Strongly Hyperbolic Unit Disk Graphs

Thomas Bläsius
Karlsruhe Institute of Technology
Karlsruhe, Germany

Tobias Friedrich
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany

Maximilian Katzmann
Karlsruhe Institute of Technology
Karlsruhe, Germany

Daniel Stephan
GSV Algorithm Consulting UG (haftungsbeschränkt) & Co. KG
Potsdam, Germany

Abstract

The class of Euclidean unit disk graphs is one of the most fundamental and well-studied graph classes with underlying geometry. In this paper, we identify this class as a special case in the broader class of *hyperbolic unit disk graphs* and introduce *strongly hyperbolic unit disk graphs* as a natural counterpart to the Euclidean variant. In contrast to the grid-like structures exhibited by Euclidean unit disk graphs, strongly hyperbolic networks feature hierarchical structures, which are also observed in complex real-world networks.

We investigate basic properties of strongly hyperbolic unit disk graphs, including adjacencies and the formation of cliques, and utilize the derived insights to demonstrate that the class is useful for the development and analysis of graph algorithms. Specifically, we develop a simple greedy routing scheme and analyze its performance on strongly hyperbolic unit disk graphs in order to prove that routing can be performed more efficiently on such networks than in general.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Computational geometry; Mathematics of computing → Graph algorithms

Keywords and phrases hyperbolic geometry, unit disk graphs, greedy routing, hyperbolic random graphs, graph classes

Funding This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Grant No. 390859508.

1 Introduction

Studying networks in terms of *graph classes* based on certain properties is a fundamental tool in graph theory. Instead of having to consider all possible graphs, we can focus on the ones in a certain class, which allows us to get a more fine-grained understanding of their structural properties and the complexity of graph problems. Additionally, it facilitates the development of more efficient algorithms that are tailored towards the characteristics of the considered networks.

Different classes can be utilized in different contexts. For example, the characteristics of wireless communication networks are captured naturally in *Euclidean unit disk graphs* [19, 38], i.e., graphs where vertices can be identified with disks of equal size in the Euclidean plane and any two are adjacent if and only if their disks intersect. In this paper, we use the following formalization. Let $G = (V, E)$ be an undirected graph. A *(Euclidean) unit disk representation* of G is a mapping $\phi \colon V \to \mathbb{R}^2$ together with a *threshold radius* R such that $\{u, v\} \in E$ if and only if the distance between $\phi(u)$ and $\phi(v)$ is at most R. Then, the graph G is a *(Euclidean) unit disk graph* if it has a unit disk representation. In such graphs, the generally NP-complete problem of finding a maximum clique can be solved in polynomial time [19, 57], and routing can be performed more efficiently than in general graphs [39].

In this paper, we study a related graph class where the Euclidean ground space is replaced with the hyperbolic plane. The result is a generalization of the Euclidean variant, containing networks with a broader range of structural properties. Formally, a graph G is a *hyperbolic unit disk graph*, if there exists a *hyperbolic unit disk representation* $\phi \colon V \to \mathbb{H}^2$ together with a threshold radius R, such that $\{u, v\} \in E$, if and only if the hyperbolic distance between the vertex representations is at most $d_{\mathbb{H}^2}(\phi(u), \phi(v)) \leq R$. We note that the threshold radius R is part of the representation and can thus depend on the graph. The choice of R does not matter in Euclidean space, as scaling R and all coordinates $\phi(\cdot)$ by the same factor yields the same adjacencies. In contrast, there is no scaling operation in the hyperbolic plane that leaves relative distances intact.[1] As a result, the size of the considered region and the threshold radius *do* have an impact on the structure of the graphs in the hyperbolic setting.

To understand this effect, which is visualized in Figure 1, first consider some region, say a disk D of radius R', in the Euclidean plane and assume we distribute vertices evenly in D. Then the resulting Euclidean unit disk graph resembles a *grid-like* structure (with a density depending on the threshold radius R and the radius R' of D). That is, in the sparse setting, we only find small cliques, while separators and treewidth as well as the diameter are large, and we observe a homogeneity among the vertices, in the sense that all neighborhoods feature similar characteristics. Essentially, as in a grid, the graph looks the same no matter from which vertex it is viewed.

As the hyperbolic plane resembles the Euclidean plane locally, we can achieve the same grid-like structures by choosing a very small radius R' and an even smaller threshold radius R (Figure 1 (left)). In fact, by scaling the Euclidean unit disk representation of a graph into a sufficiently small region, we can realize the same adjacencies in the hyperbolic plane and obtain the following.

▶ **Theorem 1.** *Every Euclidean unit disk graph is a hyperbolic unit disk graph.*

[1] Under the common assumption that the curvature is -1, such a scaling operation does not exist. The term "unit disk" is still justified as we could instead fix $R = 1$ and allow for different curvatures.

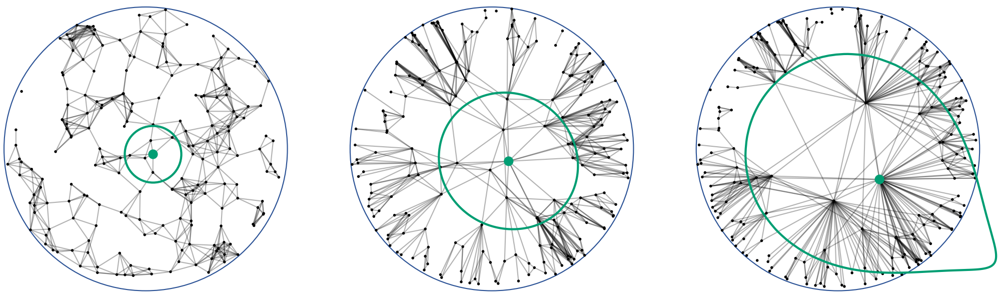

Figure 1 Hyperbolic unit disk graphs with different ground space and threshold radii. The representations have been scaled such that the ground spaces *appear* to have the same size, while their actual sizes are denoted by R'. **(Left)** The ground space is very small and the threshold radius even smaller, leading to grid-like structures. **(Center)** Ground space and threshold radius are increased, hierarchies start form but grid like structures remain. **(Right)** Ground space and threshold have the same large value, leading to hierarchical structures.

Beyond that, we can increase the radii R' and R. Then, the grid-like structures start to vanish and *hierarchical* structures begin to form (Figure 1 (center)). Eventually, we reach the *strongly hyperbolic* setting where only hierarchical and no grid-like structures remain (Figure 1 (right)). There, vertices are rather heterogeneous, with respect to their degree and what neighborhoods look like. The diameter is small, while large cliques can form. We note that the treewidth is also large, just as in grid-like graphs. However, in hierarchical graphs this is an artifact of the large cliques, while in grid-like graphs we observe large treewidth *despite* the fact that only small cliques form. In hierarchical networks vertices connect via hubs, which connect via larger hubs, and so on. The hubs explicitly exhibit the hierarchy in the graph structure. As a result, the graph looks very different when viewed from vertices on different levels in the hierarchy.

Formally, we say that a graph is a *strongly hyperbolic unit disk graph* if it admits a hyperbolic unit disk representation in which ϕ maps all vertices to points within a disk whose radius matches the threshold ($R' = R$). For better understanding, we recommend using the interactive visualization[2], which lets the user change the size of the ground space, allowing to smoothly transition between Euclidean and strongly hyperbolic unit disk graphs.

To paint the big picture, hyperbolic unit disk graphs comprise two extremes: Euclidean unit disk graphs with grid-like structures on one side and strongly hyperbolic unit disk graphs with hierarchical structures on the other. Therefore, if we want to design algorithms for grid-like structures, it makes sense to analyze them on Euclidean unit disk graphs. For hierarchical structures, strongly hyperbolic unit disk graphs are a good choice.

Related Concepts.

To the best of our knowledge, intersection graphs of hyperbolic unit disks, or hyperbolic unit balls, have so far only been considered by Kisfaludi-Bak [42]. There, for every $\rho > 0$, a graph is said to be in the graph class $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ (*UBG = unit ball graph*) if its vertices can be mapped into $\mathbb{H}^d$ such that vertices have distance at most 2ρ if and only if they are adjacent. There are two core differences compared to our definition of hyperbolic unit disk

[2] `https://thobl.github.io/hyperbolic-unit-disk-graph/`

graphs. First, it allows for higher dimensions. Secondly, it is parameterized by the radius, i.e., $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ describes an infinite family of graph classes rather than a single class.

This second difference is somewhat subtle but rather important. Consider the class $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ for a fixed radius ρ. Moreover, assume we want to study graphs in $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ that are sparse; for the sake of argument, assume constant average degree. Then, for an increasing number of vertices n, the region of $\mathbb{H}^d$ spanned by the vertices has to grow, as otherwise the density of the graph grows with n. Thus, for sufficiently large n, the radius ρ is arbitrarily small compared to the region spanned by the vertices, yielding grid-like structures (see discussion above). Thus, for fixed ρ, large graphs in $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ are grid-like rather than hierarchical. This means that asymptotic statements for the classes $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ do not translate to the hierarchical structures in the class of strongly hyperbolic unit disk graphs.

A second related concept are hyperbolic random graphs [47], which are basically random strongly hyperbolic unit disk graphs. A hyperbolic random graph is obtained by assigning each vertex a random point in a disk of radius $R \approx 2\log(n)$ and connecting two vertices if and only if their distance is at most R. This yields graphs that resemble certain real-world networks, as they have small diameter, high clustering, and a power-law degree distribution whose exponent can be adjusted using the probability distribution of the vertex positions. This is not the case for graphs in $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$ as the connection radius cannot increase with the graph size. Nevertheless, every hyperbolic random graph is a strongly hyperbolic unit disk graph and thus any statement shown for the latter also holds for the former.

Hyperbolic random graphs have also been studied in a noisy setting, where, with some small probability, distant vertices are adjacent and close vertices are not adjacent. Similarly, Kisfaludi-Bak [42] also studies a noisy variant of the class $\mathrm{UBG}_{\mathbb{H}^d}(\rho)$. It would be interesting to also study (strongly) hyperbolic unit disk graphs in a noisy setting. This is, however, beyond the scope of this paper and left for future research.

Contribution.

Beyond the generalization of Euclidean unit disk graphs to hyperbolic unit disk graphs, we identify strongly hyperbolic unit disk graphs as a natural counterpart to the Euclidean special case and provide the first insights into their structural and algorithmic properties. In particular, we study fundamental criteria relating the coordinates of vertices to their adjacency and investigate the formation of cliques (Section 2).

Using these insights, we follow up on prior empirical efforts towards understanding how an underlying hyperbolic geometry facilitates efficient routing on internet-like networks [13, 52], and utilize strongly hyperbolic unit disk graphs to obtain theoretical performance guarantees, proving that routing in such networks can be performed more efficiently than in general (Section 3). While similar results have been obtained on the grid-like Euclidean unit disk graphs [39], our analysis covers networks with hierarchical structures.

In particular, it includes hyperbolic random graphs, which are used to represent real-world complex networks like the internet [13], where routing plays an important role. By developing a simple routing scheme, which is interesting in its own right, we show that greedy routing on such graphs can be performed with a stretch of at most 3, while asymptotically almost surely requiring at most $\mathcal{O}(\log^4 n)$ bits of storage per vertex and taking $\mathcal{O}(\log^2 n)$ time per routing decision.

2 (Strongly) Hyperbolic Unit Disk Graphs

Throughout the paper we consider the *polar-coordinate model* of the hyperbolic plane $\mathbb{H}^2$. There, we have a designated *pole* $O \in \mathbb{H}^2$, together with a *polar axis*, i.e., a reference ray starting at O. A point p is identified by its *radius* $r(p)$, denoting the hyperbolic distance to O, and its *angle* $\varphi(p)$, denoting the angular distance between the polar axis and the ray from O through p. In our figures we interpret these values as polar coordinates in the Euclidean plane. The disk of radius R centered at p is denoted by $D_R(p)$. When $p = O$ we simply write D_R. The hyperbolic distance between points p and q is given by

$$d_{\mathbb{H}^2}(p,q) = \operatorname{acosh}\Big(\cosh\big(r(p)\big)\cosh\big(r(q)\big) - \sinh\big(r(p)\big)\sinh\big(r(q)\big)\cos\big(\Delta_\varphi(p,q)\big)\Big), \quad (1)$$

where $\cosh(x) = (e^x + e^{-x})/2$, $\sinh(x) = (e^x - e^{-x})/2$, and $\Delta_\varphi(p,q) = \pi - |\pi - |\varphi(p) - \varphi(q)||$ denotes the angular distance between p and q. Without loss of generality, we assume that the representation ϕ of a strongly hyperbolic unit disk graph maps the vertices into a disk of radius R that is centered at O. For the sake of readability, we typically associate a vertex v with its mapping $\phi(v)$ and denote the set of vertices lying in a region $A \subseteq D_R$ with $V(A)$.

