Odd Yao-Yao Graphs are Not Spanners

Yifei Jin
Tsinghua University
Beijing 100084, China
jin-yf13@mails.tsinghua.edu.cn

Jian Li
Tsinghua University
Beijing 100084, China
corresponding author: lijian83@mail.tsinghua.edu.cn

Wei Zhan
Princeton University
Princeton, NJ 08544, USA
weizhan@cs.princeton.edu

Abstract

It is a long standing open problem whether Yao-Yao graphs YY_k are all spanners [Li et al. 2002]. Bauer and Damian [Bauer and Damian, 2012] showed that all YY_{6k} for $k \geq 6$ are spanners. Li and Zhan [Li and Zhan, 2016] generalized their result and proved that all even Yao-Yao graphs YY_{2k} are spanners (for $k \geq 42$). However, their technique cannot be extended to odd Yao-Yao graphs, and whether they are spanners are still elusive. In this paper, we show that, surprisingly, for any integer $k \geq 1$, there exist odd Yao-Yao graph YY_{2k+1} instances, which are not spanners.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Odd Yao-Yao Graph, Spanner, Counterexample

Digital Object Identifier 10.4230/LIPIcs.SoCG.2018.49

Related Version A full version of this paper is available at https://arxiv.org/abs/1704.03132.

Funding This research is supported in part by the National Basic Research Program of China Grant 2015CB358700, the National Natural Science Foundation of China Grant 61772297, 61632016, 61761146003, and a grant from Microsoft Research Asia.

1 Introduction

Let $\mathcal{P}$ be a set of points in the Euclidean plane $\mathbb{R}^2$. The complete Euclidean graph defined on set $\mathcal{P}$ is the edge-weighted graph with vertex set $\mathcal{P}$ and edges connecting all pairs of points in $\mathcal{P}$, where the weight of each edge is the Euclidean distance between its two end points. Storing the complete graph requires quadratic space, which is very expensive. Hence, it is desirable to use a sparse subgraph to approximate the complete graph. This is a classical and well-studied topic in computational geometry (see e.g., [1, 19, 26, 32, 34]). In this paper, we study the so called *geometric t-spanner*, formally defined as follows (see e.g., [29]).

▶ **Definition 1** (Geometric t-Spanner). A graph G is a *geometric t-spanner* of the complete Euclidean graph, if (1) G is a subgraph of the complete Euclidean graph; and (2) for any pair of points p and q in $\mathcal{P}$, the shortest path between p and q in G is no longer than t times the Euclidean distance between p and q.

34th International Symposium on Computational Geometry (SoCG 2018).
Editors: Bettina Speckmann and Csaba D. Tóth; Article No. 49; pp. 49:1–49:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The factor t is called the *stretch factor* or *dilation factor* of the spanner in the literature. If the maximum degree of G is bounded by a constant k, we say that G is a *bounded-degree spanner*. The concept of geometric spanners was first proposed by L.P. Chew [10]. See the comprehensive survey by Eppstein [16] for related topics about geometric spanners. Geometric spanners have found numerous applications in wireless ad hoc and sensor networks. We refer the readers to the books by Li [24] and Narasimhan and Smid [27] for more details.

Yao graphs are one of the first approximations of complete Euclidean graphs, introduced independently by Flinchbaugh and Jones [18] and Yao [34].

▶ **Definition 2** (Yao Graph Y_k)**.** Let k be a fixed integer. Given a set of points $\mathcal{P}$ in the Euclidean plane $\mathbb{R}^2$, the Yao graph $\mathsf{Y}_k(\mathcal{P})$ is defined as follows. Let $C_u(\gamma_1, \gamma_2]$ be the cone with apex u, which consists of the rays with polar angles in the half-open interval $(\gamma_1, \gamma_2]$. For each $u \in \mathcal{P}$, $\mathsf{Y}_k(\mathcal{P})$ contains an edge connecting u to a nearest neighbor v in each cone $C_u(j\theta, (j+1)\theta]$, for $\theta = 2\pi/k$ and $j \in [0, k-1]$. We generally consider Yao graphs as undirected graphs. For a *directed Yao graph*, we add directed edge $\overrightarrow{uv}$ to the graph instead.

Molla [15] showed that Y_2 and Y_3 may not be spanners. On the other hand, it has been proven that all Y_k for $k \geq 4$ are spanners. Bose et al. [6] proved that Y_4 is a 663-spanner. Damian and Nelavalli [13] improved this to 54.6 recently. Barba et al. [2] showed that Y_5 is a 3.74-spanner. Damian and Raudonis [14] proved that the Y_6 graph is a 17.64 spanner. Li et al. [25, 26] first proved that all $\mathsf{Y}_k, k > 6$ are spanners with stretch factor at most $1/(1 - 2\sin(\pi/k))$. Later Bose et al. [6, 7] also obtained the same result independently. Recently, Barba et al. [2] reduced the stretch factor of Y_6 from 17.6 to 5.8 and improved the stretch factors to $1/(1 - 2\sin(3\pi/4k))$ for odd $k \geq 7$.

However, a Yao graph may not have bounded degree. This can be a serious limitation in certain wireless network applications since each node has very limited energy and communication capacity, and can only communicate with a small number of neighbors. To address the issue, Li et al. [26] introduced *Yao-Yao* graphs (or *Sparse-Yao* graphs in the literature). A Yao-Yao graph $\mathsf{YY}_k(\mathcal{P})$ is obtained by removing some edges from $\mathsf{Y}_k(\mathcal{P})$ as follows:

▶ **Definition 3** (Yao-Yao Graph YY_k)**.** (1) Construct the directed Yao graph, as in Definition 2. (2) For each node u and each cone rooted at u containing two or more incoming edges, retain a shortest incoming edge and discard the other incoming edges in the cone. We can see that the maximum degree in $\mathsf{YY}_k(\mathcal{P})$ is upper-bounded by $2k$.

As opposed to Yao graphs, the spanning property of Yao-Yao graphs is not well understood yet. Li et al. [26] provided some empirical evidence, suggesting that YY_k graphs are t-spanners for some sufficiently large constant k. However, there is no theoretical proof yet, and it is still an open problem [4, 23, 24, 26]. It is also listed as Problem 70 in the Open Problems Project.[1]

▶ **Conjecture 4** (see [4])**.** There exists a constant k_0 such that for any integer $k > k_0$, any Yao-Yao graph YY_k is a geometric spanner.

Now, we briefly review the previous results about Yao-Yao graphs. It is known that YY_2 and YY_3 may not be spanners since Y_2 and Y_3 may not be spanners [15]. Damian and Molla [12, 15] proved that $\mathsf{YY}_4, \mathsf{YY}_6$ may not be spanners. Bauer et al. [2] proved that YY_5 may not be spanners. On the positive side, Bauer and Damian [4] showed that for any integer

[1] `http://cs.smith.edu/~orourke/TOPP/P70.html`

$k \geq 6$, any Yao-Yao graph YY_{6k} is a spanner with the stretch factor at most 11.67 and the factor becomes 4.75 for $k \geq 8$. Recently, Li and Zhan [23] proved that for any integer $k \geq 42$, any even Yao-Yao graph YY_{2k} is a spanner with the stretch factor $6.03 + O(k^{-1})$.

