distant from a random vertex \( u \), of a vertex \( w \) of a \((u,v)\)-geodesic at distance \( \text{rad}(G) \) from \( v \), and of a \( \text{BFS}(w) \)-tree \( T_2 \) rooted at vertex \( w \). For each vertex \( x \in V \), \( k(x) := \text{ecc}_{T_2}(x) - \text{ecc}_G(x) \). Also, \( k_{\text{max}} := \max_{x \in V} k(x) \) and \( k_{\text{avg}} := \frac{1}{n} \sum_{x \in V} k(x) \).
<table>
<thead>
<tr>
<th>Network</th>
<th>$k_{max}$</th>
<th>$k_{avg}$</th>
<th>% of vertices with $k(x) = 0$</th>
<th>% of vertices with $k(x) = 1$</th>
<th>% of vertices with $k(x) = 2$</th>
<th>% of vertices with $k(x) = 3$</th>
<th>% of vertices with $k(x) = 4$</th>
<th>% of vertices with $k(x) \geq 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUTCH-ELITE</td>
<td>6</td>
<td>2.431</td>
<td>16.1</td>
<td>0</td>
<td>47.1</td>
<td>0</td>
<td>35.9</td>
<td>0</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>3</td>
<td>0.704</td>
<td>43.6</td>
<td>42.5</td>
<td>13.8</td>
<td>0.1</td>
<td>16.1</td>
<td>0</td>
</tr>
<tr>
<td>Eva</td>
<td>2</td>
<td>0.572</td>
<td>47.6</td>
<td>47.6</td>
<td>4.8</td>
<td>1</td>
<td>68.7</td>
<td>17.2</td>
</tr>
<tr>
<td>SLASHDOT</td>
<td>3</td>
<td>1.88</td>
<td>0.1</td>
<td>17.7</td>
<td>76.2</td>
<td>$\approx 6$</td>
<td>68.6</td>
<td>8.3</td>
</tr>
<tr>
<td>Loans</td>
<td>3</td>
<td>2.031</td>
<td>0.1</td>
<td>14</td>
<td>68.7</td>
<td>17.2</td>
<td>68.6</td>
<td>8.3</td>
</tr>
<tr>
<td>Twitter</td>
<td>3</td>
<td>1.821</td>
<td>3.1</td>
<td>$\approx 20$</td>
<td>68.6</td>
<td>8.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Email-Virgili</td>
<td>4</td>
<td>1.932</td>
<td>4.3</td>
<td>22.8</td>
<td>48.4</td>
<td>24.4</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>Email-Enron</td>
<td>2</td>
<td>0.903</td>
<td>22.4</td>
<td>64.8</td>
<td>12.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Email-EU</td>
<td>2</td>
<td>0.002</td>
<td>99.9</td>
<td>0.03</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wikitalk-China</td>
<td>3</td>
<td>1.791</td>
<td>$\approx 0$</td>
<td>21</td>
<td>79</td>
<td>0.008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CE-METABOLIC</td>
<td>1</td>
<td>0.349</td>
<td>65.1</td>
<td>34.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC-PPI</td>
<td>7</td>
<td>4.196</td>
<td>1.3</td>
<td>4.1</td>
<td>6.2</td>
<td>13.4</td>
<td>27.2</td>
<td>35.9</td>
</tr>
<tr>
<td>Yeast-PPI</td>
<td>4</td>
<td>2.558</td>
<td>0.7</td>
<td>5.9</td>
<td>$\approx 36$</td>
<td>51.7</td>
<td>5.7</td>
<td></td>
</tr>
<tr>
<td>Homo-Pi</td>
<td>2</td>
<td>0.612</td>
<td>41.6</td>
<td>55.5</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-GRAPH-1</td>
<td>2</td>
<td>0.887</td>
<td>19.6</td>
<td>72.2</td>
<td>8.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-GRAPH-2</td>
<td>2</td>
<td>0.833</td>
<td>25.7</td>
<td>65.3</td>
<td>$\approx 9$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-GRAPH-3</td>
<td>2</td>
<td>0.312</td>
<td>70.4</td>
<td>28</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RouteView</td>
<td>2</td>
<td>0.329</td>
<td>69.7</td>
<td>27.6</td>
<td>2.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-Caida</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITDK</td>
<td>5</td>
<td>2.702</td>
<td>0.3</td>
<td>3.4</td>
<td>28.6</td>
<td>61.4</td>
<td>6.3</td>
<td>$\approx 0$</td>
</tr>
<tr>
<td>Gnutaella-06</td>
<td>5</td>
<td>3.543</td>
<td>0.01</td>
<td>0.7</td>
<td>5.9</td>
<td>37.2</td>
<td>50.9</td>
<td>5.3</td>
</tr>
<tr>
<td>Gnutaella-24</td>
<td>6</td>
<td>4.475</td>
<td>0.02</td>
<td>0.1</td>
<td>0.7</td>
<td>8.6</td>
<td>38.3</td>
<td>46.5</td>
</tr>
<tr>
<td>Gnutaella-30</td>
<td>6</td>
<td>4.034</td>
<td>0.02</td>
<td>0.2</td>
<td>2.6</td>
<td>16.4</td>
<td>54.8</td>
<td>25.1</td>
</tr>
<tr>
<td>Gnutaella-31</td>
<td>6</td>
<td>4.251</td>
<td>0.01</td>
<td>0.1</td>
<td>1.3</td>
<td>11.6</td>
<td>48.4</td>
<td>37.8</td>
</tr>
<tr>
<td>Web-Stanford</td>
<td>28</td>
<td>0.006</td>
<td>99.9</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>0.04</td>
<td>$\approx 0$</td>
<td>0.02</td>
</tr>
<tr>
<td>Web-Notredam</td>
<td>2</td>
<td>0.935</td>
<td>7.1</td>
<td>92.3</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web-BerkStan</td>
<td>22</td>
<td>0.002</td>
<td>99.97</td>
<td>$\approx 0$</td>
<td>$\approx 0$</td>
<td>0.02</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>Amazon-1</td>
<td>7</td>
<td>0.919</td>
<td>49.7</td>
<td>21.7</td>
<td>18.6</td>
<td>8.1</td>
<td>1.1</td>
<td>0.4</td>
</tr>
<tr>
<td>Amazon-2</td>
<td>5</td>
<td>2.03</td>
<td>1.2</td>
<td>15.1</td>
<td>65</td>
<td>17.1</td>
<td>1.6</td>
<td>$\approx 0$</td>
</tr>
<tr>
<td>Road-euro</td>
<td>8</td>
<td>0.135</td>
<td>97.4</td>
<td>0.3</td>
<td>0.1</td>
<td>0.4</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td>Openflight</td>
<td>2</td>
<td>0.641</td>
<td>36.1</td>
<td>63.7</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power-grid</td>
<td>4</td>
<td>1.469</td>
<td>46.3</td>
<td>13.1</td>
<td>12.6</td>
<td>9.1</td>
<td>18.8</td>
<td></td>
</tr>
<tr>
<td>Road-pa</td>
<td>80</td>
<td>22.545</td>
<td>0.7</td>
<td>20.9</td>
<td>0.3</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 6.5: Distribution of distortion values $k(x) = ecc_{T_2}(x) - ecc_G(x)$, $x \in V$. $k_{max} := \max_{x \in V} k(x)$. $k_{avg} := \frac{1}{n} \sum_{x \in V} k(x)$. 

130
than $k_{\text{max}}$ for all graphs. In fact, $k_{\text{avg}} < 3$ in all but five graphs (\textsc{gnutella-30}, \textsc{gnutella-31}, \textsc{amazon-2}, \textsc{power-grid}, and \textsc{road-pa}). In graphs with high $k_{\text{max}}$, close inspection reveals that only small percent of vertices achieve this maximum. For example, in graph \textsc{web-stanford}, $k_{\text{max}} = 28$ was only achieved by 17 vertices. The distributions of the values of $k(v) := \text{ecc}_{T_1}(v) - \text{ecc}_G(v)$ of all graphs are listed in Table 6.3.

Similar experiments were performed following Proposition 9. For each graph $G$ in our datasets, we picked a random vertex $u \in V$ and a random vertex $v \in F(u)$. Then, we identified in a randomly picked $(u,v)$-geodesic a vertex $w$ at distance $\text{rad}(G)$ from $v$. We did not consider a vertex $c$ defined in Proposition 9(d) since, for majority of graphs in our datasets, $c$ will be a middle vertex of a geodesic between two mutually distant vertices, and working with $c$ we will duplicate previous experiments. Recall that for majority of our graphs (as found in our experiments) two BFS sweeps already identify a pair of mutually distant vertices. We know from Proposition 9 that $\text{ecc}_G(v) \geq \text{diam}(G) - 2\delta \geq 2\text{rad}(G) - 4\delta - 1$ and $\text{ecc}_G(w) \leq \text{rad}(G) + \delta$. Our experimental results are better than these theoretical bounds. In Table 6.4, we list eccentricities of $v$ and $w$ for each graph. In almost all graphs, the eccentricity of $v$ is equal to the diameter $\text{diam}(G)$. Only four graphs have $\text{ecc}_G(v) = \text{diam}(G) - 1$ and one graph (\textsc{road-pa}) has $\text{ecc}_G(v) < \text{diam}(G) - 1$. Vertex $w$ is central for 21 graphs, has eccentricity equal to $\text{rad}(G) + 1$ for 10 graphs, has eccentricity equal to $\text{rad}(G) + 2$ for one graph, and only for one remaining graph (\textsc{road-pa} network, which has large hyperbolicity) its eccentricity is equal to $\text{rad}(G) + 15$. It turns out also (see columns five and six of Table 6.4) that vertex $w$ either belongs to the center $C(G)$ or is very close to the center. The only exception is again \textsc{road-pa} network where $2\text{rad}(G) - \text{ecc}_G(w) = 32$ and $d(w, C(G)) = 21$.

For every graph $G = (V, E)$ in our datasets, we constructed also an arbitrary $BFS(w)$-tree $T_2 = (V, E')$, rooted at vertex $w$, and analyzed how well $T_2$ preserves or approximates
the eccentricities of vertices in \( G \). The value of \( k_{\text{max}} \) is at most five for 23 graphs. The average distortion \( k_{\text{avg}} \) is much smaller than \( k_{\text{max}} \) in all graphs. The distributions of the values of \( k(x) \) for all graphs are presented in Table 6.5.