2.1 Relation to Euclidean Unit Disk Graphs

We start with the proof of Theorem 1, stating that every Euclidean unit disk graph is also a hyperbolic unit disk graph. To this end, we utilize the *Poincaré disk* model of hyperbolic space. There, the infinite two-dimensional hyperbolic plane is mapped to the interior of the unit circle in the Euclidean plane, which is referred to as the Poincaré disk $\mathbb{D}$. In this model, points are identified using Cartesian coordinates. Given two points $p, q \in \mathbb{D}$, we can compute their hyperbolic distance by interpreting them as vectors and computing

$$d_{\mathbb{D}}(p,q) = 2\operatorname{asinh}\left(\frac{||p-q||}{\sqrt{(1-||p||^2)(1-||q||^2)}}\right), \quad (2)$$

where $||\cdot||$ denotes the Euclidean norm.

Proof of Theorem 1. Let $G = (V, E)$ be a Euclidean unit disk graph with representation $\phi_E \colon V \to \mathbb{R}^2$ and threshold radius R_E. To prove that G is a hyperbolic unit disk graph, we show that there exists a hyperbolic unit disk representation $\phi_H \colon V \to \mathbb{D}$ of G with threshold radius R_H. First, note that we can take a disk of radius $\rho \in (0,1)$ in the Euclidean plane and scale the coordinates ϕ_E and threshold R_E by a positive factor such that all vertices lie in this disk, while maintaining a valid Euclidean unit disk representation of G with coordinates ϕ_E^ρ and threshold radius R_E^ρ. We now set $\phi_H \coloneqq \phi_E^\rho$ for a sufficiently small ρ. In the following, we identify a vertex $v \in V$ with its coordinate $\phi_H(v)$, for the sake of readability.

To conclude the proof, it remains to show that there exists an R_H such that for two vertices $u, v \in V$, we have $d_{\mathbb{D}}(u,v) \le R_H$ if and only if their Euclidean distance is at most $d_E(u,v) \le R_E^\rho$. Note that when u and v are not adjacent, there exists a $\tau > 1$ such that $d_E(u,v) > \tau \cdot R_E^\rho$. In the following, we determine upper and lower bounds on $d_{\mathbb{D}}(u,v)$ in terms of $d_E(u,v)$ and show that, with decreasing ρ, they approach each other faster than τ approaches 1. Eventually, the bounds are sufficiently tight, such that scaling the lower bound by τ yields something larger than the upper bound, allowing us to find a threshold R_H that fits between the two.

The hyperbolic distance between u and v can be computed via Equation (2). Note that $||u - v|| = d_E(u,v)$. Moreover, since ϕ_H maps all vertices to points in a disk of radius ρ, we have $0 \leq ||u||, ||v|| \leq \rho$ and thus

$$(1-\rho^2) = \sqrt{(1-\rho^2)(1-\rho^2)} \leq \sqrt{(1-||u||^2)(1-||v||^2)} \leq 1,$$

from which we can derive that

$$2\operatorname{asinh}(d_E(u,v)) \leq d_{\mathbb{D}}(u,v) \leq 2\operatorname{asinh}\left(\frac{1}{1-\rho^2} d_E(u,v)\right).$$

Since $\operatorname{asinh}(x) \leq x$ for all $x \geq 0$, we can set $\hat{g}(\rho) := 2/(1-\rho^2)$ and simplify the upper bound to $d_{\mathbb{D}}(u,v) \leq \hat{g}(\rho) \cdot d_E(u,v)$. In order to simplify the lower bound, we use a Taylor-approximation of $\operatorname{asinh}(x)$ around 0 and express the remainder using the Lagrange form (see [4, Equations 25.2.24 and 25.2.25]), in which case there exists a $\xi \in (0,x)$ such that

$$\operatorname{asinh}(x) = x - \frac{\xi}{2(1+\xi^2)^{3/2}} \cdot x^2.$$

Since the factor is monotonically increasing for $\xi \in [0, 1/2]$, we can choose $\rho \leq 1/2$ sufficiently small such that $0 \leq \xi \leq d_E(u,v) \leq 2\rho$, allowing us to bound

$$\begin{aligned} 2\operatorname{asinh}(d_E(u,v)) &\geq 2 \cdot \left(d_E(u,v) - \frac{\rho}{(1+4\rho^2)^{3/2}} d_E(u,v)^2\right) \\ &\geq 2 \cdot \left(1 - \frac{\rho}{(1+4\rho^2)^{3/2}}\right) d_E(u,v) =: \check{g}(\rho) \cdot d_E(u,v). \end{aligned}$$

where the second inequality follows from the fact that for $\rho \leq 1/2$ we have $d_E(u,v) \leq 2\rho \leq 1$ and thus $d_E(u,v)^2 \leq d_E(u,v)$. We obtain

$$\check{g}(\rho) \cdot d_E(u,v) \leq d_{\mathbb{D}}(u,v) \leq \hat{g}(\rho) \cdot d_E(u,v).$$

Now consider the case where $\{u,v\} \in E$. Then, we have

$$\begin{aligned} & d_E(u,v) \leq R_E^\rho \\ \Leftrightarrow\quad & \hat{g}(\rho) \cdot d_E(u,v) \leq \hat{g}(\rho) \cdot R_E^\rho \\ \Rightarrow\quad & d_{\mathbb{D}}(u,v) \leq \hat{g}(\rho) \cdot R_E^\rho, \end{aligned}$$

where the first step follows from the fact that $\hat{g}(\rho) > 0$ for $\rho < 1$, and the second step is due to the above inequality. On the other hand, if $\{u,v\} \notin E$, we have

$$\begin{aligned} & d_E(u,v) > \tau \cdot R_E \\ \Leftrightarrow\quad & \check{g}(\rho) \cdot d_E(u,v) > \tau \cdot \check{g}(\rho) \cdot R_E^\rho \\ \Rightarrow\quad & d_{\mathbb{D}}(u,v) > \tau \cdot \check{g}(\rho) \cdot R_E^\rho, \end{aligned}$$

where the first step is valid since $\check{g}(\rho) > 0$ for $\rho > 0$, and the second step, again, follows from the above inequality. It follows that, if there exists a $\rho^* > 0$ such that $\hat{g}(\rho^*) < \tau \cdot \check{g}(\rho^*)$, then there also exists an $R_H \in [\hat{g}(\rho^*), \tau \cdot \check{g}(\rho^*)] \cdot R_E^{\rho^*}$, such that ϕ_H and R_H yield a valid hyperbolic unit disk graph representation of G.

Note that $\hat{g}(\rho) < \tau \cdot \check{g}(\rho)$ holds for $\rho = 0$. Since both functions are continuous on $[0,1)$, so is the function $h(\rho) = \tau \cdot \check{g}(\rho) - \hat{g}(\rho)$, with $h(0) > 0$. By applying the (ε, δ)-definition of continuity, we can derive that there exist a $\delta > 0$ such that for all $\rho \in (-\delta, \delta)$ we have $|h(0) - h(\rho)| \leq \varepsilon$ for $\varepsilon = h(0)/2 > 0$. In particular, this means that there exists a $\rho^* > 0$ such that $h(\rho^*) > 0$, implying that $\hat{g}(\rho^*) < \tau \cdot \check{g}(\rho^*)$. ◀

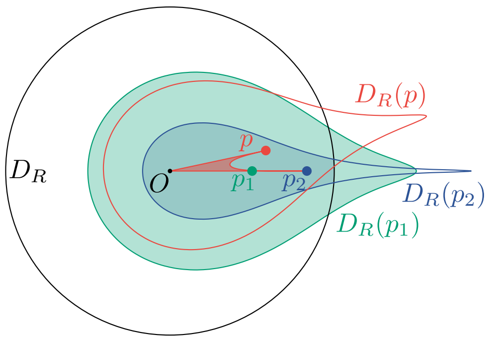

Figure 2 Visualization of the proof of Lemma 2. Point p_1 has a smaller radius than p_2, both having the same angular coordinate. Consequently, $D_R(p_1)$ (green region) is a superset of $D_R(p_2) \cap D_R$ (blue region). The triangle formed by the points p, p_2, and O is contained in $D_R(p)$ (both red).

2.2 Adjacency

Similar results to the ones described in this subsection have been determined on hyperbolic random graphs before (see, e.g., [36]). Here we verify under which requirements they also hold on strongly hyperbolic unit disk graphs. By definition, two vertices in a strongly hyperbolic unit disk graph G are adjacent, if and only if their hyperbolic distance is at most R. Consequently, we can imagine that each vertex v is equipped with a neighborhood disk $D_R(v)$. That is, $N(v) = V(D_R(v))$. The following lemma shows that moving such a neighborhood disk closer to the center of D_R only increases the region of D_R that it covers.

▶ **Lemma 2.** *Let R be a radius and let $p_1, p_2 \in D_R$ be points with $r(p_1) \leq r(p_2)$ and $\varphi(p_1) = \varphi(p_2)$. Then, $D_R(p_1) \supseteq D_R(p_2) \cap D_R$.*

Proof. Let $p \in D_R(p_2) \cap D_R$ be a point and note that $d_{\mathbb{H}^2}(p, p_2) \leq R$. Now consider the triangle spanned by the points p, p_2, and the origin O. This triangle is completely contained in the disk $D_R(p)$, as $d_{\mathbb{H}^2}(p, p_2) \leq R$ and $r(p) \leq R$, as shown in Figure 2. Since disks are convex and p_1 lies on the line from O to p_2, it is part of the triangle and therefore also contained in the disk. Consequently, $d_{\mathbb{H}^2}(p, p_1) \leq R$ and thus $p \in D_R(p_1)$. ◀

Consequently, moving a vertex towards the center does not decrease its neighborhood.

▶ **Corollary 3.** *Let G be a strongly hyperbolic unit disk graph with radius R and let v_1, v_2 be vertices with $r(v_1) \leq r(v_2) \leq R$ and $\varphi(v_1) = \varphi(v_2)$. Then, $N(v_1) \supseteq N(v_2)$.*

In the following, we investigate in greater detail under which circumstances two vertices are adjacent. Consider two vertices v_1 and v_2 in G with radii r_1 and r_2, respectively. Clearly, the two are adjacent if $r_1 + r_2 \leq R$. When $r_1 + r_2 > R$, the hyperbolic distance between them depends on their angular distance $\Delta_\varphi(v_1, v_2)$. More precisely, for vertices of fixed radii, increasing the angular distance also increases the hyperbolic distance. Let $\theta(r_1, r_2)$ denote the angular distance, such that the hyperbolic distance between v_1 and v_2 is exactly R. That is, for $\Delta_\varphi(v_1, v_2) \leq \theta(r_1, r_2)$ we have $d_{\mathbb{H}^2}(v_1, v_2) \leq R$, meaning v_1 and v_2 are adjacent. We can compute $\theta(r_1, r_2)$ by using the hyperbolic distance function in Equation (1), setting the distance equal to R, and solving for the angular distance. That is,

$$\theta(r_1, r_2) = \operatorname{acos}\left(\frac{\cosh(r_1)\cosh(r_2) - \cosh(R)}{\sinh(r_1)\sinh(r_2)}\right). \tag{3}$$

Tight asymptotic bounds on $\theta(r_1, r_2)$ have been derived before [48, Lemma 3.2]. The following lemma holds for all $R > 0$.

▶ **Lemma 4.** *Let $R > 0$ and $r_1, r_2 \in (0, R]$ with $r_1 + r_2 \geq R$ be given. Then,*

$$2\sqrt{e^{R-r_1-r_2} + e^{-R-r_1-r_2} - (e^{-2r_1} + e^{-2r_2})} \leq \theta(r_1, r_2) \leq \pi\sqrt{e^{R-r_1-r_2}}.$$

Proof. We start by applying the cosine function on both sides of Equation (3), which makes it easier to deal with the right hand side for now. This yields

$$\cos(\theta(r_1, r_2)) = \frac{\cosh(r_1)\cosh(r_2) - \cosh(R)}{\sinh(r_1)\sinh(r_2)}. \tag{4}$$

We consider the upper bound on $\theta(r_1, r_2)$ first. Note that we aim to eventually apply the inverse cosine function to revert the above step. Since this function is monotonically decreasing, we first determine a *lower* bound on $\cos(\theta(r_1, r_2))$, in order to obtain an upper bound on $\theta(r_1, r_2)$. Recall that $\cosh(x) = (e^x + e^{-x})/2$ and $\sinh(x) = (e^x - e^{-x})/2$, and note that $\sinh(x) \leq e^x/2$. Thus, the above equation can be bounded by

$$\begin{aligned}
\cos(\theta(r_1, r_2)) &\geq \frac{1/4(e^{r_1} + e^{-r_1})(e^{r_2} + e^{-r_2}) - 1/2(e^R + e^{-R})}{1/4e^{r_1+r_2}} \\
&= \frac{e^{r_1+r_2} + e^{r_1-r_2} + e^{r_2-r_1} + e^{-r_1-r_2} - 2e^R - 2e^{-R}}{e^{r_1+r_2}} \\
&= 1 - 2e^{R-r_1-r_2} + e^{-2r_1} + e^{-2r_2} + e^{-2(r_1+r_2)} - 2e^{-R-r_1-r_2}.
\end{aligned}$$

We now argue that the remaining expression can be bounded by dropping the last four terms since their sum is non-negative. First note that $e^x \geq 0$ for all $x \in \mathbb{R}$. Consequently, the second to last term is non-negative and it remains to show that $e^{-2r_1} + e^{-2r_2} \geq 2e^{-R-r_1-r_2}$, which can be done by showing that $e^{-2r_1}, e^{-2r_2} \geq e^{-R-r_1-r_2}$. In the following, we show that this is the case for e^{-2r_1}. The proof for e^{-2r_2} is analogous. Note that $r_1 - r_2 \leq R$, since $r_1, r_2 \in (0, R]$ by assumption. It follows that $r_1 \leq R + r_2$ and thus $e^{-2r_1} \geq e^{-R-r_1-r_2}$. We can conclude that $\cos(\theta(r_1, r_2)) \geq 1 - 2e^{R-r_1-r_2}$. The claimed upper bound now follows by applying the inverse cosine and observing that $\mathrm{acos}(1 - x) \leq \pi\sqrt{x/2}$ holds for all $x \in [0, 2]$.