From these positive results, it is quite tempting to believe Conjecture 4. However, we show in this paper that, surprisingly, Conjecture 4 is false for odd Yao-Yao graphs.

▶ **Theorem 5.** *For any $k \geq 1$, there exists a class of point set instances $\{\mathcal{P}_m\}_{m \in \mathbb{Z}^+}$ such that the stretch factor of $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ cannot be bounded by any constant, as m approaches infinity.*[2]

Related work. It has been proven that in some special cases, Yao-Yao graphs are spanners [11,21,22,33]. Specifically, it was shown that YY_k graphs are spanners in *civilized graphs*, where the ratio of the maximum edge length to the minimum edge length is bounded by a constant [21,22].

Besides the Yao and Yao-Yao graph, the Θ-*graph* is another common geometric t-spanner. The difference between Θ-graphs and Yao graphs is that in a Θ-graph, the nearest neighbor to u in a cone C is a point $v \neq u$ lying in C and minimizing the Euclidean distance between u and the orthogonal projection of v onto the bisector of C. It is known that except for Θ_2 and Θ_3 [15], for $k = 4$ [3], 5 [8], 6 [5], ≥ 7 [9,28], Θ_k-graphs are all geometric spanners. We note that, unfortunately, the degrees of Θ-graphs may not be bounded.

Recently, some variants of geometric t-spanners such as weak t-spanners and power t-spanners have been studied. In weak t-spanners, the path between two points may be arbitrarily long, but must remain within a disk of radius t-times the Euclidean distance between the points. It is known that all Yao-Yao graphs YY_k for $k > 6$ are weak t-spanners [20,30,31]. In power t-spanners, the Euclidean distance $|\cdot|$ is replaced by $|\cdot|^\kappa$ with a constant $\kappa \geq 2$. Schindelhauer et al. [30,31] proved that for $k > 6$, all Yao-Yao graphs YY_k are power t-spanners for some constant t. Moreover, it is known that any t-spanner is also a weak t_1-spanner and a power t_2-spanner for some t_1, t_2 depending only on t. However, the converse is not true [31].

Our counterexample is inspired by the concept of fractals. Fractals have been used to construct examples for β-skeleton graphs with unbounded stretch factors [17]. Here a β-skeleton graph is defined to contain exactly those edges ab such that no point c forms an angle $\angle acb$ greater than $\sin^{-1} 1/\beta$ if $\beta > 1$ or $\pi - \sin^{-1} \beta$ if $\beta < 1$. Schindelhauer et al. [31] used the same example to prove that there exist graphs which are weak spanners but not t-spanners. However, their examples cannot serve as counterexamples to the conjecture that odd Yao-Yao graphs are spanners.

2 Overview of our counterexample construction

We first note that both the counterexamples for YY_3 and YY_5 are not weak t-spanners [2,15]. However, Yao-Yao graphs YY_k for $k \geq 7$ are all weak t-spanners [20,30,31]. Hence, to construct the counterexamples for YY_k for $k \geq 7$, the previous ideas for YY_3 and YY_5 cannot be used. We will construct a class of instances $\{\mathcal{P}_m\}_{m \in \mathbb{Z}^+}$ such that all points in $\mathcal{P}_m$ are placed in a bounded area. Meanwhile, there exist shortest paths in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ whose lengths approach infinity as m approaches infinity.

[2] Here, m is a parameter in our recursive construction. We will explain it in detail in Section 3. Roughly speaking, m is the level of recursion, and the number of points in $\mathcal{P}_m$ increases with m.

Our example contains two types of points, called *normal points* and *auxiliary points*. Denote them by $\mathcal{P}_m^{\mathrm{n}}$ and $\mathcal{P}_m^{\mathrm{a}}$ respectively and $\mathcal{P}_m = \mathcal{P}_m^{\mathrm{n}} \cup \mathcal{P}_m^{\mathrm{a}}$. The normal points form the basic skeleton, and the auxiliary points are used to break the edges connecting any two normal points that are far apart.

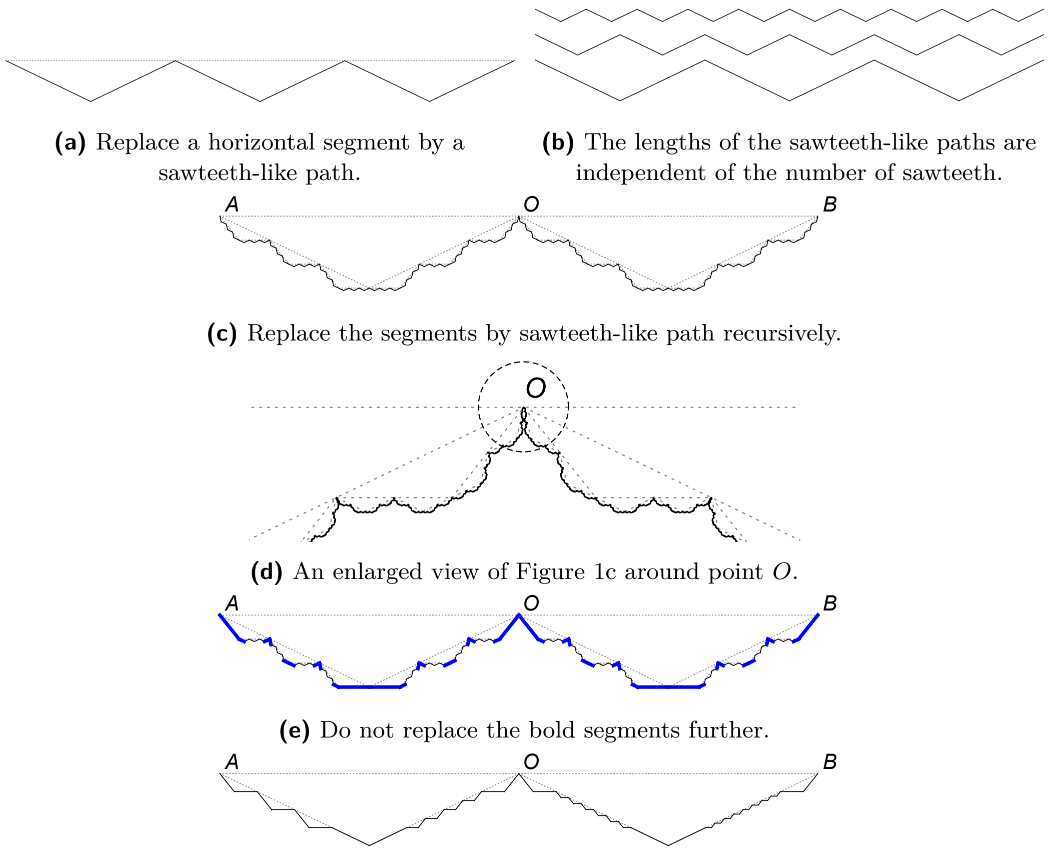
We are inspired by the concept of fractals to construct the normal points. A fractal can be contained in a bounded area, but its length may diverge. In our counterexample, the shortest path between two specific normal points is a fractal-like polygonal path. Here a polygonal path refers to a curve specified by a sequence of points and consists of the line segments connecting the consecutive points. Suppose the two specific points are A and B, AB is horizontal, and $|AB| = 1$. When $m = 0$, the polygonal path is just the line segment AB. When m increases by one, we replace each line segment in the current polygonal path by a *sawteeth-like* path (see Figure 1a). If the angle between each segment of the sawteeth-like path and the base segment (i.e., the one which is replaced) is γ, the total length of the path increases by a factor of $\cos^{-1}\gamma$. An important observation here is that the factor is independent of the number of sawteeth (see Figure 1b). If we continue this process directly, the length of the resulting path would increase to infinity as m approaches infinity since $\cos^{-1}\gamma > 1$ (see Figure 1c). However, we need to make sure that such a path is indeed in a Yao-Yao graph and it is indeed the shortest path from A to B. There are two technical difficulties we need to overcome.

