几何集合覆盖和 t-跨度图问题研究

（申请清华大学工学博士学位论文）

培 养 单 位：交 叉 信 息 院

学　　科：计 算 机 科 学 与 技 术

研 究 生：金 逸 飞

指 导 教 师：李 建 副 教 授

二〇一八年六月

Geometric Set Cover and t-Spanner

Dissertation Submitted to

Tsinghua University

in partial fulfillment of the requirement

for the degree of

Doctor of Philosophy

in

Computer Science and Technology

by

Yifei Jin

Dissertation Supervisor : Jian Li

June, 2018

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：

清华大学拥有在著作权法规定范围内学位论文的使用权，其中包括：（1）已获学位的研究生必须按学校规定提交学位论文，学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（2）为教学和科研目的，学校可以将公开的学位论文作为资料在图书馆、资料室等场所供校内师生阅读，或在校园网上供校内师生浏览部分内容；（3）根据《中华人民共和国学位条例暂行实施办法》，向国家图书馆报送可以公开的学位论文。

本人保证遵守上述规定。

（保密的论文在解密后应遵守此规定）

作者签名：_____　　导师签名：_____

日　　期：_____　　日　　期：_____

摘 要

在本论文中，我们研究了几何集合覆盖和 t-跨度图问题。在平面中给定一个带权的单位圆集合和一个点集。带权最小单位圆集覆盖问题要求得到可以覆盖给定点集中所有点的最小的带权单位圆集合。带权最小单位圆集覆盖问题是几何覆盖问题中的一个，这类问题已经被研究了二十余年。对于一般图的覆盖问题，想得到一个 $(1-\epsilon)\ln(n)$ 的解是 NP-hard 问题。但在几何背景下往往能得到更好的近似结果。根据几何图形的不同（如三角形，正方形，圆，半平面等）和几何图形带不带权重，几何覆盖问题可以衍生出很多变种。而其中绝大部分变种均为 NP-hard 的，不同的问题往往需要引入不同的策略来解决。人们已经证明了不带权（即权均为 1）的最小圆集覆盖问题是 NP-hard 的，并且给出了一个多项式时间的近似方案（PTAS）[1]。然而对于带权最小圆集覆盖问题，目前还只有一些常数近似，还不知道它是不是存在一个 PTAS。我们的工作给出了第一个 PTAS 算法。再结合一些已有的结果，我们的工作还对最长连续时间覆盖问题（maximum lifetime coverage）给出了第一个 PTAS 算法，并且提高了带权最小圆支配集问题的近似比。

我们还研究了几何 t-跨度图问题。t-跨度图是定义在一个空间点集上的图。该图保持了任意两点间在图上的最短距离是这两点间欧几里德距离的至多 t 倍的性质。几何 t-跨度图在通讯网络中有重要的应用。研究什么样的图具有 t-跨度属性是计算几何中的一个经典问题。Yao-Yao 图因其构造简单，并且每个点的度数有限，而成为一个很好的几何 t-跨度图候选体。关于 Yao-Yao 图（YY_k 图）是否具有 t-跨度属性，是一个长期困扰人们的开放问题。Bauer 和 Damian[2] 证明了所有 $\mathrm{YY}_{6k}(k >= 6)$ 都是 t-跨度图。Li 和 Zhan[3] 推广了他们的结论，证明了所有的 $\mathrm{YY}_{2k}(k >= 42)$ 都是 t-跨度图。但是这些已知的技术都不能拓展到奇数 Yao-Yao 图上。我们的工作第一次给出了结论——对于任意 $k >= 1$, 存在点集 $\mathcal{P}$，使得定点集上的奇数 Yao-Yao 图（YY_{2k+1}）不是 t-跨度图。

关键词：集合覆盖；t-跨度图；单位圆图；Yao-Yao 图

Abstract

In this dissertation, we study the geometric set cover and t-spanner problems. First, we give the first polynomial-time approximation scheme (PTAS) for the weighted unit disk set cover problem. We are given a set of weighted unit disks and a set of points in the Euclidean plane. The minimum weight unit disk cover (WUDC) problem asks for a subset of disks of minimum total weight that covers all given points. It is known that the unweighted unit disk cover problem is NP-hard and admits a polynomial-time approximation scheme (PTAS). For the weighted unit disk cover problem, several constant factor approximation algorithms have been developed. However, whether the problem admits a PTAS has been an open question. In this dissertation, we answer this question affirmatively by presenting the first PTAS for WUDC. Our result implies the first PTAS for the minimum weight dominating set problem in unit disk graphs. Combining with existing ideas, our result can also be used to obtain the first PTAS for the maximum lifetime coverage problem and an improved 4.475-approximation for the connected dominating set problem in unit disk graphs.

Second, we study the Yao-Yao graphs (also known as sparse-Yao graphs) and prove that odd Yao-Yao graphs are not spanners. It is a long standing open problem whether Yao-Yao graphs YY_k are all spanners[4]. Bauer and Damian[2] showed that all YY_{6k} for $k \geq 6$ are spanners. Li and Zhan[3] 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 dissertation, we show that, surprisingly, for any integer $k \geq 1$, there exist odd Yao-Yao graph YY_{2k+1} instances, which are not spanners.

Key Words: Set Cover; t-Spanner; Unit Disk Graph; Yao-Yao Graph; PTAS

Contents

Chapter 1 Introduction

In this dissertation, we study two important computational geometric problems. First, we study the weighted unit disk set cover problem (WUDC). Whether there exists a PTAS for WUDC or not is an open question mentioned in a number of previous papers[5-10]. We settle this question affirmatively by presenting the first PTAS. Second, we study the Yao-Yao graphs. We prove that for any integer $k \geq 1$, there exist odd Yao-Yao graph YY_{2k+1} instances which are not spanners, which is also a long standing open problem[4].

1.1 Weighted Unit Disk Set Cover

The set cover problem is a central problem in theoretical computer science and combinatorial optimization. The input of the set cover problem consists of a ground set U and collection $\mathcal{S}$ of subsets of U. Each set $S \in \mathcal{S}$ has a non-negative weight w_S. The goal is to find a subcollection $C \subseteq \mathcal{S}$ of minimum total weight such that $\bigcup C$ covers all elements of U. The approximability of the general SC problem is rather well understood: it is well known that the greedy algorithm gives an H_n-approximation ($H_n = \sum_{i=1}^{n} 1/i$) and obtaining a $(1-\epsilon)\ln n$-approximation for any constant $\epsilon > 0$ is NP-hard[11-12]. In the *geometric set cover problem*, U is a set of points in some Euclidean space $\mathbb{R}^d$, and $\mathcal{S}$ consists of geometric objects (e.g., disks, squares, triangles). In the geometric setting, we can hope for better-than-logarithmic approximations due to the special structure of $\mathcal{S}$. Most geometric set cover problems are still NP-hard, even for the very simple classes of objects such as unit disks[13-14] (see[15-16] for more examples and exceptions). Approximation algorithms for geometric set cover have been studied extensively for the past two decades, not only because of the importance of the problem per se, but also its rich connections to other important notions and problems, such as VC-dimension[17-19], ϵ-net, union complexity[20-22], planar separators[23-24], even machine scheduling problems[25].

In this work, we study the geometric set cover problem with one of the simplest class of objects − unit disks. The formal definition of our problem is as follows:

Definition 1.1 (Weighted Unit Disk Cover (WUDC)): Given a set $\mathcal{D} = \{\mathsf{D}_1, \ldots, \mathsf{D}_n\}$ of n unit disks and a set $\mathcal{P} = \{P_1, \ldots, P_m\}$ of m points in Euclidean plane $\mathbb{R}^2$, each disk D_i

has a weight $w(\mathsf{D}_i)$. The goal of WUDC is to choose a subset of disks covering all points in $\mathcal{P}$ with the minimum total weight.

WUDC generalizes the following minimum weight dominating set problem in unit disk graphs (UDG).

Definition 1.2 (Minimum Weight Dominating Set (MWDS) in UDG): Given a unit disk graph $G(V, E)$, where V is a set of weighted points in $\mathbb{R}^2$ and $(u, v) \in E$ iff $\|u - v\| \leq 1$ for any $u, v \in V$, a dominating set S is a subset of V such that for any node $v \notin S$, there is some $u \in S$ with $(u, v) \in E$. The goal in the minimum weight dominating set problem is to find a dominating set with the minimum total weight.

To see that WUDC is a generalization of MWDS, consider the following reduction. Given a dominating set instance with point set V, we create a WUDC instance by placing, for each point in $v \in V$, a point (to be covered) co-located with v and a unit disk centered at v, with weight equal to the weight of v. In this dissertation, we only state our algorithms and results in the context of WUDC.

1.1.1 Previous Results and Our Contribution

We first recall that a polynomial time approximation scheme (PTAS) for a minimization problem is an algorithm $\mathcal{A}$ that takes an input instance, a constant $\epsilon > 0$, returns a solution of value SOL such that $\mathsf{SOL} \leq (1 + \epsilon)\mathsf{OPT}$, where OPT is the optimal value, and the running time of $\mathcal{A}$ is polynomial in the size of the input for any fixed constant ϵ.

WUDC is NP-hard, even when the weights are equivalent (i.e., $w(\mathsf{D}_i) = 1$)[14]. For unweighted dominating set in unit disk graphs, Hunt et al.[1] obtained the first PTAS in unit disk graphs. For the more general disk graphs, based on the connection between geometric set cover problem and ϵ-nets, developed in[17-19], and the existence of ϵ-net of size $O(1/\epsilon)$ for halfspaces in $\mathbb{R}^3$[26] (see also[27]), it is possible to achieve a constant factor approximation. As estimated in[23], these constants are at best 20 (A recent result[28] shows that the constant is at most 13). Moreover, there exists a PTAS for unweighted disk cover and minimum dominating set via the local search technique[23-24].

For the general weighted WUDC problem, the story is longer. Ambühl et al.[5] obtained the first approximation for WUDC with a concrete constant 72. Applying the shifting technique of[29], Huang et al.[30] obtained a $(6 + \epsilon)$-approximation algorithm for WUDC. The approximation factor was later improved to $(5 + \epsilon)$[31], and to $(4 + \epsilon)$ by

several groups[6,32-33]. Willson et al improve the ratio to 3.63. ① Very recently, Zhang et al.[34] give a $(3+\epsilon)$-approximation algorithm. The *quasi-uniform sampling method*[21-22] provides another approach to achieve a constant factor approximation for WUDC (even in disk graphs). However, the constant depends on several other constants from rounding LPs and the size of *the union complexity*. Very recently, based on the separator framework of Adamaszek and Wiese[35], Mustafa et al.[36] obtained a QPTAS (Quasi-polynomial time approximation scheme) for weighted disks in $\mathbb{R}^2$ (in fact, weighted halfspaces in $\mathbb{R}^3$), thus ruling out the APX-hardness of WUDC.

Another closely related work is by Erlebach and van Leeuwen[7], who obtained a PTAS for set cover on weighted unit squares, which is the first PTAS for weighted geometric set cover on any planar objects (except those poly-time solvable cases[15-16]). Although it may seem that their result is quite close to a PTAS for weighted WUDC, as admitted in their paper, their technique is insufficient for handling unit disks and "a completely different insight is required".

In light of all the aforementioned results, it seems that we should expect a PTAS for WUDC, but it remains to be an open question (explicitly mentioned as an open problem in a number of previous papers, e.g.,[5-10]). Our main contribution in this dissertation is to settle this question affirmatively by presenting the first PTAS for WUDC.

Theorem 1.1: There is a polynomial time approximation scheme for the WUDC problem. The running time is $n^{O(1/\epsilon^9)}$.

Because WUDC is more general than MWDS, we immediately have the following corollary.

Corollary 1.1: There is a polynomial time approximation scheme for the minimum weight dominating set problem in unit disk graphs.

We note that the running time $n^{\text{poly}(1/\epsilon)}$ is nearly optimal in light of the negative result by Marx[37], who showed that an EPTAS (i.e., Efficient PTAS, with running time $f(1/\epsilon)\text{poly}(n)$) even for the unweighted dominating set in UDG would contradict the exponential time hypothesis.

Finally, in Section 2.8, we show that our PTAS for WUDC can be used to obtain improved approximation algorithms for two important problems in wireless sensor networks, the connected dominating set problem and the maximum lifetime coverage problem in UDG.

① The algorithm can be found in Du and Wan[9], who attributed the result to a manuscript by Willson et al.

1.2 Odd Yao-Yao Graphs are Not Spanners

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.,[4,38-41]). In this dissertation, we study the so called *geometric t-spanner*, formally defined as follows (see e.g.,[42]).

Definition 1.1: (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.

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[43]. See the comprehensive survey by Eppstein[44] 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[45] and Narasimhan and Smid[46] for more details.

Yao graphs are one of the first approximations of complete Euclidean graphs, introduced independently by Flinchbaugh and Jones[47] and Yao[41].

Definition 1.2 (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 point $u \in \mathcal{P}$, $\mathsf{Y}_k(\mathcal{P})$ 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[48] 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.[49] proved that Y_4 is a 663-spanner. Damian and Nelavalli[50] improved this to 54.6 recently. Barba et al.[51] showed

that Y_5 is a 3.74-spanner. Damian and Raudonis[52] proved that the Y_6 graph is a 17.64 spanner. Li et al.[4,53] first proved that all Y_k, $k > 6$ are spanners with stretch factor at most $1/(1 - 2\sin(\pi/k))$. Later Bose et al.[49,54] also obtained the same result independently. Recently, Barba et al.[51] 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.[4] introduced *Yao-Yao* graphs (or *Sparse-Yao* graphs in the literature). A Yao-Yao graph $YY_k(\mathcal{P})$ is obtained by removing some edges from $Y_k(\mathcal{P})$ as follows:

Definition 1.3 (Yao-Yao Graph YY_k): (1) Construct the directed Yao graph, as in Definition A.7. (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 $YY_k(\mathcal{P})$ 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.[4] 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[2-4,45]. It is also listed as Problem 70 in the Open Problems Project.①

Conjecture 1.1 (see[2]): 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[48]. Damian and Molla[48,55] proved that YY_4, YY_6 may not be spanners. Bauer et al.[51] proved that YY_5 may not be spanners. On the positive side, Bauer and Damian[2] showed that for any integer $k \geq 6$, 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[3] 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 A.1. However, we show in this dissertation that, surprisingly, Conjecture A.1 is false for odd Yao-Yao graphs.

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

Theorem 1.4: 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.①

Related Work It has been proven that in some special cases, Yao-Yao graphs are spanners[56-59]. 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[56-57].

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[48], for $k = 4$[60], 5[61], 6[62], ≥ 7[63-64], 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[65-67]. In power t-spanners, the Euclidean distance $|\cdot|$ is replaced by $|\cdot|^\kappa$ with a constant $\kappa \geq 2$. Schindelhauer et al.[66-67] 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[67].

Our counterexample is inspired by the concept of fractals. Fractals have been used to construct examples for β-skeleton graphs with unbounded stretch factors[68]. 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.[67] 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.

Remark 1.1: The work “A PTAS for Weighted Unit Disk Cover" has been published in ICALP 2015 and “Odd Yao-Yao graphs are Not Spanners" has been published in SoCG

① Here, m is a parameter in our recursive construction. We will explain it in detail in Section A. Roughly speaking, m is the level of recursion and the number of points in $\mathcal{P}_m$ increases with m.

2018.

Chapter 2 A PTAS for Weighted Unit Disk Cover

2.1 Our Approach - A High Level Overview

By the standard shifting technique[69], it suffices to provide a PTAS for WUDC when all disks lies in a square of constant size (we call it a block, and the constant depends on $1/\epsilon$). This idea is formalized in Huang et al.[30], as follows.

Lemma 2.1 (Huang et al.[30]): Suppose there exists a ρ-approximation for WUDC in a fixed $L \times L$ block, with running time $f(n, L)$. Then there exists a $(\rho + O(1/L))$-approximation with running time $O(L \cdot n \cdot f(n, L))$ for WUDC. In particular, setting $L = 1/\epsilon$, there exists a $(\rho + \epsilon)$-approximation for WUDC, with running time $O\left(\frac{1}{\epsilon} \cdot n \cdot f(n, \frac{1}{\epsilon})\right)$.

In fact, almost all previous constant factor approximation algorithms for WUDC were obtained by developing constant approximations for a single block of a constant size (which is the main difficulty ①). The main contribution of the paper is to improve on the previous work[5-6,30-31] for a single block, as in the following lemma.

Lemma 2.2: There exists a PTAS for WUDC in a fixed block of size $L \times L$ for $L = 1/\epsilon$. The running time of the PTAS is $n^{O(1/\epsilon^9)}$

From now on, the approximation error guarantee $\epsilon > 0$ is a fixed constant. Whenever we say a quantity is a constant, the constant may depend on ϵ. We use OPT to represent the optimal solution (and the optimal value) in this block. We use capital letters $A, B, C, \ldots$ to denote points, and small letters $a, b, c, \ldots$ to denote arcs. For two points A and B, we use AB to denote the line segment connecting A and B and use $|AB|$ to denote its length. We use D_i to denote a disk and D_i to denote its center. For a point A and a real $r > 0$, let $\mathsf{D}(A, r)$ be the disk centered at A with radius r. For a disk D_i, we use $\partial\mathsf{D}_i$ to denote its boundary. We call a segment of $\partial\mathsf{D}_i$ *an arc*.

First, we guess whether OPT contains more than C disks or not for some constant C. If OPT contains no more than C disks, we enumerate all possible combinations and choose the one which covers all points and has the minimum weight. This takes $O\left(\sum_{i=1}^{C} \binom{n}{i}\right) = O(n^C)$ time, which is polynomial.

① For the unweighted dominating set problem in a single block, it is easy to see that the optimal number of disks is bounded by a constant, which implies that we can compute the optimum in poly-time. However, for the weighted dominating set problem or WUDC the optimal solution in a single block may consist of $\Theta(n)$ disks.

The more challenging case is when OPT contains more than C disks. In this case, we guess (i.e., enumerate all possibilities) the set $\mathcal{G}$ of the C most expensive disks in OPT. There are at most a polynomial number (i.e., $O(n^C)$) possible guesses. Suppose our guess is correct. Then, we delete all disks in $\mathcal{G}$ and all points that are covered by $\mathcal{G}$. Let D_t (with weight w_t) be the cheapest disk in $\mathcal{G}$. We can see that $\mathsf{OPT} \geq Cw_t$. Moreover, we can also safely ignore all disks with weight larger than w_t (assuming that our guess is correct). Now, our task is to cover the remaining points with the remaining disks, each having weight at most w_t. We use $\mathcal{D}' = \mathcal{D} \setminus \mathcal{G}$ and $\mathcal{P}' = \mathcal{P} \setminus \mathcal{P}(\mathcal{G})$ to denote the set of the remaining disks and the set of remaining points respectively, where $\mathcal{P}(\mathcal{G})$ denote the set of points covered by at least one disk in $\mathcal{G}$.

Next, we carefully choose to include in our solution a set $\mathcal{H} \subseteq \mathcal{D}'$ of at most ϵC disks. The purpose of $\mathcal{H}$ is to break the whole instance into many (still a constant) small pieces (substructures), such that each substructure can be solved optimally, via dynamic programming. [①] One difficulty is that the substructures are not independent and may interact with each other (i.e., a disk may appear in more than one substructure). Each substructure has a direction (in the clockwise or counterclockwise) and all disks in the substructure have a partial order based on the direction. In order to apply the dynamic programming technique to all substructures simultaneously, we have to ensure the orders of the disks in different substructures are consistent with each other. Choosing $\mathcal{H}$ to ensure a globally consistent order of disks is in fact the main technical challenge of the paper.

Suppose we have a set $\mathcal{H}$ which suits our need (i.e., the remaining instance $(\mathcal{D}' \setminus \mathcal{H}, \mathcal{P}' \setminus \mathcal{P}(\mathcal{H}))$ can be solved optimally in polynomial time by dynamic programming). Let $\mathcal{S}$ be the optimal solution of the remaining instance. Our final solution is $\mathsf{SOL} = \mathcal{G} \cup \mathcal{H} \cup \mathcal{S}$. First, we can see that

$$w(\mathcal{S}) \leq w(\mathsf{OPT} - \mathcal{G} - \mathcal{H}) \leq \mathsf{OPT} - w(\mathcal{G}),$$

since $\mathsf{OPT} - \mathcal{G} - \mathcal{H}$ is a feasible solution for the instance $(\mathcal{D}' \setminus \mathcal{H}, \mathcal{P}' \setminus \mathcal{P}(\mathcal{H}))$. Hence, we have that

$$\mathsf{SOL} = w(\mathcal{G}) + w(\mathcal{H}) + w(\mathcal{S}) \leq \mathsf{OPT} + \epsilon C w_t \leq (1 + \epsilon)\mathsf{OPT},$$

where the 2nd to last inequality holds because $|\mathcal{H}| \leq \epsilon C$, and the last inequality uses the

① An individual substructure can be solved using a dynamic program similar to [5,16].

fact that $\mathsf{OPT} \geq w(\mathcal{G}) \geq Cw_t$.

Constructing $\mathcal{H}$: Now, we provide a high level sketch for how to construct $\mathcal{H} \subseteq \mathcal{D}'$. First, we partition the block into *small squares* with side length $\mu = O(\epsilon)$ such that any disk centered in a square can cover the whole square and the disks in the same square are close enough. Let the set of small squares be $\Xi = \{\Gamma_{ij}\}_{1 \leq i,j \leq K}$ where $K = L/\mu$. For a small square Γ, let $\mathsf{D}_{s_\Gamma} \in \Gamma$ and $\mathsf{D}_{t_\Gamma} \in \Gamma$ be the furthest pair of disks (i.e., $|D_{s_\Gamma} D_{t_\Gamma}|$ is maximized). We include the pair D_{s_Γ} and D_{t_Γ} in $\mathcal{H}$, for every small square $\Gamma \in \Xi$, and call the union of the two disks *the square gadget* for Γ. See Figure A.1 for an example. We only need to focus on covering the remaining points in the *uncovered region* $\mathbb{U}(\mathcal{H})$.

We consider all disks with centers in a small square Γ. The portion of those disks in the uncovered region defines two disjoint connected regions (See right hand side of Figure A.1, the two shaded regions). We call such a region, together with all relevant arcs, a *substructure* (formal definition in Section A). In fact, we can solve the disk covering problem for a single substructure optimally using dynamic programming (which is similar to the dynamic program in[5,16]). It appears that we are almost done, since ("intuitively") all square gadgets have already covered much area of the entire block, and we should be able to use similar dynamic program to handle all such substructures as well. However, the situation is more complicated (than we initially expected) since the arcs are dependent. See Figure A.2 for a "not-so-complicated" example. Firstly, there may exist two arcs (called *sibling arcs*) which belong to the same disk when the disk is centered in the *core-central area*, as shown in Figure A.1). The dynamic program has to make decisions for two sibling arcs, which belong to two different substructures (called R(emotely)-correlated substructures), together. Second, in order to carry out a dynamic program, we need a suitable order of all arcs. To ensure such an order exists, we need all substructures to interact with each other "nicely".