It remains to prove that the claimed lower bound on $\theta(r_1, r_2)$ is valid. Again, we start with Equation (4). However, this time we determine an *upper* bound on $\cos(\theta(r_1, r_2))$. First, we apply the identity

$$\cosh(x)\cosh(y) = \sinh(x)\sinh(y) + \cosh(x - y),$$

which yields

$$\begin{aligned}
\cos(\theta(r_1, r_2)) &= \frac{\sinh(r_1)\sinh(r_2) + \cosh(r_1 - r_2) - \cosh(R)}{\sinh(r_1)\sinh(r_2)} \\
&= 1 - \frac{\cosh(R) - \cosh(r_1 - r_2)}{\sinh(r_1)\sinh(r_2)}.
\end{aligned}$$

Using the definition of cosh and the fact that $\sinh(x) \leq e^x/2$ to conclude that

$$\begin{aligned}
\cos(\theta(r_1, r_2)) &\leq 1 - \frac{1/2(e^R + e^{-R}) - 1/2(e^{r_1-r_2} + e^{r_2-r_1})}{1/4e^{r_1+r_2}} \\
&= 1 - 2(e^{R-r_1-r_2} + e^{-R-r_1-r_2} - (e^{-2r_2} + e^{-2r_1}))
\end{aligned}$$

The claim then follows by applying the inverse cosine function and observing that $\mathrm{acos}(1-x) \geq \sqrt{2x}$ is valid for all $x \in [0, 2]$. ◀

We note that, while the above bounds are easier to work with than the exact function and are generally applicable due to the few constraints on the considered radii, the lower bound is still a bit tedious to work with. However, by introducing some minor requirements, we can obtain a slightly weaker bound that can be worked with more easily.

▶ **Corollary 5.** *Let $R \geq 1$ and $r_1, r_2 \in (0, R]$ with $r_1 + r_2 \geq R$ and $|r_1 - r_2| \leq R - 1$ be given. Then,*

$$\sqrt{e^{R-r_1-r_2}} \leq \theta(r_1, r_2) \leq \pi\sqrt{e^{R-r_1-r_2}}.$$

Proof. The upper bound immediately follows from Lemma 4. By utilizing the lower bound from the same lemma, we obtain

$$\begin{aligned}\theta(r_1, r_2) &\geq 2\sqrt{e^{R-r_1-r_2} + e^{-R-r_1-r_2} - (e^{-2r_1} + e^{-2r_2})}. \\ &\geq 2\sqrt{e^{R-r_1-r_2} - (e^{-2r_1} + e^{-2r_2})}. \\ &= 2\sqrt{e^{R-r_1-r_2}\left(1 - e^{-R}(e^{r_1-r_2} + e^{-(r_1-r_2)})\right)},\end{aligned}$$

where the second inequality is valid since $e^{-R-r_1-r_2} \geq 0$. To prove the claim, it thus suffices to show that the remaining negative part is at most $3/4$, which can be done as follows. First note that

$$e^{-R}(e^{r_1-r_2} + e^{-(r_1-r_2)}) = 2e^{-R} \cdot \frac{1}{2}(e^{r_1-r_2} + e^{-(r_1-r_2)}) = 2e^{-R} \cdot \cosh(r_1 - r_2).$$

Now note that $\cosh(x)$ is symmetric about the y-axis and thus $\cosh(r_1 - r_2) = \cosh(|r_1 - r_2|)$. Moreover, since $\cosh(x)$ is monotonically increasing for $x \geq 0$, we can utilize the assumption that $|r_1 - r_2| \leq R - 1$ to conclude

$$e^{-R}(e^{r_1-r_2} + e^{-(r_1-r_2)}) \leq 2e^{-R} \cdot \cosh(R - 1).$$

Finally, since $\cosh(x) = 1/2(e^x + e^{-x}) \leq e^x$ for all $x \geq 0$, we obtain

$$e^{-R}(e^{r_1-r_2} + e^{-(r_1-r_2)}) \leq 2e^{-R} \cdot e^{R-1} = 2/e \leq 3/4.$$ ◀

Apart from the above bounds, we highlight another property of the function $\theta(r_1, r_2)$, for the special case where $r_1 = r_2$.

▶ **Lemma 6.** *The function $\theta(r, r)$ is monotonically decreasing for $r \geq 0$.*

Proof. Consider the definition of $\theta(r_1, r_2)$ in Equation (3). By utilizing the fact that $r_1 = r_2 = r$, the equation simplifies to

$$\theta(r, r) = \operatorname{acos}\left(\frac{\cosh(r)^2 - \cosh(r)}{\sinh(r)^2}\right).$$

We can now apply the identities $\cosh(x)^2 = (\cosh(2x)+1)/2$ and $\sinh(x)^2 = (\cosh(2x)-1)/2$, both being valid for $x \in \mathbb{R}$, to obtain

$$\begin{aligned}\theta(r, r) &= \operatorname{acos}\left(\frac{1/2(\cosh(2r) + 1) - \cosh(r)}{1/2(\cosh(2r) - 1)}\right) \\ &= \operatorname{acos}\left(\frac{(\cosh(2r) + 1) - 2\cosh(r)}{\cosh(2r) - 1}\right) \\ &= \operatorname{acos}\left(\frac{\cosh(2r) - 1 + 2 - 2\cosh(r)}{\cosh(2r) - 1}\right) \\ &= \operatorname{acos}\left(1 - 2\frac{\cosh(r) - 1}{\cosh(2r) - 1}\right).\end{aligned}$$

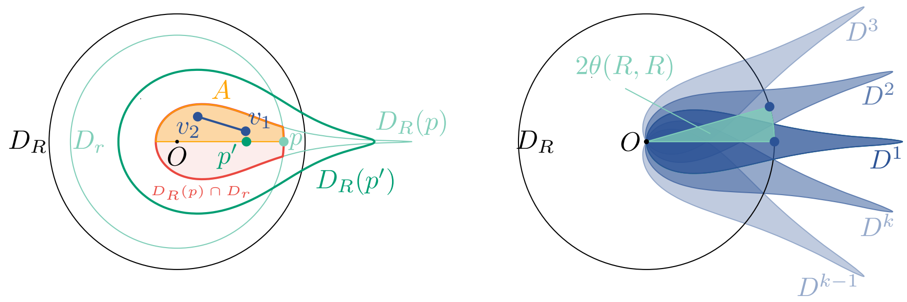

Figure 3 Covering strongly hyperbolic unit disk graphs with cliques. **(Left)** Visualization of the proof of Lemma 7. Vertices v_1, v_2 (blue) are in the half A (orange) of the region $D_R(p) \cap D_r$ (red) and are adjacent. **(Right)** Visualization of the proof of Lemma 8, showing five of the k disks $D^1, \dots, D^k$ (blue) and the angular distance between two consecutive centers (green).

Further, utilizing the fact that

$$\frac{\cosh(x) - 1}{\cosh(2x) - 1} = \frac{1}{2\cosh(x) + 2},$$

which is valid for all $x \in \mathbb{R}$, the above term can be simplified to

$$\theta(r, r) = \operatorname{acos}\left(1 - \frac{1}{\cosh(r) + 1}\right).$$

Note that $\cosh(x)$ is monotonically increasing for $x \geq 0$, and so is the argument in the inverted cosine. The claim follows as acos is monotonically decreasing. ◀

2.3 Cliques

In the following, we examine how the underlying geometry affects the formation of cliques. We start by showing that the vertices lying in $D_R(p)$, having smaller radius than p, form two cliques. More precisely, we say that a vertex set $S \subseteq V$ can be *covered by k cliques*, if there exists a partitioning $S_1, \dots, S_k$ of S such that the induced subgraphs $G[S_i]$ for $i \in [k]$ are cliques.

▶ **Lemma 7.** *Let G be a strongly hyperbolic unit disk graph with radius $R > 0$ and let $p \in D_R$ be a point with $r(p) = r$. Then, $V(D_R(p) \cap D_r)$ can be covered by two cliques.*

Proof. Assume without loss of generality that $\varphi(p) = 0$. We divide the region $D_R(p) \cap D_r$ into two halves A and A' containing all points with angles in $[0, \pi)$ and $[\pi, 2\pi)$, respectively, as illustrated in Figure 3 (left). The goal now is to show that the vertices in $V(A)$ and the ones in $V(A')$ induce a clique. More precisely, we show that this is the case for A. For symmetry reasons this then also holds for A'. Consider two vertices $v_1, v_2 \in A$ and assume without loss of generality that $\varphi(v_1) \leq \varphi(v_2)$. Since $v_2 \in A \subseteq D_R(p)$ and since by Lemma 2 moving $D_R(p)$ towards the origin increases the region of D_R that it covers, we know that v_2 is contained in the disk $D_R(p')$ for $p' = (r(v_1), 0)$ (dark green in Figure 3 (left)). It follows that $d_{\mathbb{H}^2}(p', v_2) \leq R$. Note that v_1 has the same radius as p' and that $\Delta_\varphi(p', v_2) \geq \Delta_\varphi(v_1, v_2)$. As established above, decreasing the angular distance between two points with fixed radii decreases their hyperbolic distance. Therefore, $d_{\mathbb{H}^2}(v_1, v_2) \leq d_{\mathbb{H}^2}(p', v_2) \leq R$, meaning v_1 and v_2 are adjacent. ◀

We note that the above lemma implies that for a vertex v, the neighbors with smaller radius than v form two cliques. We continue by investigating the number of cliques required to cover a strongly hyperbolic unit disk graph.

▶ **Lemma 8.** *Let G be a strongly hyperbolic unit disk graph with radius $R > 0$. Then, G can be covered by* $\max\{2\pi\sqrt{2}, 2\pi e^{R/2}\}$ *cliques.*

Proof. To prove the claim, we utilize the underlying geometry by covering the ground space D_R with a set of k disks $D^1, \ldots, D^k$, such that each $V(D^i)$ for $i \in [k]$ can be covered by two cliques. All of these disks have radius R and their centers lie on the boundary of the disk D_R. The center of the first disk has an angular coordinate of 0. All other disks D^i are placed at an angular distance of $2\theta(R, R)$ to their predecessor D^{i-1} in counterclockwise direction. See Figure 3 (right) for an illustration. As a consequence, the boundaries of two consecutive disks intersect on the boundary of D_R, which is therefore covered completely by the k disks. It follows that each vertex is contained in at least one disk D^i.