1. As m increases, the polygonal path may intersect itself. See Figure 1d. The polygonal path intersects itself around the point O. This is relatively easy to handle: we do not recurse for those segments that may cause self-intersection. See Figure 1e. We do not replace the bold segment further. We need to make sure that the total length of such segments is proportionally small (so that the total length can keep increasing as m increases).
2. In the Yao-Yao graph defined over the normal points constructed in the recursion, there may be some edges connecting points that are far apart. Actually, how to break such edges is the main difficulty of the problem. We outline the main techniques below.

First, we *do not* replace all current segments using the same sawteeth, like in the usual fractal construction. Actually, for each segment, we will choose a polygonal path such that the paths have different numbers of sawteeth and the sizes of the sawteeth in the path may not be the same. See Figure 1f. Finally, we construct them in a specific sequential order. Actually, we organize the normal points in an m-level *recursion tree* $\mathcal{T}$ and generate them in a DFS preorder traversal of the tree. We describe the details in Section 3.

Second, we group the normal points into a collection of sets such that each normal point belongs to exactly one set. We call such a set a *hinge set*. Refer to Figure 6 for an overview. Then, we specify a *total order* of the hinge sets. Call the edges in the Yao-Yao graph $\mathsf{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$ connecting any two normal points in the same hinge set or two adjacent hinge sets (w.r.t. the total order) *hinge connections* and call the other edges *long range connections*. We describe the details in Section 4.

As we will see, all possible long range connections have a relatively simple form. Then, we show that we can break all long range connections by adding a set $\mathcal{P}_m^{\mathrm{a}}$ of *auxiliary points*. Each auxiliary point has a unique *center* which is the normal point closest to it. Let the minimum distance between any two normal points in $\mathcal{P}_m^{\mathrm{n}}$ be Δ. The distance between an auxiliary point and its center is much less than Δ. Naturally, we can extend the concepts of hinge set and long range connection to include the auxiliary points. An *extended hinge set* consists of the normal points in a hinge set and the auxiliary points centered on these normal points. We will see that the auxiliary points break all long range connections and



(a) Replace a horizontal segment by a sawteeth-like path.

(b) The lengths of the sawteeth-like paths are independent of the number of sawteeth.

(c) Replace the segments by sawteeth-like path recursively.

(d) An enlarged view of Figure 1c around point O.

(e) Do not replace the bold segments further.

(f) The paths have different numbers of sawteeth and the sizes of sawteeth may not be the same.

Figure 1 The overview of the counterexample construction. Figure 1a-1f illustrate the fractal and its variants.

introduce no new long range connection. We describe the details in Section 5.

Finally, according to the process above, we can see that the shortest path between the normal points A and B in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ for $m \in \mathbb{Z}^+$ should pass through all extended hinge sets in order. Thereby, the length of the shortest path between A and B diverges as m approaches infinity. See Section 5 for the details.

3 The positions of normal points

3.1 Some basic concepts

Let $k \geq 3$ be a fixed positive integer.[3] We consider YY_{2k+1} and let $\theta = 2\pi/(2k+1)$.

▶ **Definition 6** (Cone Boundary). Consider any two points u and v. If the polar angle of $\overrightarrow{uv}$ is $j\theta = j \cdot 2\pi/(2k+1)$ for some integer $j \in [0, 2k]$, we call the ray $\overrightarrow{uv}$ a cone boundary for point u.

[3] Note that the cases $k = 1, 2$ have been proved in [15]

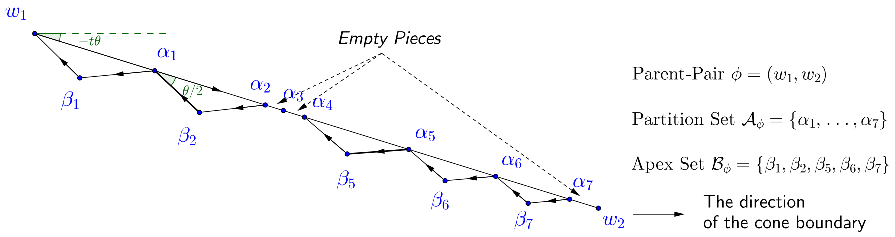

w_1
$-t\theta$
α_1
$\theta/2$
β_1
β_2
α_2
α_3
α_4
Empty Pieces
α_5
β_5
α_6
β_6
α_7
β_7
w_2
Parent-Pair $\phi = (w_1, w_2)$
Partition Set $\mathcal{A}_\phi = \{\alpha_1, \ldots, \alpha_7\}$
Apex Set $\mathcal{B}_\phi = \{\beta_1, \beta_2, \beta_5, \beta_6, \beta_7\}$
The direction of the cone boundary

Figure 2 An example of one gadget. $\phi = (w_1, w_2)$ is the parent-pair in the gadget. $\mathcal{A}_\phi = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_7\}$ is the partition set and $\mathcal{B}_\phi = \{\beta_1, \beta_2, \beta_5, \beta_6, \beta_7\}$ is the apex set. There are eight pieces, in which $w_1\alpha_1, \alpha_1\alpha_2, \alpha_4\alpha_5, \alpha_5\alpha_6, \alpha_6\alpha_7$ are non-empty pieces and $\alpha_2\alpha_3, \alpha_3\alpha_4, \alpha_7 w_2$ are empty pieces.

▶ **Property 7.** Consider two points u and v in $\mathcal{P}$. If $\overrightarrow{uv}$ consists of a cone boundary in $\mathsf{YY}_{2k+1}(\mathcal{P})$, its reverse $\overrightarrow{vu}$ is not a cone boundary.

In retrospect, this property is a key difference between odd and even Yao-Yao graphs, and our counterexample for odd Yao-Yao graphs will make crucial use of the property.

▶ **Definition 8** (Boundary Pair)**.** A boundary pair consists of two ordered points, denoted by (w_1, w_2), such that $\overrightarrow{w_1 w_2}$ is a cone boundary of point w_1.

For convenience, we refer to the word *pair* in the paper as the boundary pair defined in Definition 8. According to Property 7, if (w_1, w_2) is a pair, its reverse (w_2, w_1) is not a pair. Moreover, if a pair ϕ is $(u, \cdot)$ or $(\cdot, u)$, we say that the point u belongs to ϕ (i.e., $u \in \phi$).

Gadget One gadget G_ϕ consists of three groups of points. We explain them one by one. See Figure 2 for an example.