In particular, besides all square gadgets, we need to add into $\mathcal{H}$ a constant number of extra disks. This is done by a series of "cut" operations. A cut can either break a cycle, or break one substructure into two substructures. To capture how substructures interact, we define an auxiliary graph, called substructure relation graph $\mathfrak{S}$, in which each substructure is a node. The aforementioned R-correlations define a set of blue edges, and geometrically overlapping relations define a set of red edges. Through the cut operations, we can make blue edges form a matching, and red edges also form a matching, and $\mathfrak{S}$ acyclic (we call $\mathfrak{S}$ an acyclic 2-matching). The special structure of $\mathfrak{S}$ allows us to define an ordering of all

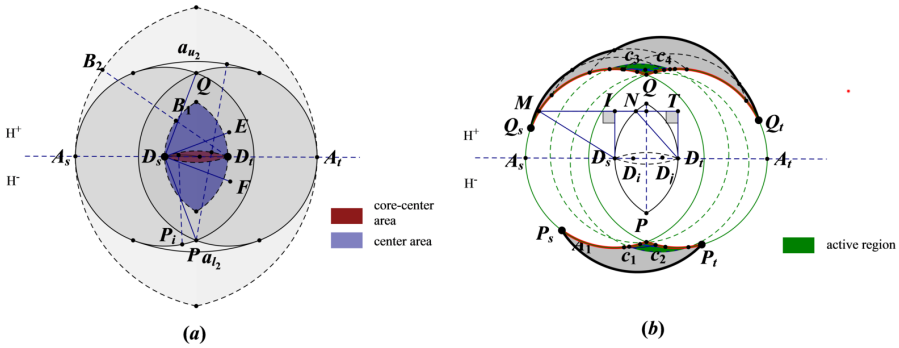


Figure 2.1 A square gadget. D_s and D_t are the furthest pair of disks in square Γ whose centers are D_s and D_t. On the left hand side, the blue region is the central area $\mathfrak{C} = \mathsf{D}(D_s, r_{st}) \cap \mathsf{D}(D_t, r_{st})$, where $r_{st} = |D_sD_t|$. The brown region is the core-central area $\mathfrak{C}_o = \mathsf{D}(P, 1) \cap \mathsf{D}(Q, 1)$. On the right hand side, the green area is the active regions, defined as $\left(\bigcup_{i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^+$ and $\left(\bigcup_{i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^-$.

arcs easily. Together with some other simple properties, we can generalize the dynamic program from one substructure to all substructures simultaneously.

2.2 Square Gadgets

We discuss the structure of a square gadget $\mathsf{Gg}(\Gamma)$ associated with the small square Γ. Recall that the square gadget $\mathsf{Gg}(\Gamma) = \mathsf{D}_s \cup \mathsf{D}_t$, where D_s and D_t are the furthest pair of disks in Γ. We can see that for any disk D_i with center in Γ, there are either one or two arcs of $\partial\mathsf{D}_i$ which are not covered by $\mathsf{Gg}(\Gamma)$. Without loss of generality, assume that D_sD_t is horizontal. The line D_sD_t divides the whole plane into two half-planes which are denoted by H^+ (the upper half-plane) and H^- (the lower half-plane). $\partial\mathsf{D}_s$ and $\partial\mathsf{D}_t$ intersect at two points P and Q. We need a few definitions which are useful throughout the paper. Figure A.1 shows an example of a square gadget.

1. (Central Area and Core-Central Area) Define the *central area* of $\mathsf{Gg}(\Gamma)$ as the intersection of the two disks $\mathsf{D}(D_s, r_{st})$ and $\mathsf{D}(D_t, r_{st})$ in the square Γ, where $r_{st} = |D_sD_t|$. We use $\mathfrak{C}(\Gamma)$ to denote it. Since D_s and D_t are the furthest pair, we can see that every other disk with center in Γ is centered in the central area $\mathfrak{C}(\Gamma)$.

   We define the *core-central area* of $\mathsf{Gg}(\Gamma)$ as the intersection of two unit disks centered at P, Q respectively. Essentially, any unit disk centered in the core-central area has four intersections with the boundary of the square gadget. Let us denote the area by $\mathfrak{C}_o(\Gamma)$.

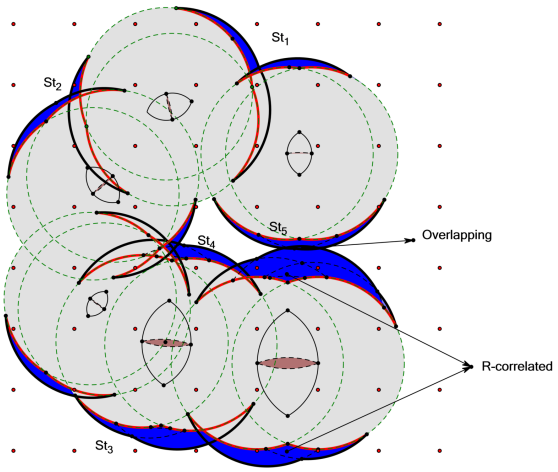


Figure 2.2 The general picture of the substructures in a block. The red points are the grid points of small squares. Dash green disks are what we have selected in $\mathcal{H}$. There are five substructures in the block.

2. (Active Region) Consider the regions

$$\left(\bigcup_{D_i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^+ \quad \text{and} \quad \left(\bigcup_{D_i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^-.$$

We call each of them an *active region* associated with square Γ. We use $\mathsf{Ar}(\Gamma)$ to denote an active region. Note that an active region is covered by the union of disks centered in the core-central area.

2.3 Substructures

Initially, $\mathcal{H}$ consists of the disks belonging to the square gadgets. In Section 2.6, we will include in $\mathcal{H}$ a constant number of extra disks. For a set S of disks, we use $\mathbb{R}(S)$ to denote the region covered by disks in S (i.e., $\cup_{\mathsf{D}_i \in S} \mathsf{D}_i$). Assuming a fixed $\mathcal{H}$, we now describe the basic structure of the uncovered region $\mathbb{R}(\mathcal{D}') - \mathbb{R}(\mathcal{H})$. [1] For ease of notation, we use $\mathbb{U}(\mathcal{H})$ to denote the uncovered region $\mathbb{R}(\mathcal{D}') - \mathbb{R}(\mathcal{H})$. Figure A.2 shows an example. Intuitively, the region consists of several "strips" along the boundary of $\mathcal{H}$. Now, we define some notions to describe the structure of those strips.

1. (Arcs) Consider a disk $\mathsf{D} \in \mathcal{D}'$ and suppose the center of D is in the square Γ. Let $\mathsf{D}_s\mathsf{D}_t$ be the square gadget $\mathsf{Gg}(\Gamma)$, and without loss of generality assume the line D_sD_t is horizontal and divides the plane into two halfplanes H^+ and H^-. D may contribute at most two *uncovered arcs*, one in H^+ and one in H^-. Let us first focus

[1] Recall that $\mathcal{D}' = \mathcal{D} \setminus \mathcal{G}$ where $\mathcal{G}$ is the set of C most expensive disks in OPT.

on H^+. In general, $\partial\mathsf{D}$ intersects $\partial\mathcal{H}$ at several points[①] in H^+. The intersections are located on $\partial\mathsf{D}$ in order in the clockwise (or counterclockwise) direction.[②] The uncovered arc is the segment of $\partial\mathsf{D}$ starting from the first intersection point and ending at the last intersection point.[③] We define the uncovered arc for H^- analogously (if $|\partial\mathsf{D} \cap \partial\mathcal{H}| \neq 0$). Figure 2.3 illustrates why we need so many words to define an arc. Essentially, some portions of an arc may be covered by some other disks in $\mathcal{H}$, and the arc is broken into several pieces. Our definition says that those pieces should be treated as a whole. In this dissertation, when we mention an *arc*, we mean an entire uncovered arc (w.r.t. the current $\mathcal{H}$). Note that both endpoints of an arc lie on the boundary of $\mathbb{R}(\mathcal{H})$.

2. (Subarcs) For an arc a, we use $a[A, B]$ to denote the closed subarc of arc a from point A to point B. Similarly, we only write $a(A, B)$ to denote the corresponding open subarc (with endpoints A and B excluded).
3. (Central Angle) Suppose arc a is part of $\partial\mathsf{D}$ for some disk D with center D. The central angle of a, denoted as $\angle(a)$ is the angle whose apex (vertex) is D and both legs (sides) are the radii connecting D and the endpoints of a. We can show that $\angle(a) < \pi$ for any arc a (See Lemma 2.19 in Appendix 2.9.1)
4. (Baseline) We use $\partial\mathcal{H}$ to denote boundary of $\mathbb{R}(\mathcal{H})$. Consider an arc a whose endpoints P_1, P_2 are on $\partial\mathcal{H}$. We say that the arc a covers a point $P \in \partial\mathcal{H}$, if P lies in the boundary between P_1 and P_2 of $\mathcal{H}$. We say a point $P \in \partial\mathcal{H}$ *can be covered* if some arc covers P. A baseline is a consecutive maximal segment of $\partial\mathcal{H}$ that can be covered. We usually use b to denote a baseline.
5. (Substructure) A substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ consists of a baseline b and the collection $\mathcal{A}$ of arcs which can cover some point in b. The two endpoints of each arc $a \in \mathcal{A}$ are on b and $\angle(a)$ is less than π. Note that every point of b is covered by some arc in $\mathcal{A}$. Figure A.3 illustrates the components of a substructure.

   Occasionally, we need a slightly generalized notion of substructure. For a set $\mathcal{A}$ of uncovered arcs, if they cover a consecutive segment of the boundary of $\mathcal{H}$, $\mathcal{A}$ also induces a substructure denoted as $\mathsf{St}[\mathcal{A}]$.

Arc Order: Now we switch our attention to the order of the arcs in a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$.

① The number must be even.

② Whether the direction is clockwise or counterclockwise does not affect the definition of arc.

③ Note that an uncovered arc may not entirely lie in the uncovered region $\mathbb{U}(\mathcal{H})$ (some portion may be covered by some disks in $\mathcal{H}$).

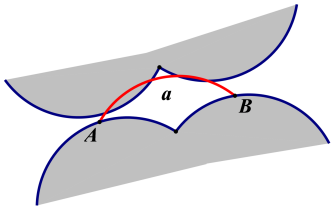


Figure 2.3 The figure gives an example of an arc. The blue curves are part of the boundary of $\mathcal{H}$. The red curve is an uncovered arc.

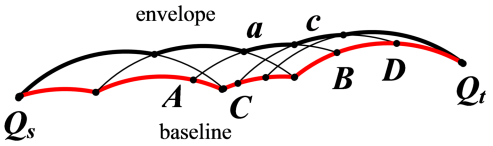


Figure 2.4 A substructure. The baseline b consists of the red arcs which are the part of consecutive boundary of $\partial\mathcal{H}$. Q_s, Q_t are the endpoints of b. The black curves are uncovered arcs. The bold black arcs form the envelope. The arc $a \prec c$ because $A \prec C$ and $B \prec D$.

Suppose the baseline b starts at point Q_s and ends up at point Q_t. Consider any two points P_1 and P_2 on the baseline b. If P_1 is closer to Q_s than P_2 along the baseline b, we say that P_1 *appears earlier* than P_2 (denoted as $P_1 \prec P_2$). Consider any two arcs a and c in $\mathcal{A}$. The endpoints of arc a are A and B, and the endpoints of arc c are C and D. All of points A, B, C, D are on the baseline b. Without loss of generality, we assume that $A \prec B$, $C \prec D$ and $A \prec C$. If $B \prec D$, we say arc a *appears earlier* than arc c (denoted as $a \prec c$). Otherwise, we say a and c are incomparable. See Figure A.3 for an example. It is easy to see that $\prec$ defines a partial order.

Adjacency: Consider two arcs a (with endpoints $A \prec B$) and c (with endpoints $C \prec D$). If $a \prec c$ and $C \prec B$, we say that a and c are *adjacent* (we can see that they must intersect exactly once), and c is the *adjacent successor* of a. Similarly, we can define the adjacent successor of subarc $a[P_1, P_2]$. If c is the adjacent successor of a, meanwhile c intersects with subarc $a[P_1, P_2]$, we say that c is the *adjacent successor* of subarc $a[P_1, P_2]$. Among all adjacent successors of $a[P_1, P_2]$, we call the one whose intersection with $a[P_1, P_2]$ is closest to P_1 the *first adjacent successor* of $a[P_1, P_2]$.

In order to carry out the dynamic program in Section A, we need to properly orient each substructure so that the (partial) order of the arcs is consistent. Our final solution in each substructure can be represented as a path (which is a segment of the boundary of the union of chosen disks). Our dynamic program essentially needs to determine such a path.

To be precise, we provide a formal definition of a *valid path*, as follows.

Definition 2.1 (A Valid Path): Consider a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$. Suppose the baseline b is oriented from Q_s to Q_t. A valid path is a path from Q_s to Q_t which consists of a sequence of subarcs $\{a_1[Q_s, Q_1],\ a_2[Q_1, Q_2],\ \ldots,\ a_k[Q_{k-1}, Q_t]\}$ (baseline segments are considered as subarcs as well). For any a_i, a_{i+1} is its adjacent successor (so $a_i \prec a_{i+1}$). Q_i is the intersection point of the arcs a_i and a_{i+1}.

Note that the baseline from Q_s to Q_t is a trivial valid path (we do not consider any coverage requirement yet). Among all the valid paths in a substructure, there is one that is maximal in terms of the coverage ability, which we call *the envelope* of the substructure.

Definition 2.2 (Envelope of a Substructure): Consider a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$. Suppose the baseline b is oriented from Q_s to Q_t. The envelope of St is the valid path $\{a_1[Q_s, Q_1]$, $a_2[Q_1, Q_2], \ldots, a_k[Q_{k-1}, Q_t]\}$ where a_{i+1} is the first adjacent successor of a_i for all $i \in [k]$.

Coverage: Consider a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$. Consider an arc a with endpoints A and B on baseline b. We use $\mathsf{b}[A, B]$ to denote the segment of b that is covered by a. We say that the region surrounded by the arc a and $\mathsf{b}[A, B]$ is *covered* by arc a and use $\mathbb{R}(a)$ to denote the region. We note that the covered region $\mathbb{R}(a)$ is with respect to the current $\mathcal{H}$. Similarly, consider a valid path Path. The region covered by Path is $\cup_{a \in \mathsf{Path}} \mathbb{R}(a)$ (the union is over all arcs in Path) and is denoted by $\mathbb{R}(\mathsf{Path})$. Finally, we define the region covered by the substructure St, denoted by $\mathbb{R}(\mathsf{St})$, to be the region covered by the envelope of St.

2.4 Simplifying the Problem

The substructures may overlap in a variety of ways. As we mentioned in Section 2.1, we need to include in $\mathcal{H}$ more disks in order to make the substructures amenable to the dynamic programming technique. However, this step is somewhat involved and we decide to postpone it to the end of the paper (Section 2.6). Instead, we present in this section what the organization of the substructures and what properties we need *after* including more disks in $\mathcal{H}$ for the final dynamic program.

Self-Intersections: In a substructure St, suppose there are two arcs a and c in $\mathcal{A}$ with endpoints A, B and C, D respectively. If $A \prec B \prec C \prec D$ and a and c cover at least one and the same point in $\mathcal{P}$, we say the substructure is *self-intersecting*. In other words, there exists

at least one point covered by two non-adjacent arcs in a self-intersecting substructure. See Figure 2.9 for an example. Self-intersections are troublesome obstacles for the dynamic programming approach. So we will eliminate all self-intersections in Section 2.6. In the rest of the section, we assume all substructures are *non-self-intersecting* and discuss their properties.

Lemma 2.3 (Single Intersection Property): For any two arcs in a non-self-intersecting substructure St, they have at most one intersection in $\mathbb{R}(\mathsf{St})$.

Proof We prove by contradiction. Suppose a_i and a_j belong to the same substructure. a_i and a_j intersect at point A and B. Since the substructure is non-self-intersecting, a_i and a_j lie on the same halfplane divided by line AB. Since the two radii of a_i and a_j are equal, the sum of central angles of a_i and a_j equals 2π. Thus, at least one central angle of a_i and a_j is no less than π, rendering a contradiction to the fact that the central angle (defined in Section A) of any arc is less than π. □

Based on the single intersection property, we can easily get the following property.

Lemma 2.4: Consider two arcs a and b in a non-self-intersecting substructure. If a and b intersect in the substructure and there is a point in the substructure being covered by two arcs a and b, then a is adjacent to b.

Order Consistency: There are two types of relations between substructures which affect how the orientations should be done. One is the *overlapping relation* and the other is *remote correlation.* See Figure A.2 for some examples.

First, we discuss the case when two substructures overlap. For an arc a, we use $\mathsf{D}(a)$ to denote the disk associated with a. For a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$, we let $\mathsf{D}(\mathsf{b})$ be the set of disks that contributes an arc to the baseline b.

Definition 2.3 (Overlapping Relation): Consider two substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$ and the point set $\mathcal{P}$. We say that St_1 and St_2 overlap when there is a point in $\mathcal{P}$ that is in $\mathbb{R}(\mathsf{St}_1) \cap \mathbb{R}(\mathsf{St}_2)$.

Our dynamic program requires the overlapping substructures satisfying the overlapping order consistency defined as follows.

Definition 2.4 (Overlapping Order Consistency): We say the overlapping order consistency holds for two overlapping substructures if their orientations are different (i.e., if one is clockwise, the other should be counterclockwise).

The other type of relation is remote correlation. As we alluded in Section 2.1, the two substructures which contain different related active regions of the same gadget interact with each other.

Definition 2.5 (Remote correlation): Consider two substructures St_u and St_l which are not overlapping. Suppose they contain two different active regions of the same gadget respectively (recalling that one gadget may have two different active regions, one in H^+, one in H^-). We say that the two substructures are *remotely correlated* or *R-correlated*. See Figure A.2.

There are two possible baseline orientations for each substructure (clockwise or anticlockwise around the center of the arc), which gives rise to four possible ways to orient both St_u and St_l. However, there are only two (out of four) of them that are consistent (thus we can do dynamic programming on them). More formally, we need the following definition:

Definition 2.6 (Remote Order Consistency): We say that remote order consistency holds for two substructures $\mathsf{St}_u(\mathsf{b}_u, \mathcal{A}_u)$ and $\mathsf{St}_l(\mathsf{b}_l, \mathcal{A}_l)$ if there is an orientation for each substructure, such that it can not happen that $a_i \prec b_i$ in substructure St_u but $a_j \succ b_j$ in St_l, where $a_i, a_j \in \partial\mathsf{D}_1, b_i, b_j \in \partial\mathsf{D}_2$ and $a_i, b_i \in \mathcal{A}_u, a_j, b_j \in \mathcal{A}_l$.

We show in the following simple lemma that the local order consistency can be easily achieved for the two substructures containing different active regions of the same square.

Lemma 2.5 (Remote Order Consistency): Consider two substructures St_u and St_l which are R-correlated. Each of them contains only one active region. Then the remote order consistency holds for the two substructures St_u and St_l.

Proof We consider two substructures St_u and St_l. The arcs not in the active regions have no influence on the order consistency since each of them only appears in one substructure. So, we only need to consider the order of the arcs in two active regions. We use the same notations as those on the RHS of Figure A.1. We orient the upper baseline from Q_s to Q_t, and the lower baseline from P_s to P_t. Suppose two arcs c_1, c_3 belong to disk D_u, and two arcs c_2, c_4 belong to the disk D_v. Assume $c_4 \prec c_3$ in substructure St_u and $c_1 \prec c_2$ in substructure St_l. There must exist another intersection point on each side of the line connecting the two intersections of $(c_1, c_2), (c_3, c_4)$. This contradicts the fact that two unit disks have at most two intersections. □

As different substructures may interact with each other, we need a dynamic program which can run over all substructures simultaneously. Hence, we need to define a globally consistent ordering of all arcs.

Definition 2.7 (Global Order Consistency): We have global order consistency if there is a way to orient each substructure, such that (1) for any pair of overlapping substructures, the local order consistency holds, and (2) for any pair of remote-correlated substructures, the remote order consistency holds.

Substructure Relation Graph $\mathfrak{S}$: we construct an auxiliary graph $\mathfrak{S}$, called the *substructure relation graph*, to capture all R-correlations and overlapping relations. Each node in $\mathfrak{S}$ represents a substructure. If two substructures are R-correlated, we add a blue edge between the two substructures. If two substructures overlap, we add a red edge.

Consider a red edge between $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$. If baseline b_1 is oriented clockwise (around the center of any of its arc), then b_2 should be oriented counterclockwise, and vise versa. The blue edge represents the same orientation relation, i.e., if $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$ are R-correlated, b_1 and b_2 should be oriented differently.

It is unclear how to orient all baselines if $\mathfrak{S}$ is an arbitrary graph. So we need to ensure that $\mathfrak{S}$ has a nice structure.

Definition 2.8 (Acyclic 2-Matching): We say the substructure relation graph $\mathfrak{S}$ is an acyclic 2-matching, if $\mathfrak{S}$ is acyclic and is composed by a blue matching and a red matching. In other words, $\mathfrak{S}$ only contains paths, and the red edges and blue edges appear alternately in each path.

If $\mathfrak{S}$ is a acyclic 2-matching, we can easily assign each substructure an orientation that achieve global order consistency .

Point-Order Consistency: Similarly to the arc order consistency, we also need to define the *point-order consistency*, which is also crucial for our dynamic program.

Definition 2.9 (Point Order Consistency): Suppose a set $\mathcal{P}_{\text{co}}$ of points is covered by both of two overlapping substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$. Consider any two points $P_1, P_2 \in \mathcal{P}_{\text{co}}$ and four arcs $a_1, a_2 \in \mathcal{A}_1, b_1, b_2 \in \mathcal{A}_2$. Suppose $P_1 \in \mathbb{R}(a_1) \cap \mathbb{R}(b_1)$ and $P_2 \in \mathbb{R}(a_2) \cap \mathbb{R}(b_2)$. But $P_1 \notin \mathbb{R}(a_2) \cup \mathbb{R}(b_2)$ and $P_2 \notin \mathbb{R}(a_1) \cup \mathbb{R}(b_1)$. We say P_1 and P_2 are point-order consistent if $a_1 \prec a_2$ in St_1 and $b_1 \prec b_2$ in St_2 (or if $a_1 \succ a_2$ in St_1 and $b_1 \succ b_2$ in St_2). We say the points in $\mathcal{P}_{\text{co}}$ satisfy point order consistency if all pairs of points in $\mathcal{P}_{\text{co}}$ are point-order consistent.

After introducing all relevant concepts, we can finally state the set of properties we need for the dynamic program.

Lemma 2.6: After choosing $\mathcal{H}$, we can ensure the following properties holds:

P1. (Active Region Uniqueness) Each substructure contains at most one active region.

P2. (Non-self-intersection) Every substructure is non-self-intersecting.

P3. (Acyclic 2-Matching) The substructure relation graph $\mathfrak{S}$ is an acyclic 2-matching, i.e., $\mathfrak{S}$ consists of only paths. In each path, red edges and blue edges appear alternately.