Since by Lemma 7 each $V(D^i)$ for $i \in [k]$ can be covered by two cliques, it suffices to show that $k \leq \max\{\pi\sqrt{2}, \pi e^{R/2}\}$ in order to finish the proof. To this end, recall that two consecutive disks are placed at an angular distance of $2\theta(R, R)$. Consequently, it takes $k = 2\pi/(2\theta(R, R)) = \pi/\theta(R, R)$ disks to cover the whole disk D_R. Using Lemma 4 we can conclude

$$\theta(R, R) \geq 2\sqrt{e^{-R} + e^{-3R} - 2e^{-2R}} = 2\sqrt{\left(e^{-R/2} - e^{-3/2 \cdot R}\right)^2} = 2e^{-R/2}(1 - e^{-R}).$$

It follows that k can be bounded by

$$k = \frac{\pi}{\theta(R, R)} \leq \frac{\pi}{2e^{-R/2}\left(1 - e^{-R}\right)} = \pi e^{R/2} \cdot \frac{1}{2\left(1 - e^{-R}\right)}. \tag{5}$$

We now distinguish between two cases depending on the size of R and start with $R < \log(2)$. Recall that the function $\theta(R, R)$ is monotonically decreasing in R (see Lemma 6). As a consequence, we have $\theta(R, R) \geq \theta(\log(2), \log(2))$. Then, it follows that

$$k \leq \frac{\pi}{\theta(\log(2), \log(2))} \leq \pi e^{\log(2)/2} \frac{1}{2\left(1 - e^{-\log(2)}\right)} = \pi\sqrt{2},$$

which we account for with the first part of the maximum. When $R \geq \log(2)$, note that we have $(1 - e^{-R}) \geq 1/2$. Consequently, we can bound the last fraction in Equation (5) by 1, which yields the claim. ◀

3 Routing

While finding a path between two vertices in an undirected network is typically rather simple, the internet is decentralized and does not allow for the use of a central data structure. Instead each vertex can only use local information to perform a *routing decision*, i.e., the decision to which vertex information is forwarded next such that it eventually reaches the target. This situation is further complicated by the fact that the internet consists of billions of vertices. In order to be able to handle a network of this scale, a routing scheme has to be optimized with respect to three criteria: the *space requirement* (the amount of information that the scheme uses to forward information), the *query time* (the time it takes to make a routing decision), and the *stretch* (how much longer the routed path is compared to a shortest path in the network, where the *length* of a path denotes the number of contained edges). Formally, a path between two vertices has *multiplicative stretch* $c \geq 1$ if it is at most c times longer than a shortest path between them. An *additive stretch* of ℓ denotes that the routed path contains at most ℓ more vertices than a shortest path. A *multiplicative stretch c with additive bound ℓ* denotes that the routed paths have stretch c *or* additive stretch ℓ. Note that this implies a multiplicative stretch of $\max\{c, 1+\ell\}$.

3.1 A Brief History of Routing

In the following, we summarize the main approaches to adjusting the trade-off between stretch and space requirements. The query times of the considered schemes are at most polylogarithmic. Figure 4 gives an overview of existing schemes.

Routing Schemes. In general networks, routing with a stretch of 1, i.e., always routing along the shortest paths, requires storing $\Theta(n^2 \log n)$ bits in total [34]. The most commonly used approach to reducing the required space is to only store shortest path information for certain vertex pairs. That is, a representation of a subgraph (typically a tree or a collection of trees) of the original graph is stored and the routing takes place on the subgraph. This is usually done by selecting a set of *landmark* (or *pivot*) vertices. Then, for each vertex only the information about how to get to the closest landmark is stored [1, 2, 3, 10, 11, 21, 26, 56, 66]. These schemes basically partition the graph based on the landmark vertices. A related approach starts with a partition and defines the landmarks afterwards [55, 59]. The scheme then routes via the landmarks closest to the source and target. This general approach can be optimized in several ways. First, the network can be partitioned with several levels of granularity, such that messages that need to travel larger distances are routed to landmarks whose associated vertex set is larger [10, 55]. Improvements for shorter distances can be obtained by storing the actual shortest path information for vertices in close vicinity of each vertex [10, 26]. Moreover, the selection of the landmarks itself can have an impact on the performance of the routing scheme. In general graphs they are typically selected at random. A more careful selection can lead to better results on Erdös-Rényi random graphs [27], or when assuming that the network has certain properties like a power-law degree distribution [18, 64]. Similarly, better results can be obtained on chordal graphs [23, 24, 25].

Closely related to routing schemes are *approximate distance oracles*. There, we are only interested in the length of a short path instead of the path itself. Again, the commonly used techniques are based on landmarks [6, 22, 53, 67], and compared to general networks, better results can be obtained when assuming that the considered graphs have certain properties like being planar [65], or being sparse (although at the expense of an increased query time) [5].

In general, routing schemes and distance oracles are based on one central data structure that holds the information required when forwarding messages, which can become an issue with increasing network size as it has been shown that, on general graphs, achieving a stretch of $c \geq 1$ requires a data structure of size $\Omega(n^{1+1/(2c+4)})$ [55]. One approach to overcoming this problem is to consider local routing schemes instead.

Local and Greedy Routing Schemes. In local routing schemes the routing information is distributed and each vertex can only use its own information to forward messages. One approach to achieving this are *interval routing schemes*, where each vertex is equipped with a mapping from its outgoing edges to a partition of the vertices in the graph and the message is forwarded along the edge whose assigned vertex set contains the target [26, 55, 61].

Another popular approach is *geographic* or *greedy* routing. There, each vertex is assigned a coordinate in a metric space and a message is routed to a neighbor that is closer to the

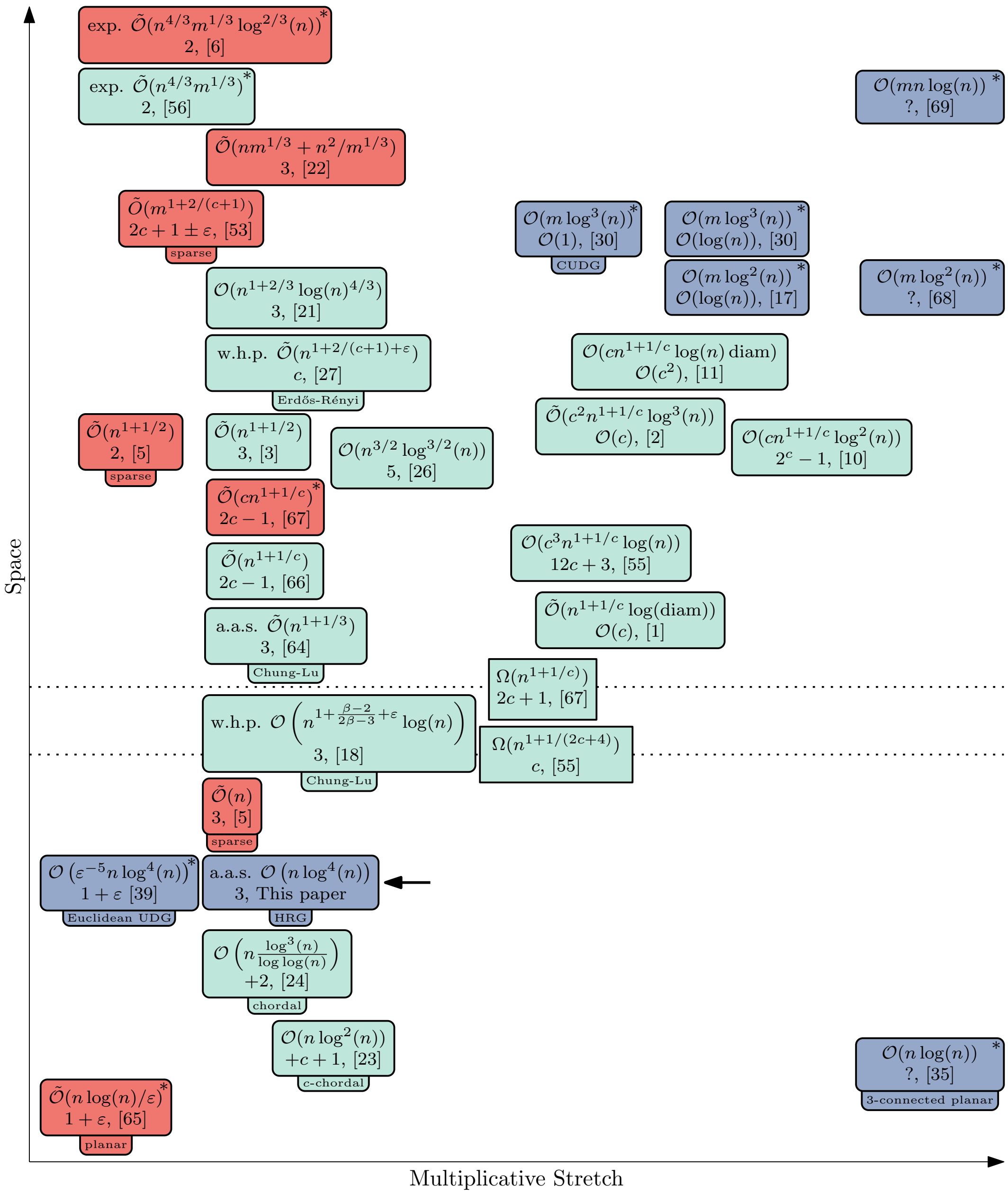

Figure 4 Distance oracles (red), routing schemes (green), and local routing schemes (blue) arranged by space requirements (first line) and multiplicative stretch (second line). Additive stretch is denoted with a preceding +. An asterisk denotes that the bound on the space requirement is adjusted to account for the total storage. Lower bounds are shown with rectangular corners.

target with respect to the metric. While initially being motivated by real-world networks with actual geographic locations [40, 62], later adaptations assigned *virtual coordinates* [58].

In addition to the previously mentioned criteria, greedy routing is also evaluated regarding the success rate, since the virtual coordinates may be assigned such that forwarding messages greedily leads to a dead end. Even simple graphs like a star with six leaves cannot be embedded in the Euclidean plane such that greedy routing always succeeds [51]. Worse yet, even if a graph admits a greedy embedding into the Euclidean plane, there are graphs that require $\Omega(n)$ bits per coordinate [8]. However, it was shown that delivery can be guaranteed on every graph when embedding it in the hyperbolic plane [44]. Unfortunately, due to the properties of hyperbolic space, this requires high precision coordinates, which leads to an increased space requirement [12]. While attempts have been made to reduce the coordinate size [29], it has been shown that this remains an open problem [12, 41]. However, in many greedy routing schemes the space per coordinate is at most poly-logarithmic [14, 17, 28, 35, 49, 68, 69].

Unfortunately, not much is known about stretch in local routing schemes. On Euclidean unit disk graphs there is a greedy approach that obtains constant stretch [30] and there exists an improvement to a stretch of $1 + \varepsilon$ while storing $\mathcal{O}(\varepsilon^{-5} \log^4(n))$ bits at each vertex, which relies on message headers that can be adapated during the routing process [39]. On graphs of bounded hyperbolicity, one can obtain an additive stretch of $\mathcal{O}(\log n)$ [32] and on general graphs a logarithmic bound on the multiplicative stretch is known [17, 30]. However, it has been observed that greedy routing schemes can achieve much better stretch in practice, which we discuss in the following.

Routing in Practice. Real-world networks rarely resemble the worst cases considered in the above mentioned results. More realistic insights can be obtained by analyzing networks whose properties resemble those of real-world graphs, like the small-world phenomenon [43]. In particular, better trade-offs between stretch and space have been obtained on sparse graphs [5, 53], and Chung-Lu random graphs [7]. There, the best known space bound of $\tilde{\mathcal{O}}(n^{1+1/2})$ for a stretch of 3 on general graphs [66], was improved to $\mathcal{O}(n^{1+(\beta-2)/(2\beta-3)+\varepsilon})$, with high probability, for power-law exponent $\beta \in (2, 3)$ and $\varepsilon > 0$ [18]. Experiments on internet-like networks further indicate that the landmark-based routing schemes due to Thorup and Zwick [66] yield a rather low stretch of about 1.1 while the information stored at the vertices is small as well [45, 46]. Similar results have been obtained in experiments on internet topologies and random graphs with power-law degree distributions [5, 18, 64].

Additionally, it was observed that greedy routing works remarkably well on internet graphs, when assuming an underlying hyperbolic geometry. There, a network is embedded into the hyperbolic plane and a message is always forwarded to the neighbor with the smallest hyperbolic distance to the target. Then, delivery is not guaranteed but experiments show that this achieves success rates of at least 97% and a stretch of about 1.1 on internet topologies [13, 52]. Partly motivated by this Krioukov et al. introduced the hyperbolic random graph (HRG) model to represent real-world networks like the internet [47]. For a generalized version it was shown that a greedy routing can succeed with constant probability, while achieving an average stretch of $1 + o(1)$, almost surely [15]. Nevertheless it remains an open question whether, in a addition to the small stretch, greedy routing on realistic representations of internet-like graphs can be implemented, such that delivery is always guaranteed, while keeping the space requirement low. In this paper, we answer this question by developing a greedy routing scheme that always succeeds with small stretch. Additionally, the space requirement is small on networks with an underlying hyperbolic geometry.

In the following, we combine standard techniques in routing to obtain a simple routing scheme and analyze its performance on strongly hyperbolic unit disk graphs. For the special case of hyperbolic random graphs, a graph model that is used to represent complex networks like the internet [47], it improves below the above mentioned performance lower bounds.

3.2 A Greedy Routing Scheme

A standard approach to routing in a decentralized setting is *greedy routing*, where the idea is to always forward a message to a neighbor of a vertex that is closer to the target.[3] When designing a greedy routing scheme, we therefore need to compute distances between vertices and select a suitable neighbor with respect to these distances. For a graph $G = (V, E)$ let $d\colon V \times V \to \mathbb{R}_{\geq 0}$ be a semi-metric on G. That is, for all $s, t \in V$ we have $d(s, t) \geq 0$, $d(s, t) = 0$ if and only if $s = t$, and $d(s, t) = d(t, s)$. We say that a greedy routing scheme routes *with respect to* d, if at s a message to t is forwarded to a neighbor v of s where $d(v, t) < d(s, t)$. Depending on d such a neighbor may not exist and the message cannot be forwarded, which is called *starvation*. In contrast, a routing scheme with guaranteed delivery is called *starvation-free* and it is known that greedy routing is starvation-free, if at every vertex $s \neq t$ there is a neighbor v with $d(v, t) < d(s, t)$ (see e.g. [70]). Moreover, we say that d is *integral* if it maps to the natural numbers, i.e., $d\colon V \times V \to \mathbb{N}$. Note that, if d is integral and routing with respect to d is starvation-free, the distance to the target decreases by at least one in each step. Thus, the length of the routed path between s and t is bounded by $d(s, t)$. When this is the case, we say that routing with respect to d is *d-bounded*.