<table>
<thead>
<tr>
<th>Network</th>
<th>( \text{diam}(G) )</th>
<th>( \text{diam}(T_1) )</th>
<th>( k_{\text{max}}^{T_1} )</th>
<th>( k_{\text{avg}}^{T_1} )</th>
<th>( \text{diam}(T_2) )</th>
<th>( k_{\text{max}}^{T_2} )</th>
<th>( k_{\text{avg}}^{T_2} )</th>
<th>( \text{diam}(T_3) )</th>
<th>( k_{\text{max}}^{T_3} )</th>
<th>( k_{\text{avg}}^{T_3} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUTCH-ELITE</td>
<td>22</td>
<td>24</td>
<td>6</td>
<td>2.35</td>
<td>24</td>
<td>6</td>
<td>2.431</td>
<td>24</td>
<td>6</td>
<td>2.083</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>8</td>
<td>8</td>
<td>2</td>
<td>0.686</td>
<td>9</td>
<td>3</td>
<td>0.704</td>
<td>8</td>
<td>2</td>
<td>0.686</td>
</tr>
<tr>
<td>EVA</td>
<td>18</td>
<td>19</td>
<td>2</td>
<td>0.571</td>
<td>19</td>
<td>2</td>
<td>0.572</td>
<td>19</td>
<td>2</td>
<td>0.571</td>
</tr>
<tr>
<td>SLASHDOT</td>
<td>12</td>
<td>14</td>
<td>3</td>
<td>1.777</td>
<td>14</td>
<td>3</td>
<td>1.88</td>
<td>12</td>
<td>2</td>
<td>0.701</td>
</tr>
<tr>
<td>LOANS</td>
<td>8</td>
<td>10</td>
<td>3</td>
<td>2.06</td>
<td>10</td>
<td>3</td>
<td>2.031</td>
<td>10</td>
<td>3</td>
<td>2.081</td>
</tr>
<tr>
<td>TWITTER</td>
<td>8</td>
<td>11</td>
<td>4</td>
<td>2.569</td>
<td>10</td>
<td>3</td>
<td>1.821</td>
<td>10</td>
<td>4</td>
<td>1.856</td>
</tr>
<tr>
<td>EMAIL-VIRGILI</td>
<td>8</td>
<td>11</td>
<td>4</td>
<td>2.729</td>
<td>10</td>
<td>4</td>
<td>1.932</td>
<td>10</td>
<td>4</td>
<td>1.906</td>
</tr>
<tr>
<td>EMAIL-ENRON</td>
<td>13</td>
<td>13</td>
<td>2</td>
<td>0.906</td>
<td>14</td>
<td>2</td>
<td>0.903</td>
<td>14</td>
<td>2</td>
<td>1.735</td>
</tr>
<tr>
<td>EMAIL-EU</td>
<td>14</td>
<td>14</td>
<td>2</td>
<td>0.002</td>
<td>14</td>
<td>2</td>
<td>0.002</td>
<td>14</td>
<td>2</td>
<td>0.002</td>
</tr>
<tr>
<td>WIKITALK-CHINA</td>
<td>8</td>
<td>9</td>
<td>3</td>
<td>2.076</td>
<td>9</td>
<td>3</td>
<td>1.791</td>
<td>8</td>
<td>2</td>
<td>0.777</td>
</tr>
<tr>
<td>CE-METABOLIC</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>1.982</td>
<td>8</td>
<td>1</td>
<td>0.349</td>
<td>8</td>
<td>2</td>
<td>1.185</td>
</tr>
<tr>
<td>SC-PP1</td>
<td>19</td>
<td>20</td>
<td>3</td>
<td>0.981</td>
<td>23</td>
<td>7</td>
<td>4.196</td>
<td>22</td>
<td>6</td>
<td>3.163</td>
</tr>
<tr>
<td>YEAST-PP1</td>
<td>11</td>
<td>12</td>
<td>3</td>
<td>1.872</td>
<td>13</td>
<td>4</td>
<td>2.558</td>
<td>12</td>
<td>3</td>
<td>1.872</td>
</tr>
<tr>
<td>HOMO-PI</td>
<td>10</td>
<td>10</td>
<td>2</td>
<td>0.747</td>
<td>10</td>
<td>2</td>
<td>0.612</td>
<td>10</td>
<td>2</td>
<td>0.747</td>
</tr>
<tr>
<td>AS-GRAH-1</td>
<td>9</td>
<td>11</td>
<td>3</td>
<td>1.791</td>
<td>10</td>
<td>2</td>
<td>0.887</td>
<td>10</td>
<td>2</td>
<td>0.886</td>
</tr>
<tr>
<td>AS-GRAH-2</td>
<td>11</td>
<td>11</td>
<td>3</td>
<td>1.124</td>
<td>11</td>
<td>2</td>
<td>0.833</td>
<td>12</td>
<td>3</td>
<td>1.272</td>
</tr>
<tr>
<td>AS-GRAH-3</td>
<td>9</td>
<td>10</td>
<td>2</td>
<td>0.828</td>
<td>10</td>
<td>2</td>
<td>0.312</td>
<td>10</td>
<td>2</td>
<td>0.312</td>
</tr>
<tr>
<td>ROUTEVIEW</td>
<td>10</td>
<td>10</td>
<td>2</td>
<td>0.329</td>
<td>10</td>
<td>2</td>
<td>0.329</td>
<td>10</td>
<td>2</td>
<td>0.329</td>
</tr>
<tr>
<td>AS-CAIDA</td>
<td>17</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ITDK</td>
<td>26</td>
<td>29</td>
<td>4</td>
<td>2.108</td>
<td>29</td>
<td>5</td>
<td>2.702</td>
<td>28</td>
<td>3</td>
<td>1.385</td>
</tr>
<tr>
<td>GNUTELLA-06</td>
<td>10</td>
<td>12</td>
<td>4</td>
<td>2.507</td>
<td>13</td>
<td>5</td>
<td>3.543</td>
<td>12</td>
<td>4</td>
<td>2.507</td>
</tr>
<tr>
<td>GNUTELLA-24</td>
<td>11</td>
<td>14</td>
<td>5</td>
<td>2.697</td>
<td>16</td>
<td>6</td>
<td>4.475</td>
<td>12</td>
<td>3</td>
<td>0.863</td>
</tr>
<tr>
<td>GNUTELLA-30</td>
<td>11</td>
<td>14</td>
<td>5</td>
<td>3.167</td>
<td>16</td>
<td>6</td>
<td>4.034</td>
<td>14</td>
<td>5</td>
<td>3.295</td>
</tr>
<tr>
<td>GNUTELLA-31</td>
<td>11</td>
<td>16</td>
<td>6</td>
<td>4.176</td>
<td>16</td>
<td>6</td>
<td>4.251</td>
<td>14</td>
<td>5</td>
<td>2.669</td>
</tr>
<tr>
<td>WEB-STANFORD</td>
<td>164</td>
<td>164</td>
<td>28</td>
<td>0.006</td>
<td>164</td>
<td>28</td>
<td>0.006</td>
<td>164</td>
<td>28</td>
<td>0.006</td>
</tr>
<tr>
<td>WEB-NOTREDA</td>
<td>46</td>
<td>46</td>
<td>2</td>
<td>0.353</td>
<td>46</td>
<td>2</td>
<td>0.353</td>
<td>46</td>
<td>2</td>
<td>0.017</td>
</tr>
<tr>
<td>WEB-BERKSTAN</td>
<td>208</td>
<td>208</td>
<td>22</td>
<td>0.002</td>
<td>208</td>
<td>22</td>
<td>0.002</td>
<td>208</td>
<td>22</td>
<td>0.002</td>
</tr>
<tr>
<td>AMAZON-1</td>
<td>47</td>
<td>47</td>
<td>6</td>
<td>0.991</td>
<td>48</td>
<td>7</td>
<td>0.919</td>
<td>47</td>
<td>7</td>
<td>1.205</td>
</tr>
<tr>
<td>AMAZON-2</td>
<td>20</td>
<td>23</td>
<td>6</td>
<td>3.735</td>
<td>22</td>
<td>5</td>
<td>2.03</td>
<td>22</td>
<td>4</td>
<td>1.274</td>
</tr>
<tr>
<td>ROAD-EURO</td>
<td>62</td>
<td>62</td>
<td>8</td>
<td>0.135</td>
<td>62</td>
<td>8</td>
<td>0.135</td>
<td>62</td>
<td>8</td>
<td>0.135</td>
</tr>
<tr>
<td>OPENFLIGHT</td>
<td>13</td>
<td>15</td>
<td>3</td>
<td>1.879</td>
<td>14</td>
<td>2</td>
<td>0.641</td>
<td>14</td>
<td>2</td>
<td>0.704</td>
</tr>
<tr>
<td>POWER-GRID</td>
<td>46</td>
<td>51</td>
<td>13</td>
<td>5.735</td>
<td>46</td>
<td>4</td>
<td>1.409</td>
<td>46</td>
<td>4</td>
<td>1.409</td>
</tr>
<tr>
<td>ROAD-PA</td>
<td>794</td>
<td>814</td>
<td>98</td>
<td>23.339</td>
<td>830</td>
<td>80</td>
<td>22.545</td>
<td>803</td>
<td>46</td>
<td>10.64</td>
</tr>
</tbody>
</table>

Table 6.6: Comparison of the three proposed eccentricity approximating BFS-trees \( T_1, T_2 \) and \( T_3 \). \( T_3 \) is a \( BFS(c^*) \)-tree rooted at a randomly picked central vertex \( c^* \in C(G) \).

In Table 6.6, we compare these two eccentricity approximating spanning trees \( T_1 \) and
$T_2$ with each other and with a third $BFS(c^*)$-tree $T_3$ which we have constructed starting from a randomly chosen central vertex $c^* \in C(G)$.

For each graph in the datasets, three values of $k_{\text{max}}$ ($k_{\text{max}}^{T_1}$, $k_{\text{max}}^{T_2}$ and $k_{\text{max}}^{T_3}$) and three values of $k_{\text{avg}}$ ($k_{\text{avg}}^{T_1}$, $k_{\text{avg}}^{T_2}$ and $k_{\text{avg}}^{T_3}$) are listed. We observe that the smallest $k_{\text{max}}$ (out of three) is achieved by tree $T_3$ in 28 graphs, by tree $T_2$ in 20 graphs and by tree $T_1$ in 20 graphs (in 14 graphs, the smallest $k_{\text{max}}$ is achieved by all three trees). The difference between the largest and the smallest $k_{\text{max}}$ of a graph is at most one for 26 graphs in the datasets. The largest difference is observed for ROAD-PA network: the largest $k_{\text{max}}$ (98) is given by tree $T_1$, the smallest $k_{\text{max}}$ (46) is given by tree $T_3$. Two other graphs have the difference larger than three: for SC-PP1 network, the largest $k_{\text{max}}$ (7) is given by tree $T_2$, the smallest $k_{\text{max}}$ (3) is given by tree $T_1$; for POWER-GRID network, the largest $k_{\text{max}}$ (13) is given by tree $T_1$, the smallest $k_{\text{max}}$ (4) is shared by remaining trees $T_2$, $T_3$. Overall, we conclude that $k_{\text{max}}$ values for trees $T_1$ and $T_2$ are comparable and generally can be slightly worse than those for tree $T_3$. Similar observations hold also for the average distortion $k_{\text{avg}}$. Note, however, that for construction of trees $T_2$ and $T_3$ one needs to know $\text{rad}(G)$ or a central vertex of $G$, which are unlikely to be computable in subquadratic time (see Statement 2).
Chapter 7

Eccentricity Approximating Trees in \((\alpha_1, \Delta)\)-Metric Graphs

Using the characteristic property of chordal graphs that they are the intersection graphs of subtrees of a tree, Erich Prisner showed that every chordal graph admits an eccentricity 2-approximating spanning tree. I.e., every chordal graph \(G\) has a spanning tree \(T\) such that \(ecc_T(v) - ecc_G(v) \leq 2\) for every vertex \(v\), where \(ecc_G(v)\) and \(ecc_T(v)\) are the eccentricities of vertex \(v\) in \(G\) and in \(T\) respectively. Using only metric properties of graphs, we extend that result to a much larger family of graphs containing among others chordal graphs, the underlying graphs of 7-systolic complexes and plane triangulations with inner vertices of degree at least 7. Furthermore, based on our approach, we propose two heuristics for constructing eccentricity \(k\)-approximating trees with small values of \(k\) for general unweighted graphs. We validate those heuristics on a set of real-world networks and demonstrate that all those networks have very good eccentricity approximating trees. Some proofs are omitted in this dissertation and can be found in [83] and [84].

The results in this chapter were obtained in collaboration with Feodor Dragan and
Ekkehard Köhler. This chapter is based on:


7.1 Introduction

A spanning tree $T$ of a graph $G$ with $d_T(u,v) - d_G(u,v) \leq k$, for all $u,v \in V$, is known as an additive tree $k$-spanner of $G$ [135] and, if it exists for a small integer $k$, then it gives a good approximation of all distances in $G$ by the distances in $T$. Many optimization problems involving distances in graphs are known to be NP-hard in general but have efficient solutions in simpler metric spaces, with well-understood metric structures, including trees. A solution to such an optimization problem obtained for a tree spanner $T$ of $G$ usually serves as a good approximate solution to the problem in $G$.

E. Prisner in [162] introduced the new notion of eccentricity approximating spanning trees. A spanning tree $T$ of a graph $G$ is called an eccentricity $k$-approximating spanning tree if $ecc_T(v) - ecc_G(v) \leq k$ holds for all $v \in V$. Such a tree tries to approximately preserve only distances from each vertex $v$ to its most distant vertices and can tolerate larger increases to nearby vertices. They are important in applications where vertices measure their degree of centrality by means of their eccentricity and would tolerate a small surplus to the actual eccentricities [162]. Note also that Nandakumar and Parthasarasthy considered in [154] eccentricity-preserving spanning trees (i.e., eccentricity 0-approximating spanning trees) and showed that a graph $G$ has an eccentricity 0-approximating spanning tree if and only if: (a) either $diam(G) = 2rad(G)$ and $|C(G)| = 1$, or $diam(G) = 2rad(G) - 1$, $|C(G)| = 2$, and those two center vertices are adjacent; (b) every vertex $u \in V \setminus C(G)$ has a neighbor $v$ such that $ecc_G(v) < ecc_G(u)$.

Every additive tree $k$-spanner is clearly eccentricity $k$-approximating. Therefore, eccentricity $k$-approximating spanning trees can be found in every interval graph for $k = 2$ [135, 148, 161], and in every asteroidal-triple–free graph [135], strongly chordal graph [41] and dually chordal graph [41] for $k = 3$. On the other hand, although for every $k$ there is a
chordal graph without a tree $k$-spanner [135, 161], yet as Prisner demonstrated in [162], every chordal graph has an eccentricity 2-approximating spanning tree, i.e., with the slightly weaker concept of eccentricity-approximation, one can be successful even for chordal graphs.

Unfortunately, the method used by Prisner in [162] heavily relies on a characteristic property of chordal graphs (chordal graphs are exactly the intersection graphs of subtrees of a tree) and is hardly extendable to larger families of graphs.

In this chapter we present a new proof of the result of [162] using only metric properties of chordal graphs. This allows us to extend the result to a much larger family of graphs which includes not only chordal graphs but also other families of graphs known from the literature.

It is known [52, 197] that every chordal graph satisfies the following two metric properties:

**$\alpha_1$-metric:** if $v \in I(u,w)$ and $w \in I(v,x)$ are adjacent, then $d_G(u,x) \geq d_G(u,v) + d_G(v,x) - 1 = d_G(u,v) + d_G(w,x)$.

**Triangle condition:** for any three vertices $u,v,w$ with $1 = d_G(v,w) < d_G(u,v) = d_G(u,w)$ there exists a common neighbor $x$ of $v$ and $w$ such that $d_G(u,x) = d_G(u,v) - 1$.