P4. (Point Order Consistency) Any point is covered by at most two substructures. The points satisfy the point order consistency.

How to ensure all these properties will be discussed in detail in Section 2.6. Now, everything is in place to describe the dynamic program.

2.5 Dynamic Programming

Suppose we have already constructed the set $\mathcal{H}$ such that Lemma A.3 holds (along with an orientation for each substructure). Without loss of generality, we can assume that the remaining disks can cover all remaining points (otherwise, either the original instance is infeasible or our guess is false). In fact, our dynamic program is inspired by, and somewhat similar to those in[5-6,16].

DP for Two Overlapping Substructures: For ease of description, we first handle the case where there are only two overlapping substructures. We will extend the DP to the general case shortly. Suppose the two substructures are $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$, b_1 is oriented from P_s to P_t and b_2 is oriented from Q_s to Q_t. See Figure 2.5 for an example.

A state of the dynamic program is a pair $\Phi = (P, Q)$ where P is an intersection point of two arcs in substructure St_1 and Q is an intersection point of two arcs in substructure St_2. Fix the state $\Phi = (P, Q)$ and consider St_1. Let b_P and t_P be the two arcs intersecting at P. Suppose $b_P \prec t_P$ with endpoints $(A, B), (C, D)$ respectively. We call arc b_P the *base-arc* and t_P the *top-arc* for point P. ① Our DP maintains that the base-arc is already paid in the subproblem.

① If P is the tail endpoint of an arc (so P is on the baseline), P only has a base-arc (no top-arc), which is the baseline arc it lies on.

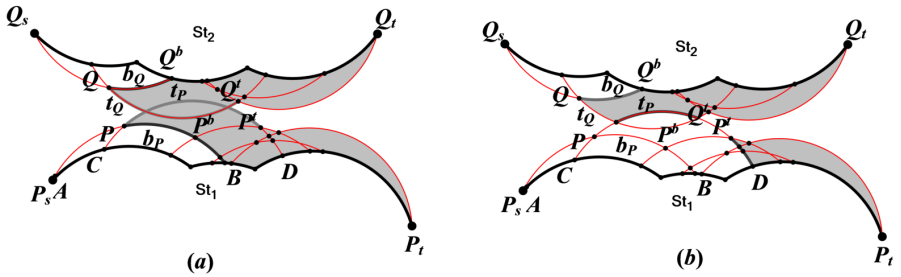


Figure 2.5 The figure explains the dynamic program of two overlapping substructures. The left figure shows the subproblem $\mathsf{OPT}(P, Q)$. The goal of $\mathsf{OPT}(P, Q)$ is to find minimum valid paths for PP_t and QQ_t respectively in set $\mathcal{A}_1[P] \cup \mathcal{A}_2[Q]$ such that the paths cover all points of $\mathcal{P}[P, Q]$. The right figure illustrates one of its four smaller subproblems $\mathsf{OPT}(P^t, Q)$.

Given state $\Phi = (P, Q)$, now we describe the subproblem associated with the state.[①] Intuitively, a feasible solution to the subproblem restricted to St_1 (resp. St_2) is a valid path starting from point P (resp. Q) and terminating at P_t (resp. Q_t). More specifically, we construct a substructure $\mathsf{St}_1^{[P]}(\mathsf{b}_1[P], \mathcal{A}_1[P])$:

- $\mathsf{b}_1[P]$ is the concatenation of subarc $b_P[P, B]$ and the original baseline segment $\mathsf{b}_1[B, P_t]$. All arcs in $\mathsf{b}_1[P]$ have cost zero.
- $\mathcal{A}_1[P]$ consists of all arcs $a' \in \mathcal{A}_1$ such that $b_P \prec a'$ (of course, with the portion covered by $\mathsf{b}_1[P]$ subtracted). The cost of each such arc is the same as its original cost.

Similarly, we consider St_2 and the intersection point Q, and construct $\mathsf{St}_2[Q]$ with baseline $\mathsf{b}_2[Q]$ and arc set $\mathcal{A}_2[Q]$. We use $\mathcal{P}(a)$ (or $\mathcal{P}(\mathcal{A})$) to denote the points covered by a (or $\mathcal{A}$) (w.r.t. the original baseline). Let the point set $\mathcal{P}_{P,Q}$ that we need to cover in the subproblem $\Phi(P, Q)$ be

$$\mathcal{P}_{P,Q} = \mathcal{P}(\mathcal{A}_1[P]) \cup \mathcal{P}(\mathcal{A}_2[Q]) - \mathcal{P}(b_P) - \mathcal{P}(b_Q). \tag{2-1}$$

We note that the minus term $-\mathcal{P}(b_P) - \mathcal{P}(b_Q)$ is not vacuous as b_P (resp. b_Q) may cover some points in $\mathcal{A}_2[Q]$ (resp. $\mathcal{A}_1[P]$), and it is important that we do not have to cover those point (this subtlety is crucial in the correctness proof of the DP). The goal for the subproblem $\Phi(P, Q)$ is to find two valid paths with minimum total weight, one from P to P_t and one from Q to Q_t, such that they together cover all points in $\mathcal{P}_{P,Q}$. Note that the weights of both base-arcs b_P and b_Q should be included in any feasible solution as well.

① Note that each state corresponds to exactly one subproblem. Hence, we do not distinguish "state" and "subproblem" in the following and use $\Phi = (P, Q)$ to represent the subproblem too.

Suppose $b_P(P, B]$ intersects its first successor at P^b (called *base-adjacent point*) and $t_P(P, D]$ intersects its first successor at P^t (called *top-adjacent point*). Similarly, we define Q^b, Q^t in St_2 in exactly the same way.

Now, computing the optimum for subproblem $\Phi(P, Q)$ reduces to computing the optima for four smaller subproblems $\mathsf{OPT}(P^b, Q)$, $\mathsf{OPT}(P^t, Q)$, $\mathsf{OPT}(P, Q^b)$ and $\mathsf{OPT}(P, Q^t)$. We define two Boolean variables I_P (reps. I_Q) indicating whether we can move from $\mathcal{P}_{P^b,Q}$ to $\mathcal{P}_{P,Q}$ without choosing a new arc. Formally, if $\mathcal{P}_{P,Q} = \mathcal{P}_{P^b,Q}$, $I_P = 0$. Otherwise, $I_P = 1$. Similarly, if $\mathcal{P}_{P,Q} = \mathcal{P}_{P,Q^b}$, $I_Q = 0$. If not, $I_Q = 1$. The dynamic programming recursion is:

$$\mathsf{OPT}(P, Q) = \min \begin{cases} \mathsf{OPT}(P, Q^b) + I_Q \cdot \infty, & \text{add no new arc ;} \\ \mathsf{OPT}(P, Q^t) + w[b_Q], & \text{add base-arc } b_Q; \\ \mathsf{OPT}(P^b, Q) + I_P \cdot \infty, & \text{add no new arc ;} \\ \mathsf{OPT}(P^t, Q) + w[b_P], & \text{add base-arc } b_P. \end{cases} \tag{2-2}$$

The optimal value we return is $\mathsf{OPT}(P_s, Q_s)$. Now, we prove the correctness of the DP in the following theorem. We note that both the point-order consistency and the subtlety mentioned above play important roles in the proof.

Theorem 2.1: Suppose that we have two overlapping substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$. Further suppose that b_1 and b_2 are oriented in a way such that the point-order consistency holds. Then, the cost of the optimal solution is equal to $\mathsf{OPT}(P_s, Q_s)$ (which is computed by (2-2)).

Proof Consider subproblem $\mathsf{OPT}(P, Q)$. As we know the optimal solution of $\mathsf{OPT}(P, Q)$ should be two valid paths. One is from P to P_t and the other is from Q to Q_t. Suppose they are $\mathsf{Path}_1 = \{a_1[P, P_1], a_2[P_1, P_2], \ldots, a_i[P_{i-1}, P_i], \ldots, a_k[P_{k-1}, P_t]\}$ and $\mathsf{Path}_2 = \{b_1[Q, Q_1], b_2[Q_1, Q_2], \ldots, b_i[Q_{i-1}, Q_i], \ldots, b_l[Q_{l-1}, Q_t]\}$. We can see that it suffices to prove that at least one of the two statements is true.

- The pair of paths $(\mathsf{Path}_1 - \{a_1[P, P_1)\}, \mathsf{Path}_2)$ is an optimal solution to subproblem $\mathsf{OPT}(P_1, Q)$.
- The pair of paths $(\mathsf{Path}_1, \mathsf{Path}_2 - \{b_1[Q, Q_1)\})$ is an optimal solution to subproblem $\mathsf{OPT}(P, Q_1)$. □

We prove by contradiction. Assume that both of the above statements are false. Suppose b_P and b_Q are the base-arcs for state $\Phi(P, Q)$, i.e., b_P intersects with a_1 at point P and b_Q intersects with b_1 at point Q. We use $\mathcal{P}(\mathsf{Path})$ to denote the point set which is covered

by path Path. Recall that $\mathcal{P}[P,Q] = \mathcal{P}(\mathcal{A}_1[P]) \cup \mathcal{P}(\mathcal{A}_2[Q]) - \mathcal{P}(b_P) - \mathcal{P}(b_Q)$. Since $\mathsf{Path}_1 \cup \mathsf{Path}_2$ is the optimal solution for $\mathsf{OPT}(P,Q)$ (hence feasible), we have that $\mathcal{P}[P,Q] = \mathcal{P}(\mathsf{Path}_1) \cup \mathcal{P}(\mathsf{Path}_2) - \mathcal{P}(b_P) - \mathcal{P}(b_Q)$. Then, the pair of paths $(\mathsf{Path}_1 - \{a_1[P,P_1)\}, \mathsf{Path}_2)$ is the optimal solution for the subproblem in which we need to pick one path from P_1 to P_t and one from Q to Q_t to cover the points in

$$\mathcal{P}(\mathsf{Path}_1) \cup \mathcal{P}(\mathsf{Path}_2) - \mathcal{P}(b_P) - \mathcal{P}(b_Q) - \mathcal{P}(a_1) = \mathcal{P}[P,Q] - \mathcal{P}(a_1).$$

If not, we can get a contradiction by replacing $\mathsf{Path}_1 - a_1[P,P_1]$ and Path_2 with the optimal solution of the above subproblem, resulting in a solution with less weight than $\mathsf{OPT}(P,Q)$ for $\Phi(P,Q)$. Since the first statement is false, we must have that

$$\mathcal{P}[P,Q] - \mathcal{P}(a_1) \neq \mathcal{P}[P_1,Q]$$

(otherwise $(\mathsf{Path}_1 - \{a_1[P,P_1)\}, \mathsf{Path}_2)$ is optimal for $\Phi(P_1,Q)$). We note that the LHS $\subseteq$ RHS. Plugging the definition (2-1), we have that

$$\mathcal{P}(\mathcal{A}_1[P]) \cup \mathcal{P}(\mathcal{A}_2[Q]) - \mathcal{P}(b_P) - \mathcal{P}(b_Q) - \mathcal{P}(a_1) \neq \mathcal{P}(\mathcal{A}_1[P_1]) \cup \mathcal{P}(\mathcal{A}_2[Q]) - \mathcal{P}(a_1) - \mathcal{P}(b_Q).$$

A careful (elementwise) examination of the above inequality shows that it is only possible if

$$\mathcal{P}(b_P) \cap \big(\mathcal{P}(\mathcal{A}_2[Q]) - \mathcal{P}(b_Q)\big) \neq \emptyset.$$

Repeating same argument, we can see that if the second statement is false, we have that

$$\mathcal{P}(b_Q) \cap (\mathcal{P}(\mathcal{A}_1[P]) - \mathcal{P}(b_P)) \neq \emptyset.$$

Hence, there exist $b_i \in \mathcal{A}_2[Q]$ and $a_j \in \mathcal{A}_1[P]$ such that $\mathcal{P}(b_P) \cap \mathcal{P}(b_i) \neq \emptyset$, and $\mathcal{P}(b_Q) \cap \mathcal{P}(a_j) \neq \emptyset$. However, this contradicts the point-order consistency because of $b_P \prec a_j$ and $b_Q \prec b_i$.

Thus, one of the two statements is true. W.l.o.g, suppose $(\mathsf{Path}_1 - \{a_1[P,P_1]\}, \mathsf{Path}_2)$ is the optimal solution to subproblem $\mathsf{OPT}(P_1,Q)$. Suppose P^t, Q^t is the top-adjacent point of P,Q. If P_1 is the top-adjacent point of P (i.e., $P_1 = P^t$), through $\mathsf{OPT}(P,Q) = \mathsf{OPT}(P_1,Q) + w[b_P]$ in DP(2-2), we can get the optimal solution of subproblem $\mathsf{OPT}(P,Q)$. If not, i.e. $P_1 \neq P^t$, P_1 and P^t are on the same arc b_P. According to (2-2), we have $\mathsf{OPT}(P^t,Q^b) = \mathsf{OPT}(P_1,Q^b)$. Then, $\mathsf{OPT}(P,Q) = \mathsf{OPT}(P^t,Q) + w[b_P] = \mathsf{OPT}(P_1,Q) + w[b_P]$. Thus, we can prove that $(\mathsf{Path}_1, \mathsf{Path}_2)$ is the optimal solution to at least one of

the subproblems $\mathsf{OPT}(P, Q^t)$ or $\mathsf{OPT}(P^t, Q)$. Thus, we can get the optimal solution for $\mathsf{OPT}(P, Q)$ by our DP.

DP for the general problem: Now, we handle all substructures together. Our goal is find a valid path for each substructure such that we minimize the total weight of all the paths. We can see that we only need to handle each path in the substructure relation graph $\mathfrak{S}$ separately (since different paths have no interaction at all). Hence, from now on, we simply assume that $\mathfrak{S}$ is a path.

Suppose the substructures are $\{\mathsf{St}_k(\mathsf{b}_k, \mathcal{A}_k)\}_{k\in[m]}$. We use A_k and B_k to denote two endpoints of b_k. Generalizing the previous section, a state for the general DP is $\Phi = \{P_k\}_{k\in[m]}$, where P_k is an intersection point in substructure St_k. We use b_{P_k}, t_{P_k}, P_k^b, P_k^t to denote the base-arc, top-arc, base-adjacent point, top-adjacent point (w.r.t. P_k) respectively. For each $k \in [m]$, we also define $\mathsf{St}_k^{[P_k]}(\mathsf{b}_k[P_k], \mathcal{A}_k[P_k])$ in exactly the same way as in the previous section. The point set we need to cover in the subproblem is:

$$\mathcal{P}\left[\{P_k\}_{k\in[m]}\right] = \bigcup_{k\in[m]} \mathcal{P}(\mathcal{A}_k[P_k]) - \bigcup_{k\in[m]} \mathcal{P}(b_{P_k}).$$

The subproblem $\mathsf{OPT}(\{P_k\}_{k\in[m]})$ is to find, for each substructure St_k, a valid path from P_k to B_k, such that all points in $\mathcal{P}[\{P_k\}_{k\in[m]}]$ can be covered and the total cost is minimized.

The additional challenge for the general case is caused by R-correlations. If two arcs (in two different substructures) belong to the same disk, we say that they are *siblings* of each other. If we processed each substructure independently, some disks would be counted twice. In order to avoid double-counting, we should consider both siblings together, i.e., select them together and pay for the disk only once in the DP.

In order to implement the above idea, we need a few more notations. We construct an auxiliary bipartite graph $\mathfrak{B}$. The nodes on one side are all disks in $\mathcal{D}' \setminus \mathcal{H}$, and the nodes on the other side are substructures. If disk D_i has an arc in the substructure St_j, we add an edge between D_i and St_j. Besides, for each arc of the baselines, we add a node to represent it and add an edge between the node and the substructure which contains the arc. Because the weight of any arc of any of the baselines is zero, it shall not induce any contradiction to regard them as independent arcs. In fact, there is a 1-1 mapping between the edges in $\mathfrak{B}$ and all arcs. See Figure 2.6 for an example.

Fix a state $\Phi = \{P_k\}_{k\in[m]}$. For any arc a in St_k (with intersection point P_k and base-arc b_{Pk}), a has three possible positions:

1. $a \prec b_{Pk}$: we label its corresponding edge with "unprocessed";

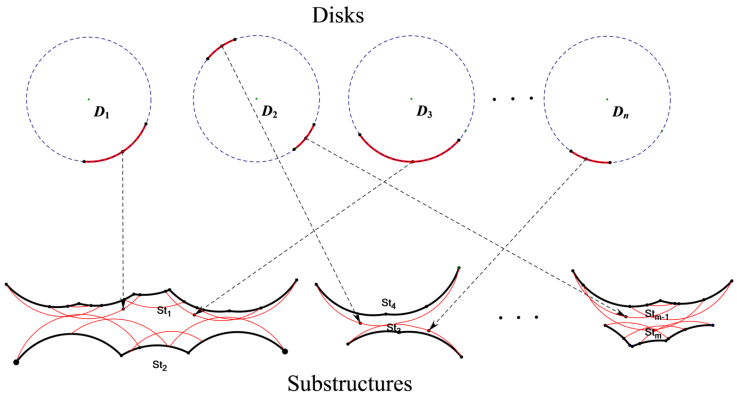


Figure 2.6 The bipartite graph which is used for marking the *ready* disks. The nodes on upper side represent the disks. The nodes on the lower side represent the substructures. If D_i has an arc in St_j, we add an arc between them.

2. $a = b_{Pk}$: we label its corresponding edge with "processing";
3. Others: we label its corresponding edge with "done".

As mentioned before, we need to avoid the situation where one arc becomes the base-arc first (i.e., being added in solution and paid once), and its sibling becomes the base-arc in a later step (hence being paid twice). With the above labeling, we can see that all we need to do is to avoid the states in which one arc is "processing" and its sibling is "unprocessed". If disk D is incident on at least one "processing" edge and not incident on any "unprocessed" edge, we say the D is *ready*. Let $\mathcal{R}$ be the set of ready disks. For each ready disk D, we use $N_{\mathrm{p}}(\mathsf{D})$ to denote the set of neighbors (i.e., substructures) of D connected by "processing" edges. We should consider all substructures in $N_{\mathrm{p}}(\mathsf{D})$ together.

Again, we need in our DP indicator variables to tell us whether a certain transition is feasible: Formally, if $\mathcal{P}[\{P_k\}_{k\in[m]}] = \mathcal{P}[[P_k][P_i^b]_{\{i\}}]$, let $I_i = 0$. Otherwise, let $I_i = 1$. Here, for ease of notation, for a set $\{e_k\}_{k\in[m]}$ and $S \subseteq [m]$, we write $[e_k][e'_i]_S = \{e_k\}_{k\in[m]\setminus S} \cup \{e'_i\}_{i\in S}$. Hence,

$$[P_k][P_i^b]_{\{i\}} = \{P_k\}_{k\in[m]\setminus i} \cup P_i^b \text{ and}$$

$$[P_k][P_i^t]_{N_{\mathrm{p}}(\mathsf{D})} = \{P_k\}_{k\in[m]\setminus N_{\mathrm{p}}(\mathsf{D})} \cup \{P_i^t\}_{i\in N_{\mathrm{p}}(\mathsf{D})}$$

Then we have the dynamic program as follows:

$$\mathsf{OPT}\left(\{P_k\}_{k\in[m]}\right) = \min\begin{cases} \min_{i\in[m]}\left\{\mathsf{OPT}\left([P_k][P_i^b]_{\{i\}}\right) + I_i\cdot\infty\right\}, & \text{add no disk} \\ \min_{\mathsf{D}\in\mathcal{R}}\left\{\mathsf{OPT}\left([P_k][P_i^t]_{N_{\mathrm{p}}(\mathsf{D})}\right) + w_{\mathsf{D}}\right\}, & \text{add disk } \mathsf{D} \end{cases} \tag{2-3}$$

Note that in the second line, the arc(s) in $N_{\mathrm{p}}(\mathsf{D})$ are base-arcs (w.r.t. state $\Phi(\{P_k\}_{k\in[m]})$.

In the rest of the section, we prove the correctness of the dynamic program. If we use the solution of a smaller subproblem $\mathsf{OPT}(\Phi')$ to compute subproblem $\mathsf{OPT}(\Phi)$, we say Φ can be reached from Φ' (denoted as $\Phi' \to \Phi$). If Φ can be reached from initial state $\Phi_0(\{P_k \mid P_k = B_k\}_{k\in[m]})$, we say the state is reachable, which is denoted by $\Phi_0 \to \Phi$.

We start with a simple consequence of our DP: there is no double-counting.

Lemma 2.7: If two arcs in the solution belong to the same disk, their weights are counted only once in the DP (A-1).

Proof From the DP, we can see that the weight of an arc is counted only when it becomes a base-arc (its label changes from "unprocessed" to "processing"). If the two sibling arcs a, b (belonging to disk D) in the solution are counted twice, there exist two states Φ_1 and Φ_2 such that (1)$\Phi_1 \to \Phi_2$, (2)a is a base-arc in Φ_1, but b is not a base-arc in Φ_1, (3) b is a base-arc in Φ_2. So, in Φ_1 or any state before that, arc b is "unprocessed". However, a can become a base-arc only when D is ready, rendering a contradiction. □

Now, we prove the correctness of the dynamic program. The proof is a generalization of Theorem 2.1.

Theorem 2.2: Suppose $\mathfrak{S}$ is a path. All baselines are oriented such that all properties in Lemma A.3 hold. Then, the optimal cost for the problem equals to $\mathsf{OPT}(\{A_k\}_{k\in[m]})$ (computed by our DP (A-1)).

Proof Suppose the set $\{\mathsf{Path}_k\}_{k\in[m]}$ of paths is the optimal solution. We need to prove (1) the final state $\Phi(\{A_k\}_{k\in[m]})$ is reachable, (2) for any reachable state $\Phi = \{P_k\}_{k\in[m]}$, $\mathsf{OPT}(\{P_k\}_{k\in[m]})$ is the optimal solution for the corresponding subproblem (that is to find one valid path from P_k to B_k for each substructure $\mathsf{St}_k^{[P_k]}$ to cover all point in $\mathcal{P}[\{P_k\}_{k\in[m]}]$, such that the total cost is minimized).

We first prove the first statement. Suppose the state Φ is reachable, we prove it can reach another state if Φ is not the final state (i.e., we do not get stuck at Φ). Let us prove it by contradiction. Assume we get stuck at state Φ. That means there is no ready disk in Φ. Note that each substructure, say St, is incident on exactly one "processing" arc, say arc a (which is the base-arc in St). a's sibling, say b (in St'), must be labeled "unprocessed" (otherwise the disk would be ready). Consider the base-arc (or "processing" arc), say a',

in St'. So we have $a' \prec b$ in St'. Again the sibling b' of a' must be an "unprocessed" arc in St. [①] So we have $b' \prec a$ in St, which contradicts the global arc-order consistency.