1. The first group is the pair $\phi = (w_1, w_2)$. We call ϕ the *parent-pair* of the gadget G_ϕ.
2. The second group is a set $\mathcal{A}_\phi$ of points on the segment of (w_1, w_2). We call the set $\mathcal{A}_\phi$ a *partition set* and call the points of $\mathcal{A}_\phi$ the *partition points* of ϕ. The set $\mathcal{A}_\phi$ divides the segment into $|\mathcal{A}_\phi| + 1$ parts, each we call a *piece* of the segment. There are two types of pieces. One is called an *empty piece* and the other a *non-empty piece*. Whether a piece is empty or not is determined in the process of the construction, which we will explain in Section 3.2.
3. For each non-empty piece, $\alpha_{i-1}\alpha_i$, we add a point β_i such that $\angle \alpha_{i-1}\beta_i\alpha_i = \pi - \theta$ and $|\alpha_{i-1}\beta_i| = |\beta_i\alpha_i|$. All β_is are on the same side of $w_1 w_2$. We call such a point β_i an *apex point* of (w_1, w_2). Let $\mathcal{B}_\phi$ be the set of apex points generated by ϕ, which is called the *apex set* of pair ϕ. $\mathcal{B}_\phi$ is the third group of points. For any empty piece, we do not add the corresponding apex point.

Consider a gadget $\mathsf{G}_\phi[\mathcal{A}_\phi, \mathcal{B}_\phi]$, where $\phi = (w_1, w_2)$. For any non-empty piece $\alpha_{i-1}\alpha_i$ and the corresponding apex point β_i, the rays $\overrightarrow{\beta_i\alpha_{i-1}}$ and $\overrightarrow{\alpha_i\beta_i}$ (note the order of the points) are cone boundaries according to their polar angles. Thus, each point $\beta_i \in \mathcal{B}_\phi$ induces two pairs (β_i, α_{i-1}) and (α_i, β_i). We call all pairs (β_i, α_{i-1}) and (α_i, β_i) induced by points in $\mathcal{B}_\phi$ the *child-pairs* of (w_1, w_2), and we say that they are *siblings* of each other. Note that there are some partition point which are not incident on any pair (e.g., α_3 in Figure 2). We call it an *isolated* point.

▶ **Definition 9** (The Order of the Child-pairs)**.** Consider a gadget $\mathsf{G}_{(w_1,w_2)}$. Suppose Φ is the set of the child-pairs of (w_1, w_2). Consider two pairs ϕ, φ in Φ. Define the order $\phi \prec \varphi$, if ϕ is closer to w_1 than φ. Here, the distance from a pair ϕ to a point w is the shortest distance from w to any point of ϕ.

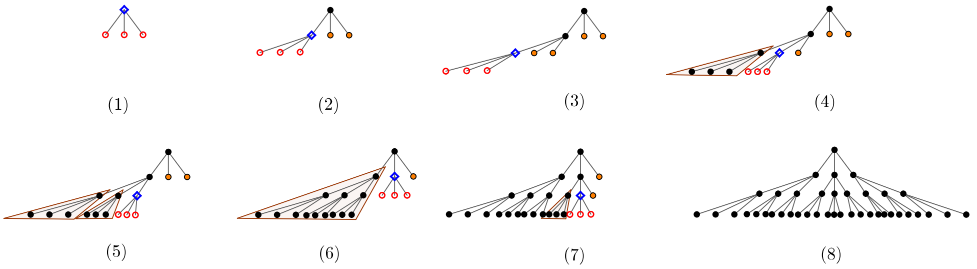

(1) (2) (3) (4) (5) (6) (7) (8)

Figure 3 The process of generating a tree according to the DFS preorder. In each subfigure, ◇ represents a node we are visiting. The nodes generated in the step are denoted by ○. ● represents a node which has already been visited. ● represents a node which has been created but not visited yet. The nodes covered by light brown triangles are related to the projection process.

3.2 The construction

In this subsection, we construct an m-level tree. When the recursion level increases by 1, we need to replace each current pair by a gadget generated by the pair. See Figure 3 for an example. The recursion can be naturally represented as a tree $\mathcal{T}$. The tree is generated according to the DFS preorder, starting from the root. W.l.o.g, we assume that the root of $\mathcal{T}$ is (μ_1, μ_2) and $\mu_1\mu_2$ is horizontal. We call a pair *a leaf-pair* if it is a leaf node in the tree and *an internal-pair* otherwise. Each time when we visit an internal-pair, we generate its gadget by two steps which are called *projection* and *refinement* and explained in detail soon. Generating its gadget is equivalent to generating its child-pairs in $\mathcal{T}$ (we, however, do not visit those children during the generation. They will be visited later according to the DFS preorder). Whether a child-pair is a leaf or not is determined when the gadget is created. Hence, note that not all leaf-pairs are at *level-m*. We call the points generated by the process *normal points* and denote the set of these points by $\mathcal{P}_m^{\mathrm{n}}$ where m is the level of the tree and n represents the word "normal".

Root gadget. Let d_0 be a large positive constant integer (see Lemma 16 for detail). Consider a pair $\phi = (\mu_1, \mu_2)$. Let $\mathcal{A}_\phi$ be its partition set which contains points

$$\alpha_i = \mu_1 \cdot \frac{d_0 - i}{d_0} + \mu_2 \cdot \frac{i}{d_0}, i \in [1, d_0 - 1].$$

For convenience, let $\alpha_0 = \mu_1, \alpha_{d_0} = \mu_2$. The points in $\mathcal{A}_\phi$ partition the segment $\mu_1\mu_2$ into d_0 pieces with equal length $|\mu_1\mu_2|/d_0$. All pieces in the root gadget are non-empty. For each piece $\alpha_{i-1}\alpha_i$, we add an apex point β_i below $\mu_1\mu_2$. Let $\mathcal{B}_\phi = \{\beta_i\}_{i\in[1,d_0]}$ be the apex set.

Projection and refinement. The projection and refinement generate the partition points of pair ϕ. The purpose of the projection is to restrict all possible *long range connections* to a relatively simple form. See Section 4 for the details. The purpose of the refinement is to make the sibling pairs have relatively the same length, hence, make it possible to repeat the projection process recursively. Formally speaking, the refinement maintains the following property over the construction.

▶ **Property 10.** We call the segment connecting the two points of the pair the *segment of the pair* and call the length of that segment the *length of the pair.* Consider an internal-pair ϕ. Suppose φ is a sibling of ϕ. The length of pair φ is at least half of the length of pair ϕ.

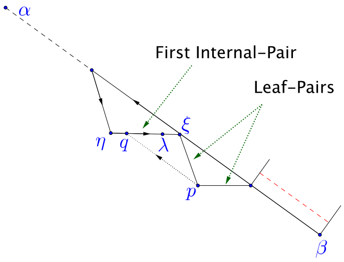
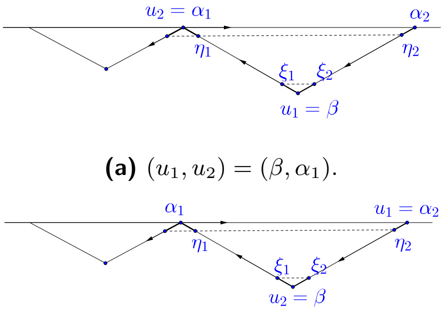


Figure 4 An example of the projection for the first internal-pair $\phi = (\eta, \xi)$.



(a) $(u_1, u_2) = (\beta, \alpha_1)$.