A graph $G$ satisfying the $\alpha_1$-metric property is called an $\alpha_1$-metric graph. If an $\alpha_1$-metric graph $G$ satisfies also the triangle condition then $G$ is called an $(\alpha_1, \Delta)$-metric graph. We prove that every $(\alpha_1, \Delta)$-metric graph $G = (V,E)$ has an eccentricity 2-approximating spanning tree and that such a tree can be constructed in $O(|V||E|)$ total time. As a consequence, we get that the underlying graph of every 7-systolic complex (and, hence, every plane triangulation with inner vertices of degree at least 7 and every chordal graph) has an eccentricity 2-approximating spanning tree. Figure 7.1 presents a taxonomy of

---

1A more general concept of $\alpha_i$-metric was introduced in [197], however, in this chapter are interested only in the case when $i = 1$. 137
graphs used in this chapter. Note that in this case, the exact value of \( \delta \)-hyperbolicity is unknown for \( \alpha_1 \)-metric graphs.

The rest of this chapter is organized as follows. In Section 7.2, we present additional notions and notations and some auxiliary results. In Section 7.3, some useful properties of the eccentricity function on \((\alpha_1, \Delta)\)-metric graphs are described. Our eccentricity approximating spanning tree is constructed and analyzed in Section 7.4. In Section 7.5, the algorithm for the construction of an eccentricity approximating spanning tree developed in Section 7.4 for \((\alpha_1, \Delta)\)-metric graphs is generalized and validated on some real-world networks. Our experiments show that all those real-world networks have very good eccentricity approximating trees. Section 7.6 concludes the chapter with a few open questions.

![Figure 7.1: The relationships between the different graph types used in this chapter.](image)

### 7.2 Preliminaries

We denote by \( C_k \) an *induced cycle* of length \( k \) by \( C_k \) (i.e., it has \( k \) vertices) and by \( W_k \) an *induced wheel* of size \( k \) which is a \( C_k \) with one extra vertex universal to \( C_k \). For a vertex \( v \)
of $G$, $N_G(v) = \{u \in V : uv \in E\}$ is called the open neighborhood, and $N_G[v] = N_G(v) \cup \{v\}$ the closed neighborhood of $v$. The distance between a vertex $v$ and a set $S \subseteq V$ is defined as $d_G(v,S) = \min_{u \in S} d_G(u,v)$ and the set of furthest (most distant) vertices from $v$ is denoted by $F(v) = \{u \in V : d_G(u,v) = ecc_G(v)\}$.

An induced subgraph of $G$ (or the corresponding vertex set $A$) is called convex if for each pair of vertices $u, v \in A$ it includes the interval $I(v,u)$ of $G$ between $u, v$. An induced subgraph $H$ of $G$ is called isometric if the distance between any pair of vertices in $H$ is the same as their distance in $G$. In particular, convex subgraphs are isometric. The disk $D(x,r)$ with center $x$ and radius $r \geq 0$ consists of all vertices of $G$ at distance at most $r$ from $x$. In particular, the unit disk $D(x,1) = N[x]$ comprises $x$ and the neighborhood $N(x)$. For an edge $e = xy$ of a graph $G$, let $D(e,r) := D(x,r) \cup D(y,r)$.

By the definition of $\alpha_1$-metric graphs clearly, such a graph cannot contain any isometric cycles of length $k > 5$ and any induced cycle of length 4. The following results characterize $\alpha_1$-metric graphs and the class of chordal graphs within the class of $\alpha_1$-metric graphs. Recall that a graph is chordal if all its induced cycles are of length 3.

**Theorem 4 ([197]).** $G$ is a chordal graph if and only if it is an $\alpha_1$-metric graph not containing any induced subgraphs isomorphic to cycle $C_5$ and wheel $W_k$, $k \geq 5$.

**Theorem 5 ([197]).** $G$ is an $\alpha_1$-metric graph if and only if all disks $D(v,k)$ ($v \in V$, $k \geq 1$) of $G$ are convex and $G$ does not contain the graph $W_6^{++}$ (see Figure 7.2) as an isometric subgraph.

![Figure 7.2: Forbidden isometric subgraph $W_6^{++}$.](image-url)
**Theorem 6** ([96, 179]). All disks $D(v, k)$ ($v \in V$, $k \geq 1$) of a graph $G$ are convex if and only if $G$ does not contain isometric cycles of length $k > 5$, and for any two vertices $x, y$ the neighbors of $x$ in the interval $I(x, y)$ are pairwise adjacent.

A graph $G$ is called a *bridged graph* if all isometric cycles of $G$ have length three [96]. The class of bridged graphs is a natural generalization of the class of chordal graphs. They can be characterized in the following way.

**Theorem 7** ([96, 179]). $G = (V, E)$ is a bridged graph if and only if the disks $D(v, k)$ and $D(e, k)$ are convex for all $v \in V$, $e \in E$, and $k \geq 1$.

As a consequence of Theorem 5, Theorem 6 and Theorem 7 we obtain the following equivalences.

**Lemma 10.** For a graph $G = (V, E)$ the following statements are equivalent:

(a) $G$ is an $\alpha_1$-metric graph not containing an induced $C_5$;

(b) $G$ is a bridged graph not containing $W_6^{++}$ as an isometric subgraph;

(c) The disks $D(v, k)$ and $D(e, k)$ of $G$ are convex for all $v \in V$, $e \in E$, and $k \geq 1$, and $G$ does not contain $W_6^{++}$ as an isometric subgraph.

**Proof.** By Theorem 5, if $G$ is an $\alpha_1$-metric graph then all disks $D(v, k)$ ($v \in V$, $k \geq 1$) of $G$ are convex and $G$ does not contain the graph $W_6^{++}$ as an isometric subgraph. If, in addition, $G$ does not contain induced subgraphs isomorphic to $C_5$ then, by Theorem 6, $G$ is a bridged graph. Hence, (a) implies (b). Also, (b) implies (c), by Theorem 7, and (c) implies (a), by Theorem 5 and since a graph where $D(e, 1)$ is convex for each $e \in E$ cannot contain an induced $C_5$.

\[ \square \]
As we will show now the class of \((\alpha_1, \Delta)\)-metric graphs contains all graphs described in Lemma 10. An induced \(C_5\) is called \textit{suspended} in \(G\) if there is a vertex in \(G\) which is adjacent to all vertices of the \(C_5\).

**Theorem 8.** A graph \(G\) is \((\alpha_1, \Delta)\)-metric if and only if it is an \(\alpha_1\)-metric graph where for each induced \(C_5\) there is a vertex \(v \in V\) such that \(C_5 \subseteq N(v)\), i.e., every induced \(C_5\) is suspended.

We will also need the following fact.

**Lemma 11.** Let \(G = (V, E)\) be an \((\alpha_1, \Delta)\)-metric graph, let \(K\) be a complete subgraph of \(G\), and let \(v\) be a vertex of \(G\). If for every vertex \(z \in K\), \(d(z, v) = k\) holds, then there is a vertex \(v'\) at distance \(k - 1\) from \(v\) which is adjacent to every vertex of \(K\).

We note here, without going into the rich theory of systolic complexes, that the underlying graph of any 7-systolic complex is exactly a bridged graph not containing a 6-wheel \(W_6\) as an induced (equivalently, isometric) subgraph (see [55] for this fact and a relation of 7-systolic complexes with \(\text{CAT}(0)\) complexes). Hence, the class of \((\alpha_1, \Delta)\)-metric graphs contains the underlying graphs of 7-systolic complexes and hence all plane triangulations with inner vertices of degree at least 7 [55] (vertices that are not on the outerface are called inner vertices).

### 7.3 Eccentricity function on \((\alpha_1, \Delta)\)-metric graphs

In what follows, by \(C(G)\) we denote not only the set of all central vertices of \(G\) but also the subgraph of \(G\) induced by this set. We say that the eccentricity function \(ecc_G(v)\) on \(G\) is \textit{unimodal} if every vertex \(u \in V \setminus C(G)\) has a neighbor \(v\) such that \(ecc_G(v) < ecc_G(u)\). In other words, every local minimum of the eccentricity function \(ecc_G(v)\) is a global minimum.
on $G$. It this section we will often omit subindex $G$ since we deal only with a graph $G$ here. A spanning tree $T$ of $G$ will be built only in the next section.

In this section, we will show that the eccentricity function $eccc_G(v)$ on an $(\alpha, \Delta)$-metric graph $G$ is almost unimodal and that the radius of the center $C(G)$ of $G$ is at most 2.

**Lemma 12.** Let $G$ be an $\alpha_1$-metric graph and $x$ be its arbitrary vertex with $ecc(x) \geq \text{rad}(G) + 1$. Then, for every vertex $z \in F(x)$ and every neighbor $v$ of $x$ in $I(x, z)$, $ecc(v) \leq ecc(x)$ holds.

*Proof.* Assume, by way of contradiction, that $ecc(v) > ecc(x)$ and consider an arbitrary vertex $u \in F(v)$. Since $x$ and $v$ are adjacent, necessarily, $d(v, u) = ecc(v) = ecc(x) + 1 = d(u, x) + 1$, i.e., $x \in I(v, u)$. By the $\alpha_1$-metric property,

$$d(u, z) \geq d(u, x) + d(v, z) = ecc(v) - 1 + ecc(x) - 1 = 2ecc(x) - 1 \geq 2\text{rad}(G) + 1.$$ 

The latter gives a contradiction to $d(u, z) \leq \text{diam}(G) \leq 2\text{rad}(G)$.

**Theorem 9.** Let $G$ be an $(\alpha_1, \Delta)$-metric graph and $x$ be an arbitrary vertex of $G$. If

(i) $ecc(x) > \text{rad}(G) + 1$ or

(ii) $ecc(x) = \text{rad}(G) + 1$ and $\text{diam}(G) < 2\text{rad}(G)$,

then there must exist a neighbor $v$ of $x$ with $ecc(v) < ecc(x)$.

*Proof.* For every neighbor $v$ of $x$, we define the set $S_v$ as the most distant vertices from $x$ which have $v$ on their shortest path from $x$. Formally,

$$S_v := \{z \in F(x) : v \in I(x, z)\}.$$
Choose a neighbor $v$ of $x$ which maximizes $|S_v|$. We claim that $ecc(v) < ecc(x)$. We know, by Lemma 12, that $ecc(v) \leq ecc(x)$. Assume $ecc(v) = ecc(x)$ and consider an arbitrary vertex $u \in F(v)$.

Suppose first that $x \in I(v, u)$. Then, $d(u, z) \geq d(u, x) + d(v, z) = 2ecc(x) - 2$ holds for every $z \in S_v$ by the $\alpha_1$-metric property. Hence, if $ecc(x) > rad(G) + 1$ then $d(u, z) > 2rad(G)$ and thus a contradiction to $d(u, z) \leq diam(G) \leq 2rad(G)$ arises. If, on the other hand, case (ii) applies, i.e., $ecc(x) = rad(G) + 1$ and $diam(G) < 2rad(G)$, then it follows that $d(u, z) > 2rad(G) > diam(G)$ and again a contradiction arises.

Now consider the case that $x \notin I(v, u)$. Then $ecc(v) = ecc(x)$ implies that $d(u, x) = d(u, v)$ and $u \in F(x)$. By the triangle condition, there must exist a common neighbor $w$ of $x$ and $v$ such that $w \in I(x, u) \cap I(v, u)$. Since $u$ belongs to $S_w$ but not to $S_v$, then, by the maximality of $|S_v|$, there must exist a vertex $z \in F(x)$ which is in $S_v$ but not in $S_w$. Thus, $d(w, z) > d(v, z)$ and $v \in I(w, z)$ must hold. Now, the $\alpha_1$-metric property applied to $v \in I(w, z)$ and $w \in I(v, u)$ gives $d(u, z) \geq d(u, w) + d(v, z) = 2ecc(x) - 2$.

As before we get $d(u, z) > 2rad(G) \geq diam(G)$, if $ecc(x) > rad(G) + 1$ (case (i)), and $d(u, z) \geq 2rad(G) > diam(G)$, if $ecc(x) = rad(G) + 1$ and $diam(G) < 2rad(G)$ (case (ii)). These contradictions complete the proof. ■

Note that the requirement in Theorem 9 that $G$ satisfies the triangle condition cannot be removed. The statement is not true for arbitrary $\alpha_1$-metric graphs (see Figure 7.3).

![Figure 7.3: An $\alpha_1$-metric graph $G$ with $rad(G) = 2$, $diam(G) = 3 < 2rad(G)$, and with a vertex of eccentricity $3 = rad(G) + 1$ that has no neighbor with smaller eccentricity. The numbers next to vertices show their eccentricities.](image-url)
For each vertex $v \in V \setminus C(G)$ of a graph $G$ we can define a parameter

$$loc(v) = \min\{d(v,x) : x \in V, ecc(x) < ecc(v)\}$$

and call it the locality of $v$. We define the locality of any vertex from $C(G)$ to be 1.

Theorem 9 says that if a vertex $v$ with $loc(v) > 1$ exists in an $(\alpha_1, \Delta)$-metric graph $G$ then $diam(G) = 2rad(G)$ and $ecc(v) = rad(G) + 1$. I.e., only in the case that $diam(G) = 2rad(G)$ the eccentricity function can be not unimodal on $G$.