Now, we prove the second statement. We consider state $\Phi = \{P_k\}_{k\in[m]}$. Suppose the optimal solution for subproblem $\Phi\{P_k\}_{k\in[m]}$ is the set of paths $\mathsf{PS} = \{\mathsf{Path}_k\}_{k\in[m]}$ where $\mathsf{Path}_k = (a_{k_1}, a_{k_2} \ldots, a_{k_n})$. Consider the states $\Phi_{N_{\mathrm{p}}(\mathsf{D})} := [P_k][P_i^t]_{N_{\mathrm{p}}(\mathsf{D})}$ for all $\mathsf{D} \in \mathcal{R}$. Obviously, we can see from our DP that $\Phi_{N_{\mathrm{p}}(\mathsf{D})} \to \Phi$. Define for each $\mathsf{D} \in \mathcal{R}$, a set of paths

$$\mathsf{PS}_{N_{\mathrm{p}}(\mathsf{D})} = \{\mathsf{Path}_k\}_{k\in[m]-N_{\mathrm{p}}(\mathsf{D})} \cup \{\mathsf{Path}_i - \{a_{i_1}\}\}_{i\in N_{\mathrm{p}}(\mathsf{D})}.$$

It suffices to prove that there exists at least one $\mathsf{D} \in \mathcal{R}$ such that $\mathsf{PS}_{N_{\mathrm{p}}(\mathsf{D})}$ is the optimal solution for $\mathsf{OPT}(\Phi_{N_{\mathrm{p}}(\mathsf{D})})$.

Consider a substructure St_i. Suppose the intersection point in St_i of Φ is P_i and the base-arc at point P_i is b_{Pi}. For each $i \in [m]$, let $\mathsf{St}_{\pi(i)}$ be the only substructure (if any) overlapping with St_i. So $b_{P\pi(i)}$ is the base-arc of $\mathsf{St}_{\pi(i)}$. Using exactly the same exchange argument in Theorem 2.1, we can show that if $\mathsf{PS}_{N_{\mathrm{p}}(\mathsf{D})}$ is not the optimal solution for $\mathsf{OPT}(\Phi_{N_{\mathrm{p}}(\mathsf{D})})$, there exists some $i \in N_{\mathrm{p}}(\mathsf{D})$ such that $\mathcal{E}_i$ happens, where $\mathcal{E}_i$ is the following event: there exists an arc $\beta_{\pi(i)}$ in $\mathsf{St}_{\pi(i)}$ with $\beta_{\pi(i)} \succ b_{P\pi(i)}$ such that

$$\mathcal{P}(b_{Pi}) \cap \mathcal{P}(\beta_{\pi(i)}) \neq \emptyset.$$

We use $\mathcal{E}_i$ to denote the above event. Thus if there is no $\mathsf{D} \in \mathcal{R}$ such that $\mathsf{PS}_{N_{\mathrm{p}}(\mathsf{D})}$ is the optimal solution for $\mathsf{OPT}(\Phi_{N_{\mathrm{p}}(\mathsf{D})})$, we have

$$\bigwedge_{\mathsf{D}\in\mathcal{R}} \left(\bigvee_{i\in N_{\mathrm{p}}(\mathsf{D})} \mathcal{E}_i \right) = \text{True}. \tag{2-4}$$

Converting the conjunctive normal form (CNF) to the disjunctive normal form (DNF), we get

$$\bigvee_{(k_1,\ldots,k_{|\mathcal{R}|})\in\Pi_{\mathsf{D}\in\mathcal{R}} N_{\mathrm{p}}(\mathsf{D})} \left(\bigwedge_{i\in\mathcal{R}} \mathcal{E}_{k_i} \right) = \text{True}, \qquad \square$$

where $\Pi_{\mathsf{D}\in\mathcal{R}} N_{\mathrm{p}}(\mathsf{D})$ means the Cartesian product of all $N_{\mathrm{p}}(\mathsf{D})$ in $\mathcal{R}$. We call each $\bigwedge_{i\in|\mathcal{R}|} \mathcal{E}_{k_i}$ a clause (note that k_i indexes a substructure). If we can prove that every clause is false, then obviously, (2-4) is false, resulting in a contradiction.

① To see that b' is in St, note that $\mathfrak{S}$ is a path and St is only R-correlated with St' (and vice versa).

Now, we show that every clause is false. First, we consider the case that both end nodes of $\mathfrak{S}$ are incident to red edges. W.l.o.g., suppose the two nodes of $\mathfrak{S}$ are St_1 and St_2. Thus, they are not R-correlated with other substructures. We know if one substructure St_i is not R-correlated with others, every clause must contain the corresponding event $\mathcal{E}_i$ (since the disk corresponding to its base-arc must be ready and in $\mathcal{R}$). Hence, every clause contains $\mathcal{E}_1$ and $\mathcal{E}_2$. Moreover, for each pair $(\mathsf{St}_i, \mathsf{St}_j)$ of R-correlated substructures, each clause should contain either $\mathcal{E}_i$ or $\mathcal{E}_j$. Suppose the length of the path $\mathfrak{S}$ is ℓ. Because red and blue edges alternates, there are $\frac{\ell-1}{2}$ R-correlated substructure pairs and $\frac{\ell+1}{2}$ overlapping substructure pairs. We should select $\frac{\ell-1}{2}+2$ terms in each clause. Because $\frac{\ell-1}{2}+2 > \frac{\ell+1}{2}$, there exists a pair of overlapping substructures $(\mathsf{St}_i, \mathsf{St}_{\pi(i)})$ such that both $\mathcal{E}_i$ and $\mathcal{E}_{\pi(i)}$ appear in the clause. To make the clause true, we must have

$$\mathcal{E}_i = (\mathcal{P}(b_{Pi}) \cap \beta_{\pi(i)} \neq \emptyset) = \text{True} \quad \text{and} \quad \mathcal{E}_{\pi(i)} = (\mathcal{P}(b_{P\pi(i)}) \cap \beta_i \neq \emptyset) = \text{True},$$

where $\beta_{\pi(i)} \succ b_{P\pi(i)}$ and $\beta_i \succ b_{Pi}$. It yields a contradiction to the point-order consistency.

Next, we consider the remaining case where at least one end of the path $\mathfrak{S}$ is a blue edge, meaning the substructure on the end does not overlap with any other substructure. W.l.o.g., suppose the end node is St_1 and it is R-correlated with St_2. The event $\mathcal{E}_1 = \mathcal{P}(b_{P1}) \cap \mathcal{P}(\beta_{\pi(1)}) \neq \emptyset)$ is always false since $\beta_{\pi(1)}$ does not exist. So all clause containing $\mathcal{E}_1$ is false. To make each of the remaining clauses true, $\mathcal{E}_2$ must be true, and the case reduces to the previous case (by simply omitting node St_1). So the same argument again renders a contradiction. This completes the proof of the theorem.

2.6 Constructing $\mathcal{H}$

In this section, we describe how to construct the set $\mathcal{H}$ in details. We first include in $\mathcal{H}$ all square gadgets. Hence, the boundary of $\mathcal{H}$ consists of several closed curves, as shown in Figure A.2. $\mathcal{H}$ and all uncovered arcs define a set of substructures.

First, we note that there may exist a closed curve that all points on the curve are covered by some arcs (or informally, we have a cyclic substructure, with the baseline being a cycle). We need to break all such baseline cycles by including a constant number of extra arcs into $\mathcal{H}$. This is easy after we introduce the label-cut operation in Section 2.6.1, and we will spell out all details then. Note that we cannot choose some arbitrary arcs on the envelope of the cycle since it may ruin some good properties we want to maintain.

From now on, we assume that all baselines are simple paths. Now, each closed curve on $\partial\mathbb{R}(\mathcal{H})$ contains one or more baselines. So, we have an initial set of well defined substructures. The main purpose of this section is to cut these initial substructures such that Lemma A.3 holds.

We will execute a series of operations for constructing $\mathcal{H}$. We first provide below a high level sketch of our algorithm, and outline how the substructures and the substructure relation graph $\mathfrak{S}$ evolve along with the operations.

- (Section 2.6.1) First, we deal with active regions. Sometimes, two active region may overlap significantly and become inseparable (formally defined later), they essentially need to be dealt as a single active region. In this case, we merge the two active regions together (we do not need to do anything, but just to pretend that there is only one active region). We can also show that one active region can be merged with at most one other active region. For the rest of cases, two overlapping active region are separable, and we can cut them into at most two non-overlapping active regions, by adding a small number of extra disks in $\mathcal{H}$. After the merging and cutting operations, each substructure contains at most one active region. Hence, the substructures satisfy the property (P1) in Lemma A.3. Moreover, we show that if any substructure contains an active region, the substructure is limited in a small region.
- (Section 2.6.2) We ensure that each substructure is non-self-intersecting by a simple greedy algorithm. After this step, (P2) is satisfied.
- (Section 2.6.3) In this step, we ensure that substructure relation graph $\mathfrak{S}$ is a acyclic 2-matching (P3). The step has three stages. First, we prove that the set of blue edges forms a matching. Second, we give an algorithm for cutting the substructures which overlap with two or more other substructures. After the cut, each substructure overlaps with no more than one other substructure. So after the first two stages, we can see that $\mathfrak{S}$ is composed of a blue matching and a red matching. At last, we prove that the blue edges and red edges cannot form a cycle, establishing $\mathfrak{S}$ is acyclic.
- (Section 2.6.4) The goal of this step is to ensure the point-order consistency (P4). We first show there does not exist a point covered by more than two substructures, when $\mathfrak{S}$ is an acyclic 2-matching. Hence, we only need to handle the case of two overlapping substructures. We show it is enough to break all cycles in a certain planar directed graph. Again, we can add a few more disks to cut all such cycles.

- (Section 2.6.5) Lastly, we show that the number of disks added in $\mathcal{H}$ in the above four steps is $O(K^2)$, where $K = \frac{L}{\mu}$ and L and μ are side lengths of block and small square respectively.

2.6.1 Merging and Cutting Active Regions

If two active regions overlap in the same substructure, we need to either merge them into a new one or cut them into two non-overlapping new ones. As we know, each gadget may have two active regions. Suppose active regions Ar_1 and Ar_2 belong to the same gadget Gg, while Ar'_1 and Ar'_2 belong to a different gadget Gg'. Due to R-correlations, we need consider the four active regions together.

First, we consider the case where Ar_1 overlaps with Ar'_1, and Ar_2 overlaps with Ar'_2. We need the following important concept *order-separability*, which characterizes how the two sets of arcs overlap.

Definition 2.10 (Order-separability)： Consider a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$. $\mathcal{A}_1, \mathcal{A}_2$ are two disjoint subsets of $\mathcal{A}$. If $\mathcal{A}_1, \mathcal{A}_2$ satisfy that

$$a \prec b, \text{ for any } a \in \mathcal{A}_1 \text{ and } b \in \mathcal{A}_2, \tag{2-5}$$

or

$$a \succ b, \text{ for any } a \in \mathcal{A}_1 \text{ and } b \in \mathcal{A}_2, \tag{2-6}$$

we say that $\mathcal{A}_1, \mathcal{A}_2$ are order-separable.

We use $\mathcal{A}_1$, $\mathcal{A}'_1$, $\mathcal{A}_2$, $\mathcal{A}'_2$ to denote the set of arcs associated with active regions Ar_1, Ar'_1, Ar_2, Ar'_2, respectively. If $\mathcal{A}_1$ and $\mathcal{A}'_1$ are not order-separable, we say the pair $(\mathsf{Ar}_1, \mathsf{Ar}'_1)$ is a *mixture*. If both of $(\mathsf{Ar}_1, \mathsf{Ar}'_1)$ and $(\mathsf{Ar}_2, \mathsf{Ar}'_2)$ are mixtures, we say the two pairs form a *double-mixture*. When they are double-mixture, we merge them simultaneously. It only means that we regard the two active regions $(\mathsf{Ar}_1, \mathsf{Ar}'_1)$ as a new single active region, and $(\mathsf{Ar}_2, \mathsf{Ar}'_2)$ as another single active region.

To show an active region cannot grow unbounded, we prove that the merge operations do not generate chain reactions. The rough idea is that if two active regions form a mixture, their corresponding small squares must be adjacent and their core-central areas must overlap. Due to the special shape of core-central areas (a narrow spindle shape), the overlapping can only happen between two of them, not more.

Lemma 2.8: Consider two non-empty small squares Γ, Γ'. Suppose the square gadgets in the two squares are $\mathsf{Gg}(\Gamma) = (\mathsf{D}_s, \mathsf{D}_t)$ and $\mathsf{Gg}(\Gamma') = (\mathsf{D}'_s, \mathsf{D}'_t)$. The active region pairs $(\mathsf{Ar}_1, \mathsf{Ar}_2)$ and $(\mathsf{Ar}'_1, \mathsf{Ar}'_2)$ are associated with gadget $\mathsf{Gg}(\Gamma)$ and $\mathsf{Gg}(\Gamma')$ respectively. $(\mathsf{Ar}_1, \mathsf{Ar}_2)$ and $(\mathsf{Ar}'_1, \mathsf{Ar}'_2)$ form a double-mixture. Then, the following statements hold:

1. Their corresponding squares Γ, Γ' are adjacent;
2. The core-central areas of $\mathsf{Gg}(\Gamma)$ and $\mathsf{Gg}'(\Gamma')$ overlap;
3. The angle between D_sD_t and $D'_sD'_t$ is $O(\epsilon)$
4. Neither of the two core-central areas can overlap with any small squares other than Γ and Γ'.

Next, we consider the case where only one of $(\mathsf{Ar}_1, \mathsf{Ar}'_1)$ and $(\mathsf{Ar}_2, \mathsf{Ar}'_2)$ is a mixture. However, we show that it is impossible as follows. We use notation $\mathsf{Ar}(\mathcal{A})$ to denote an active region with arc set $\mathcal{A}$.

Lemma 2.9: Suppose active region pairs $(\mathsf{Ar}_1(\mathcal{A}_1), \mathsf{Ar}_2(\mathcal{A}_2))$ and $(\mathsf{Ar}'_1(\mathcal{A}'_1), \mathsf{Ar}'_2(\mathcal{A}'_2))$ are associated with gadget Gg and Gg' respectively. If $\mathcal{A}_1$ and $\mathcal{A}'_1$ are order-separable, then $\mathcal{A}_2$ and $\mathcal{A}'_2$ are also order-separable.

The proofs of Lemma 2.8 and Lemma 2.9 are elementary planar geometry and we defer them to Appendix 2.9.2.

Cutting Overlapping Active Regions: After the merging stage, even if any two active regions overlap in the same substructure, their arcs are order-separable. We define *label-cut* operation as below to further separate them such that the baselines of all the active regions are non-overlapping.

We consider two overlapping active regions $\mathsf{Ar}_1(\mathcal{A}_1)$ and $\mathsf{Ar}_2(\mathcal{A}_2)$ in a substructure $\mathsf{St}(\mathcal{A})$. Suppose $\mathcal{A}_1, \mathcal{A}_2 \subset \mathcal{A}$ and $\mathcal{A}_1, \mathcal{A}_2$ are order-separable. We can add two consecutive arcs in the envelope of A into $\mathcal{H}$. Then, St is cut into two substructures St_1 and St_2 with disjoint baselines, such that $\mathbb{R}(\mathsf{Ar}_1) \subset \mathbb{R}(\mathsf{St}_1)$ and $\mathbb{R}(\mathsf{Ar}_2) \subset \mathbb{R}(\mathsf{St}_2)$. We call the process *label-cut*. In other words, if we assign arcs in $\mathcal{A}_1$ one kind of label and arcs in $\mathcal{A}_2$ a different kind of label, after the label-cut operation, the arcs with different labels belong to different new substructures.

Lemma 2.10: Consider a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ and two subsets $\mathcal{A}'_1, \mathcal{A}'_2$ of $\mathcal{A}$. There exists a label-cut when $\mathcal{A}'_1$ and $\mathcal{A}'_2$ are order-separable.

Figure 2.7 illustrates the process of construction. Note that $\mathcal{A}'_1$ and $\mathcal{A}'_2$ are order-separable. We travel along the envepole of $\mathcal{A}$. We defer its proof in Appendix 2.9.2.

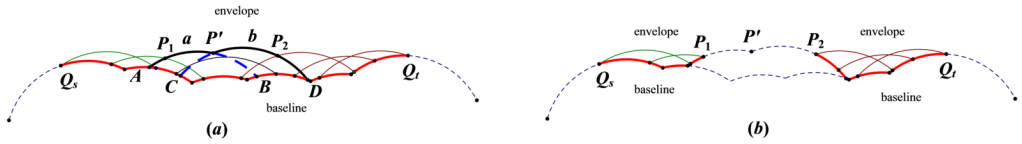


Figure 2.7 The example of label-cut. The left hand side illustrates the whole substructure before cutting. The arcs have two different labels. One is green and the other is brown. The bold black subarcs are what we select in the envelope. The right hand side illustrates that each of the two separable substructures induced by the label-cut operation only contains arcs with the same label.

After the label-cut operation, in each substructure, the baselines for all active regions are not overlapping. Thus, if any substructure contains more than one active region, consider any two of them, say $\mathcal{A}_1$ and $\mathcal{A}_2$. Note that $\mathcal{A}_1$ and $\mathcal{A}_2$ are not overlapping. Thus, we can add into $\mathcal{H}$ one arc a along the envelope which satisfies $a_1 \prec a \prec a_2, \forall a_1 \in \mathcal{A}_1, a_2 \in \mathcal{A}_2$. After the addition of arc a, $\mathcal{A}_1$ and $\mathcal{A}_2$ are separated into two different new substructures. Repeat the above step whenever one substructure contains more than one active region. This establishes the active region uniqueness property (P1).

Limiting the size of substructure which contains an active region: Now, we discuss how to make substructure which contains an active region bounded inside a small region. This property is particular useful later when we show the substructure relation graph $\mathfrak{S}$ is acyclic.

Suppose the gadget of square Γ is $(\mathsf{D}_s, \mathsf{D}_t)$. The line D_sD_t divides the plane into two halfplanes H^+ and H^-. D_s and D_t intersect at points P and Q. The boundary of disk $\mathsf{D}(P, 2)$ is tangent to D_s and D_t at point Q_s and Q_t respectively. $\mathsf{D}(Q_s, 1)$ and $\mathsf{D}(Q_t, 1)$ intersect at point D. See Figure 2.8. We call D the dome-point of gadget $\mathsf{Gg}(\Gamma)$ and use $\mathsf{Dom}(\Gamma^+)$ to denote the region $(\mathsf{D}(D, 1)) - \mathsf{D}_s - \mathsf{D}_t) \cap H^+$. Similarly, we use $\mathsf{Dom}(\Gamma^-)$ to denote the region $(\mathsf{D}(D, 1)) - \mathsf{D}_s - \mathsf{D}_t) \cap H^-$. $\mathsf{Dom}(\Gamma^+)$ and $\mathsf{Dom}(\Gamma^-)$ covers the active region associated with Γ. Formally speaking, we give the following lemma.

Lemma 2.11: Consider the substructure St which contains an active region of square Γ. The substructure can be cut into at most three smaller substructures, by doing label-cut at most twice. At most one of them contains the active region. Moreover, this new substructure (if any) is covered by the region $\mathsf{Dom}(\Gamma^+)$ (or $\mathsf{Dom}(\Gamma^-)$) associated with Γ.

We defer the proof to Appendix 2.9.2.

There may exist a substructure which contains a merged active region (i.e. an active region which is the union of two initial active regions). Based on Lemma 2.8, the arcs

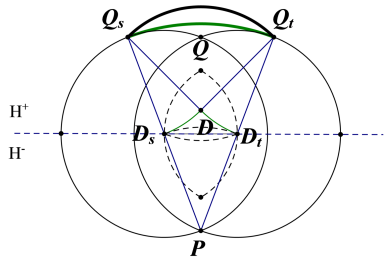


Figure 2.8 The farthest disk pair of square Γ is $(\mathsf{D}_s, \mathsf{D}_t)$. Suppose D_s and D_t intersect at point P, Q. The disk $\mathsf{D}(P, 2)$ is tangent to D_s and D_t at point Q_s and Q_t respectively. $\mathsf{D}(Q_s, 1)$ and $\mathsf{D}(Q_t, 1)$ intersect at D. The active region of Γ in H^+ is totally covered by $\mathsf{Dom}(\Gamma^+)$.

of the two initial active regions belong to two adjacent small squares. Suppose the two squares are Γ, Γ', with square gadgets $(\mathsf{D}_s, \mathsf{D}_t)$ and $(\mathsf{D}'_s, \mathsf{D}'_t)$ respectively. The dome-points of Γ and Γ' in H^+ are respectively D and D'. We apply the operations in Lemma 2.11 for each initial active region. Since the angle between D_sD_t and $D'_sD'_t$ is $O(\epsilon)$, obviously, the substructure containing the merged active region is small as well, i.e. bounded in $\mathsf{Dom}(\Gamma^+) \cup \mathsf{Dom}(\Gamma'^+)$.

Since each substructure which contains an active region is small enough, we can show the following lemma, which will be useful for proving the acyclicity of the substructure relation graph $\mathfrak{S}$.

Lemma 2.12 (Highly parallel arcs): Suppose substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ contains an active region. The central angle of any arc in St is no more than $O(\epsilon)$. Meanwhile, there exists a line l such that the angle between l and the tangent line at any point of any arc $a \in \mathcal{A}$ is at most $O(\epsilon)$.

The correctness is directly followed by Lemma 2.8. Actually, for the initial active region, the line D_sD_t satisfy the property. For the merged active region, we know the angle between D_sD_t and $D'_sD'_t$ is $O(\epsilon)$ based on Lemma 2.8. Hence, we can still see that the line D_sD_t satisfies the property.

To summarize, after all operations in this subsection, we can ensure that each substructure contains at most one active region (i.e., (P1) in Lemma A.3). Moreover, we have that each substructure which contains an active region is small enough (so that Lemma 2.12 holds).

Handling cyclic substructures: At the end of this subsection, we deal with the problem we left in the very beginning of Section 2.6, to break all cyclic baselines. Note that this step should be done in the beginning. First, we consider that case that there exists a point

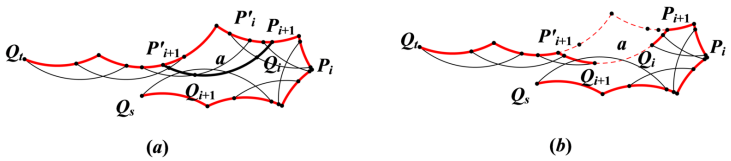


Figure 2.9 The process to avoid self-intersection. The left hand side is a self-intersection substructure. We search from point Q_s along the envelope. Let arc set $\mathcal{A}_i$ be $\{a_1, a_2, \ldots, a_i\}$. Then we have a set $\{\mathsf{St}_i[\mathcal{A}_i]\}_{i\in[k]}$ of substructures. If St_i is non-self-intersecting but St_{i+1} is self-intersecting, we add arc a_{i+1} in $\mathcal{H}$. The right hand side illustrates the two new substructures after the cut.

on the baseline which cannot be covered by any arc of any active region. We can include any envelope arc that covers the point into $\mathcal{H}$, which is enough to break the cycle. This is essentially a label-cut and does not separate any single connected active region into disconnected pieces. Then we consider the case where every point on the baseline is covered by some arc of active regions. Note that the merge operation only depends on the local property of two active regions, thus, we can merge active regions even in a "cyclic substructure". Assume that we have done all merge operations. Based on Lemma 2.12, we know one active region is very small and thus cannot cover all points on a closed curve. We pick one active region. Using the same operation as Lemma 2.11 (do two label-cuts), we can essentially isolate the active region and cut the original cyclic baseline to two new baselines.