(b) $(u_1, u_2) = (\alpha_2, \beta)$.

Figure 5 The two cases for refinement.

– **Projection.** Consider a pair (β, α) with the set Φ being its child-pairs. We decide whether a pair in Φ is a leaf-pair or an internal-pair after introducing the process projection and refinement. There is a clear order between the leaf-pairs and the internal-pairs. We provide that the property of the order here and prove it in the full version.

▶ **Property 11.** Consider a pair (β, α) with the set Φ of its child-pairs. For $\phi_1, \phi_2, \phi_3 \in \Phi$, if $\phi_1 \prec \phi_2 \prec \phi_3$ and ϕ_1 and ϕ_3 are two internal-pairs, then ϕ_2 is an internal-pair.

Next, we apply the projection to an internal-pair $\phi \in \Phi$ since only internal-pairs have children. We explain the projection formally below. We define the first internal-pair in direction $\overrightarrow{\beta\alpha}$ as the *first internal-pair* of Φ. Let $\mathcal{T}_\phi$ be the set of points in the subtree rooted at ϕ. Depending on whether ϕ is the first internal-pair of Φ, there are two cases.

- Pair ϕ is the first internal-pair of Φ: In Figure 4, suppose pair $\phi = (\eta, \xi)$ is the first internal child-pair of (β, α) and the length of ϕ is δ. Point ξ is the partition point in ϕ. First, we add a point λ on the segment of ϕ such that $|\xi\lambda| = \delta/d_0$. Second, for each leaf-pair $\varphi \prec \phi$, project the apex point in φ to the segment of ϕ along the direction $\overrightarrow{\beta\alpha}$, [4] e.g., project p to q in Figure 4. Note that the length of leaf-pair φ is at least $\delta/2$ according to Property 10. Thus, there is no point between λ and ξ as long as $d_0 > 2$. Formally, we denote the operation by

$$\widehat{\mathcal{A}}_\phi \leftarrow \mathsf{Proj}\left[\bigcup\nolimits_{\varphi \prec \phi, \varphi \in \Phi} \mathcal{T}_\varphi\right] \cup \lambda. \tag{1}$$

- Pair ϕ is not the first internal-pair: According to the DFS preorder, we have already constructed the subtrees rooted at $\varphi \prec \phi$. We project all points $p \in \bigcup_{\varphi \prec \phi, \varphi \in \Phi} \mathcal{T}_\varphi$ to the segment of ϕ along the direction $\overrightarrow{\beta\alpha}$. Let the partition set $\widehat{\mathcal{A}}_\phi$ of ϕ be the set of the projected points falling inside the segment of ϕ. If several points overlap, we keep only one of them. Formally, we denote the operation by

$$\widehat{\mathcal{A}}_\phi \leftarrow \mathsf{Proj}\left[\bigcup\nolimits_{\varphi \prec \phi, \varphi \in \Phi} \mathcal{T}_\varphi\right]. \tag{2}$$

– **Refinement.** The partition points of ϕ divide the segment of ϕ into pieces. In order to maintain Property 10, we need to ensure that all non-empty pieces of ϕ have approximately

[4] If the projected point falls outside the segment of ϕ, we do not need to add a normal point.

the same length. We call this process the *refinement* operation. After the projection, we obtain a candidate partition set $\widehat{\mathcal{A}}_\phi$ of ϕ (defined in (1) and (2)). However, note that the length between the pieces may differ a lot.

W.l.o.g., suppose pair ϕ has unit length, $|\widehat{\mathcal{A}}_\phi| = n$ and $n > d_0$.[5] Suppose $\phi = (u_1, u_2)$. We distinguish two cases based on whether the first point u_1 is a partition point or an apex point.

- If u_1 is an apex point, we mark the piece incident on u_2. See Figure 5a for an illustration, in which $(u_1, u_2) = (\beta, \alpha_1)$ and piece $\alpha_1\eta_1$ is the marked piece.
- If u_1 is a partition point, we mark the pieces incident on u_1 and u_2. See Figure 5b for an illustration, in which $(u_1, u_2) = (\alpha_2, \beta)$ and piece $\alpha_2\eta_2$ and $\xi_2\beta$ are the marked pieces.

We do not add any point in the marked pieces under refinement. Consider two sibling pairs (β, α_1) and (α_2, β) where β is an apex point. Suppose $\alpha_1\eta_1, \alpha_2\eta_2, \xi_2\beta$ are the marked pieces of the two pairs and ξ_1 is the point on the segment $\beta\alpha_1$ which is projected to ξ_2.[6] Then $|\beta\xi_1| = |\beta\xi_2|$ and $|\alpha_1\eta_1| = |\alpha_2\eta_2|$ after the refinement. See Figure 5 for an example.

Denote the length of the ith piece (defined by $\widehat{\mathcal{A}}_\phi$) by δ_i. Let $\delta_o = 1/n^2$. Except for the marked pieces, for each piece which is at least twice longer than δ_o, we place $\lfloor \delta_i/\delta_o \rfloor - 1$ equidistant points on the piece, which divide the piece into $\lfloor \delta_i/\delta_o \rfloor$ equal-length parts.

We call this process the *refinement* and denote the resulting point set by

$$\mathcal{A}_\phi \leftarrow \mathsf{Refine}[\widehat{\mathcal{A}}_\phi]. \tag{3}$$

Note that The number of points added in the refinement process is at most $O(n^2)$ since the segment of pair ϕ has unit length and $\delta_o \geq 1/n^2$. We call each piece whose length is less than δ_o a *short piece*. The short pieces remain unchanged before and after the refinement. Moreover, the refinement does not introduce any new short piece for the pair. In the full version, we prove that the sum of lengths of the short pieces is less than $1/d_0$.

Deciding Emptiness, Leaf-Pairs and Internal-Pairs We defer the details to the full version and just give the definitions here. In the full version, we will prove Property 10 and provide more properties of the construction.

Consider a pair ϕ whose apex point is β and partition point is α.[7] We let the piece incident on the apex point β and the short pieces be empty and the other pieces be non-empty.

For each non-empty piece, we generate one apex point. As we have discussed before, the apex set $\mathcal{B}_\phi$ induces the set Φ of child-pairs of ϕ. Let the three pairs closest to α and two pairs closest to β be *leaf-pairs*. We do not further expand the tree from the leaf-pairs. Let the other pairs be the *internal-pairs*.

Overall, after the projection and refinement process, we can generate the gadget for any pair in the tree. We denote this process by

$$\mathsf{G}_\phi \leftarrow \mathsf{Proj\text{-}Refn}(\phi). \tag{4}$$

In the full version, we prove that the Property 10 and 11 hold under the construction and provide some other properties about the construction.

[5] If $n \leq d_0$, we repeatedly split the inner pieces (i.e., all pieces except for the two pieces incident on the points of ϕ) into two equal-length pieces until the number of the points in $\widehat{\mathcal{A}}_\phi$ is larger than d_0.

[6] Point ξ_1 must exist since ξ_2 is a projected point and there is no point in the marked piece $\xi_2\beta$.

[7] Note that the first point of a pair can be either apex point or partition point. Here, $\phi = (\alpha, \beta)$ or $\phi = (\beta, \alpha)$ depending on whether first point of ϕ is apex point or not.