Given a connected graph G, a natural choice for determining a distance between s and t is to use the length of a shortest path between them, which we denote with $d_G(s, t)$. Routing with respect to d_G is starvation-free and yields perfect stretch. However, d_G cannot be computed while simultaneously keeping the required space and query time low [34]. Therefore, we relax the constraint on routing with respect to exact graph distances and use upper bounds instead. This can be achieved by taking a subgraph G' of G and routing on G with respect to $d_{G'}$. The stretch of the resulting routing scheme depends on how well the distances in G' approximate the distances in G. Unfortunately, finding a subgraph with good stretch is hard in general [16, 54]. However, instead of routing with respect to the distances in a single subgraph, we can combine the distances in multiple subgraphs. To obtain a good stretch, it then suffices to find low-stretch subgraphs for small parts of the graph. To formalize this, we use a (c, ℓ, k)-*graph-cover* $\mathcal{C}$ of G, which is a collection of subgraphs of G, such that for all $s, t \in V$ there exists a connected subgraph G' in $\mathcal{C}$ with $d_{G'}(s, t) \leq c \cdot d_G(s, t)$ or $d_{G'}(s, t) \leq d_G(s, t) + \ell$, and every vertex $v \in V$ is contained in at most k graphs in $\mathcal{C}$. We say that $\mathcal{C}$ has multiplicative stretch c with additive bound ℓ. For two vertices s and t we define $d_{\mathcal{C}}(s, t) = \min_{G' \in \mathcal{C}} d_{G'}(s, t)$.

▶ **Lemma 9.** *Let G be a graph and let $\mathcal{C}$ be a (c, ℓ, k)-graph-cover of G. Then, greedy routing on G with respect to $d_{\mathcal{C}}$ has multiplicative stretch c with additive bound ℓ.*

Proof. Let $s \neq t$ be two vertices. To prove the claim, we need to show that an s-t-path obtained by greedily routing with respect to $d_{\mathcal{C}}$ has length at most $c \cdot d_G(s, t)$ or $d_G(s, t) + \ell$. To this end, we prove that the resulting routing scheme is $d_{\mathcal{C}}$-bounded. The claim then follows, due to the fact that consequently the routed s-t-path has length at most $d_{\mathcal{C}}(s, t) = \min_{G' \in G} d_{G'}(s, t)$ and the fact that there exists a $G' \in G$ with $d_{G'}(s, t) \leq c \cdot d_G(s, t)$ or $d_{G'}(s, t) \leq d_G(s, t) + \ell$ by assumption.

Since $d_{\mathcal{C}}$ is the minimum of integral semi-metrics, it is itself an integral semi-metric. Therefore, it suffices to show that routing with respect to $d_{\mathcal{C}}$ is starvation-free, which is the case, if for every two vertices $s \neq t$ there exists a neighbor v of s in G with $d_{\mathcal{C}}(v, t) < d_{\mathcal{C}}(s, t)$. Consider the connected subgraph $G' \in \mathcal{C}$ for which $d_{G'}(s, t) = d_{\mathcal{C}}(s, t)$. Then, there exists

[3] The term *greedy* often refers to routing to a neighbor *closest* to the target. For us *closer* is sufficient.

a shortest path from s to t in G'. For the successor v of s on this path, it holds that $d_{G'}(v,t) = d_{G'}(s,t) - 1$ and thus $d_{\mathcal{C}}(v,t) \leq d_{G'}(s,t) - 1 = d_{\mathcal{C}}(s,t) - 1 < d_{\mathcal{C}}(s,t)$. Finally, since G' is a subgraph of G, it follows that v is also a neighbor of s in G. ◀

To show that $d_{\mathcal{C}}$ can be computed efficiently, we use *distance labeling schemes* [33]. Such a scheme implements a semi-metric d by assigning each vertex a *distance label*, such that for two vertices s, t we can compute $d(s,t)$ by looking at their distance labels only. The *label size* of a distance labeling scheme denotes the maximum number of bits required to represent the label of a vertex. The *query time* denotes the time it takes to compute d using the labels. Given a graph-cover $\mathcal{C}$, a distance labeling scheme that implements $d_{\mathcal{C}}$ can be obtained by combining distance labeling schemes for the contained subgraphs.

▶ **Lemma 10.** *Let G be a graph and let $\mathcal{C}$ be a (c, ℓ, k)-graph-cover of G such that for every $G' \in \mathcal{C}$ there exists a distance labeling scheme that implements $d_{G'}$ with label size λ and query time q. Then, there exists a distance labeling scheme for G that implements $d_{\mathcal{C}}$ with label size $\mathcal{O}(k(\lambda + \log k + \log n))$ and query time $\mathcal{O}(kq)$.*

Proof. We assign each subgraph $G' \in \mathcal{C}$ a unique *graph-ID* in $[|\mathcal{C}|]$ and compute the distance labels for all vertices in G'. By combining the distance labels with the corresponding graph-ID, we obtain an *identifiable distance label* that can be used to uniquely identify to which graph a distance label belongs. The label of a vertex v is then obtained by collecting the identifiable distance labels of v for all subgraphs that v is contained in and sorting them by graph-ID.

The label size can now be bounded as follows. Since each vertex v is contained in at most k subgraphs, we can conclude that $|\mathcal{C}| \leq kn$. Therefore, the graph-IDs can be encoded using $\mathcal{O}(\log k + \log n)$ bits. Moreover, by assumption the distance labels in the subgraphs can be represented using λ bits. It follows that a single identifiable distance label takes $\mathcal{O}(\lambda + \log k + \log n)$ bits. Again, since every vertex is contained in at most k subgraphs, v's label consists of at most k identifiable distance labels. Consequently, the label size is bounded by $\mathcal{O}(k(\lambda + \log k + \log n))$.

It remains to bound the query time. Given the collection of identifiable distance labels of two vertices, we can identify the ones with matching graph-IDs in time $\mathcal{O}(k)$, since they are sorted by graph-ID. For each match we compute the distance in the corresponding subgraph in time q. Afterwards the minimum distance can be found in $\mathcal{O}(k)$ time. It follows that $d_{\mathcal{C}}$ can be computed in time $\mathcal{O}(kq)$. ◀

In order to perform a routing decision efficiently, we want to avoid performing a linear search over all neighbors. To this end, we need to be able to identify a neighbor directly, which can be done by assigning each neighbor v of s a unique *port* $p_s(v) \colon N(s) \to \{1, \dots, n\}$. Finding a neighbor of s that is closer to a target t with respect to a semi-metric d then boils down to determining the corresponding port. To this end, we can use a *port labeling scheme* that implements d. Such a scheme assigns each vertex in a graph a *port label* such that we can determine the port of a neighbor of s that is closer to t with respect to d, by only looking at the port labels of s and t. The corresponding label sizes and query times are defined analogous to how they are defined for distance labels.

Given a graph-cover $\mathcal{C}$, we can combine distance and port labels of the subgraphs in the cover, to obtain a port labeling scheme that implements $d_{\mathcal{C}}$.

▶ **Lemma 11.** *Let G be a graph and let $\mathcal{C}$ be a (c, ℓ, k)-graph-cover of G such that for every $G' \in \mathcal{C}$ there exist distance and port labeling schemes that implement $d_{G'}$ with label size λ and query time q. Then, there exists a port labeling scheme for G that implements $d_{\mathcal{C}}$ with label size $\mathcal{O}(k(\lambda + \log k + \log n))$ and query time $\mathcal{O}(kq)$.*

Proof. For every vertex s in G we fix a port assignment for the neighbors of s. Afterwards, we assign the same ports in the subgraphs G' of G in $\mathcal{C}$ that s is contained in. More precisely, if v is a neighbor of s in G', then the port $p_s(v)$ is identical in G and G'. As a consequence, we can use a port labeling scheme in G' to determine the port of a neighbor of s in G.

Now consider the distance labeling scheme described in Lemma 10, where we assigned each subgraph $G' \in \mathcal{C}$ a unique graph-ID and computed distance labels for all vertices in all subgraphs to obtain identifiable distance labels. In addition, we now compute port labels for all vertices in all subgraphs. By combining them with the previously obtained identifiable distance labels, we obtain *identifiable distance port labels*. As before, the label of a vertex v then consists of the collection of identifiable distance port labels of v in all subgraphs that v is contained in, sorted by graph-ID.

We continue the proof by showing that, given the labels of two vertices $s \neq t$, we can compute the port of a neighbor of s in G that is closer to t with respect to $d_{\mathcal{C}}$. As described in the proof of Lemma 10, we can use the labels to find the graph-ID of a subgraph G' of G for which $d_{G'}(s,t) = d_{\mathcal{C}}(s,t)$. We then use the corresponding port labels of s and t to determine the port $p_s(v)$ of a neighbor v of s that is closer to t with respect to $d_{G'}$. Clearly, we have $d_{\mathcal{C}}(v,t) \leq d_{G'}(v,t) < d_{G'}(s,t) = d_{\mathcal{C}}(s,t)$. Moreover, since $p_s(v)$ is identical in G' and G, it follows that $p_s(v)$ is a suitable port in G.

The label size can be bounded as follows. By Lemma 10 we can encode all k identifiable distance labels stored at a vertex using $\mathcal{O}(k(\lambda + \log k + \log n))$ bits. Since a single identifiable distance label is extended with a port label that takes at most λ bits, it follows that the label size increases by an additive $\mathcal{O}(k\lambda)$, yielding a size of $\mathcal{O}(k(\lambda + \log k + \log n))$ bits.

It remains to bound the query time. Again, as described in the proof of Lemma 10, determining the graph-ID of the subgraph G' for which $d_{G'}(s,t) = d_{\mathcal{C}}(s,t)$ takes $\mathcal{O}(kq)$ time. Computing the port $p_s(v)$ of a suitable neighbor v of s then takes an additional time q. Consequently, the query time is $\mathcal{O}(kq)$. ◀

We are now ready to combine the above results to obtain our greedy routing scheme. To this end, we need to find (c,ℓ,k)-graph-covers with small values for c, ℓ, and k, as well as distance and port labeling schemes with small label sizes and query times, as all of these properties affect the performance of the routing scheme. While distance labeling schemes require large labels in general graphs [33], better results can be obtained by restricting the graph-cover to only contain trees as subgraphs. Such a cover is then called *tree-cover*. Tree-covers are standard in routing [9, 11, 26, 30, 37, 63, 66], and while it is known that greedy routing with respect to $d_{\mathcal{C}}$ for a (c,ℓ,k)-tree-cover $\mathcal{C}$ is starvation-free (see e.g., [37]), we also know that the resulting routing scheme has stretch c with additive bound ℓ due to Lemma 9. Moreover, for trees there are distance and port labeling schemes with $\mathcal{O}(\log^2 n)$ bit labels and constant query time [31, 66]. Together with Lemma 11 we obtain the following theorem.

▶ **Theorem 12.** *Given a (c,ℓ,k)-tree-cover of a graph G, greedy routing on G can be implemented such that the routing scheme is starvation-free, has stretch c with additive bound ℓ, stores $\mathcal{O}(k(\log^2 n + \log k))$ bits at each vertex, and takes $\mathcal{O}(k)$ time for a routing decision.*

To complete our scheme, we propose an algorithm that computes a tree-cover with bounded stretch. It is an adaptation of an algorithm previously used to compute graph spanners [20]. We start with the following lemma, describing a situation that is exploited by our algorithm.

▶ **Lemma 13.** *Let $G = (V,E)$ be a graph, $u, v \in V$, and let H be an induced subgraph that contains all vertices on a shortest uv-path P in G. Let T be a partial shortest-path* *tree in H rooted at t that contains u and v. Then, for every vertex w in T that lies on P, $d_T(u,v) \le d_G(u,v) + 2d_H(t,w)$.*

Proof. Let t' be the lowest common ancestor of u and v and consider the paths P_u and P_v from t' to u and v, respectively. Note that P_u and P_v are shortest paths in H as they are descending paths in a shortest-path tree. Thus, $d_T(u,v) = |P_u| + |P_v| = d_H(t',u) + d_H(t',v)$.