2.6.2 Eliminating Self-intersections

The goal of this part is to add a few more disks into $\mathcal{H}$ so that any substructure is non-self-intersecting (Recall the definition in Lemma 2.3). Note that after merging and cutting process in Section 2.6.1, substructures which contain active region are non-self-intersecting. We just need to process the substructures without active regions in this part.

We use a simple greedy approach. We consider one substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$. Suppose the endpoints of b are Q_s and Q_t, and the envelope is $\{a_1[Q_s, Q_1],\ldots, a_k[Q_{k-1}, Q_t]\}$, where Q_i is the intersection point of a_{i-1} and a_i. We denote the endpoints of a_i on the baseline b by P_i and P'_i (Note that $P_1 = Q_s$). Let arc set $\mathcal{A}_i$ be $\{a_1, a_2, \ldots, a_i\}$. Then we have a set $\{\mathsf{St}_i[\mathcal{A}_i]\}_{i\in[k]}$ of substructures, where $\mathsf{St}_i[\mathcal{A}_i]$ is the substructure induced by arc set $\mathcal{A}_i$. We consider the arcs lying on the envelope one by one and check whether we should add it into $\mathcal{H}$ or not. Concretely, we add $\mathsf{D}(a_{i+1})$ in $\mathcal{H}$ if the following condition holds:

- St_i is non-self-intersecting, but St_{i+1} is self-intersecting,

The addition of $\mathsf{D}(a_{i+1})$ cuts the substructure into two smaller substructures. One of which is certainly non-self-intersecting. The other is induced by arc set $\mathcal{A} = \{a_{i+2}, \ldots, a_k\}$. See Figure 2.9. We repeat the above process until there is no self-intersection in all substructures. Furthermore, we can easily prove the following nice property. The proof can be found in Appendix 2.9.3.

Lemma 2.13: In each of the above iterations, one substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ is cut into at most two new substructures. Any original arc in $\mathcal{A}$ cannot be cut into two pieces, each of which belongs to a different new substructure.

To summarize, we have obtained the non-self-intersection property ((P2) in Lemma A.3).

2.6.3 Ensuring that $\mathfrak{S}$ is an Acyclic 2-Matching

We discuss how to add some extra disks in $\mathcal{H}$ to make $\mathfrak{S}$ an acyclic 2-matching ((P3) in Lemma A.3).

Blue edges: First we show that the set of blue edges form a matching.

Lemma 2.14: Two blue edges cannot be incident to the same node.

Proof Before the merge operation, the set of active region pairs forms a matching. To see this, note that our merge operations always apply to a double-mixture (which corresponds to merging two blue edges into one). Moreover, any cut operation cannot break one active region into two, thus has no effect on any blue edge. Hence, after all merge and cut operations, the set of active region pairs is still a matching. □

Red edges: Then, we prove that any node which has more than one incident red edges can be cut such that each new node(i.e., substructure) only has at most one incident red edge.

First, we prove a simple yet useful geometric lemma stating that a point cannot be covered by three or more substructures. Note that from now on, all substructures have no self-intersections.

Lemma 2.15: We are given a substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ and an arc $a \in \mathcal{A}$. Consider two arcs $b_1, b_2 \notin \mathcal{A}$. If b_1, b_2 cover the same point on a, b_1, b_2 should belong to the same substructure.

Intuitively, if the two disks corresponding to b_1 and b_2 cover the same point, they should be close enough such that their corresponding square gadgets overlap (which implies b_1 and b_2 share the same baseline). First we prove that the minimum distance between any two disks in two different substructure should not be too small, i.e., their overlapping region cannot be too large. The proof can be found in Appendix 2.9.4.

In fact, essentially the same proof can be used to prove that any two different substructures cannot both intersect with subarc whose central angle is $O(\epsilon)$, as in the following corollary.

Corollary 2.1: Suppose $\mathsf{St}(\mathsf{b}, \mathcal{A})$ is a substructure without any self-intersection. Consider an arc $a \in \mathcal{A}$ and two arcs $b_1, b_2 \notin \mathcal{A}$. Suppose a' is a subarc of a whose central angle is $O(\epsilon)$. If both b_1, b_2 cover some part of a', b_1, b_2 should belong to the same substructure.

Combining with Lemma 2.12, we can easily see the following lemma.

Lemma 2.16: Any substructure which contains an active region cannot overlap with two or more different substructures.

Then we show how to cut the substructure $\mathsf{St}(\mathsf{b}, \mathcal{A})$ which overlaps with more than one other substructures. Note that such substructure does not contain an active region based on Lemma 2.16. Suppose the envelope of St is $\{a_1[Q_s, Q_1], \ldots, a_k[Q_{k-1}, Q_t]\}$. St overlaps with k substructures $\mathsf{St}_i(\mathsf{b}_i, \mathcal{A}_i), i = 1, 2, \ldots, k$.

If St_i overlaps with St, there exists an arc $a \in \mathsf{St}_i$ intersecting some envelope arc of St. Thus, the envelope St is subdivided into several segments by those intersection points. We can label those segments as follows:

- If the segment is covered by some arc in St_i, we label it as 'i'.
- If there is no arc in any St_i covering the segment, we label it as '0'.

See Figure 2.10 for an example. According to the Lemma 2.15, we know there is no point on the envelope covered by two substructures. Thus the above labeling scheme is well defined.

Traversing those segments along the envelope, we obtain a label sequence. First, for each label i, we identify those maximal consecutive subsequence, which consists of only letter 0 and i, and starts with and end with i, and replace the subsequence by a single letter i. We obtain a compressed sequence. In fact, each letter, say i ($i \neq 0$), in the compressed sequence corresponds to one or more segments labeled with either i or 0, and

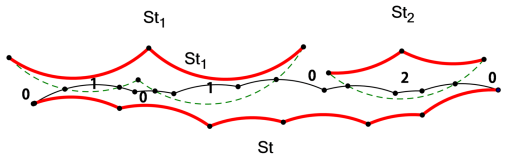


Figure 2.10 The arcs in substructures St_1 and St_2 cut the envelope of St into 7 segments. The sequence of the labels for those segments is 0101020. The compressed label sequence is 01020. So we have 5 l-segments.

the first and last one must be labeled with i. We call the concatenation of those segments an l(ong)-segment. Of course, a letter 0 in the compressed sequence corresponding to a segment with label 0. Actually, the sequence is Davenport-Schinzel sequence of order 2[70], because two segments with different labels cannot intersect (because the baselines of two substructures cannot intersect). For example, the pattern "1212" should never appear. Thus, the length of the compressed sequence, i.e., the number of l-segments, is at most $O(k)$.

Now, we discuss how to cut St into several new ones based on l-segments. Keep in mind that our goal is to make sure each new substructure only overlap with one substructure of $\{\mathsf{St}_i\}_{i\in[k]}$. The cut operation is again a simple greedy procedure. Consider two consecutive l-segments. Suppose they are $\{a_i[Q_{i-1}, Q_i], a_{i+1}[Q_i, Q_{i+1}], \ldots, a_j[Q_{j-1}, Q_j]\}$ and $\{a_{j+1}[Q_j, Q_{j+1}], \ldots, a_k[Q_{k-1}, Q_k]\}$. We add into $\mathcal{H}$ the last arc a_j of former l-segment and the first arc a_{j+1} of the later l-segment. St is thus cut into two new ones. Repeat the above step for all two consecutive l-segments in order.

We still need to show that after the cut, every new substructure overlap at most one of $\{\mathsf{St}_i\}_{i\in[k]}$. Consider one new substructure. Notice such an original arc in $\mathcal{A}$ can only belong to one new substructure. By our cut operation, all envelope arcs of the new substructure can intersect at most one of $\{\mathsf{St}_i\}_{i\in[k]}$. Hence, the new substructure can overlap at most one St_i.

Blue edges and red edges: After the above operations, the set of all blue edges forms a matching, while the set of all red edges also forms a matching. To show $\mathfrak{S}$ is an acyclic 2-matching, it suffices to prove $\mathfrak{S}$ contains no cycle. So, the rest of the section is devoted to prove the following lemma.

Lemma 2.17: Suppose the side length of square is μ, where $\mu = O(\epsilon)$ and the block contains $K \times K$ small squares, where $K = C_0/\epsilon^2$ and C_0 is an appropriate constant. Then, after all operations stated in this section, there is no cycle in $\mathfrak{S}$.

If there is a cycle Cyc in $\mathfrak{S}$, the red edges and blue edges alternate in Cyc, which correspond to a sequence of substructures, each containing an active region (since it is R-correlated with another active region). Now, we provide a high level explanation why Lemma 2.17 should hold. Each active region is associated with a small square. A small square is very small (comparing to a unit disk or the whole block), and an active region is also very small. We pick a point in each small square and each substructure. If two substructures in Cyc overlap, the two points in them are also very close (i.e., $O(\epsilon)$). So we can pretend the two points as one point (or we just pick a point in their overlapping region). For each active region Ar, we connect the point in Ar and the point in the small square associated with Ar. Since Both the small square and the active region are very small, the distance between two point is about $1 - O(\epsilon)$. Thus, a cycle Cyc would present itself geometrically as a polygon. We can show the angle between two adjacent edges of the polygon is close to π. So the size of the polygon cannot be not small (it takes a lot of edges to wrap a loop). However, the polygon cannot be much larger than the block. By contradiction, we prove that there is no cycle in $\mathfrak{S}$.

Now, we formally prove Lemma 2.17. We first prove a geometric lemma which will be useful for bounding the angle between the two adjacent edges of the aforementioned polygon.

Lemma 2.18: Consider two substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_1, \mathcal{A}_2)$ in Cyc. Suppose St_1 and St_2 overlap (there is a red edge between them). For any two arcs $a \in \mathcal{A}_1$ and $b \in \mathcal{A}_2$, suppose that $\mathsf{D}(a)$ and $\mathsf{D}(b)$ overlap and their intersection points are A and B. The central angles of a, b are θ_a, θ_b respectively. The centers of $\mathsf{D}(a), \mathsf{D}(b)$ are D_a, D_b. Then $\angle AD_aB$ (or $\angle AD_bB$) is at most $(\theta_a + \theta_b)$.

Proof We distinguish a few cases depending on whether the intersection points A and B lie on a or b or none of them. All cases are depicted in Figure 2.11.

- Both A and B lie on one arc (see Figure (a)(b)). W.l.o.g., suppose they lie on arc a. Obviously, $\angle AD_aB$ is no more than θ_a (or θ_b) .
- If one intersection is on neither a nor b (see Figure (c)), we prove that the case cannot happen. Suppose A is on neither of a and b. The endpoint A_1 of a is covered by disk $\mathsf{D}(b)$ and the endpoint A_2 of a is covered by disk $\mathsf{D}(a)$. Thus the baseline b_1 must intersect with b_2 which contradicts the fact that St_1 and St_2 are two different substructures.

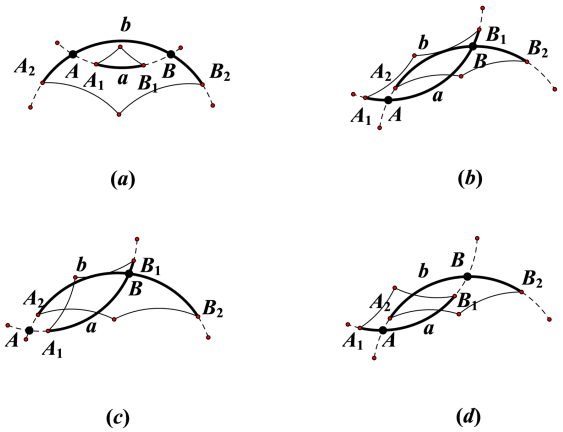


Figure 2.11 The four kinds of cases for two overlapping arcs.

- If one intersection point is on a but not on b and the other intersection point is on b but not on a (see Figure (d)), it is easy to see that B_1 is covered by arc b, and A_2 is covered by a since their baselines do not intersect. Thus, the length of arc AB (w.r.t. $\mathsf{D}(a)$) is no more than the sum of lengths of a and b. So $\angle AD_aB$ is at most $\theta_a + \theta_b$.

The above cases are exhaustive, thus our proof is completed. □

Based on Lemma 2.12, Lemma 2.18, we can prove Lemma 2.17 below:

Proof of Lemma 2.17: Consider two substructures St_1 and St_2. Suppose St_1 contains the active region Ar_1 , and St_2 contains the active region Ar_2. The centers of arcs of Ar_1 and Ar_2 locate in small squares Γ_1 and Γ_2 respectively. The square gadgets of the two squares are $\mathsf{Gg}(\mathsf{D}_{s_1}, \mathsf{D}_{t_1})$ and $\mathsf{Gg}(\mathsf{D}_{s_2}, \mathsf{D}_{t_2})$ respectively. If St_1 and St_2 overlap, there must exist an arc a in St_1 and an arc b in St_2 such that a and b intersect. Suppose $\mathsf{D}(a)$ and $\mathsf{D}(b)$ intersect at points A and B. The center of $\mathsf{D}(a)$ and $\mathsf{D}(b)$ are D_a and D_b.

Based on Lemma 2.12, we know both central angle of a and b are no more than $O(\epsilon)$. According to Lemma 2.18, the central angle of the arc AB is at most $O(\epsilon)$. It means angle between the tangent lines of $\mathsf{D}(a)$ and $\mathsf{D}(b)$ at point A is no more than $O(\epsilon)$. Thus, $\angle D_aAD_b$ is at least $\pi - O(\epsilon)$.

We know that all disks in the same active region are centered in one small square or two adjacent small squares. Moreover Lemma 2.12 implies all disk centers should lie in one or two squares. Hence, the distance between (any point in) the square and (any point in) its active region is at least $1 - O(\epsilon)$. Construct the aforementioned polygon. Consider two adjacent edges XY and YZ in the polygon. We consider two cases:

1. Y is in the intersection of two substructures St_1 and St_2. We can easily see that (1)

$|YA| = O(\epsilon)$; (2) $|XD_a| = O(\epsilon)$; (3) $|ZD_b| = O(\epsilon)$. Hence, we can see $\angle XYZ$ is at least $\pi - O(\epsilon)$.

2. Y is in a small square Γ and X and Z are in the two substructures associated with Γ. Since both substructures are bound in an $O(\epsilon)$ size region (by Lemma 2.12), we can see that $\angle XYZ$ is at least $\pi - O(\epsilon)$ as well.

Hence, we can see the polygon contains at least $\Omega(2\pi/\epsilon)$ nodes and the diameter of the polygon is at least $\Omega(1/\epsilon)$. But this cannot be larger than the diameter of a block, rendering a contradiction.

□

To summarize, we have ensured that $\mathfrak{S}$ is an acyclic 2-matching ((P3) in Lemma A.3).

2.6.4 Ensuring Point Order Consistency

We have ensured that the set of red edges is a matching in $\mathfrak{S}$. Hence, one substructure can overlap with at most one other substructure. Therefore, if we can guarantee that the points which are covered by any pair of overlapping substructures satisfy order consistence, then all points in $\mathcal{P}$ satisfy order consistency (after all, point-order consistency is defined over a pair of substructures).

Consider two overlapping substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$ and a set $\mathcal{P}_{\text{co}}$ of points covered by $\mathcal{A}_1 \cup \mathcal{A}_2$. Suppose $P_1, P_2 \in \mathcal{P}_{\text{co}}$ and $a_1, a_2 \in \mathcal{A}_1, b_1, b_2 \in \mathcal{A}_2$. Recall point-order consistency requires that when

- $P_1 \in \mathbb{R}(a_1) \cap \mathbb{R}(b_1)$ and $P_2 \in \mathbb{R}(a_2) \cap \mathbb{R}(b_2)$
- $P_1 \notin \mathbb{R}(a_2) \cup \mathbb{R}(b_2)$ and $P_2 \notin \mathbb{R}(a_1) \cup \mathbb{R}(b_1)$.

then

- $(a_1 \prec a_2) \Leftrightarrow (b_1 \prec b_2)$.

It is helpful to consider the following directed planar graph $\mathfrak{D}$ induced by all arcs in $\mathcal{A}_1 \cup \mathcal{A}_2$ in the uncovered region $\mathbb{U}(\mathcal{H})$. Regard each intersection point of arcs in $\mathcal{A}_1 \cup \mathcal{A}_2$ as a node. Each subarc is an directed edge with a direction consistent with its baseline. We use $A \to B$ to denote that there is a directed edge between nodes A and B in $\mathfrak{D}$. If there is no directed cycle in $\mathfrak{D}$, we can verify that all conditions listed above hold. Indeed, suppose the condition is not satisfied, which means $a_1 \prec a_2$, $b_1 \succ b_2$, $a_1, a_2 \in \mathcal{A}_1$ and $b_1, b_2 \in \mathcal{A}_2$. Suppose a_1, b_1 intersect at (A_1, B_1) and $A_1 \to B_1$, meanwhile a_2, b_2 intersect at (A_2, B_2) and $A_2 \to B_2$. Since $a_1 \prec a_2$, there exists a path in St_1 which goes from B_1 to A_2. Similarly, there exists a path in St_2 from B_2 to A_1. Thus, the two paths and a_1, a_2 form

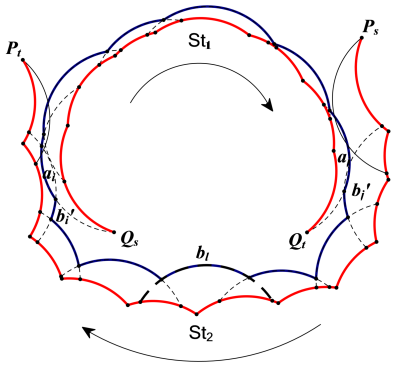


Figure 2.12 Substructure $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$ are overlapping. b_1 starts from Q_s and ends up with Q_t and b_2 starts from P_s and ends up with P_t. There are two paths forming a cycle.

a directed cycle.

So all we have to do is to break all cycles in $\mathfrak{D}$. When $\mathfrak{D}$ contains a cycle, we can cut the cycle through adding an arc on the envelope into $\mathcal{H}$. See Figure 2.12 for an example. Only arcs $\mathcal{A}_1$ (or in $\mathcal{A}_2$) cannot form a cycle. So if there is a cycle, the cycle must pass through the envelope of $\mathcal{A}_1$ and $\mathcal{A}_2$. Moreover, based on Lemma 2.12, if St_1 and St_2 form a cycle, either St_1 or St_2 does not contains any active region. W.l.o.g., suppose it is St_2. Based on these observations, we have our algorithms as follows:

Suppose the envelope of St_1 is $\mathsf{Path}_1 = \{a_1, a_2, \ldots, a_k\}$. a_i and a_j are the first and last arcs respectively which intersect St_2. Suppose the arc $b_{i'} \in \mathcal{A}_2$ overlaps with a_i, (if there is more than one such arc, we select a minimal one, w.r.t. the arc ordering) and the arc $b_{j'} \in \mathcal{A}_2$ overlaps with a_j (if there is more than one such arc, we select a maximal one). Since $a_i \prec a_j$ and they do not satisfy point-order consistency, we have $b_{i'} \succ b_{j'}$. We can see that all arcs between $b_{i'}$ and $b_{j'}$ cannot intersect with Path_1. So we can select one arc between $b_{i'}$ and $b_{j'}$ to add in $\mathcal{H}$ for cutting St_2 into two. After the cut, any cycle in $\mathfrak{D}$ can be broken.

After cutting St_2, St_1 overlaps with both of the new substructures obtained from St_2. Then, we encounter the same situation as in Section 2.6.3 (a node in $\mathfrak{S}$ has two incident red edges). We can apply the operation in Section 2.6.3 to cut St_1 such that the set of red edges in $\mathfrak{S}$ is still a matching.

2.6.5 The number of disks in $\mathcal{H}$

Finally, we count collectively the total number of disks that we have added in $\mathcal{H}$. First, we add the square gadget for each nonempty small square in $\mathcal{H}$. The number of the disks is $O(K^2)$, where $K = L/\mu = O(1/\epsilon^2)$. In order to cut overlapping active regions, the

number of disks we add in $\mathcal{H}$ is bounded by the number of active regions. Since there are $O(K^2)$ gadgets, we add $O(K^2)$ disks in Section 2.6.1. In Section 2.6.2, to ensure that each substructure is non-self-intersecting and contains at most one active region, we design a greedy algorithm. We can see that each arc we added in $\mathcal{H}$ covers at least one intersection point of two disks in $\mathcal{H}'$, where $\mathcal{H}'$ is the set $\mathcal{H}$ before this step. The algorithm guarantees that each arc which we add in $\mathcal{H}$ do not cover the same intersection point on the boundary of $\mathcal{H}'$. Since the union complexity of unit disks is linear[71] and $\mathcal{H}'$ contains at most $O(K^2)$ disks, there are at most $O(K^2)$ intersection points on $\partial\mathcal{H}'$. So, we add at most $O(K^2)$ in this step. In Section 2.6.3, we break the cycles in $\mathfrak{S}$. The number of disks we add $\mathcal{H}$ is proportional to the number of substructures. So, again, we add at most $O(K^2)$ disks. Similarly, in Section 2.6.4, we also add at most $O(K^2)$ disks. To summarize, we have added at most $O(K^2)$ disks in $\mathcal{H}$.

2.7 Time Complexity

The time complexity contains three parts. The first part is to enumerate all combination of $\mathcal{G}$. We set $C = O(\frac{K^2}{\epsilon})$ (since we need $C > |\mathcal{H}|/\epsilon$). Since $K = O(\frac{1}{\epsilon^2})$, the number of combinations is bounded by $n^C = n^{O(1/\epsilon^5)}$. The second part is the construction of set $\mathcal{H}$. It is easy to see that the time cost for each operation (i.e., label-cut) is no more than $O(n^2)$. Thus, the time cost is $O(K^2n^2) = O(n^2/\epsilon^4)$. The last part is the dynamic program. There are at most $O(K^2)$ substructures and at most $O(n^2)$ intersection points in each substructure. Thus, the number of total states is at most $O((n^2)^{K^2})$. For each recursion, the time cost is at most $O(n)$. Thus, the overall time complexity of the dynamic program is $O(n^{2K^2+1}) = n^{O(1/\epsilon^4)}$.

Overall, the total time cost is $n^{O(1/\epsilon^5)} \cdot \max\{n^2/\epsilon^4, n^{O(1/\epsilon^4)}\}) = n^{O(1/\epsilon^9)}$. This finishes the proof of Lemma A.2.

2.8 Applications

The weighted dominating set problem (MWDS) in unit disk graphs has numerous applications in the areas of wireless sensor networks[9]. In this section, we show that our PTAS for WUDC can be used to obtain better approximation algorithms for two important problems in this domain.