Observe that the center $C(G)$ of a graph $G = (V,E)$ can be represented as the intersection of all the disks of $G$ of radius $rad(G)$, i.e., $C(G) = \bigcap\{D(v,rad(G)) : v \in V\}$. Consequently, the center $C(G)$ of an $\alpha_1$-metric graph $G$ is convex (in particular, it is connected), as the intersection of convex sets is always a convex set. In general, any set $C_{\leq i}(G) := \{z \in V : ecc(z) \leq rad(G) + i\}$ is a convex set of $G$ as $C_{\leq i}(G) = \bigcap\{D(v,rad(G) + i) : v \in V\}$.

**Corollary 3.** In an $\alpha_1$-metric graph $G$, all sets $C_{\leq i}(G), i \in \{0, \ldots, diam(G) - rad(G)\}$, are convex. In particular, the center $C(G)$ of an $\alpha_1$-metric graph $G$ is convex.

The following result gives bounds on the diameter and the radius of the center of an $(\alpha_1, \Delta)$-metric graph. Previously it was known that the diameter (the radius) of the center of a chordal graph is at most 3 (at most 2, respectively) [53].

**Theorem 10.** Let $G$ be an $(\alpha_1, \Delta)$-metric graph. Then, $diam(C(G)) \leq 3$ and $rad(C(G)) \leq 2$.

As chordal graphs are $(\alpha_1, \Delta)$-metric graphs, we get the following corollary.

**Corollary 4** ([53]). Let $G$ be a chordal graph. Then, $diam(C(G)) \leq 3$ and $rad(C(G)) \leq 2$.

For our next arguments we need a generalization of the set $S_2(s,t)$, as used in the proof of Theorem 10. We define a slice of the interval $I(u,v)$ from $u$ to $v$ for $0 \leq k \leq d(u,v)$ to be the set $S_k(u,v) = \{w \in I(u,v) : d(w,u) = k\}$. 144
**Theorem 11.** Let $G$ be an $(\alpha_1, \Delta)$-metric graph. Then, in every slice $S_k(u,v)$ there is a vertex $x$ that is universal to that slice, i.e., $S_k(u,v) \subseteq N[x]$. In particular, if $\text{diam}(G) = 2\text{rad}(G)$, then $\text{diam}(C(G)) \leq 2$ and $\text{rad}(C(G)) \leq 1$.

### 7.4 Eccentricity approximating spanning tree construction

In this section, we construct an eccentricity approximating spanning tree and analyze its quality for $(\alpha_1, \Delta)$-metric graphs. Here, we will use sub-indices $G$ and $T$ to indicate whether the distances or the eccentricities are considered in $G$ or in $T$. However, $I(u,v)$ will always mean the interval between vertices $u$ and $v$ in $G$.

#### 7.4.1 Tree construction for unimodal eccentricity functions

First consider the case when the eccentricity function on $G$ is unimodal, i.e., every non-central vertex of $G$ has a neighbor with smaller eccentricity. We will need the following lemmas.

**Lemma 13 ([80]).** Let $G$ be an arbitrary graph. The eccentricity function on $G$ is unimodal if and only if, for every vertex $v$ of $G$, $\text{ecc}_G(v) = d_G(v, C(G)) + \text{rad}(G)$.

**Proof.** Let $v$ be an arbitrary vertex of $G$ and let $v'$ be a vertex of $C(G)$ closest to $v$, i.e., $d_G(v, C(G)) = d_G(v, v')$. Consider a vertex $u \in F(v)$. We have

$$\text{ecc}_G(v) = d_G(v, u) \leq d_G(v, v') + d_G(v', u) \leq d_G(v, C(G)) + \text{rad}(G).$$

First assume that the eccentricity function on $G$ is unimodal. We will show that $\text{ecc}_G(v) \geq d_G(v, C(G)) + \text{rad}(G)$ by induction on $k = d_G(v, C(G))$. If $k = 0$ then $v \in C(G)$ and hence $\text{ecc}_G(v) = \text{rad}(G)$. If $k = d_G(v, C(G)) > 0$, then, by unimodality, there must
exist a neighbor $x$ of $v$ such that $\text{ecc}_G(v) = \text{ecc}_G(x) + 1$. By the inductive hypothesis,

$$\text{ecc}_G(v) = \text{ecc}_G(x) + 1 = d_G(x, C(G)) + \text{rad}(G) + 1 \geq d_G(v, C(G)) + \text{rad}(G)$$

as $d_G(v, C(G)) \leq d_G(v, x') \leq d_G(x, x') + 1 = d_G(x, C(G)) + 1$ (here, $x'$ is a vertex of $C(G)$ closest to $x$). Assume now that $\text{ecc}_G(v) = d_G(v, C(G)) + \text{rad}(G)$ holds for every vertex $v$ of $G$. Consider a neighbor $x$ of $v$ on a shortest path from $v$ to a vertex of $C(G)$ closest to $v$. Since $d_G(v, C(G)) = d_G(x, C(G)) + 1$, we get $\text{ecc}_G(v) = \text{ecc}_G(x) + 1$. ■

**Lemma 14** ([22]). Let $G$ be an arbitrary $\alpha_1$-metric graph. Let $x, y, v, u$ be vertices of $G$ such that $v \in I(x, y)$, $x \in I(v, u)$, and $x$ and $v$ are adjacent. Then $d(u, y) = d(u, x) + d(v, y)$ holds if and only if there exist a neighbor $x'$ of $x$ in $I(x, u)$ and a neighbor $v'$ of $v$ in $I(v, y)$ with $d_G(x', v') = 2$; in particular, $x'$ and $v'$ lie on a common shortest path of $G$ between $u$ and $y$.

We construct a spanning tree $T$ of $G$ as follows. First find the center $C(G)$ of $G$ and pick an arbitrary central vertex $c$ of the graph $C(G)$, i.e., $c \in C(C(G))$. Compute a breadth-first-search tree $T'$ of $C(G)$ started at $c$. Expand this tree $T'$ to a spanning tree $T$ of $G$ by identifying for every vertex $v \in V \setminus C(G)$ its parent vertex in the following way: among all neighbors $x$ of $v$ with $\text{ecc}_G(x) = \text{ecc}_G(v) - 1$ pick that vertex which is closest to $c$ in $G$ (break ties arbitrarily).

**Lemma 15.** Let $G$ be an $(\alpha_1, \Delta)$-metric graph whose eccentricity function is unimodal. Then, for a tree $T$ constructed as described above and every vertex $v$ of $G$, $d_G(v, c) = d_T(v, c)$ holds, i.e., $T$ is a shortest-path-tree of $G$ started at $c$.

As a consequence of Lemma 13 and Lemma 15, we get the following result.
Lemma 16. Let \( G \) be an \((\alpha_1, \Delta)\)-metric graph whose eccentricity function is unimodal. Then, for a tree \( T \) constructed as described above and for every vertex \( v \) of \( G \), \( \text{ecc}_T(v) \leq \text{ecc}_G(v) + \text{rad}(C(G)) \) holds.

Proof. Let \( v \) be an arbitrary vertex of \( G \), \( v' \) be a vertex of \( C(G) \) closest to \( v \) in \( T \), and \( u \) be a vertex most distant from \( v \) in \( T \), i.e., \( \text{ecc}_T(v) = d_T(v, u) \). By Lemma 13 and by the construction of \( T \), \( d_G(v, v') = d_T(v, v') \) and \( v' \) is a vertex of \( C(G) \) closest to \( v \) in \( G \). We have

\[
\text{ecc}_T(v) = d_T(v, u) \leq d_T(v, v') + d_T(v', c) + d_T(c, u),
\]

where \( c \in C(C(G)) \) is the root of the tree \( T \) (see the construction of \( T \)). Since \( d_G(v, v') = d_T(v, v') \), \( d_T(v', c) = d_G(v', c) \leq \text{rad}(C(G)) \), and \( d_T(c, u) = d_G(c, u) \leq \text{rad}(G) \) (by Lemma 15 and the fact that \( c \in C(C(G)) \)), we obtain \( \text{ecc}_T(v) \leq d_G(v, v') + \text{rad}(C(G)) + \text{rad}(G) = \text{ecc}_G(v) + \text{rad}(C(G)) \), as \( d_G(v, v') + \text{rad}(G) = d_G(v, C(G)) + \text{rad}(G) = \text{ecc}_G(v) \) by Lemma 13.

\[
\Box
\]

7.4.2 Tree construction for eccentricity functions that are not unimodal

Consider now the case when the eccentricity function on \( G \) is not unimodal, i.e., there is at least one vertex \( v \notin C(G) \) in \( G \) which has no neighbor with smaller eccentricity. By Theorem 9, \( \text{ecc}_G(v) = \text{rad}(G) + 1 \), \( \text{diam}(G) = 2\text{rad}(G) \) and \( v \) has a neighbor with the eccentricity equal to \( \text{ecc}_G(v) \). We will need the following weaker version of Lemma 13.

Lemma 17. Let \( G = (V, E) \) be an \((\alpha_1, \Delta)\)-metric graph. Let \( v \) be an arbitrary vertex of \( G \) and \( v' \) be an arbitrary vertex of \( C(G) \) closest to \( v \). Then,

\[
d_G(v, C(G)) + \text{rad}(G) - 1 \leq \text{ecc}_G(v) \leq d_G(v, C(G)) + \text{rad}(G).
\]
Furthermore, there is a shortest path $P := (v' = x_0, x_1, \ldots, x_\ell = v)$, connecting $v$ with $v'$, for which the following holds:

(a) if $\text{ecc}_G(v) = d_G(v, C(G)) + \text{rad}(G)$
then $\text{ecc}_G(x_i) = d_G(x_i, C(G)) + \text{rad}(G) = i + \text{rad}(G)$ for each $i \in \{0, \ldots, \ell\}$;

(b) if $\text{ecc}_G(v) = d_G(v, C(G)) + \text{rad}(G) - 1$
then $\text{ecc}_G(x_i) = d_G(x_i, C(G)) - 1 + \text{rad}(G) = i - 1 + \text{rad}(G)$ for each $i \in \{3, \ldots, \ell\}$
and $\text{ecc}_G(x_1) = \text{ecc}_G(x_2) = \text{rad}(G) + 1$.

In particular, if $\text{ecc}_G(v) = \text{rad}(G) + 1$ then $d_G(v, C(G)) \leq 2$.

Now we are ready to construct an eccentricity approximating spanning tree $T$ of $G$ for the case when the eccentricity function is not unimodal. We know that $\text{diam}(G) = 2\text{rad}(G)$ in this case and, therefore, $C(G) \subseteq S_{\text{rad}(G)}(x, y)$ for any diametral pair of vertices $x$ and $y$, i.e., for $x, y$ with $d_G(x, y) = \text{diam}(G)$. By Theorem 11 and since $C(G)$ is convex, there is a vertex $c \in C(G)$ such that $C(G) \subseteq N[c]$. First we find such a vertex $c$ in $C(G)$ and build a tree $T'$ by making $c$ adjacent with every other vertex of $C(G)$. Then, we expand this tree $T'$ to a spanning tree $T$ of $G$ by identifying for every vertex $v \in V \setminus C(G)$ its parent vertex in the following way: if $v$ has a neighbor with eccentricity less than $\text{ecc}_G(v)$, then among all such neighbors pick that vertex which is closest to $c$ in $G$ (break ties arbitrarily); if $v$ has no neighbors with eccentricity less than $\text{ecc}_G(v)$ (i.e., $\text{ecc}_G(v) = \text{rad}(G) + 1$ by Theorem 9), then among all neighbors $x$ of $v$ with $\text{ecc}_G(x) = \text{ecc}_G(v) = \text{rad}(G) + 1$ pick again that vertex which is closest to $c$ in $G$ (break ties arbitrarily).

**Lemma 18.** Let $G$ be an $(\alpha_1, \Delta)$-metric graph whose eccentricity function is not unimodal. Then, for a tree $T$ constructed as described above and every vertex $v$ of $G$, $d_T(v, c) = d_G(v, c)$ holds.
Proof. Assume, by way of contradiction, that \( d_G(v, c) < k := d_T(v, c) \) and let \( v \) be a vertex with such a condition that has smallest eccentricity \( ecc_G(v) \). We may assume that \( ecc_G(v) > rad(G) + 1 \). Indeed, every \( v \) with \( ecc_G(v) = rad(G) + 1 \) either has a neighbor in \( C(G) \) or has a neighbor with a neighbor in \( C(G) \) (see Lemma 17). Therefore, if \( d_G(v, c) < d_T(v, c) \) then, by the construction of \( T \), necessarily \( d_G(v, c) = 2 \), \( d_T(v, c) = 3 \) and the neighbor \( x \) of \( v \) on the path of \( T \) between \( v \) and \( c \) must have the eccentricity equal to \( rad(G) + 1 = ecc_G(v) \). But then, for a common neighbor \( w \) of \( v \) and \( c \) in \( G \), \( ecc_G(w) \leq rad(G) + 1 \) must hold and hence vertex \( v \) cannot choose \( x \) as its parent in \( T \), since \( w \) is a better choice.

So, let \( ecc_G(v) > rad(G) + 1 \). By Lemma 17, there must exist a shortest path in \( G \) between \( v \) and \( c \) such that the neighbor \( w \) of \( v \) on this path has eccentricity \( ecc_G(w) = ecc_G(v) - 1 \). Hence, by the construction of \( T \), \( ecc_G(x) = ecc_G(v) - 1 \) must hold for the neighbor \( x \) of \( v \) on the path of \( T \) between \( v \) and \( c \). By the minimality of \( ecc_G(v) \), we have \( d_G(x, c) = d_T(x, c) = k - 1 \). Since \( d_G(w, c) = d_G(v, c) - 1 < k - 1 \), a contradiction arises; again \( v \) cannot choose \( x \) as its parent in \( T \), since \( w \) is a better choice.