Observe that clearly $d_H(t',u) \le d_H(t,u)$. Moreover, by the triangle inequality, we have $d_H(t,u) \le d_H(t,w) + d_H(w,u)$. Analogously for v, we obtain $d_H(t',v) \le d_H(t,w) + d_H(w,v)$. Thus, we get

$$\begin{aligned} d_T(u,v) &= d_H(t',u) + d_H(t',v) \le d_H(t,w) + d_H(w,u) + d_H(t,w) + d_H(w,v) \\ &= d_H(u,v) + 2d_H(t,w), \end{aligned}$$

where the last equality holds as w lies on a shortest uv-path P in G, which is also a shortest uv-path in H, since H is an induced subgraph of G that contains all vertices of P. For the same reason, we get $d_H(u,v) = d_G(u,v)$, which proves the claim. ◀

Consider the setting as in the above lemma, let w be chosen such that $d_H(t,w)$ is minimal and let $\xi = 2d_H(t,w)/d_G(u,v)$. Then, it holds that $d_T(u,v) \le (1+\xi)d_G(u,v)$. That is, T has stretch $(1+\xi)$. The following algorithm computes a tree-cover with the same stretch.

Let G be the input graph. The algorithm operates in phases, starting with phase 0. For each phase i, we define a radius $r_i = b^i$, for a base $b > 1$. Then, for $a > 0$, we choose a vertex t in the current graph and compute the partial shortest-path tree with root t containing all vertices with distance at most $(1+a)r_i$ from t. Afterwards, we delete all vertices with distance at most r_i to t from the current graph. This is iterated until all vertices are deleted. Afterwards, phase i is done and we restore the original input graph G before starting phase $i+1$. This process is stopped, once the whole graph is deleted after processing the first tree in a phase. The output of the algorithm is the set of all trees computed during execution. Since the algorithm **pro**duces **t**ree-covers **o**f **n**etworks, we call it PROTON.

Note that PROTON has several degrees of freedom. We can choose the parameters $a > 0$ and $b > 1$, as well as the order in which the roots of the partial shortest-path trees are selected. The following lemma holds independent of the root selection strategy.

▶ **Lemma 14.** *The tree-cover computed by PROTON has stretch $(1 + 2b/a)$ with additive bound 2.*

Proof. Let $\mathcal{C}$ be the tree-cover computed by PROTON, let $G = (V,E)$ be the input graph, and let $u \ne v \in V$ be two arbitrary vertices. We have to show that $\mathcal{C}$ contains a tree T that includes u and v such that $d_T(u,v) \le (1+2b/a)d_G(u,v)$ or $d_T(u,v) \le d_G(u,v) + 2$.

Let i be minimal such that $d_G(u,v) \le ar_i$. Assume for now that PROTON did not stop before phase i; we deal with the other case later. As phase i continues until all vertices are deleted, at one point a vertex w on a shortest uv-path in G is deleted. Let H be the current graph before that happens for the first time and let T be the partial shortest-path tree computed in H rooted at t. To show that T is the desired tree, we aim to apply Lemma 13.

First note that H is an induced subgraph of G that contains all vertices on a shortest uv-path of G. Moreover, T contains u and v for the following reason. As w is deleted, we have that $d_H(t,w) \le r_i$. Moreover, as w lies on a shortest uv-path, the distance from w to either u or v cannot exceed $d_H(u,v) = d_G(u,v)$. Thus, by the triangle inequality and the above choice of i, we have $d_H(t,u) \le d_H(t,w) + d_H(w,u) \le r_i + ar_i = (1+a)r_i$, which implies that u is a vertex of T. Analogously, v is also contained in T.

With this, we can apply Lemma 13, yielding stretch $(1+\xi)$ for $\xi = 2d_H(t,w)/d_G(u,v)$. To bound ξ, recall that we chose i minimal such that $d_G(u,v) \le ar_i$. Thus, if $i > 0$, then $d_G(u,v) > ar_{i-1} = \frac{a}{b}r_i$. Together with the fact that $d_H(t,w) \le r_i$, we obtain $\xi \le 2\frac{b}{a}$, as desired. In the special case that $i = 0$ we have $r_i = 1$ and therefore $d_H(t,w) \le 1$. Thus, Lemma 13 directly yields $d_T(u,v) \le d_G(u,v) + 2$, which is covered by the additive bound 2.

Finally, we assumed above that PROTON did not stop before phase i and it remains to consider the case where it stops in phase $j < i$. In this case, let T be the tree we get in phase j, which includes all vertices of G. Let t be the root of T. As all vertices have distance at most r_j from t, we get $d_T(u,v) \le 2r_j$. Moreover, as i was chosen minimal such that $d_G(u,v) \le ar_i$, we have $d_G(u,v) > ar_j$. Together with the previous inequality, this gives a stretch of $2/a$, which is smaller than the desired $(1+2b/a)$, as $b > 1$. ◀

3.3 Performance on Strongly Hyperbolic Unit Disk Graphs

Throughout the remainder of the paper, we utilize strongly hyperbolic unit disk graphs to investigate the performance of routing on networks with underlying hyperbolic geometry. This is not only interesting since routing is one of the most fundamental graph problems, but is also particularly relevant on complex networks like the internet, which has previously been observed to fit well into the hyperbolic plane [13].

While PROTON computes a (c,ℓ,k)-tree-cover with bounded stretch, the value k, i.e., the maximum number of trees that a vertex is contained in, depends on the structure of the considered graph. In the following, we show that k is small on networks with an underlying hyperbolic geometry. In our analysis, we consider the *radially increasing* root selection strategy that selects the vertices in order of increasing distance to the origin of the hyperbolic plane, and prove the following theorem. There, $\operatorname{diam}(G)$ denotes the *diameter* of G, i.e., the length of the longest shortest path in G.

▶ **Theorem 15.** *Let G be a strongly hyperbolic unit disk graph with radius $R > 0$. Given the disk representation of G, $a > 0$, and $b > 1$, the PROTON algorithm with the radially increasing root selection strategy computes a (c,ℓ,k)-tree-cover of G with*

$$c = 1 + 2b/a,\ \ \ell = 2\ ,\ \text{and}\ k = \pi e\left(\frac{1+a}{b-1}(b^2\operatorname{diam}(G) - 1)R + 2\left(\log_b(\operatorname{diam}(G)) + 2\right)\right).$$

First note that the correctness of the claimed stretch immediately follows from Lemma 14. However, bounding k is more involved. To that end, we first compute an upper bound that holds for a given phase and afterwards sum over all phases.

Consider the roots of the partial shortest-path trees that contain a vertex v in a given phase, which we refer to as the *roots of v*. We partition the hyperbolic disk into radial bands and compute an upper bound on the number of roots of v in each band, see Figure 5 for an illustration. We then utilize two key ingredients. First, since v is contained in the partial shortest-path trees of its roots, the length of the path between v and a root is bounded, and so is the angular distance between them. Consequently, all roots in a band lie in a bounded angular interval (blue areas in Figure 5). Secondly, roots cannot be adjacent as they would otherwise delete each other, which means that the hyperbolic distance between them has to be sufficiently large. For roots in the same band, this can only be achieved if their angular distance is large. Consequently, each root in a band reserves a portion of the angular interval (red areas in Figure 5) that no other root can lie in, from which we can derive an upper bound on the number of roots that lie in the band.

The following lemma bounds the angular distance between a vertex u and another vertex u_k, assuming that there exists a path of length k between them that consists only of

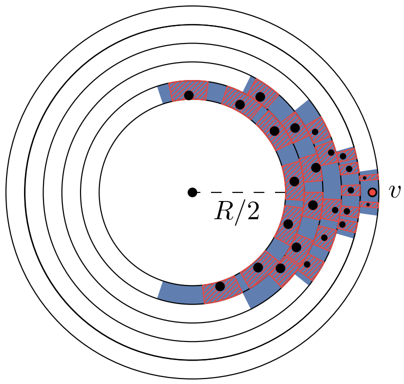

Figure 5 The hyperbolic disk is divided into radial bands. The roots (black vertices) of v (red dot) in a band lie in an angular interval Φ of bounded width (blue). Each root reserves a portion of that interval (red) that no other root can lie in. All vertices with radius at most $R/2$ are removed after processing the first root.

vertices whose radius is not smaller than the one of u. In particular, this applies to roots of v: In a given phase, the length of the paths considered in the partial shortest-path trees is bounded. Moreover, when the partial shortest-path tree of a root ρ of v is computed, all vertices of smaller radii than ρ have been deleted (since roots are considered in order of increasing radius), meaning the path from ρ to v cannot contain vertices of smaller radius.

▶ **Lemma 16.** *Let G be a strongly hyperbolic unit disk graph with radius R and let u be a vertex with $r(u) \geq R/2$. Further, let $P = (u, u_1, \dots, u_k)$ be a path with $r(u) \leq r(u_i)$ for all $i \in [k]$. Then, $\Delta_\varphi(u, u_k) \leq k \cdot \pi e^{R/2 - r(u)}$.*

Proof. For convenience, we define $u_0 = u$. Then, $\Delta_\varphi(u, u_k)$ can be bounded by

$$\Delta_\varphi(u, u_k) \leq \sum_{i=1}^{k} \Delta_\varphi(u_{i-1}, u_i).$$

Note that u_{i-1} and u_i are adjacent and recall that $\theta(r(u_{i-1}), r(u_i))$ denotes the maximum angular distance between them, such that this is the case. Thus,

$$\Delta_\varphi(u, u_k) \leq \sum_{i=1}^{k} \theta(r(u_{i-1}), r(u_i)).$$

Since $R/2 \leq r(u) \leq r(u_i)$ for all $i \in [k]$ is a precondition of this lemma, we have $r(u_{i-1}) + r(u_i) \geq R$ for all $i \in [k]$. Consequently, we can apply Lemma 4 to bound $\theta(r(u_{i-1}), r(u_i))$, which yields

$$\Delta_\varphi(u, u_k) \leq \sum_{i=1}^{k} \pi e^{(R - r(u_{i-1}) - r(u_i))/2} \leq \sum_{i=1}^{k} \pi e^{(R - r(u) - r(u))/2} = k \cdot \pi e^{R/2 - r(u)},$$

where the second inequality is valid since $r(u) \leq r(u_i)$ for all $i \in [k]$. ◀

The second key ingredient is a lower bound on the minimum angular distance between two non-adjacent vertices in a radial band of fixed width in the hyperbolic disk. We note that in order to obtain a bound that is easy to work with, we aim to utilize Corollary 5. However, this requires that R is not too small. For now, we assume that this requirement is met and afterwards resolve the constraint in the analysis of the algorithm.

▶ **Lemma 17.** *Let G be a strongly hyperbolic unit disk graph with radius $R \geq 1$ and let $r \geq R/2$ be a radius. Further, let u, v be non-adjacent vertices with $r(u), r(v) \in [r, r+\tau]$ for $\tau \in [0, R-1]$. Then, $\Delta_\varphi(u, v) \geq e^{R/2-(r+\tau)}$.*

Proof. Recall that $\theta(r(u), r(v))$ denotes the maximum angular distance such that u and v are adjacent. Since the two vertices are not adjacent in our case, we can derive that $\Delta_\varphi(u, v) > \theta(r(u), r(v))$. We now aim to apply Corollary 5 in order to obtain a lower bound on $\theta(r(u), r(v))$. To this end, we first validate that its preconditions are met. Since $r(u), r(v) \geq r \geq R/2$, we have $r(u) + r(v) \geq R$. Moreover, by assumption we know that $r(u), r(v) \in [r, r+\tau]$ for $\tau \in [0, R-1]$, which implies that $|r(u) - r(v)| \leq \tau \leq R-1$. Consequently, we can apply Corollary 5 to conclude that

$$\begin{aligned} \theta(r(u), r(v)) &\geq e^{(R-r(u)-r(v))/2} \\ &\geq e^{(R-2r-2\tau)/2} \\ &= e^{R/2-(r+\tau)}, \end{aligned}$$

where the second inequality is valid, since by assumption $r(u), r(v) \leq r + \tau$. ◀

We can now combine the two key ingredients to compute an upper bound on the number of the roots of v in a given phase i, which we denote by $\boldsymbol{\rho}_i(v)$.

▶ **Lemma 18.** *Let G be a strongly hyperbolic unit disk graph with radius $R > 0$. Let the disk representation of G, $a > 0$, and $b > 1$ be given and consider phase i of the* Proton *algorithm. Then, for every vertex v it holds that $|\boldsymbol{\rho}_i(v)| \leq \pi e(R(1+a)b^i + 2)$.*

Proof. In the following, we aim to utilize Lemmas 16 and 17, both of which require that the considered vertices have a radius of at least $R/2$ and one additionally assumes that $R \geq 1$ Therefore, we first argue about the case where these conditions are not met. First note that after the first root in a phase is processed, all vertices with radius at most $R/2$ are removed since (if they exist in the first place) they form a clique. Additionally, when $R < 1$, the whole graph can be covered by few cliques. More precisely, by Lemma 8 a strongly hyperbolic unit disk graph with radius R can be covered by $\max\{2\pi\sqrt{2}, 2\pi e^{R/2}\}$ cliques. In particular, for $R < 1$, this yields a bound of $2\pi\sqrt{e}$. Since processing each root removes at least one such clique from the graph, the number of roots in the phase is bounded by the number of cliques. It follows, considering the first clique in $D_{R/2}$ and the remaining cliques when $R < 1$, that we can bound the roots of v in phase i as $|\boldsymbol{\rho}_i(v)| \leq 1 + 2\pi\sqrt{e} \leq 2\pi e$, which we account for with the $+2$ in the lemma statement.