2.8.1 Connected Dominating Set in UDG

The goal for the *minimum-weighted connected dominating set* problem (MWCDS) is to find a dominating set which induces a connected subgraph and has the minimum total weight. Clark et al.[14] proved that MWCDS in unit disk graphs is NP-hard. Ambühl et al.[5] obtained the first constant factor approximation algorithm for MWCDS(the constant is 94). The ratio was subsequently improved in a series of papers[6,30-31]. The best ratio known is 7.105[9] pp.78.

One way to compute an approximation solution for MWCDS is to first compute minimum weighted dominating set (MWDS) and then connect the dominating set using a *node-weighted steiner tree* (NWST)[30,72]. The optimal MWDS value is no more than the optimal MWCDS value. After zeroing out the weight of all terminals, the optimal NWST value (for any set of terminals) is also no more than the optimal MWCDS value. Hence, if there is an α-approximation for MWDS(or equivalently WUDC) and a β-approximation for NWST, then there is an $\alpha + \beta$ factor approximation algorithm for MWCDS.

Zou et al.[72] show that there exists a 2.5ρ-approximation for NWST if there exists a ρ-approximation for the classical edge-weighted *minimum steiner tree problem*. The current best ratio for minimum steiner tree is 1.39[73]. Thus, there exists a 3.475-approximation for NWST. Combining with our PTAS for WUDC, we obtain the following improved result for MWCDS.

Theorem 2.3: There exists a polynomial-time $(4.475 + \epsilon)$-approximation for MWCDS for any fixed constant $\epsilon > 0$.

2.8.2 Maximum Lifetime Coverage in UDG

The *maximum lifetime coverage problem* (MLC) is a classical problem in wireless sensor networks: Given n targets $t_1, \ldots, t_n$ and m sensors $s_1, \ldots s_m$, each covering a subset of targets, find a family of sensor cover $S_1, \ldots, S_p$ with time lengths $\tau_1, \ldots, \tau_p$ in $[0, 1]$, respectively, to maximize $\tau_1 + \ldots + \tau_p$ subject to that the total active time of every sensor is at most 1. MLC is known to be NP-hard[74]. Berman et al.[75] reduced MLC to the *minimum weight sensor cover* (MSC) problem through Garg-Könemann technique[76]. In particular, they proved that if MSC has a ρ-approximation, then MLC has a $(1 + \epsilon)\rho$-approximation for any $\epsilon > 0$. Ding et al.[33] noted that, if all sensors and targets lie in the Euclidean plane and all sensors have the same covering radius, any approximation result for WUDC can be converted to almost the same approximation for MLC. Hence, the current best known

result for MLC is a $(3+\epsilon)$-approximation[34]. Using our PTAS, we obtained the first PTAS for MLC.

Theorem 2.4: There exists a PTAS for MLC when all sensors and targets lie in the Euclidean plane and all sensors have the same covering radius.

Let us mention one more variant of MLC, called maximum lifetime connected coverage problem, studied by Du et al.[77]. The problem setting is the same as MLC, except that each sensor cover S_i should induce a connected subgraph. They obtained a $(7.105+\epsilon)$-approximation when the communication radius R_c is no less two times the sensing radius R_s. Essentially, they showed that an α-approximation for WUDC and a β-approximation for NWST imply an $\alpha+\beta$-approximation algorithm for the connected MLC problem. Using our PTAS, we can improve the approximation ratio to $(4.475+\epsilon)$.

2.9 Delayed Proofs of PTAS for WUDC

2.9.1 Delayed Proofs in Section A

Lemma 2.19: The central angle of any uncovered arc is less than π.

Proof We only need prove the arc of a gadget is less than π. As we union all gadgets and add more and more disks in $\mathcal{H}$, the central angle of an arc only becomes smaller. So, now we fix a square gadget $\mathsf{Gg}(\Gamma)$. Consider the substructure above line D_sD_t. See the right hand side of Figure A.1 for an example. Suppose an arc a with endpoints M and M' on the boundary of Gg. The center of the arc is in the central area of Gg. If the central angle $\angle(a) \geq \pi$, its center should lie in the cap region bounded by a and the chord MM'. Without loss of generality, we suppose M is closer to the line D_sD_t than M'. Draw a auxiliary line at M which is parallel to D_sD_t. If the line does not intersect the central area, the center a locates below the line MM' (Hence, outside the cap region), thus the central angle is less than π. If the auxiliary line intersects the boundary of central area at point N. Then, we can see that

$$|MN| = \sqrt{|MD_s|^2 - x^2} + |D_sD_t| - \sqrt{|ND_t|^2 - x^2},$$

where x is the vertical distance between point M and line D_sD_t. It is not difficult to see that $|MN| > 1$. It means that the center of the arc locates below the line MN (Otherwise, the distance from the center to M is larger than $|NM|$, which is larger than 1, rendering a contradiction). So, the central angle of any arc is less than π. □

2.9.2 Delayed Proofs in Section 2.6.1

Lemma 2.8 *Consider two non-empty small squares* Γ, Γ'. *Suppose the square gadgets in the two squares are* $\mathsf{Gg}(\Gamma) = (\mathsf{D}_s, \mathsf{D}_t)$ *and* $\mathsf{Gg}(\Gamma') = (\mathsf{D}'_s, \mathsf{D}'_t)$. *The active region pairs* $(\mathsf{Ar}_1, \mathsf{Ar}_2)$ *and* $(\mathsf{Ar}'_1, \mathsf{Ar}'_2)$ *are associated with gadget* $\mathsf{Gg}(\Gamma)$ *and* $\mathsf{Gg}(\Gamma')$ *respectively.* $(\mathsf{Ar}_1, \mathsf{Ar}_2)$ *and* $(\mathsf{Ar}'_1, \mathsf{Ar}'_2)$ *form a double-mixture. Then, the following statements hold:*

1. *Their corresponding squares* Γ, Γ' *are adjacent;*
2. *The core-central areas of* $\mathsf{Gg}(\Gamma)$ *and* $\mathsf{Gg}'(\Gamma')$ *overlap;*
3. *The angle between* D_sD_t *and* $D'_sD'_t$ *is* $O(\epsilon)$
4. *Neither of the two core-central areas can overlap with any small squares other than* Γ *and* Γ'.

Proof Suppose the core-central areas of Gg and Gg' are $\mathfrak{C}_o$ and $\mathfrak{C}'_o$ respectively. Since $(\mathsf{Ar}_1, \mathsf{Ar}_2)$ and $(\mathsf{Ar}'_1, \mathsf{Ar}'_2)$ form a double-mixture, there exists at least one disk centered in Γ' which appears in Ar_1. Thus, the disk is centered in $\mathfrak{C}_o$. It means that $\mathfrak{C}_o$ and $\mathfrak{C}'_o$ overlap. Hence, Γ and Γ' are adjacent. See Figure 2.13.

It is easy to see that any core-central area can overlap with at most two squares. Since $\mathfrak{C}_o$ and $\mathfrak{C}'_o$ overlap, at least one of them overlaps with both Γ and Γ'. Without loss of generality, suppose $\mathfrak{C}_o$ overlap with both Γ and Γ'. We only need to prove $\mathfrak{C}'_o$ cannot overlap with other squares other than Γ and Γ'.

Our proof needs a useful notion, called *apex angle*. Consider a square gadget. See the left one of Figure A.1 (and we use the notations there). The line D_sE and D_sF are the tangent lines to the boundary of core-central area at point D_s. We define *apex angle* $\theta_{\mathfrak{C}_o}$ to be $\angle ED_sF$. We notice that $\theta_{\mathfrak{C}_o}$ only depends on the distance $|D_sD_t|$. When the side length of a small square is ϵ, $\theta_{\mathfrak{C}_o}$ is $O(\epsilon)$. In fact, even the core-central area is completely determined by D_s and D_t. Thus, we can generalize the concept *core-central area* by using any two overlapping disks in $\mathcal{H}$.

Now go back to our proof. See Figure 2.13. The segments D_sD_t and $D'_sD'_t$ cannot intersect because they belong to different small squares. Suppose the squares Γ and Γ' are $A_1A_2A_5A_6$ and $A_2A_3A_4A_5$, respectively. A_2A_5 is the common side. W.l.o.g, suppose D'_s is closer to line A_2A_5 than D'_t. If any disk centered in $\mathfrak{C}'_o$ can appear in Ar_1 but outside the disk D'_s (i.e., outside the disks $\mathsf{D}_s \cup \mathsf{D}_t \cup \mathsf{D}'_s$), the core-central areas defined by D'_sD_t [①] and $\mathfrak{C}'_o$ should overlap nontrivial (not only touch at point D'_s). Thus, the angle $\angle D_tD'_sD'_t$ should be less than $O(\epsilon)$. It means angle between D_sD_t and $D'_sD'_t$ is at most $O(\epsilon)$. Moreover,

① D'_s and D_t do not locate in the same square. This is core-central area in the generalized sense..

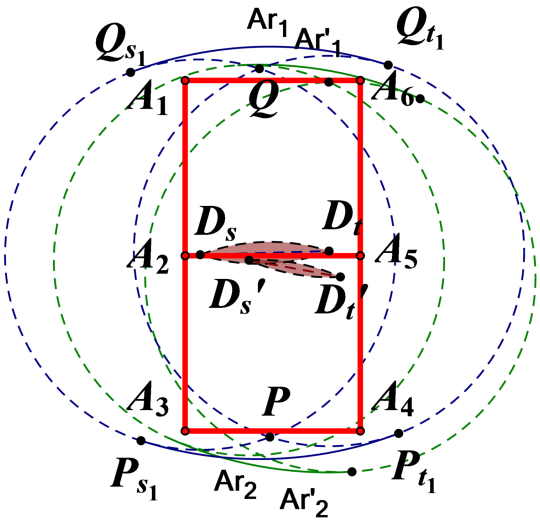


Figure 2.13 The case that arcs in two different active regions are not order-separable. The two adjacent squares are $\Gamma = A_1A_2A_5A_6$ and $\Gamma' = A_2A_3A_4A_5$. $(\mathsf{D}_s, \mathsf{D}_t)$ is the square gadget in Γ and $(\mathsf{D}'_s, \mathsf{D}'_t)$ is the square gadget in Γ'. The active regions Ar_1 and Ar_2 belong to the $\mathsf{Gg}(\Gamma)$, while the Ar'_1 and Ar'_2 belong to the $\mathsf{Gg}'(\Gamma')$.

angle between line $D'_sD'_t$ and A_5A_4 is no less than $\frac{\pi}{2} - O(\epsilon) > O(\epsilon)$. Thus, $\mathfrak{C}'_o$ cannot intersect A_5A_4. Hence, $\mathfrak{C}'_o$ cannot overlap other squares. □

Lemma 2.9 *Suppose active region pairs* $(\mathsf{Ar}_1(\mathcal{A}_1), \mathsf{Ar}_2(\mathcal{A}_2))$ *and* $(\mathsf{Ar}'_1(\mathcal{A}'_1), \mathsf{Ar}'_2(\mathcal{A}'_2))$ *are associated with gadget* Gg *and* Gg' *respectively. If* $\mathcal{A}_1$ *and* $\mathcal{A}'_1$ *are order-separable, then* $\mathcal{A}_2$ *and* $\mathcal{A}'_2$ *are also order-separable.*

Proof We prove the lemma by contradiction. We only consider the arcs which have siblings. Suppose $\mathcal{A}_1$ and $\mathcal{A}'_1$ are order-separable but $\mathcal{A}_2$ and $\mathcal{A}'_2$ are not order separable. W.l.o.g., assume $a \prec a', \forall a \in \mathcal{A}_1, \forall a' \in \mathcal{A}'_1$. Since $\mathcal{A}_2$ and $\mathcal{A}'_2$ are not order-separable, there exist arcs $b_1, b_2 \in \mathcal{A}_2$ and $b' \in \mathcal{A}'_2$ such that $b_1 \prec b' \prec b_2$. (If not, there exist arcs $b \in \mathcal{A}_2$ and $b'_1, b'_2 \in \mathcal{A}'_2$ such that $b'_1 \prec b \prec b'_2$.) Suppose a_2 is the sibling of b_2, a' is the sibling of b'. Thus, $a' \prec a_2$ based on the same proof to Lemma 2.5. It yields a contradiction to the assumption that $a \prec a', \forall a \in \mathcal{A}_1, \forall a' \in \mathcal{A}'_1$. □

Lemma 2.10 *Consider a substructure* $\mathsf{St}(\mathsf{b}, \mathcal{A})$ *and two subsets* $\mathcal{A}'_1, \mathcal{A}'_2$ *of* $\mathcal{A}$. *There exists a label-cut when* $\mathcal{A}'_1$ *and* $\mathcal{A}'_2$ *are order-separable.*

Proof We only discuss the case that some arcs of $\mathcal{A}'_1$ are adjacent to some arcs of $\mathcal{A}'_2$. (If not, we can trivially select one arc a along the envelope which satisfy $a'_1 \prec a \prec a'_2, \forall a'_1 \in$

$\mathcal{A}'_1, a'_2 \in \mathcal{A}'_2$ to separate them.) Along the envelope of substructure St, there exist two consecutive arcs such that one is in $\mathcal{A}'_1$ but the other is in $\mathcal{A}'_2$. we use a, b to denote the two arcs and add them in $\mathcal{H}$. Suppose the endpoints of a and b on b are (A, B), (C, D) respectively. a and b intersect at P'. Let region $\mathbb{R}_{co}$ to be $\mathbb{R}(a) \cup \mathbb{R}(b)$. It is easy to see that $\mathbb{R}_{co}$ separate the St into two new ones. Suppose the endpoints of baseline b are (Q_s, Q_t). Regard $\mathsf{b}[Q_s, A] \cup a[A, P']$ as the baseline b_1 of St_1 and $b[P', D] \cup \mathsf{b}[D, Q_t]$ as the baseline b_2 of St_2. The subarcs of $\mathcal{A}$ in region $\mathbb{R}(\mathsf{St}) - \mathbb{R}_{co}$ are separated two different group based on the two different baselines. We denote them by $\mathcal{A}_1$ and $\mathcal{A}_2$. Since $a' \prec b, \forall a' \in \mathcal{A}'_1$ and $b' \succ a, \forall b' \in \mathcal{A}'_2$, a' cannot intersect with $b[P', D]$ and b' cannot intersect with $a[A, P']$. Thus, no subarc in $\mathcal{A}'_1$ belongs to $\mathcal{A}_2$ and no subarc in $\mathcal{A}'_2$ belongs to $\mathcal{A}_1$. So, we construct two suitable substructures $\mathsf{St}_1(\mathsf{b}_1, \mathcal{A}_1)$ and $\mathsf{St}_2(\mathsf{b}_2, \mathcal{A}_2)$. Figure 2.7 illustrates the process of construction. □

Lemma 2.11 *Consider the substructure* St *which contains an active region of square* Γ. *The substructure can be cut into at most three smaller substructures, by doing label-cut twice. At most one of them contains the active region. Moreover, this new substructure (if any) is bounded by the region* $\mathsf{Dom}(\Gamma^+)$ *associated with* Γ.

Proof See Figure 2.8. We use the same notations defined in Section 2.6.1. The entire active region $[\bigcup_{i:D_i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)] \cap H^+$ is a sub-region of $(\mathsf{D}(P, 2) - \mathsf{D}_s - \mathsf{D}_t) \cap H^+$. Thus, any disk centered in $\mathsf{D}(Q_s, 1) \cap \mathsf{D}(Q_t, 1) \cap H^+$ can cover the active region. So, if there is a disk centered in $\mathsf{D}(Q_s, 1) \cap \mathsf{D}(Q_t, 1) \cap H^+$, we add it into $\mathcal{H}$ such that the entire active region is covered. If not, we prove we can cut St such that the new substructure containing the active region is bounded by the region $(\mathsf{D}(D, 1) - \mathsf{D}_s - \mathsf{D}_t) \cap H^+$. The disks centered in $\mathbb{R}(\Gamma) \cap \mathsf{D}(Q_s, 1) - \mathsf{D}(Q_s, 1) \cap \mathsf{D}(Q_t, 1))$ cover point Q_s. Thus, the arcs of these disks are order-separable to the arcs in active region. Similarly, the arcs of disks centered in $\mathbb{R}(\Gamma) \cap \mathsf{D}(Q_t, 1) - \mathsf{D}(Q_t, 1) \cap \mathsf{D}(Q_s, 1)$ are also order-separable to the arcs in active region. Thus, we can add two disks in $\mathcal{H}$ to label-cut St into at most three new substructures. Obviously, there is only one of them containing arcs centered in $\Gamma - \mathsf{D}(Q_s, 1) \cup \mathsf{D}(Q_t, 1)$. Hence, this new substructure is bounded in the region $(\mathsf{D}(D, 1) - \mathsf{D}_s - \mathsf{D}_t) \cap H^+$. □

2.9.3 Delayed proofs in Section 2.6.2

Lemma 2.13 *In each of the above iterations, one substructure* $\mathsf{St}(\mathsf{b}, \mathcal{A})$ *is cut into at most two new substructures. Any original arc in* $\mathcal{A}$ *cannot be cut into two pieces, each of which belongs to a different new substructure.*

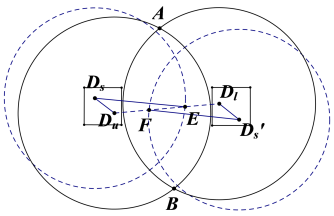


Figure 2.14 D_u and D_l intersect at point A and B. Suppose the side length of square is μ. If the central angle of arc $a[A, B]$ is more than $2\sqrt{2\mu}$. $\mathsf{D}_s, \mathsf{D}'_s$ should overlap, where $\mathsf{D}_s \in \mathsf{Gg}(\Gamma_u), \mathsf{D}'_s \in \mathsf{Gg}(\Gamma_l)$.

Proof Figure 2.9 gives an explanation about the change of substructures before and after the process. Suppose the envelope is $\{a_1[Q_s, Q_1], \ldots, a_k[Q_{k-1}, Q_t]\}$. After we add the disk $\mathsf{D}(a_i)$ in $\mathcal{H}$, the baselines of the two new substructures are $\mathsf{b}_1 = \mathsf{b}[Q_s, P_{i+1}] \cup a_{i+1}[P_{i+1}, Q_i]$ and $\mathsf{b}_2 = a_{i+1}[Q_{i+1}, P'_{i+1}] \cup \mathsf{b}[P'_{i+1}, Q_t]$. Since a_{i+1} is an arc lying on the envelope, there does not exist an arc b with endpoints (A, B) such that $A \prec P_{i+1} \prec P'_{i+1} \prec B$. So the endpoints of any arc in St_1 cannot be in $\mathbb{R}(\mathsf{St}_2)$. Thus, any arc cannot be separated into two substructures. □

2.9.4 Delayed proofs in Section 2.6.3

Lemma 2.20: Consider two disks D_u and D_l in squares Γ_u, Γ_l respectively. Suppose $\mathsf{D}_u \notin \mathsf{Gg}(\Gamma_u), \mathsf{D}_l \notin \mathsf{Gg}(\Gamma_l)$ and D_u and D_l intersect at points A and B. Suppose the side length of square is μ. If $\angle AD_uB$ (or $\angle AD_lB$) is more than $2\sqrt{2\mu}$, $\mathsf{Gg}(\Gamma_u)$ and $\mathsf{Gg}(\Gamma_l)$ should overlap. Moreover, if A (resp. B) is in $\mathbb{U}(\mathcal{H})$, the two arcs which intersect at point A (resp. B) are in the same substructure.

Proof Suppose D_u and D_l intersect at point A and B. $\mathsf{D}_s \in \mathsf{Gg}(\Gamma_u)$, $\mathsf{D}'_s \in \mathsf{Gg}(\Gamma_l)$. Thus, $\mathsf{D}_u, \mathsf{D}_s$ locate in the same square Γ_u and $\mathsf{D}_l, \mathsf{D}'_s$ locate in the same square. The line D_uD_l intersect disk D_s at point E and intersect disk D'_s at point F. $|D_uE|$ and $|D_lF|$ are no less than $1 - \mu$ according to the triangle inequality. Suppose $\angle AD_uB$ is θ. Then the length $|D_uD_l|$ equals $2\cos(\theta/2)$. Since $\theta > 2\sqrt{2\mu}$, we have $|D_uE| + |D_lF| > |D_uD_l|$, i.e. D_s and D'_s overlap. Figure 2.14 illustrates the situation. If A (or B) is in $\mathbb{U}(\mathcal{H})$, it is obvious that the two arcs which intersect at A (or B) can cover the same point on the baseline (i.e., the intersection point of D_s and D'_s). Hence, they are in the same substructure. □

Based on the Lemma 2.20, we prove Lemma 2.15 as follows.

Lemma 2.15 *We are given a substructure* $\mathsf{St}(\mathsf{b}, \mathcal{A})$ *and an arc* $a \in \mathcal{A}$. *Suppose two*

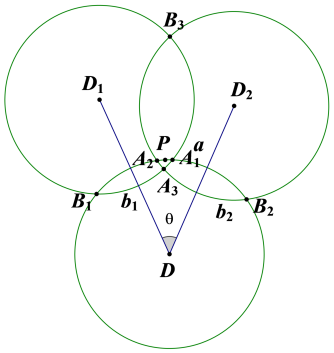


Figure 2.15 Consider a point P on D. a, b_1, b_2 which belong to disks $\mathsf{D}, \mathsf{D}_1, \mathsf{D}_2$ respectively, can cover the point P. D and D_1 intersect at A_1 and B_1, meanwhile D and D_2 intersect at A_2 and B_2. Then, D_1 and D_2 belong to the same substructure.

arcs $b_1, b_2 \notin \mathcal{A}$. If b_1, b_2 cover the same point on a, b_1, b_2 should belong to the same substructure.

Proof We prove the lemma by contradiction. Suppose the arcs a, b_1, b_2 are parts of $\mathsf{D}, \mathsf{D}_1, \mathsf{D}_2$ respectively. Consider a point P on D that can be covered by all of a, b_1, b_2. D and D_1 intersect at A_1 and B_1, meanwhile D and D_2 intersect at A_2 and B_2. See Figure 2.15. Since D and D_1 belong to different substructures, we know $\angle A_1DB_1$ is less than $2\sqrt{2\mu}$ based on Lemma 2.20. Similarly, $\angle A_2DB_2$ is less than $2\sqrt{2\mu}$. It is easy to see that $\angle D_1DD_2$ is less than $2\sqrt{2\mu}$. $\angle D_1DD_2$ is $O(\sqrt{\epsilon})$ when $\mu = O(\epsilon)$. $|D_1D_2|$ is less than 1 as $|DD_1|$ and $|DD_2|$ are more than 1 and less than 2. Suppose A_3 and B_3 are the two intersection points of D_1 and D_2. Thus, we have $\angle A_3D_1B_3 > \pi/3 > 2\sqrt{2\mu}$. Moreover, P is in $\mathbb{U}(\mathcal{H})$. By Lemma 2.20, we can see that it contradicts the fact that D_1 and D_2 belong to two different substructures. □

2.10 Final Remarks

Much of the technicality comes from the fact that the substructures interact with each other in a complicated way and it is not easy to ensure a globally consistent order. The reader may wonder what if we choose more than two disks (but still a constant) in a small square, hoping that the uncovered regions become separated and more manageable. We have tried several other ways, like choosing a constant number of disks in the convex hull of the centers in a small square. However, these seemed to only complicate, not to simplify, the matter.