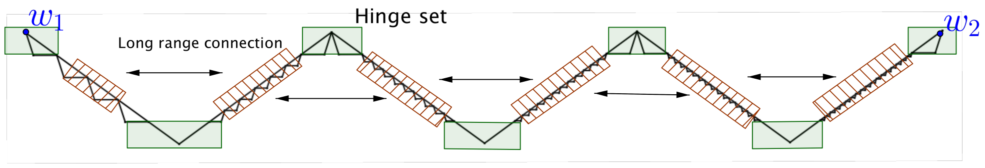
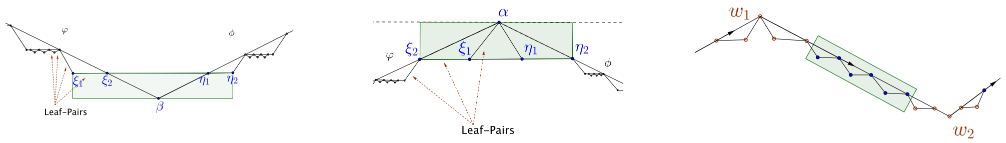


Figure 6 The overview of the hinge set decomposition. Roughly speaking, each set of points covered by a green rectangle 🟩 is a hinge set. Recursively, we can further decompose the points covered by shadowed rectangle ▨ into hinge sets.



(a) The hinge set centered on an apex point.

(b) The hinge set centered on a partition point.

(c) The hinge set consisting of the leaf-pairs at *level-m*.

Figure 7 The hinge sets centered on a point in an internal-pair.

4 Hinge set decomposition of the normal points

We decompose $\mathcal{P}_m^{\mathrm{n}}$ into a collection of sets of points such that each normal point belongs to exactly one set. We call these sets *hinge sets*. Briefly speaking, each hinge set is a set of points which are close geometrically. See Figure 6 for an overview. Consider a pair $\hat{\phi}$ at *level-l*, $l < m - 1$ with partition point set $\mathcal{A}_{\hat{\phi}}$ and apex point set $\mathcal{B}_{\hat{\phi}}$. Let $\hat{\Phi}$ be the set of the child-pairs of $\hat{\phi}$. Formally, the hinge sets are defined as follows.

- The hinge set centered on a point $\beta \in \mathcal{B}_{\hat{\phi}}$ which belongs to one or two internal-pairs in $\hat{\Phi}$: We denote the two pairs by φ and ϕ. The hinge set centered at β includes: β itself, the child-pair of φ closest to β (denoted by (ξ_1, ξ_2) in Figure 7a) and the child-pair of ϕ closest to β (denoted by (η_1, η_2) in Figure 7a).[8]
- The hinge set centered on a point $\alpha \in \mathcal{A}_{\hat{\phi}}$ which belongs to one or two internal-pairs in $\hat{\Phi}$: We denote the two pairs by φ and ϕ. The hinge set centered on α includes: α itself, the two child-pairs closest to α of φ and ϕ respectively (denoted by (ξ_2, ξ_1) and (η_1, η_2) in Figure 7b).[9]

W.l.o.g., we process the points μ_1 and μ_2 in the root pair in the same way as the partition points in $\mathcal{A}_{(\mu_1,\mu_2)}$. So far, some points at *level-m* still do not belong to any hinge set.

- The hinge set consisting of the leaf-pairs at *level-m*: Consider any pair $\phi = (w_1, w_2)$ at *level-*$(m-1)$. Define the set difference of $\mathcal{A}_\phi \cup \mathcal{B}_\phi$ and the hinge sets centered on w_1 and w_2 as a hinge set.[10] See Figure 7c.

Overall, we decompose the points $\mathcal{P}_m^{\mathrm{n}}$ into a collection of hinge sets.

▶ **Lemma 12.** *Each point p in $\mathcal{P}_m^n$ belongs to exactly one hinge set.*

[8] Note that ξ_1, ξ_2, η_1, η_2 only belong to leaf-pairs.
[9] There is a degenerated case in which α is an isolated point which does not belong to any internal-pair in $\hat{\Phi}$. We process the case in the full version.
[10] Although these points form the leaf-pairs at *level-m*, these leaf-pairs are the "candidate internal-pairs" to generate the points at *level-*$(m+1)$.

Order of the hinge sets. We define the total order of all hinge sets. We denote the order by "$\prec_h$", which is different from the previous order "$\prec$". The $\prec_h$ is in fact consistent with the order of traversing the fractal path from μ_1 to μ_2. Rigorously, we define $\prec_h$ in the full version.

▶ **Definition 13** (Long range connection)**.** We call an edge connecting two points in two non-adjacent hinge sets a *long range connection*.

If there is no long range connection, the total order of the hinge sets corresponds to the ordering of the shortest path from μ_1 to μ_2 in the final construction. It means that each hinge set has at least one point on the shortest path between μ_1 and μ_2 and the order of these points is consistent with $\prec_h$. However, there indeed exist long range connections among normal points. Fortunately, the long range connections in $\mathsf{YY}_{2k+1}(\mathcal{P}^{\mathrm{n}}_m)$ have relatively simple form. After introducing some auxiliary points (in Section 5), we can cut the long range connections without introducing any new long range connections.

Now, we examine long range connections in $\mathsf{YY}_{2k+1}(\mathcal{P}^{\mathrm{n}}_m)$. We give the sketch below and defer the details to the full version. Intuitively, these properties result from Property 7 which does not hold for even Yao-Yao graphs.

- First, we claim that if we can cut all long range connections between the points in $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ for any two sibling pairs ϕ and φ, then there is no long range connection.
- Then, we consider the long range connections between two subtrees of any two sibling pairs. Consider two sibling pairs ϕ and φ with subtrees $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ respectively. Suppose p belongs to $\mathcal{T}_\phi$ and q belongs to $\mathcal{T}_\varphi$. We prove that if directed edge $\overrightarrow{pq}$ is in $\mathsf{YY}_{2k+1}(\mathcal{P}^{\mathrm{n}}_m)$, then $\phi \prec \varphi$.
- Finally, we consider the possible long range connection from $\mathcal{T}_\phi$ to $\mathcal{T}_\varphi$ for $\phi \prec \varphi$. Suppose p belongs to $\mathcal{T}_\phi$. Note that the points of $\mathcal{T}_\varphi$ are located in at most two cones of p based on Property 10. We prove that for each point p, only one of the two cones may contain a long range connection. Moreover, there exists a cone boundary pu such that (1) the long range connection is on the cone boundary if pu is in the cone, (2) or there is a point $q \in \mathsf{G}_\varphi$ on pu and $|pq| < |pv|$ for any point v in the cone if pu is not in the cone.[11]

5 The positions of auxiliary points

We discuss how to use the auxiliary points to cut the long range connections in the Yao-Yao graph $\mathsf{YY}_{2k+1}(\mathcal{P}^{\mathrm{n}}_m)$. Denote the set of auxiliary points by $\mathcal{P}^{\mathrm{a}}_m$. Let $\mathcal{P}_m = \mathcal{P}^{\mathrm{n}}_m \cup \mathcal{P}^{\mathrm{a}}_m$. First, we consider a simple example to see how auxiliary points work. Consider three points u, v and w. Line uv is horizontal, and $\angle wvu = \angle wuv = \theta/2$. The point ξ_1 and ξ_2 are two points on segment uw and vw respectively. $\xi_1\xi_2$ is horizontal. See Figure 8. Note that the polar angles of a cone in the Yao-Yao graph belong to a left side half-open interval in the counterclockwise direction. Thus, uv is in the YY_{2k+1} graph, which is the shortest path between u and v. However, we can add an auxiliary point r close to v and $\angle rvu < \theta/2$. Then, the shortest path between uv becomes $u\xi_1\xi_2 rv$.