For the remaining roots of v we can now assume that $R \geq 1$ and that all vertices have radius at least $R/2$. Furthermore, it suffices to show that there are at most $\pi e R(1+a)b^i$ such roots. We cover the remainder of the disk with $R/2$ bands of radial width 1, where the jth band (for $j \in \{0, \dots, R/2-1\}$) contains all points with radius in $[R/2+j, R/2+j+1]$, see Figure 5. The claim then follows if we can bound the number of roots of v in a single band by $2\pi e(1+a)b^i$.

Let $\boldsymbol{\rho}_{i,j}(v)$ denote the roots of v that lie in the jth band (see Figure 6). We first bound the angular distance between v and a root in $\boldsymbol{\rho}_{i,j}(v)$, and with that the width of the angular interval Φ (blue region in Figure 6) that contains all of them. Afterwards, we show that each root reserves a portion of Φ that no other root can be in. An upper bound on $|\boldsymbol{\rho}_{i,j}(v)|$ is then obtained by the quotient of the widths of the two intervals.

Consider a root $\rho \in \boldsymbol{\rho}_{i,j}(v)$. Since the roots are processed in order of increasing radius, all vertices of radius at most $r(\rho)$ have been removed before. Consequently, the path from ρ

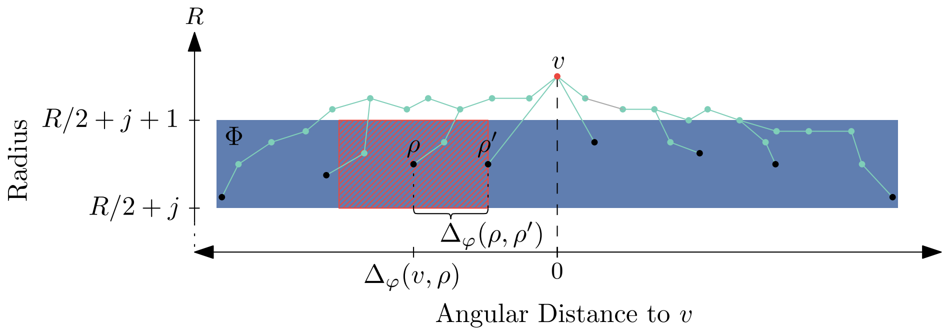

Figure 6 A vertex v (red dot) and the roots (black vertices) that are contained in the jth band and are connected to v (via the green paths). All roots lie in the angular interval Φ (blue region). Other than ρ, no root can lie in the red region.

to v in the partial shortest-path tree rooted at ρ consists only of vertices with radius at least $r(\rho)$. Moreover, in phase i the depth of this tree is $(1+a)b^i$, which means that the path between ρ and v is at most this long. Therefore, we can apply Lemma 16 to conclude that the maximum angular distance between v and a root ρ is at most

$$\max_{\rho \in \boldsymbol{\rho}_{i,j}(v)} \Delta_\varphi(v, \rho) \le \max_{\rho \in \boldsymbol{\rho}_{i,j}(v)} (1+a)b^i \cdot \pi e^{R/2 - r(\rho)} \le (1+a)b^i \cdot \pi e^{-j},$$

where the last inequality stems from the fact that $r(\rho) \ge R/2 + j$ holds for all $\rho \in \boldsymbol{\rho}_{i,j}(v)$. Moreover, since roots cannot be adjacent (as they would otherwise delete each other) and all roots in $\boldsymbol{\rho}_{i,j}(v)$ have their radii in $[R/2+j, R/2+j+1]$, we can apply Lemma 17 to conclude that the minimum angular distance between two roots $\rho, \rho' \in \boldsymbol{\rho}_{i,j}(v)$ is at least

$$\min_{\rho \ne \rho' \in \boldsymbol{\rho}_{i,j}(v)} \Delta_\varphi(\rho, \rho') \ge e^{R/2 - (R/2+j+1)} = e^{-(j+1)}.$$

Note that the angular interval Φ extends to both angular directions from v. Therefore, the number of roots in $\boldsymbol{\rho}_{i,j}(v)$ can be bounded by

$$\begin{aligned} |\boldsymbol{\rho}_{i,j}(v)| &\le 2 \cdot \frac{\max_{\rho \in \boldsymbol{\rho}_{i,j}(v)} \Delta_\varphi(v, \rho)}{\min_{\rho \ne \rho' \in \boldsymbol{\rho}_{i,j}(v)} \Delta_\varphi(\rho, \rho')} \\ &\le \frac{2(1+a)b^i \cdot \pi e^{-j}}{e^{-(j+1)}} \\ &= 2\pi e(1+a)b^i. \end{aligned}$$

◀

With this we are now ready to bound the number k of trees that a vertex is contained in, for tree-covers produced by the Proton algorithm.

Proof of Theorem 15. First note that the values for c and ℓ hold for any graph due to Lemma 14. It remains to show that the stated bound on k is valid. To that end, we make use of Lemma 18, which states that v is contained in at most $\pi e(R(1+a)b^i + 2)$ trees in phase i, and sum over all phases. Since the radius of the shortest-path trees that are removed from the graph in two consecutive phases increases by a factor of b and the algorithm terminates when the first tree in a phase deletes the whole graph, there are at most $\lceil \log_b(\mathrm{diam}(G)) \rceil$ phases. Thus,

$$
\begin{aligned}
k &= \sum_{i=0}^{\log_b(\mathrm{diam}(G))+1} \pi e(R(1+a)b^i + 2) \\
&= \pi e \left(R(1+a) \left(\sum_{i=0}^{\log_b(\mathrm{diam}(G))+1} b^i \right) + 2\left(\log_b(\mathrm{diam}(G)) + 2\right) \right).
\end{aligned}
$$

Note that the remaining sum is a partial sum of a geometric series with $b > 1$, which can be computed as $\sum_{i=0}^{x} b^i = (b^{x+1} - 1)/(b-1)$. We obtain

$$
\begin{aligned}
k &= \pi e \left(R(1+a) \frac{b^{\log_b(\mathrm{diam}(G))+2} - 1}{b-1} + 2\left(\log_b(\mathrm{diam}(G)) + 2\right) \right) \\
&= \pi e \left(\frac{1+a}{b-1} (b^2 \,\mathrm{diam}(G) - 1) R + 2\left(\log_b(\mathrm{diam}(G)) + 2\right) \right).
\end{aligned}
$$ ◀

Since hyperbolic random graphs are a special case of strongly hyperbolic unit disk graphs, where vertices are distributed in a disk of radius $R = \mathcal{O}(\log n)$ and since these graphs have a diameter of $\mathcal{O}(\log n)$ asymptotically almost surely [50], we obtain the following corollary.

▶ **Corollary 19.** *Let G be a hyperbolic random graph. Given the disk representation of G, $a > 0$, and $b > 1$, the* Proton *algorithm with the radially increasing root selection strategy computes a (c, ℓ, k)-tree-cover of G with $c = 1 + 2b/a$, $\ell = 2$, and, asymptotically almost surely*

$$
k = \mathcal{O}\left(\frac{(1+a)b^2}{b-1} \cdot \log^2 n \right).
$$

Together with Theorem 12, it follows that greedy routing on hyperbolic random graphs can be implemented such that the resulting scheme is starvation-free and has stretch $1 + 2b/a$ with additive bound 2. Moreover, by setting $a = b = 2$ we obtain a multiplicative stretch of 3, and can derive that the scheme, asymptotically almost surely, stores $\mathcal{O}(\log^4 n)$ bits at each vertex and takes $\mathcal{O}(\log^2 n)$ time per query, which improves upon the performance lower bound for general graphs.

4 Experiments

We designed a routing scheme that utilizes hierarchical structures and showed that it has small stretch, space requirements, and query times on strongly hyperbolic unit disk graphs. To evaluate how well our results translate to real-world networks, we performed experiments on 50 graphs from the Network Data Repository [60], with sizes ranging from 14 k to over 2.3 M vertices. Since we do not have unit disk representations for these, we used the degrees of the vertices as a proxy for their place in the hierarchical structure. That is, the root selection strategy processed the vertices by decreasing degree. For each graph, we computed a tree-cover using the Proton algorithm with parameters $a = b = 2$, and sampled 10 k vertex pairs for which the path obtained by our routing scheme was compared to a shortest path between them. Figure 7 shows boxplots aggregating our observations.

As expected, the maximum observed stretch is 3. However, this stretch occurred only rarely. In all networks most of the sampled routes had a stretch of at most 1.5 and in 16 of the 50 graphs the median stretch was 1. At the same time, the number of trees that a vertex was contained in on average remained small. In 42 of the 50 networks this number was less than 50, even in networks with over 2.3 M vertices.

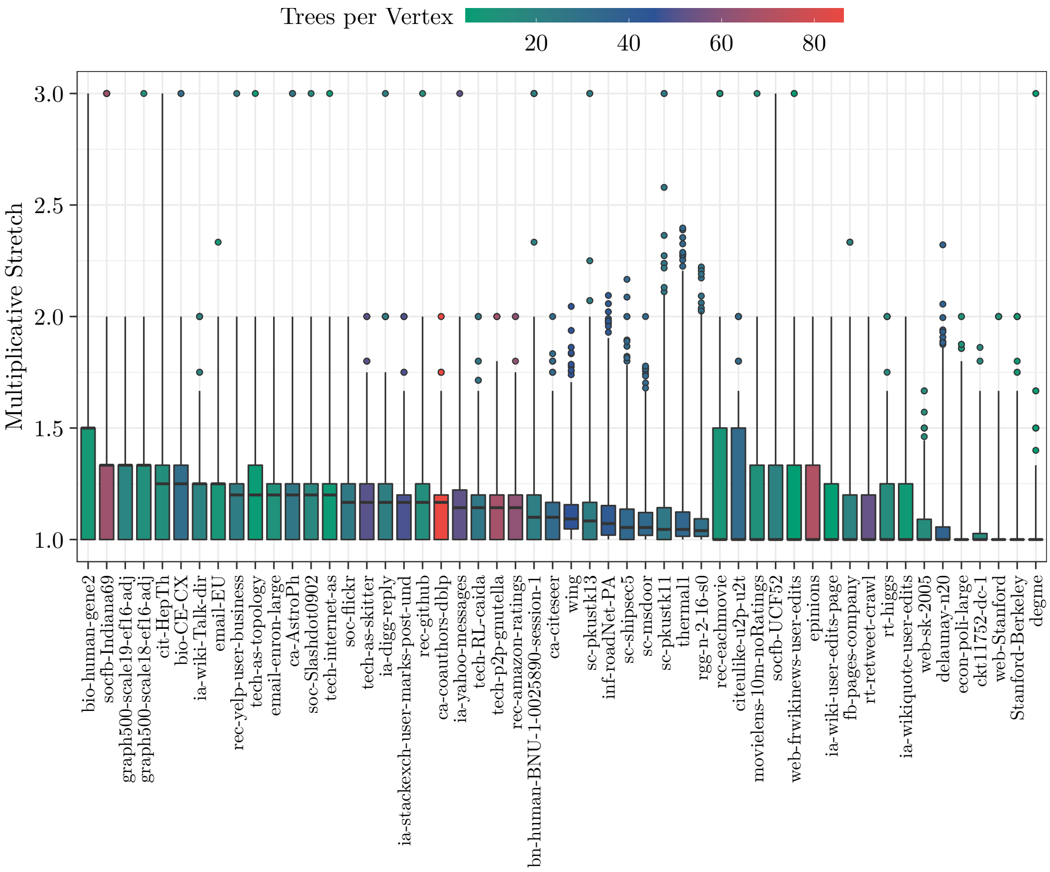

Figure 7 Multiplicative stretch when routing with a tree-cover obtained using PROTON with $a = b = 2$. Colors show the number of trees that an average vertex is contained in. Boxes denote the interquartile range extending to the 25th and 75th percentile with horizontal bars showing the median. Whiskers extend to 0.1% and 99.9%, while dots show values beyond that.

References

1 Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. Routing with improved communication-space trade-off. In *Distributed Computing, 18th International Conference, DISC 2004, Amsterdam, The Netherlands, October 4-7, 2004, Proceedings*, volume 3274, pages 305–319, 2004. doi:10.1007/978-3-540-30186-8_22.

2 Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. On space-stretch trade-offs: Upper bounds. In *Annual ACM Symposium on Parallelism in Algorithms and Architectures*, volume 2006, pages 217–224, 2006. doi:10.1145/1148109.1148144.

3 Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, Noam Nisan, and Mikkel Thorup. Compact name-independent routing with minimum stretch. *ACM Trans. Algorithms*, 4(3), 2008. doi:10.1145/1367064.1367077.