We believe our result and insight are useful to tackle other problems involving unit disks or unit disk graphs. On the other hand, our approach strongly relies on the special properties of unit disks and does not seem to generalize to arbitrary disks with disparate radius. Obtaining a PTAS for the weighted disk cover problem with arbitrary disks is still a central open problem in this domain. An interesting intermediate step would be to consider the special case where the ratio between the longest radius and the shortest radius is bounded.

Chapter 3 Odd Yao-Yao Graphs are Not Spanners

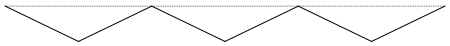
3.1 Overview of our Counterexample Construction

We first note that both the counterexamples for YY_3 and YY_5 are not weak t-spanners[48,51]. However, Yao-Yao graphs YY_k for $k \geq 7$ are all weak t-spanners[65-67]. 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.

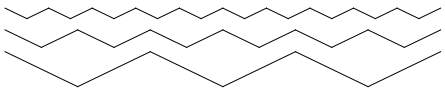
Our example contains two types of points, called *normal points* and *auxiliary points*. Denote them by $\mathcal{P}_m^{\mathrm{n}}$ and $\mathcal{P}_m^{\mathrm{a}}$ respectively and $\mathcal{P}_m = \mathcal{P}_m^{\mathrm{n}} \cup \mathcal{P}_m^{\mathrm{a}}$. The normal points form the basic skeleton, 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 A.4(a)). 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 A.4(b)). If we repeat this process directly, the length of the resulting path would increase to infinity as m approaches infinity since $\cos^{-1}\gamma > 1$ (see Figure A.4(c)). 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.

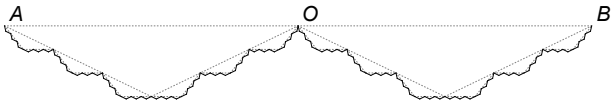
1. As m increases, the polygonal path may intersect itself. See Figure A.4(d). The polygonal path intersects itself around the point O. This is relatively easy to handle: we do not recurse for those segments that may cause self-intersection. See Figure A.4(e). We do not replace the bold segment further. We need to make sure



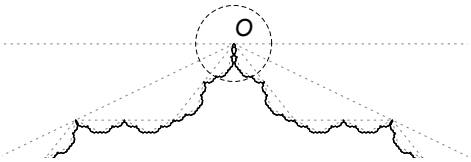
(a) Replace a horizontal segment by a sawteeth-like path.



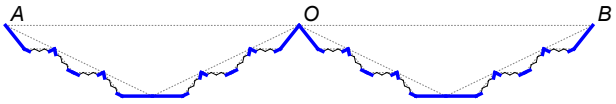
(b) The lengths of the sawteeth-like paths are independent of the number of sawteeth.



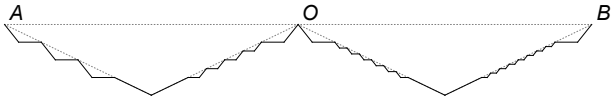
(c) Replace the segments by sawteeth-like path recursively.



(d) An enlarged view of Figure A.4(c) 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 3.1 The overview of the counterexample construction. Figure A.4(a)-A.4(f) illustrate the fractal and its variants.

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 A.4(f). 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 A.

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 A.5 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 3.3.

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 introduce no new long range connection. We describe the details in Section 3.4.

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. We describe the details in Section 3.5.

3.2 The Positions of Normal Points

In this section, we describe the positions of normal points. Note that, in the section, we only care about the positions of the points. The segments in any figure of this section are used to illustrate the relative positions of the points. Those segments may not represent the edges in Yao or Yao-Yao graphs. See Figure 3.2 for an overview of the positions of the normal points.

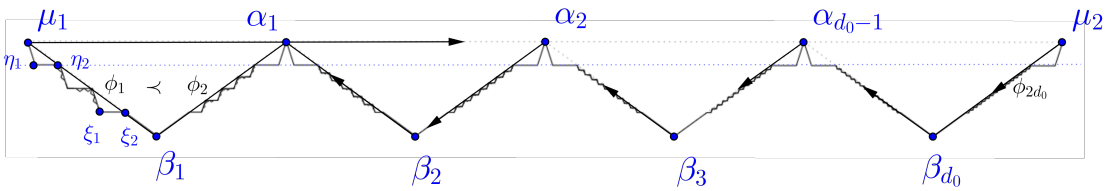


Figure 3.2 The overview of the positions of normal points. There exists a point at each intersection of these segments. $\mu_1\mu_2$ is horizontal. $\{\alpha_1, \alpha_2, \ldots, \alpha_{d_0-1}\}$ partitions the segment $\mu_1\mu_2$ into d_0 equal parts. For each β_i, $\angle\alpha_{i-1}\beta_i\alpha_i = \pi - \theta$ and $|\alpha_{i-1}\beta_i| = |\beta_i\alpha_i|$. We call $\{\alpha_1, \alpha_2, \ldots, \alpha_{d_0-1}\}$ the partition set and $\{\beta_1, \beta_2, \ldots, \beta_{d_0}\}$ the apex set of pair (μ_1, μ_2).

3.2.1 Some Basic Concepts

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

Definition 3.1 (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.

Note that in an odd Yao-Yao graph, if $\overrightarrow{uv}$ is a cone boundary, its reverse $\overrightarrow{vu}$ is not a cone boundary. In retrospect, this property is a key difference between odd Yao-Yao graphs and even Yao-Yao graphs, and our counterexample for odd Yao-Yao graphs will make crucial use of the property. We make it explicit as follows.

Property 3.2: Consider two points u and v in $\mathcal{P}$. If $\overrightarrow{uv}$ is a cone boundary in $\mathrm{YY}_{2k+1}(\mathcal{P})$, its reverse $\overrightarrow{vu}$ is not a cone boundary.

Definition 3.3 (Boundary Pair): A boundary pair consists of two ordered points, denoted by (w_1, w_2), such that $\overrightarrow{w_1w_2}$ is a cone boundary of point w_1.

For convenience, we refer to the word *pair* in the dissertation as the boundary pair defined in Definition A.10. According to Property A.4, if (w_1, w_2) is a pair, its reverse (w_2, w_1) is not a pair. For a pair $\phi = (w_1, w_2)$, we call w_1 the first point in ϕ and w_2 the second point in ϕ. 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: Now, we introduce the concept of a gadget generated by a pair $\phi = (w_1, w_2)$. Such a gadget is a collection of points which is a superset of ϕ (see Figure A.6). If the recursive level m increases by 1, we use a gadget generated by pair ϕ to replace ϕ.

① Note that the cases $k = 1, 2$ have been proved in [48].

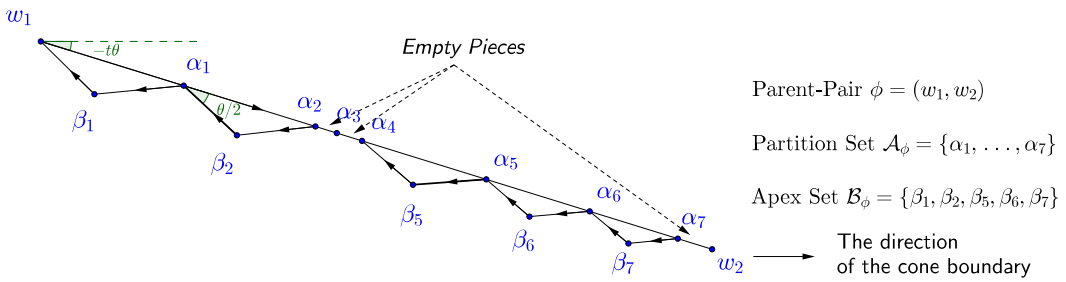


Figure 3.3 An example of one gadget. $\phi = (w_1, w_2)$ is the parent-pair in the gadget. $\mathcal{A}_\phi = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_7\}$ is the partition set and $\mathcal{B}_\phi = \{\beta_1, \beta_2, \beta_5, \beta_6, \beta_7\}$ is the apex set. There are eight pieces, in 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.

One gadget Gg_ϕ consists of three groups of points. We explain them one by one. See Figure A.6 for an example.

1. The first group is the pair $\phi = (w_1, w_2)$. We call the pair the *parent-pair* of the gadget Gg_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. For example, in Figure A.6, $\{\alpha_1, \alpha_2, \ldots, \alpha_7\}$ (here, $|w_1\alpha_i| < |w_1\alpha_j|$ if $i < j$) is a partition set of (w_1, w_2). 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 A.0.0.2.
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 $|\alpha_{i-1}\beta_i| = |\beta_i\alpha_i|$.[①] All β_is are on the same side of w_1w_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. In Figure A.6, $\{\beta_1, \beta_2, \beta_5, \beta_6, \beta_7\}$ is an apex set of (w_1, w_2).

We summarize the above construction in the following definition.

Definition 3.4 (Gadget): A gadget Gg_ϕ generated by a pair ϕ is a set of points which consists of the pair ϕ, a partition set $\mathcal{A}_\phi$ and an apex set $\mathcal{B}_\phi$ of ϕ. We denote the gadget by $\mathsf{Gg}_\phi[\mathcal{A}_\phi, \mathcal{B}_\phi]$.

① Note that the subscript i of β_i is consistent with the subscript of the piece $\alpha_{i-1}\alpha_i$. Hence, the subscripts may not be consecutive among all β_is.

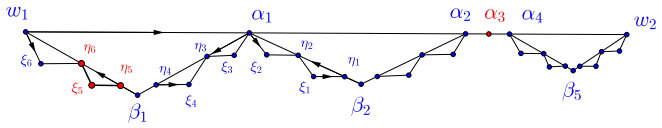


Figure 3.4 An example of the gadgets which are generated in a recursive manner. α_3 is an isolated partition point. The arrow of a segment indicates the order of two points in the pair. For example, the arrow from w_1 to w_2 indicates that (w_1, w_2) is a pair.

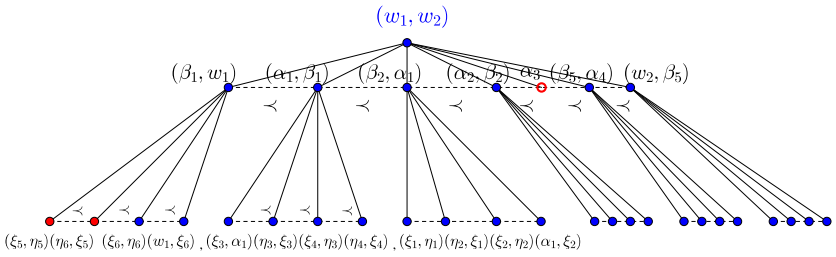


Figure 3.5 The recursion tree of our construction. Each node of the tree represents a pair (e.g., (β_1, ω_1)) or a point (e.g., α_3) in Figure 3.4. Pair (w_1, w_2) is the root at *level*-0. Any pair at *level*-$(i+1)$ is generated from a pair at *level*-i.

Consider a gadget $\mathsf{Gg}_\phi[\mathcal{A}_\phi, \mathcal{B}_\phi]$, 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.[①] 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. Now, we define the order of the child-pairs of pair (w_1, w_2), based on their distances to w_1. Here, the distance from a point w to a pair ϕ is the shortest distance from w to any point of ϕ.

Definition 3.5 (The Order of the Child-pairs): Consider a gadget $\mathsf{Gg}_{(w_1,w_2)}$. 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.

For example, in Figure A.6, $(\beta_2, \alpha_1) \prec (\alpha_5, \beta_5)$. We emphasize that the order of the child-pairs depends on the direction of their parent-pair.

3.2.2 The Recursion Tree

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. The recursion

① Suppose the polar angle of w_1w_2 is $-t\theta$. Note that $(2k+1)\theta = 2\pi$. Then, we can obtain that the polar angle of $\overrightarrow{\alpha_i\beta_i}$ is $(k-t+1)\theta$ and the polar angle of $\overrightarrow{\beta_i\alpha_{i-1}}$ is $(k-t)\theta$. Hence, $\overrightarrow{\alpha_i\beta_i}$ and $\overrightarrow{\beta_i\alpha_{i-1}}$ are cone boundaries.

can be naturally represented as a tree $\mathcal{T}$. Each node of the tree represents either a pair or a point. To avoid confusion, we use *point* to express a point in $\mathbb{R}^2$ and *node* to express a vertex in the tree. The pair (μ_1, μ_2) is the root of the tree (*level*-0). The child-pairs of (μ_1, μ_2) are the child-nodes of the root (they are at *level*-1). Recursively, each child-pair of a pair ϕ is a child-node of the node ϕ in $\mathcal{T}$. Besides, there are some *partition points* of the empty pieces (e.g., the point α_3 in Figure 3.4) which may not belong to any pair. We call it an *isolated* point. Let an isolated point be a leaf in $\mathcal{T}$ and the parent of such a point be its parent pair. For example, the parent of α_3 is the pair (ω_1, ω_2) in Figure 3.4. We provide the recursion tree in Figure 3.5, which corresponds to the points in Figure 3.4.

The nodes with the same parent are siblings. According to Definition A.11, we define a total order "$\prec$"of them. In tree $\mathcal{T}$, if "$\varphi \prec \phi$", we place φ to the left of ϕ, (e.g., (ξ_5, η_5) is at the left of (η_6, ξ_5) in Figure 3.5). However, "$\varphi \prec \phi$" does not mean that φ is on the left hand side of ϕ geometrically. For example, in Figure 3.4, pair $(\xi_5, \eta_5) \prec (\eta_6, \xi_5)$ in the tree $\mathcal{T}$, but in the Euclidean plane, point η_5 is on the right side of η_6.

For a pair ϕ (corresponding to a node in $\mathcal{T}$), we use $\mathcal{T}_\phi$ to denote the subtree rooted at ϕ (including ϕ), or all the points involved in the subtree.

Our counterexample $\mathcal{P}_m$ corresponds to a recursion tree with m levels. We have not yet specified how to choose the partition set for each gadget and decide which pieces are empty for each pair. We will do it in the next subsection. We note that we *do not* construct the tree level by level, but rather according to the DFS preorder.

3.2.3 The Construction

Now, we describe the process of generating the m-level recursion tree $\mathcal{T}$. See Figure A.7 for an example. We call a pair *a leaf-pair* if it is a leaf node in the tree and *an internal-pair* otherwise. W.l.o.g, we assume that the root of $\mathcal{T}$ is (μ_1, μ_2) and $\mu_1\mu_2$ is horizontal. The tree is generated according to the DFS preorder, starting from the root. When we are visiting an internal-pair, we generate its gadget. Note that generating its gadget is equivalent to generating its children in $\mathcal{T}$. We, however, do not visit those children immediately after their generation. They will be visited later according to the DFS preorder. Whether a pair is a leaf or not is determined as the gadget being created. Note that not all leaf-pairs are at *level-m*.

The process generating the gadget for an internal-pair includes two steps, which are called *projection* and *refinement*. We will explain the detail soon. We denote the

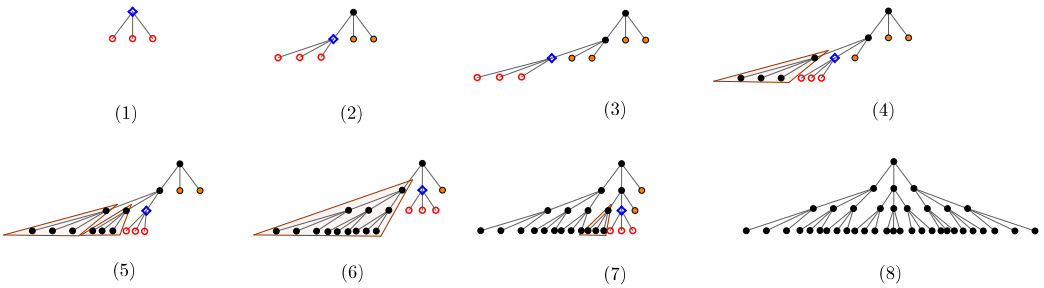


Figure 3.6 The process of generating a tree according to the DFS preorder. In each subfigure, ◇ represents a node we are visiting. The nodes generated in the step are denoted by ○. ● represents a node which has already been visited. ● represents a node which has been created but not visited yet. The nodes covered by light brown triangles are related to the projection process.

Algorithm 1: GenGadget(ϕ): Generate the Normal Points in $\mathcal{T}_\phi$

```
if φ is a leaf-pair then
    Return ;
else
    Gg_φ ← Proj-Refn(φ);
foreach child-pair φ of φ do
    GenGadget(φ) ;
```

procedure to construct the recursion tree $\mathcal{T}$ by GenGadget(ϕ) and the pseudocode can be found in Algorithm 1. We call the points generated by Algorithm 1 *normal points* and denote them by $\mathcal{P}^{\mathrm{n}}_m$ 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.[①] 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. See Figure 3.7 for an example.

Projection and Refinement: Let $\mathcal{T}_\phi$ be the set of points in the subtree rooted at ϕ. The

① d_0 depends on k, but on the number of points. We will determine the exact value of d_0 in Section 3.5.

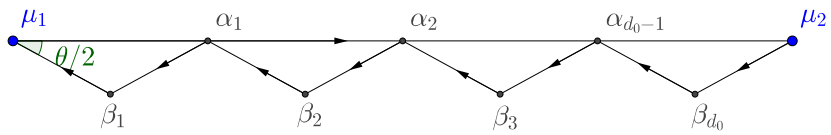


Figure 3.7 The root gadget $\mathsf{Gg}_\phi(\mathcal{A}_\phi, \mathcal{B}_\phi)$ where $\phi = (\mu_1, \mu_2)$. $\mu_1\mu_2$ is horizontal. $\mathcal{A}_\phi$ is the equidistant partition. Each piece is non-empty.

projection and refinement process for ϕ is slightly more complicated, as it depends on the subtrees $\mathcal{T}_\varphi$ rooted at the siblings φ of ϕ such that $\varphi \prec \phi$. Recall that when we visit a pair ϕ (and generate Gg_ϕ) in the tree $\mathcal{T}$ according to the DFS preorder, we have already visited all pairs $\varphi \prec \phi$.

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 3.3 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 3.6: 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.

– 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. We provide the property of the order here and prove it in the end of the section.

Property 3.7: 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 describe the projection operation for a pair $\phi \in \Phi$. W.l.o.g., suppose ϕ is an internal-pair since only internal-pairs have children. We define the first internal-pair in direction $\overrightarrow{\beta\alpha}$ as the *first internal-pair* of 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 3.8, 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

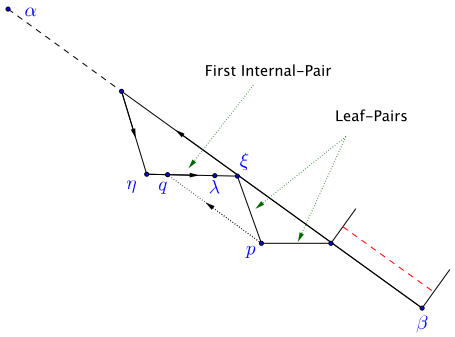


Figure 3.8 An example of the projection for the first internal-pair $\phi = (\eta, \xi)$. First, we add a point λ such that $|\lambda\xi| = \delta/d_0$ where δ is the length $|\eta\xi|$. Second, for each leaf-pair $\varphi \prec \phi$, project its apex point p to the segment of ϕ along the direction $\overrightarrow{\beta\alpha}$, i.e., add the point q in the figure.

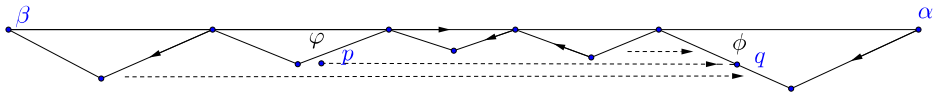


Figure 3.9 The projection for pair ϕ. Here, p is a point in subtree $\mathcal{T}_\varphi$. q is a projection point of p, i.e., the point on segment of pair ϕ such that pq is parallel to $\beta\alpha$. The set $\mathsf{Proj}[\bigcup_{\varphi\prec\phi,\varphi\in\Phi}\mathcal{T}_\varphi]$ consists of all projection points of $\bigcup_{\varphi\prec\phi,\varphi\in\Phi}\mathcal{T}_\varphi$ on segment of ϕ.

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 $\overrightarrow{\beta\alpha}$, [①] e.g., project p to q in Figure 3.8. Note that the length of leaf-pair φ is at least $\delta/2$ according to Property 3.6. Thus, there is no point between λ and ξ as long as $d_0 > 2$. Formally, we denote the operation by

$$\widehat{\mathcal{A}}_\phi \leftarrow \mathsf{Proj}\left[\bigcup\nolimits_{\varphi\prec\phi,\varphi\in\Phi}\mathcal{T}_\varphi\right] \cup \lambda. \tag{3-1}$$

- 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. See Figure 3.9 for an example. Formally, we denote the operation by

$$\widehat{\mathcal{A}}_\phi \leftarrow \mathsf{Proj}\left[\bigcup\nolimits_{\varphi\prec\phi,\varphi\in\Phi}\mathcal{T}_\varphi\right]. \tag{3-2}$$

① If the projected point falls outside the segment of ϕ, we do not need to add a normal point.

– Refinement: After the projection, we obtain a candidate partition set $\widehat{\mathcal{A}}_\phi$ of ϕ (defined in (3-1) and (3-2)). However, note that the length between the pieces may differ a lot. In order to maintain Property 3.6, we add some other points to ensure that all non-empty pieces of ϕ have approximately the same length. We call this process the *refinement* operation.

W.l.o.g., suppose pair ϕ has unit length and $|\widehat{\mathcal{A}}_\phi| = n$ and $n > d_0$. ①

Suppose $\phi = (u_1, u_2)$. We distinguish into 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 3.10(a) 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 3.10(b) 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_2.② Then $|\beta\xi_1| = |\beta\xi_2|$ and $|\alpha_1\eta_1| = |\alpha_2\eta_2|$ after the refinement. See Figure 3.10 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 pieces③, for each other 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

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

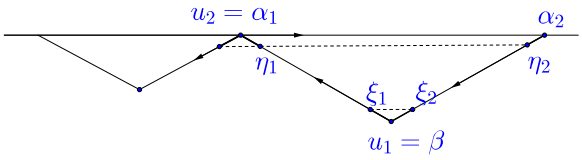
The number of points added in the refinement process is at most $O(n^2)$ since the segment of pair ϕ has unit length and $\delta_o \geq 1/n^2$. We call each piece whose length is less than δ_o a *short piece*. The short pieces remain unchanged before and after the refinement. Moreover, the refinement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

Deciding Emptiness, Leaf-Pairs and Internal-Pairs: Next, we discuss the principle

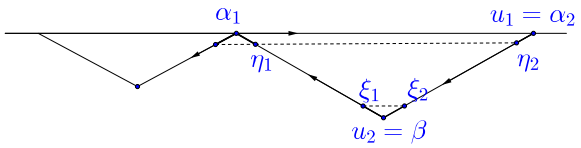
① If $n \leq d_0$, we repeatedly split the inner pieces (i.e., all pieces except for the two pieces incident on the points of ϕ) into two equal-length pieces until the number of the points in $\widehat{\mathcal{A}}_\phi$ is larger than d_0.