As a consequence of Lemma 17 and Lemma 18, we get the following result.

**Lemma 19.** Let \( G \) be an \((\alpha_1, \Delta)\)-metric graph with \( diam(G) = 2rad(G) \). Then, for a tree \( T \) constructed as described above and every vertex \( v \) of \( G \), \( ecc_T(v) \leq ecc_G(v) + 2 \) holds.

Proof. Let \( v \) be an arbitrary vertex of \( G \) and \( u \) be a vertex most distant from \( v \) in \( T \), i.e., \( ecc_T(v) = d_T(v, u) \). We have

\[
  ecc_T(v) = d_T(v, u) \leq d_T(v, c) + d_T(c, u) = d_G(v, c) + d_G(c, u) \leq d_G(v, c) + rad(G)
\]

\[
  \leq d_G(v, C(G)) + 1 + rad(G) \leq ecc_G(v) + 2
\]
since \( d_G(c, u) \leq \text{ecc}_G(c) = \text{rad}(G) \), \( d_G(v, c) \leq d_G(v, C(G)) + 1 \) (recall that \( C(G) \subseteq N[c] \)), and \( d_G(v, C(G)) - 1 + \text{rad}(G) \leq \text{ecc}_G(v) \) (by Lemma 17).

Our main result is the following theorem. It combines Theorem 10, Lemma 16 and Lemma 19; the complexity follows straightforward.

**Theorem 12.** Every \((\alpha_1, \Delta)\)-metric graph \( G = (V, E) \) has an eccentricity 2-approximating spanning tree. Furthermore, such a tree can be constructed in \( O(|V||E|) \) total time.

As two consequences we have the following corollaries for two important subclasses of \((\alpha_1, \Delta)\)-metric graphs.

**Corollary 5.** If \( G \) is the underlying graph of a 7-systolic complex then \( G \) has an eccentricity 2-approximating spanning tree. In particular, every plane triangulation with inner vertices of degree at least 7 has an eccentricity 2-approximating spanning tree.

**Corollary 6 ([162]).** Every chordal graph has an eccentricity 2-approximating spanning tree.

Note that the result of Corollary 6 (and hence of Theorem 12) is sharp as there are chordal graphs that do not have any eccentricity 1-approximating spanning tree [162].

### 7.5 Experimental results for some real-world networks

Here, we analyze if the eccentricity terrain of a network resembles the eccentricity terrain of a tree. Recall that in trees, the eccentricity of a vertex can range between \( \text{rad}(T) \) and at least \( 2\text{rad}(T) - 1 \) (as \( \text{diam}(T) \geq 2\text{rad}(T) - 1 \)), every vertex \( v \in V(T) \setminus C(T) \) has a neighbor \( u \) such that \( \text{ecc}_T(v) = \text{ecc}_T(u) + 1 \) (i.e., the eccentricity function on trees is unimodal), and the center \( C(T) \) of a tree consists of one vertex or two adjacent vertices. We have seen that in \((\alpha_1, \Delta)\)-metric graphs, the eccentricity function is almost unimodal, the
Table 7.1: Graph datasets and their parameters: $|V|$ is the number of vertices; $|E|$ is the number of edges; $size(G) = |V| + |E|$; $d$ is the average degree; $diam(G)$ is the diameter; $rad(G)$ is the radius. Most networks listed in this table are available at [205, 201, 206, 204].

| Network       | Ref.  | $|V|$  | $|E|$  | $\log_2(size(G))$ | $d$   | $diam(G)$ | $rad(G)$ |
|---------------|-------|-------|-------|-------------------|-------|-----------|----------|
| EMAIL         | [112] | 1133  | 5451  | 12.68             | 9.6   | 8         | 5        |
| FACEBOOK      | [143] | 4039  | 88234 | 16.49             | 43.7  | 8         | 4        |
| DUTCH-ELITE   | [78]  | 3621  | 4310  | 12.95             | 2.4   | 22        | 12       |
| JAZZ          | [108] | 198   | 2742  | 11.52             | 27.7  | 6         | 4        |
| EVA           | [159] | 4475  | 4664  | 13.16             | 2.1   | 18        | 10       |
| AS-GRAF-1     | [205] | 3015  | 5156  | 12.95             | 3.4   | 9         | 5        |
| AS-GRAF-2     | [205] | 4885  | 9276  | 13.79             | 3.8   | 11        | 6        |
| AS-GRAF-3     | [205] | 5357  | 10328 | 13.94             | 3.9   | 9         | 5        |
| YEAST-PI      | [67]  | 1728  | 11003 | 13.64             | 12.7  | 12        | 7        |
| MACAQUE-BRAIN-1| [156] | 45    | 463   | 8.99              | 11.3  | 4         | 2        |
| MACAQUE-BRAIN-2| [151] | 350   | 5198  | 12.44             | 29.7  | 4         | 3        |
| E-COLI-METABOLIC | [147] | 242  | 376   | 9.27              | 3.1   | 16        | 9        |
| C-ELEGANS-METABOLIC | [91] | 453  | 4596  | 12.3              | 8.9   | 7         | 4        |
| YEAST-TRANSCRIPTION | [150] | 321 | 711   | 10.01             | 4.4   | 9         | 5        |
| US-AIRLINES   | [26]  | 332   | 2126  | 11.26             | 12.8  | 6         | 3        |
| POWER-GRID    | [192] | 4941  | 6594  | 13.49             | 2.7   | 46        | 23       |
| WORD-ADJACENCY| [158] | 112   | 425   | 9.07              | 7.6   | 5         | 3        |
| FOOD          | [205] | 135   | 596   | 9.51              | 8.8   | 4         | 3        |

The eccentricity of a vertex can range between $rad(G)$ and at least $2rad(G) - 2$ (as $diam(G) \geq 2rad(G) - 2$), $diam(C(G)) \leq 3$, $rad(C(G)) \leq 2$, and the center $C(G)$ is convex and hence connected. Furthermore, every $(\alpha_1, \Delta)$-metric graph $G$ admits an eccentricity 2-approximating spanning tree, which provides a strong evidence that the eccentricity terrain of $G$ resembles the eccentricity terrain of a tree.

In this section, we analyze vertex localities and centers in a collection of real-world networks/graphs coming from a number of different domains. Additionally, based on what we learned from $(\alpha_1, \Delta)$-metric graphs in Section 7.4, we propose two heuristics for constructing eccentricity approximating trees in general graphs and analyze their performance on our set of real-world networks. Some of those networks are not connected, but they
usually have one very large connected component and a few very small components. In this case, we consider only a largest connected component. Note that all our networks are unweighted and we ignore directions of edges if a network is originally directed. A summary of basic statistical properties of largest connected components of the networks in our datasets is given in Table 7.1.

7.5.1 Datasets

First we describe the investigated networks.

Social networks.

EMAIL [201, 112]: This network represents the email interchanges between members of the university of Rovira i Virgili, Tarragona.

FACEBOOK [204, 143]: This network has 4039 users who belong to the ego networks (the network of friendship between a user’s friends) of 10 people. Two vertices (users) are connected if they are Facebook friends.

DUTCH-ELITE [26, 78]: This is a network data on the administrative elite in the Netherlands, April 2006. Vertices represent persons and organizations that are most important to the Dutch government (2-mode network). An edge connects a person-vertex and an organization-vertex if the corresponding person belongs to the corresponding organization.

JAZZ [201, 108]: In this network, vertices represent different Jazz musicians and two vertices are connected if the two musicians have played together.

EVA [26, 159]: This network presents corporate ownership information as a social network. Two vertices are connected with an edge if one is the owner of the other.

Internet graphs.

AS-GRAPHs [205]: Those graphs represent the Autonomous Systems topology of the Internet. In each graph, a vertex represents an autonomous system, and two vertices are
connected if the two autonomous systems share at least one physical connection. In this work, we use three AS graphs: AS-GRAPH-1, AS-GRAPH-2, and AS-GRAPH-3 for which the data were collected in November 1997, April 1999, and July 1999, respectively.

**Biological networks.**

Protein Interaction (PI) Networks: Generally, in a PI network, the vertices represent different proteins and the edges represent the connections between the interacting proteins. We consider the protein interaction networks of the Escherichia coli [46] and the Yeast [67].

Neural Networks: In those networks, neurons (vertices) are connected together through synapsis (edges). We analyze two different brain area networks of the Macaque monkey [151, 156].

Metabolic Networks: Metabolic networks are represented by metabolites (vertices) such as amino acids and biochemical reactions (directed edges). In this datasets, we have the Escherichia coli [147] and the Caenorhabditis elegans [91] metabolic networks.

Transcription Networks: Networks in which vertices are genes and edges represent different interactions (interrelationships) between genes. We analyze the Yeast transcription network [150].

**Other networks.**

US-AIRLINES [26]: The transportation network of airlines in the United States. The original graph from [26] is weighted. We ignored the weights in our experiments.

POWER-GRID [206, 192]: This network represents the topology of the Western States Power Grid of the United States.

WORD-ADJACENCY [206, 158]: The adjacency network of commonly occurring adjectives and nouns in the novel David Copperfield by Charles Dickens. An edge connects any adjacent pair of words.

FOOD [205]: This network represents the food predatory interactions among different
species in the Ythan Eastuary environment. Vertices represent species, and a directed edge links two vertices if one species preys on the other.

### 7.5.2 Analysis of vertex localities and centers

Recall that the locality of a vertex $v \notin C(G)$ (with respect to the eccentricity function) is the distance from $v$ to a closest vertex with smaller eccentricity:

$$loc(v) = \min\{d(v, x) : x \in V, ecc(x) < ecc(v)\}.$$ 

The locality of any central vertex $u \in C(G)$ is defined to be 1. Hence, the eccentricity function is unimodal on $G$ if and only if all vertices of $G$ have locality 1. Given a graph $G = (V, E)$, we can define its eccentricity layering $\mathcal{EL}(G) = (C_0(G), \ldots, C_{\text{diam}(G) - \text{rad}(G)}(G))$ to be a partition of the vertex set $V$ into layers $C_k(G) = \{v \in V : ecc(v) = \text{rad}(G) + k\}$, $k = 0, 1, \ldots, \text{diam}(G) - \text{rad}(G)$. Clearly, $C_0(G) = C(G)$. The layer of each vertex $u$ with respect to the eccentricity layering, denoted by $\text{layer}(u)$, is $k$ if $u \in C_k(G)$.

Theorem 9 from Section 7.3 says that if a vertex $v$ with $loc(v) > 1$ exists in an $(\alpha_1, \Delta)$-metric graph $G$ then $\text{diam}(G) = 2\text{rad}(G)$ and $\text{ecc}(v) = \text{rad}(G) + 1$, i.e., $\text{layer}(v) = 1$.

We analyzed vertex localities in the graphs from our dataset. It turns out that the eccentricity function is unimodal for the graphs JAZZ, MACAQUE-BRAIN-2, US-AIRLINES and FOOD. For all graphs, except MACAQUE-BRAIN-1 (78%), E-COLI-METABOLIC (86%) and WORD-ADJACENCY (77%), at least 90% of vertices (in most cases close to 100%) have locality 1 (see Table 7.2).

In Figure 7.4, we show also the distribution of vertices with $loc(\cdot) > 1$ over different layers of the eccentricity layering in each graph of the datasets. As in the case of $(\alpha_1, \Delta)$-metric graphs, in the graphs of our datasets (with the exception of POWER-GRID), the vertices with locality greater than 1 also tend to reside in the layers $C_k(G)$ with smaller $k$. 

154
Table 7.2: Percent of vertices with localities equal to 1 and larger than 1 in each graph of the datasets.

<table>
<thead>
<tr>
<th>Network</th>
<th>% of ver with $\text{loc}(\cdot) = 1$</th>
<th>% of ver with $\text{loc}(\cdot) &gt; 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMAIL</td>
<td>$\approx 95%$</td>
<td>$\approx 5%$</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>$\approx 98%$</td>
<td>$\approx 2%$</td>
</tr>
<tr>
<td>DUTCH-ELITE</td>
<td>$\approx 97%$</td>
<td>$\approx 3%$</td>
</tr>
<tr>
<td>JAZZ</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>EVA</td>
<td>$\approx 99%$</td>
<td>$\approx 1%$</td>
</tr>
<tr>
<td>AS-GRAHP-1</td>
<td>$\approx 99%$</td>
<td>$\approx 1%$</td>
</tr>
<tr>
<td>AS-GRAHP-2</td>
<td>$\approx 98%$</td>
<td>$\approx 2%$</td>
</tr>
<tr>
<td>AS-GRAHP-3</td>
<td>$\approx 98%$</td>
<td>$\approx 2%$</td>
</tr>
<tr>
<td>E-coli-PI</td>
<td>$\approx 90%$</td>
<td>$\approx 10%$</td>
</tr>
<tr>
<td>YEAST-PI</td>
<td>$\approx 95%$</td>
<td>$\approx 5%$</td>
</tr>
<tr>
<td>MACAQUE-brain-1</td>
<td>$\approx 78%$</td>
<td>$\approx 22%$</td>
</tr>
<tr>
<td>MACAQUE-brain-2</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>E-coli-metabolic</td>
<td>$\approx 86%$</td>
<td>$\approx 14%$</td>
</tr>
<tr>
<td>C-elegans-metabolic</td>
<td>$\approx 98%$</td>
<td>$\approx 2%$</td>
</tr>
<tr>
<td>YEAST-transcription</td>
<td>$\approx 91%$</td>
<td></td>
</tr>
<tr>
<td>US-AIRLINES</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>POWER-GRID</td>
<td>$\approx 99%$</td>
<td>$\approx 2%$</td>
</tr>
<tr>
<td>Word-Adjacency</td>
<td>$\approx 77%$</td>
<td>$\approx 23%$</td>
</tr>
<tr>
<td>Food</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>
We know that $\text{diam}(G) \geq 2\text{rad}(G) - 2$ holds for every $\alpha_1$-metric graph $G$ [197]. From Table 7.1, we see that for all graphs in our datasets $2\text{rad}(G) - \text{diam}(G)$ is at most 2 as well. For graphs Facebook, Macaque-brain-1, US-Airlines and POWER-Grid, in fact, $\text{diam}(G) = 2\text{rad}(G)$ holds.