4 M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables*. Applied mathematics series. Dover Publications, 1965.

5 R. Agarwal, P. B. Godfrey, and S. Har-Peled. Approximate distance queries and compact routing in sparse graphs. In *2011 Proceedings IEEE INFOCOM*, pages 1754–1762, 2011. doi:10.1109/INFCOM.2011.5934973.

6 Rachit Agarwal and Philip Brighten Godfrey. Brief announcement: a simple stretch 2 distance oracle. In Panagiota Fatourou and Gadi Taubenfeld, editors, *ACM Symposium on Principles of Distributed Computing, PODC '13*, pages 110–112, 2013. doi:10.1145/2484239.2484277.

7 William Aiello, Fan Chung, and Linyuan Lu. A random graph model for power law graphs. *Experimental Mathematics*, 10(1):53–66, 2001. doi:10.1080/10586458.2001.10504428.

8 Patrizio Angelini, Giuseppe Di Battista, and Fabrizio Frati. Succinct greedy drawings do not always exist. *Networks*, 59(3):267–274, 2012. doi:10.1002/net.21449.

9 B. Awerbuch, S. Kutten, and D. Peleg. On buffer-economical store-and-forward deadlock prevention. *IEEE Transactions on Communications*, 42(11):2934–2937, 1994. doi:10.1109/26.328973.

10 Baruch Awerbuch, Amotz Bar-Noy, Nathan Linial, and David Peleg. Improved routing strategies with succinct tables. *Journal of Algorithms*, 11(3):307 – 341, 1990. doi:10.1016/0196-6774(90)90017-9.

11 Baruch Awerbuch and David Peleg. Routing with polynomial communication-space trade-off. *SIAM J. Discrete Math.*, 5:151–162, 05 1992. doi:10.1137/0405013.

12 Thomas Bläsius, Tobias Friedrich, Maximilian Katzmann, and Anton Krohmer. Hyperbolic embeddings for near-optimal greedy routing. *ACM J. Exp. Algorithmics*, 25, 2020. doi:10.1145/3381751.

13 Marián Boguñá, Fragkiskos Papadopoulos, and Dmitri Krioukov. Sustaining the internet with hyperbolic mapping. *Nature Communications*, 1(1):62, 2010. doi:10.1038/ncomms1063.

14 Arthur Brady and Lenore Cowen. *Compact Routing on Power Law Graphs with Additive Stretch*, pages 119–128. 2006. doi:10.1137/1.9781611972863.12.

15 Karl Bringmann, Ralph Keusch, Johannes Lengler, Yannic Maus, and Anisur Rahaman Molla. Greedy routing and the algorithmic small-world phenomenon. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, page 371–380, 2017. doi:10.1145/3087801.3087829.

16 Leizhen Cai. Np-completeness of minimum spanner problems. *Discrete Applied Mathematics*, 48(2):187–194, 1994. doi:10.1016/0166-218X(94)90073-6.

17 M. Camelo, D. Papadimitriou, L. Fàbrega, and P. Vilà. Geometric routing with word-metric spaces. *IEEE Communications Letters*, 18(12):2125–2128, 2014. doi:10.1109/LCOMM.2014.2364213.

18 Wei Chen, Christian Sommer, Shang-Hua Teng, and Yajun Wang. A compact routing scheme and approximate distance oracle for power-law graphs. *ACM Trans. Algorithms*, 9(1), 2012. doi:10.1145/2390176.2390180.

19 Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990. doi:10.1016/0012-365X(90)90358-O.

20 Edith Cohen. Fast algorithms for constructing t-spanners and paths with stretch t. *SIAM Journal on Computing*, 28(1):210–236, 1998. doi:10.1137/S0097539794261295.

21 Lenore J Cowen. Compact routing with minimum stretch. *Journal of Algorithms*, 38(1):170 – 183, 2001. doi:10.1006/jagm.2000.1134.

22 D. Dor, S. Halperin, and U. Zwick. All pairs almost shortest paths. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 452–461, 1996. doi:10.1109/SFCS.1996.548504.

23 Yon Dourisboure. Compact routing schemes for generalised chordal graphs. *J. Graph Algorithms Appl.*, 9(2):277–297, 2005. URL: https://www.emis.de/journals/JGAA/accepted/2005/Dourisboure2005.9.2.pdf.

24 Yon Dourisboure and Cyril Gavoille. Improved compact routing scheme for chordal graphs. In *Distributed Computing*, pages 252–264, 2002.

25 Feodor Dragan, Chenyu Yan, and Irina Lomonosov. Collective tree spanners of graphs. In *SIAM Journal on Discrete Mathematics*, volume 20, 10 2004. doi:10.1007/978-3-540-27810-8_7.

26 Tamar Eilam, Cyril Gavoille, and David Peleg. Compact routing schemes with low stretch factor. *Journal of Algorithms*, 46(2):97 – 114, 2003. doi:10.1016/S0196-6774(03)00002-6.

27 M. Enachescu, M. Wang, and A. Goel. Reducing maximum stretch in compact routing. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 336–340, 2008. doi:10.1109/INFOCOM.2008.76.

28 D. Eppstein and M. T. Goodrich. Succinct greedy geometric routing using hyperbolic geometry. *IEEE Transactions on Computers*, 60(11):1571–1580, 2011. doi:10.1109/TC.2010.257.

29 David Eppstein and Michael T Goodrich. Succinct greedy graph drawing in the hyperbolic plane. In *International Symposium on Graph Drawing*, pages 14–25, 2008. doi:10.1007/978-3-642-00219-9_3.

30 R. Flury, S. V. Pemmaraju, and R. Wattenhofer. Greedy routing with bounded stretch. In *IEEE INFOCOM 2009*, pages 1737–1745, 2009. doi:10.1109/INFCOM.2009.5062093.

31 Ofer Freedman, Paweł Gawrychowski, Patrick K. Nicholson, and Oren Weimann. Optimal distance labeling schemes for trees. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, page 185–194, 2017. doi:10.1145/3087801.3087804.

32 Cyril Gavoille and Olivier Ly. Distance labeling in hyperbolic graphs. In *Algorithms and Computation*, volume 3827, pages 1071–1079, 2005. doi:10.1007/11602613_106.

33 Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. *Journal of Algorithms*, 53(1):85–112, 2004. doi:10.1016/j.jalgor.2004.05.002.

34 Cyril Gavoille and Stéphane Pérennès. Memory requirement for routing in distributed networks. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, page 125–133, 1996. doi:10.1145/248052.248075.

35 Michael T. Goodrich and Darren Strash. Succinct greedy geometric routing in the euclidean plane. In *Algorithms and Computation*, pages 781–791, 2009.

36 Luca Gugelmann, Konstantinos Panagiotou, and Ueli Peter. Random hyperbolic graphs: degree sequence and clustering. In *International Colloquium on Automata, Languages, and Programming*, pages 573–585. Springer, 2012.

37 R. Houthooft, S. Sahhaf, W. Tavernier, F. De Turck, D. Colle, and M. Pickavet. Fault-tolerant greedy forest routing for complex networks. In *2014 6th International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 1–8, 2014. doi:10.1109/RNDM.2014.7014924.

38 M.L. Huson and A. Sen. Broadcast scheduling algorithms for radio networks. In *Proceedings of MILCOM '95*, volume 2, pages 647–651 vol.2, 1995. doi:10.1109/MILCOM.1995.483546.

39 Haim Kaplan, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth. Routing in unit disk graphs. *Algorithmica*, 80(3):830–848, 2018. doi:10.1007/s00453-017-0308-2.

40 Brad Karp and H. T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, page 243–254, 2000. doi:10.1145/345910.345953.

41 Zoltán Király and Sándor Kisfaludi-Bak. A succinct tree coding for greedy navigation. Technical Report TR-2015-02, Egerváry Research Group, Budapest, 2015. egres.elte.hu.

42 Sándor Kisfaludi-Bak. Hyperbolic intersection graphs and (quasi)-polynomial time. In *Symposium on Discrete Algorithms (SODA)*, pages 1621–1638, 2020. doi:10.1137/1.9781611975994.100.

43 Jon Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing (STOC)*, page 163–170, 2000. doi:10.1145/335305.335325.

44 R. Kleinberg. Geographic routing using hyperbolic space. In *Proceedings of the IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, page 1902–1909, 2007. doi:10.1109/INFCOM.2007.221.

45 D. Krioukov, K. Fall, and X. Yang. Compact routing on internet-like graphs. In *IEEE INFOCOM 2004*, volume 1, page 219, 2004. doi:10.1109/INFCOM.2004.1354495.

46 Dmitri Krioukov, k c claffy, Kevin Fall, and Arthur Brady. On compact routing for the internet. *SIGCOMM Comput. Commun. Rev.*, 37(3):41–52, 2007. doi:10.1145/1273445.1273450.

47 Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. Hyperbolic geometry of complex networks. *Phys. Rev. E*, 82:036106, 2010. `doi:10.1103/PhysRevE.82.036106`.

48 Anton Krohmer. *Structures & Algorithms in Hyperbolic Random Graphs*. Doctoral thesis, Universität Potsdam, 2016.

49 R. B. Muhammad. A distributed geometric routing algorithm for ad hocwireless networks. In *Fourth International Conference on Information Technology (ITNG'07)*, pages 961–963, 2007. `doi:10.1109/ITNG.2007.4`.

50 Tobias Müller and Merlijn Staps. The diameter of kpkvb random graphs. *Advances in Applied Probability*, 51(2):358–377, 2019. `doi:10.1017/apr.2019.23`.

51 Christos H. Papadimitriou and David Ratajczak. On a conjecture related to geometric routing. *Theoretical Computer Science*, 344(1):3–14, 2005. `doi:10.1016/j.tcs.2005.06.022`.

52 F. Papadopoulos, D. Krioukov, M. Boguna, and A. Vahdat. Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. In *2010 Proceedings IEEE INFOCOM*, pages 1–9, 2010. `doi:10.1109/INFCOM.2010.5462131`.

53 M. Patrascu, L. Roditty, and M. Thorup. A new infinity of distance oracles for sparse graphs. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 738–747, 2012. `doi:10.1109/FOCS.2012.44`.

54 David Peleg and Alejandro A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989. `doi:10.1002/jgt.3190130114`.

55 David Peleg and Eli Upfal. A trade-off between space and efficiency for routing tables. *J. ACM*, 36(3):510–530, 1989. `doi:10.1145/65950.65953`.

56 Mihai Pătraşcu and Liam Roditty. Distance oracles beyond the thorup–zwick bound. *SIAM Journal on Computing*, 43(1):300–311, 2014. `doi:10.1137/11084128X`.

57 Vijay Raghavan and Jeremy Spinrad. Robust algorithms for restricted domains. *Journal of Algorithms*, 48(1):160–172, 2003. `doi:https://doi.org/10.1016/S0196-6774(03)00048-8`.

58 Ananth Rao, Sylvia Ratnasamy, Christos Papadimitriou, Scott Shenker, and Ion Stoica. Geographic routing without location information. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, page 96–108, 2003. `doi:10.1145/938985.938996`.

59 Liam Roditty and Roei Tov. Close to linear space routing schemes. *Distributed Computing*, 29(1):65 – 74, 2016. `doi:10.1007/s00446-015-0256-5`.

60 Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015. URL: `http://networkrepository.com`.

61 N. Santoro and R. Khatib. Labelling and implicit routing in networks. *The Computer Journal*, 28(1):5–8, 1985. `doi:10.1093/comjnl/28.1.5`.

62 H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–257, 1984. `doi:10.1109/TCOM.1984.1096061`.

63 M. Tang, H. Chen, G. Zhang, and J. Yang. Tree cover based geographic routing with guaranteed delivery. In *2010 IEEE International Conference on Communications*, pages 1–5, 2010. `doi:10.1109/ICC.2010.5502391`.

64 Mingdong Tang, Jing Yang, and Guoqing Zhang. Compact routing on random power law graphs. In *8th IEEE International Symposium on Dependable, Autonomic and Secure Computing, DASC 2009*, pages 575–578, 12 2009. `doi:10.1109/DASC.2009.133`.

65 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004. `doi:10.1145/1039488.1039493`.

66 Mikkel Thorup and Uri Zwick. Compact routing schemes. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, page 1–10, 2001. `doi:10.1145/378580.378581`.

67 Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005. `doi:10.1145/1044731.1044732`.

68 C. Westphal and G. Pei. Scalable routing via greedy embedding. In *IEEE INFOCOM 2009*, pages 2826–2830, 2009. doi:10.1109/INFCOM.2009.5062240.

69 Huaming Zhang and Swetha Govindaiah. Greedy routing via embedding graphs onto semi-metric spaces. In *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*, pages 58–69, 2011.

70 Huaming Zhang and Swetha Govindaiah. Greedy routing via embedding graphs onto semi-metric spaces. *Theoretical Computer Science*, 508:26–34, 2013. doi:10.1016/j.tcs.2012.01.049.