The positions of the auxiliary points Let Δ be the minimum distance between any two normal points and n be the number of the normal points. Recall that we partition the root pair μ_1, μ_2 into d_0 equidistant pieces. Let γ be a very small angle, such as $\gamma = \theta d_0^{-1}$. Let

[11] Since a cone is a half-open interval in the counterclockwise direction, each cone has one boundary outside the cone.

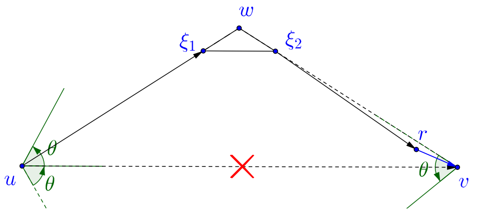


Figure 8 A simple example to explain how an auxiliary point cuts a long range connection.

$\sigma = \max\{\sin(\theta/2-\gamma)/\sin\gamma, \sin^{-1}(\theta/2-\gamma)\}+\epsilon$ for some small $\epsilon > 0$. Let $\chi = d_0\sigma^n\Delta^{-1}$. Roughly speaking, $\chi \gg d_0 > \sigma > 1$.

Now, we explain the positions of the auxiliary points formally. Inspired by the example in Figure 8, we call the normal point closest to an auxiliary point the *center* of the auxiliary point. Then, we find the *candidate centers* to add auxiliary points.

▶ **Definition 14** (Candidate center)**.** Consider a pair (v_1, v_2) and a set Φ of its child-pairs. $\phi, \varphi \in \Phi$ and $\varphi \prec \phi$. p is a point in $\mathcal{T}_\varphi$ and q is the projected point of p in $\mathcal{T}_\phi$. $r \in \mathsf{G}_\phi$ is the closest point to p on segment pq.[12] We call r a *candidate center* of auxiliary points.

We traverse $\mathcal{T}$ in the DFS preorder. Each time we reach a pair ϕ, we find all candidate centers in G_ϕ and add auxiliary points centered on them.[13] Moreover, let the order of ϕ in the DFS preorder w.r.t. $\mathcal{T}$ be κ. The distance between the auxiliary point and its center just depends on κ. Let $\phi = (w_2, w_1)$ and (v_1, v_2) be the parent-pair of ϕ. There are two cases according to $\angle(v_1v_2, w_1w_2) = \theta/2$ or $-\theta/2$.

- $\angle(v_1v_2, w_1w_2) = \theta/2$ (see Figure 9a and 9b):
  - If $q = w_1$, do not add auxiliary point.
  - If $q = w_2$, we add the point η such that $\angle(w_2w_1, w_2\eta) = -\gamma$ and $|w_2\eta| = \sigma^\kappa\chi^{-1}$.
  - Otherwise, we add two points η_1 and η_2 centered on q such that $\angle(w_2w_1, q\eta_1) = \angle(w_2w_1, \eta_2q) = -\gamma$ and $|q\eta_1| = |\eta_2q| = \sigma^\kappa\chi^{-1}$.
- $\angle(v_1v_2, w_1w_2) = -\theta/2$ (see Figure 9c and 9d):
  - If $q = w_1$, do not add auxiliary point.
  - If $q = w_2$, we add the point η such that $\angle(w_2w_1, w_2\eta) = \gamma$ and $|w_2\eta| = \sigma^\kappa\chi^{-1}$.
  - If p and q are in the same hinge set (i.e., p, q are the points ξ_1, ξ_2 in Figure 9c or 9d), we add two points η_1 and η_2 centered on q such that $\angle(w_2w_1, q\eta_1) = \angle(w_2w_1, \eta_2q) = \gamma$ and $|q\eta_1| = |\eta_2q| = \sigma^\kappa\chi^{-1} + \epsilon_0$ where ϵ_0 is much less than the distance between any two points in $\mathcal{P}_m$.[14]
  - Otherwise, we add two points η_1 and η_2 centered on q such that $\angle(w_2w_1, q\eta_1) = \angle(w_2w_1, \eta_2q) = \gamma$ and $|q\eta_1| = |\eta_2q| = \sigma^\kappa\chi^{-1}$.

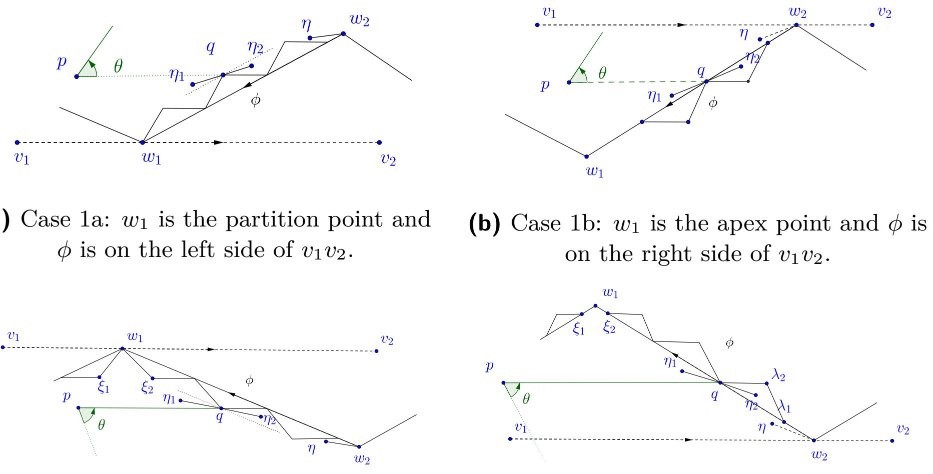
Let *extended hinge set* consist of the normal points in the hinge set and the auxiliary points centered on these normal points. The concept of long range connections can be extended to the extended hinge sets. Then, there is no long range connection in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$.

▶ **Lemma 15.** *There is no long range connection in* $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$.

[12] r may not be q. See the full version for the proof of the existence of r.

[13] Note that the candidate centers belong to G_ϕ, may not belong to ϕ itself.

[14] It is slightly different from the first case. We add two auxiliary points with distance slightly larger than $\sigma^\kappa\chi^{-1}$. The reason is that the cone is half-open half-close in the counterclockwise direction. It will help a lot to unify the proof in the same framework.



(a) Case 1a: w_1 is the partition point and ϕ is on the left side of v_1v_2.

(b) Case 1b: w_1 is the apex point and ϕ is on the right side of v_1v_2.

(c) Case 2a: w_1 is the partition point and ϕ is on the right side of v_1v_2.

(d) Case 2b: w_1 is the apex point and ϕ is on the left side of v_1v_2.

Figure 9 The auxiliary points for each point. Here $\phi = (w_2, w_1)$ and $q \in \mathsf{G}_\phi$. η_1 and η_2 are two auxiliary points centered on q. Note that $|\eta_1 q|$ and $|\eta_2 q|$ are very small in fact.