As we have mentioned earlier, the center $C(G)$ of any $\alpha_1$-metric graph $G$ is connected (in fact, it is convex and hence isometric). We analyzed centers of all graphs from our datasets (see Table 7.3). The centers of most of the graphs (except Dutch-Elite, As-Graph-2, E-coli-metabolic and Yeast-transcription) turned out to be connected as well. By Theorem 10, we know that $\text{diam}(C(G)) \leq 3$ and $\text{rad}(C(G)) \leq 2$ holds for every $(\alpha_1, \Delta)$-metric graph $G$. As the centers of the graphs in our datasets are not necessarily isometric (distance-preserving) subgraphs, we used notions of \textit{weak diameter} $\overline{\text{diam}}(C(G))$ and \textit{weak radius} $\overline{\text{rad}}(C(G))$ to measure their centers, where $\overline{\text{diam}}(C(G)) = \max\{d_G(x, y) : x, y \in C(G)\}$ and $\overline{\text{rad}}(C(G)) = \min\{\max\{d_G(x, y) : y \in C(G)\} : x \in C(G)\}$. 

156
Table 7.3: The weak diameters and weak radii of the centers of each graph in the datasets.

| Network            | $diam(C(G))$ | $rad(C(G))$ | Connected? | $|V(G)|$ |
|--------------------|--------------|-------------|------------|--------|
| EMAIL              | 4            | 3           | yes        | 215/1133 |
| FACEBOOK           | 0            | 0           | yes        | 1/4039  |
| DUTCH-ELITE        | 4            | 4           | no         | 3/3621  |
| JAZZ               | 3            | 2           | yes        | 56/198  |
| EVA                | 3            | 2           | yes        | 15/4475 |
| AS-Graph-1         | 2            | 1           | yes        | 32/3015 |
| AS-Graph-2         | 4            | 3           | no         | 531/4885|
| AS-Graph-3         | 2            | 2           | yes        | 10/5357 |
| E-coli-PI          | 2            | 2           | yes        | 6/126   |
| YEAST-PI           | 5            | 3           | yes        | 53/1728 |
| Macaque-brain-1    | 0            | 0           | yes        | 1/45    |
| Macaque-brain-2    | 3            | 2           | yes        | 194/350 |
| E-coli-metabolic   | 5            | 3           | no         | 5/242   |
| C-elegans-metabolic| 4            | 2           | yes        | 17/453  |
| YEAST-TRANSCRIPTION| 3            | 3           | yes        | 3/321   |
| US-AIRLINES        | 0            | 0           | yes        | 1/332   |
| POWER-GRID         | 0            | 0           | yes        | 1/4941  |
| Word-Adjacency     | 2            | 1           | yes        | 4/112   |
| Food               | 3            | 2           | yes        | 53/135  |

Interestingly, the graphs with $diam(G) = 2rad(G)$ in our datasets (i.e., FACEBOOK, Macaque-brain-1, US-AIRLINES and POWER-GRID) have single vertex centers. The centers of all graphs have small weak diameter and weak radius: the weak radius is 0 in 4 graphs, 1 in 2 graphs, 2 in 7 graphs, 3 in 5 graphs, and 4 in 1 graph (DUTCH-ELITE).

### 7.5.3 Eccentricity approximating tree construction and analysis

We say that a tree $T$ is an eccentricity $k$-approximating tree for a graph $G$ if for every vertex $v$ of $G$, $|ecc_T(v) - ecc_G(v)| \leq k$ holds. If $T$ is a spanning tree, then $ecc_T(v) \geq ecc_G(v)$, for all $v \in V$, and this new definition agrees with the definition of an eccentricity $k$-approximating spanning tree.

Our goal in this section is to propose a heuristic for constructing an eccentricity $k$-
approximating tree for general graphs such that the value of $k$ is as small as possible. In our construction of an eccentricity 2-approximating spanning tree for an $(\alpha_1, \Delta)$-metric graph $G$, two main ingredients were crucial: 1. the eccentricity function on $G$ is almost unimodal and the vertices with locality larger than 1 reside only in layer $\mathcal{C}_1(G)$; 2. the radius of the center $C(G)$ is at most 2. Our eccentricity 2-approximating spanning tree was a shortest-path-tree starting at a vertex $c \in C(C(G))$.

Although the weak radius of each graph in our datasets is relatively small (for 13 graphs it was at most 2, for 5 graphs at most 3, and only for Dutch-Elite it was 4; see Table 7.3), for some graphs, a small number of vertices with locality 2, 3 or 4 exists and those vertices may reside also at eccentricity layers $\mathcal{C}_k(G)$ with $k > 1$ (see Figure 7.4).

Based on what we learned from $(\alpha_1, \Delta)$-metric graphs in Section 7.4 and on what we observed about vertex localities and centers in the graphs in our datasets, we propose two heuristics for constructing eccentricity approximating trees in general graphs. Both heuristics try to mimic the construction for $(\alpha_1, \Delta)$-metric graphs that we used in Section 7.4.

Our first heuristic, named $EAST$, constructs an Eccentricity Approximating Spanning Tree $T_{EAST}$ as a shortest-path-tree starting at a vertex $c \in C(C(G))$. We identify an arbitrary vertex $c \in C(C(G))$ as the root of $T_{EAST}$, and for each other vertex $v$ of $G$ define its parent in $T_{EAST}$ as follows: among all neighbors of $v$ in $I(v, c)$ choose a vertex with minimum eccentricity (break ties arbitrarily). A formal description of this construction is given in Algorithm EAST.

Our second heuristic, named $EAT$, constructs for a graph $G$ an Eccentricity Approximating Tree $T_{EAT}$ (not necessarily a spanning tree; it may have a few edges not present in graph $G$) as follows. We again identify an arbitrary vertex $c \in C(C(G))$ as the root of $T_{EAT}$ and make it adjacent in $T_{EAT}$ to all other vertices of $C(G)$ (clearly, some of these edges might not be contained in $G$). Then, for each vertex $v \in V \backslash C(G)$, we find a vertex $u$ with
**Algorithm EAST.** Eccentricity $k$-approximating spanning trees for general graphs.  
**Input.** A graph $G = (V, E)$.  
**Output.** An eccentricity $k$-approximating spanning tree $T = (V, U)$ of $G$.  

1. $U \leftarrow \emptyset$  
2. pick a vertex $c \in C(G)$ with the minimum distance to every other vertex in $C(G)$  
3. for every $v \in V \setminus \{c\}$ do  
4. among all neighbors of $v$ in $I(v, c)$ choose a vertex $u$ with minimum eccentricity  
5. add edge $uv$ to $U$  
6. return $T = (V, U)$

**Algorithm EAT.** Eccentricity $k$-approximating trees for general graphs.  
**Input.** A graph $G = (V, E)$.  
**Output.** An eccentricity $k$-approximating tree $T = (V, U)$ of $G$.  

1. $U \leftarrow \emptyset$  
2. pick a vertex $c \in C(G)$ with the minimum distance to every other vertex in $C(G)$  
3. for every $u \in C(G) \setminus \{c\}$ do  
4. add edge $uc$ to $U$  
5. for every $v \in V \setminus C(G)$ do  
6. among all vertices $\{u \in V : d_G(u, v) = loc(v) \text{ and } ecc_G(u) < ecc_G(v)\}$  
7. choose a vertex $u$ which is closest to $c$  
8. add edge $uv$ to $U$  
9. return $T = (V, U)$

$ecc_G(u) < ecc_G(v)$ which is closest to $v$, and if there is more than one such vertex, we pick the one which is closest to $c$. In other words, among all vertices $\{u \in V : d_G(u, v) = loc(v) \text{ and } ecc_G(u) < ecc_G(v)\}$, we choose a vertex $u$ which is closest to $c$ (break ties arbitrarily). Such a vertex $u$ becomes the parent of $v$ in $T_{EAT}$. Clearly, if $loc(v) > 1$ then edge $uv$ of $T_{EAT}$ is not present in $G$. A formal description of this construction is given in Algorithm EAT.

We tested both heuristics on our set of real-world networks. Experimental results obtained are presented in Table 7.4 and Table 7.5.

Table 7.4 demonstrates the quality of the spanning tree $T$ constructed by Algorithm EAST for each graph $G$ in the datasets. Algorithm EAST was able to produce an eccen-
Table 7.4: A spanning tree $T$ constructed by Algorithm EAST: for each vertex $u \in V$, $k(u) = ecc_T(u) - ecc_G(u)$; $k_{max} = \max_{u \in V} k(u)$; $k_{avg} = \frac{1}{n} \sum_{u \in V} k(u)$.

<table>
<thead>
<tr>
<th>Network</th>
<th>diam($G$)</th>
<th>diam($T$)</th>
<th>$k_{max}$</th>
<th>$k_{avg}$</th>
<th>% of ver w $k(0) = 0$</th>
<th>% of ver w $k(1) = 1$</th>
<th>% of ver w $k(2) = 2$</th>
<th>% of ver w $k(3) = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMAIL</td>
<td>8</td>
<td>10</td>
<td>3</td>
<td>1.774</td>
<td>$\approx 0.79%$</td>
<td>$\approx 27.8%$</td>
<td>$\approx 64.61%$</td>
<td>$\approx 6.8%$</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>8</td>
<td>8</td>
<td>2</td>
<td>0.69</td>
<td>51.9%</td>
<td>27.6%</td>
<td>20.5%</td>
<td></td>
</tr>
<tr>
<td>DUTCH-ELITE</td>
<td>22</td>
<td>24</td>
<td>6</td>
<td>2.083</td>
<td>$\approx 17.45%$</td>
<td>$\approx 0%$</td>
<td>$\approx 61.28%$</td>
<td></td>
</tr>
<tr>
<td>JAZZ</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>1.742</td>
<td>$\approx 1.52%$</td>
<td>$\approx 22.72%$</td>
<td>$\approx 75.76%$</td>
<td></td>
</tr>
<tr>
<td>EVA</td>
<td>18</td>
<td>19</td>
<td>2</td>
<td>0.575</td>
<td>$\approx 47.59%$</td>
<td>$\approx 47.26%$</td>
<td>$\approx 5.14%$</td>
<td></td>
</tr>
<tr>
<td>AS-Graph-1</td>
<td>9</td>
<td>10</td>
<td>2</td>
<td>0.64</td>
<td>$\approx 35.78%$</td>
<td>$\approx 64.18%$</td>
<td>$\approx 0.03%$</td>
<td></td>
</tr>
<tr>
<td>AS-Graph-2</td>
<td>11</td>
<td>12</td>
<td>3</td>
<td>1.272</td>
<td>$\approx 64.4%$</td>
<td>$\approx 31.87%$</td>
<td>$\approx 3.7%$</td>
<td></td>
</tr>
<tr>
<td>AS-Graph-3</td>
<td>9</td>
<td>10</td>
<td>2</td>
<td>0.312</td>
<td>$\approx 70.38%$</td>
<td>$\approx 28%$</td>
<td>$\approx 0.03%$</td>
<td></td>
</tr>
<tr>
<td>E-COLI-PI</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>0.769</td>
<td>$\approx 34.92%$</td>
<td>$\approx 55.17%$</td>
<td>$\approx 11.9%$</td>
<td></td>
</tr>
<tr>
<td>YEAST-PI</td>
<td>12</td>
<td>13</td>
<td>4</td>
<td>0.972</td>
<td>$\approx 28%$</td>
<td>$\approx 50.23%$</td>
<td>$\approx 18.5%$</td>
<td>$\approx 2.89%$</td>
</tr>
<tr>
<td>MACAQUE-BRAIN-1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>0.222</td>
<td>77.78%</td>
<td>22.22%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACAQUE-BRAIN-2</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>1.489</td>
<td>$\approx 1.71%$</td>
<td>$\approx 47.7%$</td>
<td>$\approx 50.57%$</td>
<td></td>
</tr>
<tr>
<td>E-COLI-METABOLIC</td>
<td>16</td>
<td>17</td>
<td>4</td>
<td>1.132</td>
<td>$\approx 34.71%$</td>
<td>$\approx 33.1%$</td>
<td>$\approx 17.77%$</td>
<td>$\approx 13.22%$</td>
</tr>
<tr>
<td>C-ELEAGANS-METABOLIC</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>0.349</td>
<td>$\approx 65.12%$</td>
<td>$\approx 34.88%$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>YEAST-TRANSCRIPT.</td>
<td>9</td>
<td>10</td>
<td>3</td>
<td>1.121</td>
<td>$\approx 33.96%$</td>
<td>$\approx 26.79%$</td>
<td>$\approx 32.4%$</td>
<td>$\approx 6.85%$</td>
</tr>
<tr>
<td>US-AIRLINES</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>100%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POWER-GRID</td>
<td>46</td>
<td>46</td>
<td>4</td>
<td>1.409</td>
<td>$\approx 46.35%$</td>
<td>$\approx 13.13%$</td>
<td>$\approx 12.61%$</td>
<td>$\approx 9.11%$</td>
</tr>
<tr>
<td>WORD-ADJACENCY</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>0.411</td>
<td>$\approx 58.93%$</td>
<td>$\approx 41.07%$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FOOD</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>1.629</td>
<td>$\approx 1.48%$</td>
<td>$\approx 34.07%$</td>
<td>$\approx 64.44%$</td>
<td></td>
</tr>
</tbody>
</table>

Eccentricity $k$-approximating spanning tree with $k = 0$ for 1 graph (US-AIRLINES), $k = 1$ for 3 graphs, $k = 2$ for 8 graphs, $k = 3$ for 3 graphs, $k = 4$ for 3 graphs, and $k = 6$ for 1 graph (DUTCH-ELITE). According to the criteria from [154] for the existence of eccentricity-preserving (i.e., eccentricity 0-approximating) spanning trees (see Introduction), graph US-AIRLINES has an eccentricity 0-approximating spanning tree and Algorithm EAST succeeded to construct such a spanning tree. Algorithm EAST succeeded to construct optimal spanning trees also for graphs MACAQUE-BRAIN-1, C-ELEAGANS-METABOLIC and WORD-ADJACENCY (those graphs do not satisfy the criteria for admitting eccentricity 0-approximating spanning trees and EAST constructs for them eccentricity 1-approximating spanning trees). For every graph $G = (V, E)$ in our datasets and for each corresponding constructed spanning tree $T$, we computed $k(u) = ecc_T(u) - ecc_G(u)$, for each vertex $u \in V$. Using this, for each graph $G$ and spanning tree $T$ we determined $k_{max} = \max_{u \in V} k(u)$ (the
maximum difference between $ecc_T(u)$ and $ecc_G(u)$) and $k_{avg} = \frac{1}{n} \sum_{u \in V} k(u)$ (the average difference). Although $k_{max}$ is greater than 2 for 7 graphs of the datasets, the average difference $k_{avg}$ is smaller than 2 for all but one graph (DUTCH-ELITE) and is smaller than 1 for 10 graphs. Overall, the constructed trees preserve vertex eccentricities of the graphs with a high level of accuracy. If we consider, for example, graph AS-GRAph-3 and its spanning tree constructed by EAST, we have $k_{max} = 2$ but about 70% of vertices preserved their eccentricities ($k(\cdot) = 0$), about 28% of vertices increased their eccentricity only by one ($k(\cdot) = 1$), and only the remaining 2% of vertices increased their eccentricity by two ($k(\cdot) = 2$); hence, the average difference $k_{avg}$ is 0.312. If we consider the graph DUTCH-ELITE and its spanning tree constructed by EAST, we have $k_{max} = 6$ but about 79% of vertices increased their eccentricity only by two ($k(\cdot) \leq 2$), resulting in the average difference $k_{avg} = 2.083$, which is rather small even for (DUTCH-ELITE).

Table 7.5 demonstrates the quality of the (not necessarily spanning) tree $T$ constructed by Algorithm EAT for each graph $G$ in the datasets. The flexibility of being able to use edges in $T$ that are not present in $G$ allowed the algorithm to get even better approximations of vertex eccentricities in graphs by vertex eccentricities in trees. Algorithm EAT was able to produce an eccentricity $k$-approximating tree with $k = 0$ for 3 graphs (FACEBOOK, MACAQUE-BRAIN-1, US-AIRLINES), with $k = 1$ for all other graphs except POWER-GRID (which has $k = 3$). For every graph $G = (V, E)$ in our datasets and for each correspondingly constructed tree $T$, we computed $k(u) = ecc_T(u) - ecc_G(u)$, for each vertex $u \in V$, and then $k_{max} = \max_{u \in V} k(u)$, $k_{min} = \min_{u \in V} k(u)$, and $k_{avg} = \frac{1}{n} \sum_{u \in V} k(u)$. Interestingly, the difference between $ecc_T(u)$ and $ecc_G(u)$ falls in the range $[-1, 0]$ for 8 graphs, in the range $[0, 1]$ for 7 graphs, and only for one graph (POWER-GRID) it falls in the range $[-3, 0]$ (we excluded FACEBOOK, MACAQUE-BRAIN-1, US-AIRLINES in these counts as for them $k_{max} = k_{min} = 0$). For 7 graphs, more than 98% of the vertices preserved their
Table 7.5: A tree $T$ constructed by Algorithm EAT: for each vertex $u \in V$, $k(u) = ecc_T(u) - ecc_G(u)$; $k_{\text{max}} = \max_{u \in V} k(u)$; $k_{\text{min}} = \min_{u \in V} k(u)$; $k_{\text{avg}} = \frac{1}{n} \sum_{u \in V} k(u)$.

<table>
<thead>
<tr>
<th>Network</th>
<th>$diam(G)$</th>
<th>$diam(T)$</th>
<th>$[k_{\text{min}}, k_{\text{max}}]$</th>
<th>$k_{\text{avg}}$</th>
<th>$%$ of ver $w$ $k(\cdot) = 0$</th>
<th>$%$ of ver $w$ $k(\cdot) = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMAIL</td>
<td>8</td>
<td>8</td>
<td>$[-1, 0]$</td>
<td>0.0009</td>
<td>99.91%</td>
<td>0.09%</td>
</tr>
<tr>
<td>FACEBOOK</td>
<td>8</td>
<td>8</td>
<td>[0, 0]</td>
<td>0</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>DUTCH-ELITE</td>
<td>22</td>
<td>21</td>
<td>$[-1, 0]$</td>
<td>-0.771</td>
<td>22.92%</td>
<td>77.18%</td>
</tr>
<tr>
<td>JAZZ</td>
<td>6</td>
<td>6</td>
<td>$[-1, 0]$</td>
<td>-0.015</td>
<td>98.48%</td>
<td>1.52%</td>
</tr>
<tr>
<td>EVA</td>
<td>18</td>
<td>18</td>
<td>$[-1, 0]$</td>
<td>-0.36</td>
<td>64.02%</td>
<td>35.98%</td>
</tr>
<tr>
<td>AS-GRAH-1</td>
<td>9</td>
<td>10</td>
<td>[0, 1]</td>
<td>0.62</td>
<td>37.98%</td>
<td>62.02%</td>
</tr>
<tr>
<td>AS-GRAH-2</td>
<td>11</td>
<td>12</td>
<td>[0, 1]</td>
<td>0.949</td>
<td>5.04%</td>
<td>94.96%</td>
</tr>
<tr>
<td>AS-GRAH-3</td>
<td>9</td>
<td>10</td>
<td>[0, 1]</td>
<td>0.248</td>
<td>75.53%</td>
<td>24.47%</td>
</tr>
<tr>
<td>E-COLI-PI</td>
<td>5</td>
<td>6</td>
<td>[0, 1]</td>
<td>0.595</td>
<td>40.48%</td>
<td>59.52%</td>
</tr>
<tr>
<td>YEAST-PI</td>
<td>12</td>
<td>12</td>
<td>$[-1, 0]$</td>
<td>-0.168</td>
<td>83.22%</td>
<td>16.78%</td>
</tr>
<tr>
<td>MACAQUE-BRAIN-1</td>
<td>4</td>
<td>4</td>
<td>[0, 0]</td>
<td>0</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>MACAQUE-BRAIN-2</td>
<td>4</td>
<td>4</td>
<td>$[-1, 0]$</td>
<td>-0.003</td>
<td>99.71%</td>
<td>0.29%</td>
</tr>
<tr>
<td>E-COLI-METABOLIC</td>
<td>16</td>
<td>15</td>
<td>$[-1, 0]$</td>
<td>-0.624</td>
<td>37.6%</td>
<td>62.4%</td>
</tr>
<tr>
<td>C-ELEGANS-METABOLIC</td>
<td>7</td>
<td>8</td>
<td>[0, 1]</td>
<td>0.342</td>
<td>65.78%</td>
<td>34.22%</td>
</tr>
<tr>
<td>YEAST-TRANSCRIPTION</td>
<td>9</td>
<td>9</td>
<td>[0, 1]</td>
<td>0.019</td>
<td>98.13%</td>
<td>1.87%</td>
</tr>
<tr>
<td>US-AIRLINES</td>
<td>6</td>
<td>6</td>
<td>[0, 0]</td>
<td>0</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>POWER-GRID</td>
<td>46</td>
<td>43</td>
<td>$[-3, 0]$</td>
<td>-1.309</td>
<td>56.34%</td>
<td>0%</td>
</tr>
<tr>
<td>WORD-ADJACENCY</td>
<td>5</td>
<td>6</td>
<td>[0, 1]</td>
<td>0.152</td>
<td>84.82%</td>
<td>15.18%</td>
</tr>
<tr>
<td>FOOD</td>
<td>4</td>
<td>4</td>
<td>$[-1, 0]$</td>
<td>-0.015</td>
<td>98.52%</td>
<td>1.48%</td>
</tr>
</tbody>
</table>

Eccentricities ($|k(\cdot)| = 0$). For POWER-GRID, more than 56% of vertices preserved their eccentricities.

### 7.6 Concluding remarks

We conclude the paper with some immediate questions building off our results.

1. Can our result on eccentricity 2-approximating spanning trees for $(\alpha_1, \Delta)$-metric graphs be extended to arbitrary $\alpha_1$-metric graphs?

2. Can our result be generalized to arbitrary $\alpha_2$-metric graphs (possibly with some generalization of the triangle condition)?
3. If we drop the requirement for a tree to be a spanning tree, does every \((\alpha_1, \Delta)\)-metric graph (in particular, every chordal graph) admit an eccentricity \(k\)-approximating tree with \(k < 2\)?

More generally, we are interested in the following questions.

4. What is the complexity of the problem: Given a graph \(G\) and an integer \(k\), check if \(G\) admits an eccentricity \(k\)-approximating (spanning) tree?

We suspect that this problem is NP-complete. So, it is natural to ask:

5. Can this problem be efficiently approximated? Is a constant factor approximation possible?

6. Do our heuristics for general graphs provide any provable good approximation?

In Chapter 6 ([61]), we show that every \(\delta\)-hyperbolic graph \(G = (V, E)\) admits an eccentricity \(O(\delta)\)-approximating spanning tree constructible in \(O(\delta|E|)\) time. Projecting that general result to chordal graphs, one gets only the existence of an eccentricity 7-approximating spanning tree.
Chapter 8

Eccentricity $k$-Approximating Tree Algorithms - Comparison

In this chapter, we compare the qualities of the eccentricity $k$-approximating spanning trees for general graphs produced using the three algorithms introduced in Chapters 6 and 7 for a few network dataset examples. The algorithms are summarized as follows:

8.1 The algorithms

1. Algorithm 1: Given a graph $G = (V, E)$, an eccentricity $k$-approximating spanning tree $T_1 = (V, E')$ is constructed as an arbitrary $BFS(c)$-tree, rooted at vertex $c$. Vertex $c$ is a middle vertex of a $(u, v)$-geodesic, where $u$ and $v$ are a pair of mutually distant vertices. This algorithm was introduced in Section 7.3.2.

2. Algorithm 2: Given a graph $G = (V, E)$, an eccentricity $k$-approximating spanning tree $T_2 = (V, E')$ is constructed as an arbitrary $BFS(w)$-tree, rooted at vertex $w$. Vertex $w$ is a vertex of a $(u, v)$-geodesic and that is at distant $rad(G)$ from vertex
Table 8.1: Comparison of the main characteristics of the three eccentricity $k$-approximating tree algorithms.

<table>
<thead>
<tr>
<th></th>
<th>Algorithm 1</th>
<th>Algorithm 2</th>
<th>Algorithm 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spanning?</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Preprocessing?</td>
<td>requires computing $\text{rad}(G)$ in advanced</td>
<td>requires computing vertex eccentricities in advanced</td>
<td></td>
</tr>
<tr>
<td>Root</td>
<td>$c \in \rho(u,v)$ \ if \ $u, v$ are mutually distant</td>
<td>$w \in \rho(u,v)$ \ if \ $v \in F(u)$ \ &amp; \ $d(w,v) = \text{rad}(G)$</td>
<td>$c \in C(C(G))$</td>
</tr>
<tr>
<td>Construction</td>
<td>$O(d</td>
<td>E</td>
<td>)$</td>
</tr>
<tr>
<td>Eccentricity distortion</td>
<td>$3\delta + 1$</td>
<td>$6\delta + 1$</td>
<td></td>
</tr>
</tbody>
</table>

$v$, where $u$ is an arbitrary chosen vertex and $v \in F(u)$. I.e., $v$ is a vertex at most distant from $u$. This algorithm was introduced in Section 7.3.2.