We defer the detailed proof to the full version. First, we prove that if for any two sibling pairs ϕ and φ in $\mathcal{T}$ at *level-l* for $l \leq m-1$, there is no long range connection between the points in $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$, then there is no long range connection (see Claim 17 in the full version). Next, for any $l \leq m-1$, we examine any two points p and q which belong to $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ respectively, where ϕ and φ are two sibling pairs at *level-l* . We give a necessary condition when there is a edge pq in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ (see Lemma 25 in the full version). We also provide some useful properties about the construction (see Property 24 in the full version). These properties indicate that for any possible long range connection, we can always find a point r (see point r in Figure 8 for an example) to break the connection (see Lemma 26 in the full version).

Finally, we can prove that the shortest paths between μ_1 and μ_2 in the root pair should pass through all extended hinge sets in order, hence, diverges as m approaches infinity.

▶ **Lemma 16.** *The length of the shortest path between μ_1 and μ_2 in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ for $k \geq 3$ is at least ρ^m, for some $\rho = (1-O(d_0^{-1}))\cdot\cos^{-1}(\theta/2)$. Thus, by setting $d_0 > \lceil 6(1-\cos(\theta/2))^{-1}\rceil$, the length diverges as m approaches infinity.*

References

1 Franz Aurenhammer. Voronoi diagrams – survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

2 Luis Barba, Prosenjit Bose, Mirela Damian, Rolf Fagerberg, Wah Loon Keng, Joseph O'Rourke, André van Renssen, Perouz Taslakian, Sander Verdonschot, and Ge Xia. New and improved spanning ratios for Yao graphs. *JoCG*, 6(2):19–53, 2015.

3 Luis Barba, Prosenjit Bose, Jean-Lou De Carufel, André van Renssen, and Sander Verdonschot. On the stretch factor of the Θ_4-graph. In *Workshop on Algorithms and Data Structures*, pages 109–120. Springer, 2013.

4 Matthew Bauer and Mirela Damian. An infinite class of sparse-Yao spanners. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 184–196. SIAM, 2013.

5 Nicolas Bonichon, Cyril Gavoille, Nicolas Hanusse, and David Ilcinkas. Connections between Θ-graphs, Delaunay triangulations, and orthogonal surfaces. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 266–278. Springer, 2010.

6 Prosenjit Bose, Mirela Damian, Karim Douïeb, Joseph O'rourke, Ben Seamone, Michiel Smid, and Stefanie Wuhrer. $\pi/2$-angle Yao graphs are spanners. *International Journal of Computational Geometry & Applications*, 22(01):61–82, 2012.

7 Prosenjit Bose, Anil Maheshwari, Giri Narasimhan, Michiel Smid, and Norbert Zeh. Approximating geometric bottleneck shortest paths. *Computational Geometry*, 29(3):233–249, 2004.

8 Prosenjit Bose, Pat Morin, André van Renssen, and Sander Verdonschot. The Θ_5-graph is a spanner. *Computational Geometry*, 48(2):108–119, 2015.

9 Prosenjit Bose, André van Renssen, and Sander Verdonschot. On the spanning ratio of theta-graphs. In *Workshop on Algorithms and Data Structures*, pages 182–194. Springer, 2013.

10 Paul Chew. There is a planar graph almost as good as the complete graph. In *Proceedings of the second annual symposium on Computational geometry*, pages 169–177. ACM, 1986.

11 Mirela Damian. A simple Yao-Yao-based spanner of bounded degree. *arXiv preprint arXiv:0802.4325*, 2008.

12 Mirela Damian, Nawar Molla, and Val Pinciu. Spanner properties of $\pi/2$-angle Yao graphs. In *Proc. of the 25th European Workshop on Computational Geometry*, pages 21–24. Citeseer, 2009.

13 Mirela Damian and Naresh Nelavalli. Improved bounds on the stretch factor of Y_4. *Computational Geometry*, 62:14–24, 2017.

14 Mirela Damian and Kristin Raudonis. Yao graphs span Θ-graphs. In *Combinatorial Optimization and Applications*, pages 181–194. Springer, 2010.

15 Nawar M El Molla. Yao spanners for wireless ad hoc networks. Master's thesis, Villanova University, 2009.

16 David Eppstein. Spanning trees and spanners. *Handbook of computational geometry*, pages 425–461, 1999.

17 David Eppstein. Beta-skeletons have unbounded dilation. *Computational Geometry*, 23(1):43–52, 2002.

18 BE Flinchbaugh and LK Jones. Strong connectivity in directional nearest-neighbor graphs. *SIAM Journal on Algebraic Discrete Methods*, 2(4):461–463, 1981.

19 K Ruben Gabriel and Robert R Sokal. A new statistical approach to geographic variation analysis. *Systematic Biology*, 18(3):259–278, 1969.

20 Matthias Grünewald, Tamás Lukovszki, Christian Schindelhauer, and Klaus Volbert. Distributed maintenance of resource efficient wireless network topologies. In *European Conference on Parallel Processing*, pages 935–946. Springer, 2002.

21 Lujun Jia, Rajmohan Rajaraman, and Christian Scheideler. On local algorithms for topology control and routing in ad hoc networks. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 220–229. ACM, 2003.

22 Iyad A Kanj and Ge Xia. On certain geometric properties of the Yao-Yao graphs. In *Combinatorial Optimization and Applications*, pages 223–233. Springer, 2012.

23 Jian Li and Wei Zhan. Almost all even Yao-Yao graphs are spanners. *The 24rd Annual European Symposium on Algorithms*, 2016.

24 Xiang-Yang Li. *Wireless Ad Hoc and Sensor Networks: Theory and Applications*. Cambridge, 6 2008.

25 Xiang-Yang Li, Peng-Jun Wan, and Yu Wang. Power efficient and sparse spanner for wireless ad hoc networks. In *Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on*, pages 564–567. IEEE, 2001.

26 Xiang-Yang Li, Peng-Jun Wan, Yu Wang, and Ophir Frieder. Sparse power efficient topology for wireless networks. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, pages 3839–3848. IEEE, 2002.

27 Giri Narasimhan and Michiel Smid. *Geometric spanner networks*. Cambridge University Press, 2007.

28 Jim Ruppert and Raimund Seidel. Approximating the d-dimensional complete euclidean graph. In *Proceedings of the 3rd Canadian Conference on Computational Geometry (CCCG 1991)*, pages 207–210, 1991.

29 Jörg-Rüdiger Sack and Jorge Urrutia. *Handbook of computational geometry*. Elsevier, 1999.

30 Christian Schindelhauer, Klaus Volbert, and Martin Ziegler. Spanners, weak spanners, and power spanners for wireless networks. In *International Symposium on Algorithms and Computation*, pages 805–821. Springer, 2004.

31 Christian Schindelhauer, Klaus Volbert, and Martin Ziegler. Geometric spanners with applications in wireless networks. *Computational Geometry*, 36(3):197–214, 2007.

32 Godfried T Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern recognition*, 12(4):261–268, 1980.

33 Yu Wang, Xiang-Yang Li, and Ophir Frieder. Distributed spanners with bounded degree for wireless ad hoc networks. *International Journal of Foundations of Computer Science*, 14(02):183–200, 2003.

34 Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, 11(4):721–736, 1982.