3. Algorithm 3: Given a graph $G = (V,E)$, an eccentricity $k$-approximating spanning tree $T_{EAST} = (V,E')$ is constructed as shortest-path-tree, rooted at vertex $c \in C(C(G))$. For each other vertex $v$ of $G$, we define its parent in $T_{EAST}$ as follows: among all neighbors of $v$ in $I(v,c)$ choose a vertex with minimum eccentricity (break ties arbitrarily). This algorithm was introduced in Section 8.5.3.

In Table 8.1, we present a brief comparison of the three algorithms.
Table 8.2: Graph datasets and their parameters: $|V|$: number of vertices, $|E|$: number of edges; $|C(G)|$: number of central vertices; $\text{deg}$: average degree; $\text{rad}(G)$: graph’s radius; $\text{diam}(G)$: graph’s diameter; $\text{diam}_G(C(G))$: diameter of the graph’s center; connected?: indicates whether or not the center of the graph is connected.

| Network           | Type                | Ref.   | $|V|$  | $|E|$  | $|C(G)|$ | $\text{deg}$ | $\text{rad}(G)$ | $\text{diam}(G)$ | $\text{diam}_G(C(G))$ | connected? |
|-------------------|---------------------|--------|-------|-------|---------|-------------|----------------|----------------|----------------------|-----------|
| DUTCH-ELITE       | Social              | [26]   | 3621  | 4310  | 3       | 2.4        | 12            | 22             | 4                    | no        |
| FACEBOOK          | Social              | [143]  | 4039  | 88244 | 1       | 43.7       | 4             | 8              | 0                    | yes       |
| EVA               |                     | [26]   | 4475  | 4664  | 15      | 2.1        | 10            | 18             | 3                    | yes       |
| SLASHDOT          |                     | [142]  | 77360 | 905468| 1       | 13.1       | 6             | 12             | 0                    | yes       |
| EMAIL-VIRGILI     |                     | [112]  | 1133  | 5451  | 215     | 9.6        | 5             | 8              | 4                    | yes       |
| EMAIL-ENRON       | Communication       | [142, 133] | 33696 | 180811 | 248    | 10.7       | 7             | 13             | 2                    | yes       |
| EMAIL-EU          |                     | [141]  | 224832 | 680720 | 1    | 3         | 7             | 14             | 0                    | yes       |
| AS-GRAF-1         |                     | [205]  | 3015  | 5156  | 32      | 3.4        | 5             | 9              | 2                    | yes       |
| AS-GRAF-2         |                     | [205]  | 4885  | 9576  | 531     | 3.8        | 6             | 11             | 4                    | no        |
| AS-GRAF-3         | Internet            | [205]  | 5357  | 10328 | 10      | 3.9        | 5             | 9              | 2                    | yes       |
| ROUTEVIEW         |                     | [200]  | 10515 | 21455 | 2      | 4.1        | 5             | 10             | 2                    | no        |
| ITDK              |                     | [198]  | 190914 | 607610 | 155    | 6.4        | 14            | 26             | 4                    | yes       |
| CE-METABOLIC      |                     | [91]   | 484   | 4598  | 17      | 8.9        | 4             | 7              | 2                    | yes       |
| YEAST-PPI         | Biological          | [45]   | 2224  | 6609  | 57      | 6           | 6             | 11             | 4                    | no        |
| HOMO-PI           |                     | [180]  | 16635 | 115364| 135     | 13.87      | 5             | 10             | 2                    | no        |
| WEB-STANFORD      | Web                 | [142]  | 255265| 2234572| 1      | 15.2       | 82            | 164            | 0                    | yes       |
| WEB-NOTREDAM      |                     | [14]   | 325729 | 1497134 | 12    | 6.8        | 46            | 23             | 2                    | no        |
| AMAZON-1          | Product co-purchasing | [196]  | 334563 | 925872 | 21     | 5.5        | 24            | 47             | 1                    | no        |
| AMAZON-2          |                     | [196]  | 400727 | 3200440 | 194   | 11.7       | 11            | 20             | 5                    | no        |
| US-AIRLINES       | Infrastructure      | [20]   | 332   | 2126  | 1       | 12.8       | 3             | 6              | 0                    | yes       |
| POWER-GRID        |                     | [192]  | 4941  | 6594  | 1       | 2.7        | 23            | 46             | 0                    | yes       |

8.2 Comparison

We run the three algorithms for constructing eccentricity $k$-approximating trees on a set of real-world network datasets. The main parameters of each network is listed in Table 8.2. In Table 8.3, we introduce the results of the three algorithms for each network.

You can see in Table 8.3 that the three algorithms provide three trees with identical (or similar) qualities for each network. For nine networks (out of 21), the values of $k_{\text{max}}$ are equal and the values of $k_{\text{avg}}$ are very close. Those nine networks are the DUTCH-ELITE, EVA, EMAIL-EU, AS-GRAF-3, ROUTEVIEW, HOMO-PI, WEB-STANFORD, WEB-NOTREDAM, and US-AIRLINES.

For networks SLASHDOT, EMAIL-VIRGILI, EMAIL-ENRON, ITDK, and AMAZON-2, algorithm 3 constructs eccentricity $k$-approximating trees with slightly smaller values...
Table 8.3: Comparison of the four eccentricity $k$-approximating algorithms:

<table>
<thead>
<tr>
<th>Network</th>
<th>Algorithm 1</th>
<th>Algorithm 2</th>
<th>Algorithm 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$k_{max}$</td>
<td>$k_{avg}$</td>
<td>$k_{max}$</td>
</tr>
<tr>
<td>Dutch-Elite</td>
<td>6</td>
<td>2.35</td>
<td>6</td>
</tr>
<tr>
<td>Facebook</td>
<td>2</td>
<td>0.686</td>
<td>3</td>
</tr>
<tr>
<td>EVA</td>
<td>2</td>
<td>0.571</td>
<td>2</td>
</tr>
<tr>
<td>Slashdot</td>
<td>3</td>
<td>1.777</td>
<td>3</td>
</tr>
<tr>
<td>EMAIL-VIRGILI</td>
<td>4</td>
<td>2.729</td>
<td>4</td>
</tr>
<tr>
<td>EMAIL-enron</td>
<td>2</td>
<td>0.906</td>
<td>2</td>
</tr>
<tr>
<td>EMAIL-Eu</td>
<td>2</td>
<td>0.002</td>
<td>2</td>
</tr>
<tr>
<td>AS-Graph-1</td>
<td>3</td>
<td>1.791</td>
<td>2</td>
</tr>
<tr>
<td>AS-Graph-2</td>
<td>3</td>
<td>1.124</td>
<td>2</td>
</tr>
<tr>
<td>AS-Graph-3</td>
<td>2</td>
<td>0.828</td>
<td>2</td>
</tr>
<tr>
<td>Routeview</td>
<td>2</td>
<td>0.329</td>
<td>2</td>
</tr>
<tr>
<td>Itdk</td>
<td>4</td>
<td>2.108</td>
<td>5</td>
</tr>
<tr>
<td>CE-Metabolic</td>
<td>3</td>
<td>1.982</td>
<td>1</td>
</tr>
<tr>
<td>yeast-PPI</td>
<td>3</td>
<td>1.872</td>
<td>4</td>
</tr>
<tr>
<td>Homo-PI</td>
<td>2</td>
<td>0.747</td>
<td>2</td>
</tr>
<tr>
<td>web-Stanford</td>
<td>28</td>
<td>0.006</td>
<td>28</td>
</tr>
<tr>
<td>Web-Notredam</td>
<td>2</td>
<td>0.935</td>
<td>2</td>
</tr>
<tr>
<td>Amazon-1</td>
<td>6</td>
<td>0.991</td>
<td>7</td>
</tr>
<tr>
<td>Amazon-2</td>
<td>6</td>
<td>3.735</td>
<td>5</td>
</tr>
<tr>
<td>US-airlines</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Power-grid</td>
<td>13</td>
<td>5.735</td>
<td>4</td>
</tr>
</tbody>
</table>

for $k_{max}$. In fact, the only case for which the eccentricity $k$-approximating tree constructed by algorithm 3 have a $k_{max}$ value that is higher than that of Algorithm 2 is in network AS-GRAPH-2. This is probably due to the fact that Algorithm 3 takes into account the locality of the vertices when choosing which edges to add to the tree. This helps in avoiding increasing the eccentricity of each vertex by too much.
Chapter 9

Conclusions and Future Directions

Structural properties provide global metric quantities that can be used to describe a graph. Metric graph properties are based on the notion of shortest paths between vertices defined as the path through the smallest cost. Examples include the small-world property, the graph diameter, the graph average path length, and the vertex eccentricities. Another graph structural property that has been commonly investigated is the $\delta$-hyperbolicity, which is a geometric quantity that measures if a graph is negatively curved (hyperbolic). The graph $\delta$-hyperbolicity is related to its underlying hyperbolic geometry. In graphs, the $\delta$-hyperbolicity can be interpreted as a parameter that measures how close the metric structure of a graph is to the metric structure of a tree.

In this dissertation, we study the metric tree-likeness in graphs using Gromov’s $\delta$-hyperbolicity. We propose theoretical and practical applications that exploit this property. We mainly investigate the relationship between the $\delta$-hyperbolicity and the core-periphery structure in graphs (Part I). Moreover, we propose an eccentricity $k$-approximating (spanning) trees for two graph classes: the $\delta$-hyperbolic graphs and the $(\alpha_1, \Delta)$ graphs (Part II).
We propose two core-periphery partitioning models for $\delta$-hyperbolic graphs using their eccentricity-based bending property. We also empirically investigate the correlation between a graph’s $\delta$-hyperbolicity and its diameter and size. Then we use this analysis to classify graphs with respect to their $\delta$-hyperbolicity into three categories: strongly-hyperbolic graphs, hyperbolic graphs, and non-hyperbolic graphs.

Moreover, to reduce the time needed to compute the $\delta$-hyperbolicity in graphs we propose a method that can reduce the size of the input graph to only a subset that maximizes its $\delta$-hyperbolicity using the vertex local dominance relationship. Furthermore, we show that the $\delta$-hyperbolicity of a graph can be found in a set of quadruples that are in close proximity. We show that this set concentrates in the core. This implies that the core of a graph is the part that is responsible for maximizing its $\delta$-hyperbolicity.

We propose two heuristics for constructing eccentricity $k$-approximating trees with small values of $k$ that is related to $\delta$ for general unweighted graphs. Also using only metric properties of graphs, we show that every chordal graph ($\delta$-hyperbolicity $\leq 1$) admits an eccentricity 2-approximating spanning tree. Furthermore, we extend this result to a much larger family of graphs.

Finally, we compare the qualities of the eccentricity $k$-approximating tree algorithms proposed in this work. Throughout this dissertation, we analyze the $\delta$-hyperbolicity of several types of real-world and artificial networks using the heuristics proposed in each chapter.

We discuss open questions at the end of each chapter. Moreover, future work related to $\delta$-hyperbolicity in graphs is two fold. First, it will be interesting to identify and explore any relationship that may exist between the $\delta$-hyperbolicity value and other graph’s structural parameters. For example, trees (0-hyperbolic) have a clustering coefficient of zero. On the other hand, complete graphs (also 0-hyperbolic) have a clustering coefficient.
of one. Does this imply that graphs with high clustering (or low clustering) have strong hyperbolicity? It is interesting to explore the connection between $\delta$-hyperbolicity and the clustering coefficient of a graph in general.

Similarly, what is the relationship between $\delta$-hyperbolicity of a graph and its modularity. We know that the hyperbolicity of a graph is caused by quadruples that belong to its core. This motivates investigating what is the effect of the existence of multiple cores (belonging to different communities) on its hyperbolicity. In other words, is it the location of the core as most central to the network is what affects the hyperbolicity? Will this be affected by the existence of multiple cores?

Finally, can we use our eccentricity $k$-approximating trees to compute or approximate the value of the $\delta$-hyperbolicity of a graph? In the case of $(\alpha_1, \Delta)$-metric graphs, we obtained stronger upper bounds on the value of the distortion for approximating vertex eccentricities. Does this imply that knowing other structural properties of a $\delta$-hyperbolic graph will help in obtaining better upper bound distortions than what we provide?
Bibliography


[78] de Nooy W. The network data on the administrative elite in the netherlands in April-June 2006.


179


[139] Kunegis J. Prosper loans, KONECT, the Koblenz Network Collection, 2016.


[187] Williams V.V. Hardness of easy problems: basing hardness on popular conjectures such as the strong exponential time hypothesis, IPEC 2015, pp. 17–29.


[201] [http://deim.urv.cat/~aarenas/data/welcome.htm](http://deim.urv.cat/~aarenas/data/welcome.htm).

[202] [http://konect.uni-koblenz.de/networks/](http://konect.uni-koblenz.de/networks/).

[203] [http://digg.cs.tufts.edu/samples/](http://digg.cs.tufts.edu/samples/).

[204] [https://snap.stanford.edu/data/](https://snap.stanford.edu/data/).


[206] [http://www-personal.umich.edu/~mejn/netdata/](http://www-personal.umich.edu/~mejn/netdata/).