② Point ξ_1 must exist since ξ_2 is a projected point and there is no point in the marked piece $\xi_2\beta$.

③ Keeping the marked pieces unchanged maintains Property 3.8 and helps a lot to decompose the normal points into hinge sets. See Section 3.3 for details.



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



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

Figure 3.10 The two cases for refinement. The first case is $\phi = (\beta, \alpha_1)$ in which the first point is an apex point. We mark the piece incident on α_1, i.e., the piece $\alpha_1\eta_1$. The second case is $\phi = (\alpha_2, \beta)$, in which the first point is a partition point. We mark the two pieces incident on α_2 and β respectively, i.e., the pieces $\alpha_2\eta_2$ and $\xi_2\beta$. Note that after refinement, $|\beta\xi_1| = |\beta\xi_2|$ and $|\alpha_1\eta_1| = |\alpha_2\eta_2|$ since there is no point added on the marked pieces after refinement.

to decide whether a piece is empty or non-empty. See Figure 3.11 for an illustration. Consider a pair ϕ whose apex point is β and partition point is α.[①] 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. The apex set $\mathcal{B}_\phi$ induces the set Φ of child-pairs of ϕ. The types of these pairs are determined as follows. 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*. Naturally, there is no leaf-pair between any two internal-pairs among pairs in Φ. Hence, Property 3.7 maintains in the construction. Further, we can see that each other normal point belongs to at most two pairs, except for the two points in the root pair.

For convenience, we call the piece generating two leaf-pairs *a near-empty piece* and generating one leaf-pair and one internal-pair *a half-empty piece*. Note that the near-empty and half-empty pieces are special non-empty pieces. We can see that, except for the near-empty piece incident on the partition point in ϕ (e.g., $\mu_1\xi_{d_0}$ in Figure 3.11), the maximum length among the non-empty pieces is at most twice longer than the minimum one according to refinement.

① 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.

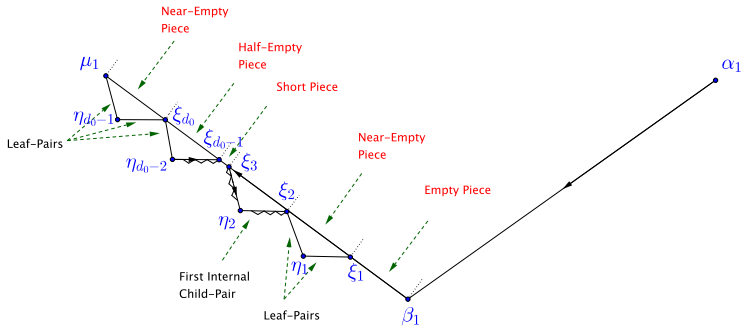


Figure 3.11 The figure illustrates the emptiness of each piece. Consider the pair (β_1, μ_1) with partition points after refinement. Segment $\mu_1\xi_{d_0}$ and $\xi_1\xi_2$ are two near-empty pieces and $\beta_1\xi_1$ is an empty piece. Pair (μ_1, η_{d_0-1}) and $(\eta_{d_0-1}, \xi_{d_0})$, $(\xi_{d_0}, \eta_{d_0-2})$, (ξ_2, η_1), (η_1, ξ_1) are the five leaf-pairs. $\xi_{d_0-1}\xi_{d_0}$ is the half-empty piece. Pair (η_2, ξ_2) is the first internal-pair.

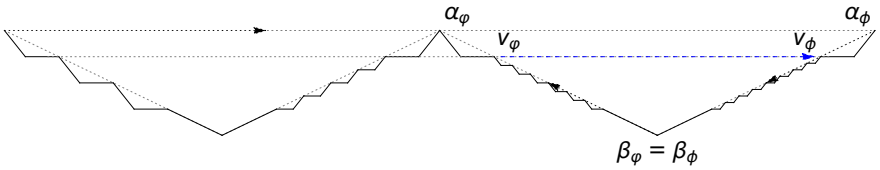


Figure 3.12 The figure illustrates Property 3.8. After projection and refinement, $|\alpha_\phi v_\phi| = |\alpha_\varphi v_\varphi|$.

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

$$\mathsf{Gg}_\phi \leftarrow \mathsf{Proj\text{-}Refn}(\phi). \tag{3-4}$$

Property 3.8: Consider two sibling pairs ϕ and φ. Suppose both of them are internal-pairs and have partition point sets $\mathcal{A}_\phi$ and $\mathcal{A}_\varphi$ respectively. Suppose $\alpha_\phi \in \phi$ and $\alpha_\varphi \in \varphi$, and both α_ϕ and α_φ are partition points. The point in $\mathcal{A}_\phi$ closest to α_ϕ is v_ϕ. Meanwhile, the point in $\mathcal{A}_\varphi$ closest to α_φ is v_φ. Then $|\alpha_\phi v_\phi| = |\alpha_\varphi v_\varphi|$. See Figure 3.12 for an example.

Proof W.l.o.g., we prove that any two adjacent siblings satisfy the property. Suppose ϕ and φ are adjacent siblings. W.l.o.g., assume $\varphi \prec \phi$. ϕ has the candidate partition set $\widehat{\mathcal{A}}_\phi$ after projection. Suppose the point in $\widehat{\mathcal{A}}_\phi$ closest to α_ϕ is $\hat{v}_\phi$. According to the projection, we know $|\alpha_\phi \hat{v}_\phi| = |\alpha_\varphi v_\varphi|$. Since we do not add any new point between α_ϕ $\hat{v}_\phi$ after refinement, $\hat{v}_\phi$ and v_ϕ are the same point. Hence, any two adjacent siblings have the property. □

Corollary 3.9: Consider a pair ϕ with partition point set $\mathcal{A}_\phi$. Suppose $\alpha \in \phi$ and α is a partition point. Among all pieces determined by $\mathcal{A}_\phi$, the piece incident on α has the maximum length.

Proof It is not difficult to check the root pair holds the property. Then, consider a pair $\hat{\phi}$ with its child pair set $\hat{\Phi}$. We prove that any pair in $\hat{\Phi}$ holds the property when $\hat{\phi}$ holds the property. Suppose ϕ is the first internal-pair in $\hat{\Phi}$, the corollary is trivially true for ϕ according to the projection process (the first projection case). Otherwise, according to Property 3.8 and the projection process, we know that for any ϕ in $\hat{\Phi}$, the piece incident on the partition point in ϕ has the same length. Thus, any pair in $\hat{\Phi}$ holds the property. □

Now, we prove Property 3.6 that we claimed at the beginning of the construction.

Proof of Property 3.6: Consider an arbitrary pair with partition point α and the set Φ of its child-pairs. Consider an internal-pair $\phi \in \Phi$. Note that the length of a child-pair is determined by its corresponding partition piece. According to the construction, except for the near-empty piece incident on α, the length of any non-empty piece is at most twice and at least half the length of another one. Thus, except for the sibling pairs generated by the piece incident on α, the length of any pair $\varphi \prec \phi$ is at least half of the length of ϕ. Finally, the piece incident on α only induces two leaf-pairs and has the maximum length among other empty pieces of ϕ according to Corollary 3.9. Hence, we have proven the property. ◄

Finally, we summarize the properties of half-empty, near-empty, and empty pieces below.

Property 3.10: Consider an internal-pair ϕ with partition point set $\mathcal{A}_\phi$. Suppose the length of ϕ is δ. The pieces determined by $\mathcal{A}_\phi$ have the following properties.

- The sum of lengths of empty pieces is less than $2\delta/d_0$.
- There are two near-empty pieces with sum of lengths less than $3\delta/d_0$.
- There is one half-empty piece with length less than δ/d_0.
- The sum of lengths of empty, near-empty and half-empty pieces is less than $6\delta/d_0$.

Proof Consider the first property. Suppose $\beta \in \phi$ and β is an apex point. There are two kinds of empty pieces. One is the short pieces and the other is a piece incident on β (denoted by $\xi\beta$). First, the sum of lengths of the short pieces is less than δ/d_0. Because the length of each short piece is less than δ/n^2 and there are less than n short pieces where $n > d_0$ is the number of partition points in $\widehat{\mathcal{A}}$ after projection and before refinement. On

the other hand, we prove that the length of $\xi\beta$ is less than δ/d_0. If β is the first point in ϕ (refer to $\phi = (\beta, \alpha_1)$ in Figure 3.10), according to refinement (the first refinement case), the length of $\xi\beta$ is less than δ/d_0. Next consider the case that β is the second point in $\phi = (\alpha_1, \beta)$ in Figure 3.10). Suppose ϕ shares the point β with its sibling φ. Hence, ϕ and φ share the point β and β is the first point in φ. Denote the piece of φ incident on β by $\eta\beta$. In this case, we have that ϕ and φ have the same length and $|\xi\beta| = |\eta\beta|$. Since β is the first point in φ, we have proven that $|\eta\beta| \leq \delta/d_0$. Thus, $|\xi\beta| \leq \delta/d_0$.

Consider the second property. Suppose α is a partition point and β is an apex point, and $\alpha, \beta \in \phi$. First, consider the near-empty piece incident on α. If ϕ is the first internal-pair of its parent, according to projection, we add a point λ on the segment of ϕ such that $|\lambda\alpha| = \delta/d_0$. Otherwise, according to Property 3.6 and 3.8, we know the piece incident on α is at most $2\delta/d_0$. Second, we consider the other near-empty piece closer to β. Its length is no more than δ/d_0 based on refinement. Thus, the sum of lengths of near-empty pieces is less than $3\delta/d_0$.

For the third property, through refinement, the length of half-empty piece is less than δ/d_0. Above all, we get the fourth property. □

3.3 Hinge Set Decomposition of the Normal Points

All points introduced so far are referred to as normal points and their positions have been defined exactly. Recall that we denote the set of normal points by $\mathcal{P}_m^{\mathrm{n}}$. In this section, decompose $\mathcal{P}_m^{\mathrm{n}}$ into a collection of sets of points such that each normal point exactly belongs to one set. We call these sets *hinge sets*. See Figure A.5 for an overview of the hinge set decomposition.

Based on the hinge sets, the edges among normal points in $\mathsf{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$ can be organized in the clear way. For convenience, we regard the Yao-Yao graph as a directed graph. Recall the construction of the directed Yao-Yao graph in Algorithm 2. Note that $C_u(\gamma_1, \gamma_2]$ represents the cone with apex u and consisting of the rays with polar angles in the half-open interval $(\gamma_1, \gamma_2]$ in counterclockwise. We call the first iteration (line 2 to 5) the *Yao-step* and call the second iteration (line 6 to 9) the *Reverse-Yao step*.

Then, we define a *total order* among hinge sets. We call an edge in the Yao-Yao graph *a hinge connection* which connects any two points in the same hinge set or in two adjacent hinge sets w.r.t. the total order. Call other edges *long range connections*. In Section 3.4, we prove that we can break all long range connections without introducing new ones by

Algorithm 2: Construct the Yao-Yao graph

Data: A point set $\mathcal{P}$ and an integer $k \geq 2$

Result: $\mathsf{YY}_{2k+1}(\mathcal{P})$

Initialize: $\theta = 2\pi/(2k+1)$ and two empty graphs Y_{2k+1} and YY_{2k+1} ;
foreach *point* u *in* $\mathcal{P}$ **do**
  foreach j *in* $[0, 2k]$ **do**
    Select v in $C_u(j\theta, (j+1)\theta]$ such that $|uv|$ is the shortest ;
    Add edge $\overrightarrow{uv}$ into Y_{2k+1} ;
foreach *point* u *in* $\mathcal{P}$ **do**
  foreach j *in* $[0, 2k]$ **do**
    Select v in $C_u(j\theta, (j+1)\theta]$, $\overrightarrow{vu} \in \mathsf{Y}_{2k+1}$ such that $|uv|$ is the shortest ;
    Add edge $\overrightarrow{vu}$ into YY_{2k+1} ;
return YY_{2k+1} ;

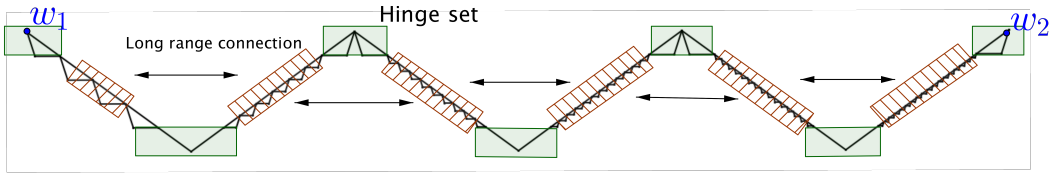


Figure 3.13 The overview of hinge set decomposition. Roughly speaking, each set of points covered by a green rectangle □ is a hinge set. Recursively, we can further decompose the points covered by shadowed rectangle ▨ into hinge sets. The hinge connections are the edges between any two points in a hinge set or between two adjacent hinge sets. The other edges in the Yao-Yao graph are long range connections.

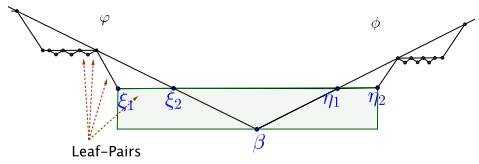
.

adding some auxiliary points. In Section 3.5, we show that, in the graph with only hinge connections, the shortest path between the two points of the root pair approaches infinity.

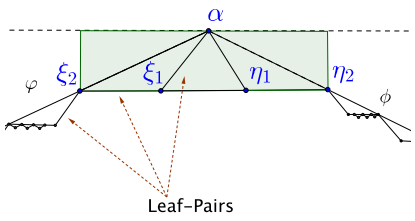
3.3.1 Hinge Set Decomposition

We discuss the process to decompose the set $\mathcal{P}_m^{\mathrm{n}}$ into *hinge sets* such that each point in $\mathcal{P}_m^{\mathrm{n}}$ belongs to exactly one hinge set. Briefly speaking, each hinge set is a set of points which are close geometrically.

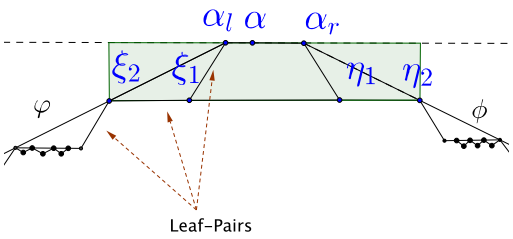
Consider a pair $\hat{\phi}$ at *level-l* ($l < m-1$) with partition point set $\mathcal{A}_{\hat{\phi}}$ and apex point set $\mathcal{B}_{\hat{\phi}}$. Denote the set of the child-pairs of $\hat{\phi}$ by $\hat{\Phi}$. Recall that we say a point u belongs to ϕ



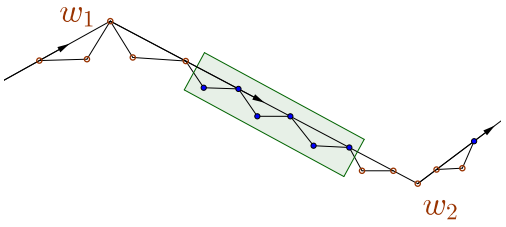
(a) The hinge set centered on an apex point



(b) The hinge set centered on a partition point



(c) Merge two hinge sets to a new one



(d) The hinge set consisting of the leaf-pairs at *level-m*. (w_1, w_2) is a pair at *level-*$(m-1)$

Figure 3.14 The hinge sets centered on a point in an internal-pair.

(i.e., $u \in \phi$), if the ϕ is $(u, \cdot)$ or $(\cdot, u)$. Just for convenient to describe, we call some point *center* of a hinge set and other points *affiliated point*. Formally, the hinge sets are defined as follows.

- The hinge set centered on a point $\beta \in \mathcal{B}_{\hat{\phi}}$ such that β belongs to one or two internal-pairs in $\hat{\Phi}$:① We denote the two internal-pairs by φ and ϕ.② See Figure 3.14(a) for an illustration. The hinge set centered at β includes: β itself, the child-pair of φ closest to β (denote the pair by (ξ_1, ξ_2)) and the child-pair of ϕ closest to β (denoted the pair by (η_1, η_2)). $\xi_1, \xi_2, \eta_1, \eta_2$ are affiliated points. According to the way to determine the leaf-pairs (see Section A), they only belong to leaf-pairs.
- The hinge set centered on a point $\alpha \in \mathcal{A}_{\hat{\phi}}$ such that α belongs to one or two internal-pairs in $\hat{\Phi}$, or α is an isolated partition point:
  - First, suppose α belongs to one or two internal-pairs in $\hat{\Phi}$, which we denote as φ and ϕ.③ See Figure 3.14(b). The hinge set centered on α includes: α itself, the two child-pairs closest to α of φ and ϕ (denote the pairs by (ξ_2, ξ_1)

① β must belong to two child-pairs of $\hat{\phi}$ since each β induces two pairs. However, β may belong to two leaf-pairs (i.e., do not belong to any internal-pair). In this case, β is affiliated to a hinge set centered on other point.

② If one of the two child-pairs is a leaf-pair, let $\varphi = \emptyset$.

③ If α belongs to only one internal-pair of $\hat{\Phi}$, let $\varphi = \emptyset$.

and (η_1, η_2)) respectively. $\xi_1, \xi_2, \eta_1, \eta_2$ are affiliated points which only belong to leaf-pairs.

– Second, α is an isolated point in $\mathcal{A}_{\mathcal{T}}$, i.e., α is an end point of a short piece and does not belong to any internal-pair in $\hat{\Phi}$. See Figure 3.14(c). Then, for each direction of segment of $\hat{\phi}$, we find the closest non-isolated point in $\mathcal{A}_{\hat{\phi}}$. Denote them by α_l and α_r. Merge the two hinge sets centered on α_l and α_r as a new one and add α to the new hinge set.

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. ① See Figure 3.14(d).

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

Lemma 3.11: Each point p in $\mathcal{P}_m^{\mathrm{n}}$ belongs to exactly one hinge set.

Proof First, we prove that any two hinge sets are not overlapping. It means that any point in a hinge set does not belong to any other hinge set. First, consider a point λ which only belongs to a leaf-pairs i.e., an affiliated point in the first two type hinge sets (see $\xi_1, \xi_2, \eta_1, \eta_2$ in Figure 3.14(a) 3.14(b) 3.14(c)) or a point in a third type hinge set. It has unique parent-pair ϕ such that $\lambda \in \mathsf{Gg}_\phi$. Let $\phi = (\alpha, \beta)$. If φ is the closest child-pair to α, λ belongs to the hinge set centered on α. Or if φ is the closest child-pair to β, λ belongs to the hinge set centered on β. Otherwise, λ belongs to a third type hinge set. It is not difficult to check that the three cases do non overlap. Thus, point λ belongs to at most one hinge set. Next, consider a point λ which belongs to some internal-pair or is an isolated partition point. Then, λ can only belong to the first two type hinge sets. λ cannot be a an affiliated point for any hinge set since affiliated point only belongs to leaf-pairs. Besides, according to the definition, the hinge set centered on λ is unique. Therefore, λ belongs to at most one hinge set.

On the other hand, we prove that each point in $\mathcal{P}_m^{\mathrm{n}}$ belongs to at least one hinge set. First, any point of *level-m* belongs to a hinge set according to the third case. Second, consider a point λ on *level-*l, $l < m$. If λ belong to any internal-pair, it should be a center

① Although these points form the leaf-pairs at *level-m*, these leaf-pairs are the "candidate internal-pairs" to generate the points at *level-*$(m+1)$.

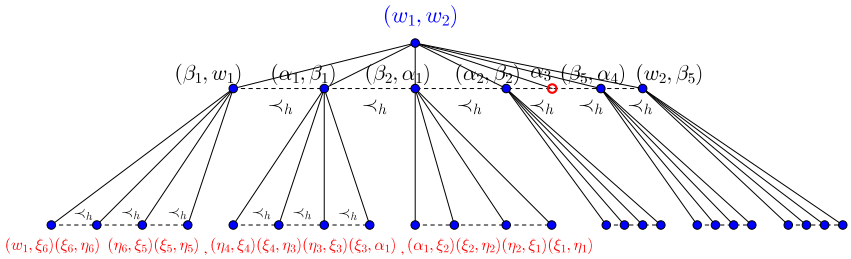


Figure 3.15 The illustration for order $\prec_h$. In the example, notice that the order of the *level*-2 in $\mathcal{T}^R$ is different with the order in $\mathcal{T}$ (see Figure 3.5).[①]

of a hinge set. If λ is an isolated partition point, it merges two hinge sets and belongs to the new hinge set. If λ only belongs to a leaf-pair φ and φ is a child-pair of $\phi = (\alpha, \beta)$, then, based on the way to determine the leaf-pairs, λ belongs to hinge set centered on α or β.

Overall, each point in $\mathcal{P}_m^{\mathrm{n}}$ belongs to exactly one hinge set. □

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 $\prec_h$ below. For comparison, in Figure 3.15, we reorganize the tree in Figure 3.5 according to the order $\prec_h$.

First, consider the root pair (μ_1, μ_2). We denote the hinge set centered on μ_1 by Λ_{μ_1} and denote the hinge set centered on μ_2 by Λ_{μ_2}. Define Λ_{μ_1} as the first hinge set and Λ_{μ_2} as the last hinge set w.r.t. $\prec_h$. Then, $\Lambda_{\mu_1} \prec_h \Lambda_{\mu_2}$.

Second, we define the orders of other hinge sets. Consider an internal-pair ϕ with parent pair (w_1, w_2) (or (w_2, w_1)) and $\Lambda_{w_1} \prec_h \Lambda_{w_2}$. Note that there are two hinge sets centered on the points in ϕ respectively. We call the one closer to (in Euclidean distance) Λ_{w_1} the *former hinge set* of ϕ, denoted by $\Lambda_\phi^{(-)}$. Call the other the *latter hinge set* of ϕ, denoted by $\Lambda_\phi^{(+)}$. Let $\Lambda_{w_1} \prec_h \Lambda_\phi^{(-)} \prec_h \Lambda_\phi^{(+)} \prec_h \Lambda_{w_2}$. Besides, recall that for any internal-pair ϕ at *level*-$(m-1)$, the points in $\mathcal{A}_\phi \cup \mathcal{B}_\phi$ but not in $\Lambda_\phi^{(-)} \cup \Lambda_\phi^{(+)}$ also form a hinge set. We denote it by Λ_ϕ and define $\Lambda_\phi^{(-)} \prec_h \Lambda_\phi \prec_h \Lambda_\phi^{(+)}$.

Note that we have organized all pairs in the recursion tree $\mathcal{T}$. We can transform the tree consisting of all **internal nodes** of $\mathcal{T}$ to a topological equivalent tree $\mathcal{T}^R$ which has a different ordering of the nodes. The order of the sibling pairs in $\mathcal{T}^R$ is determined by their

① Note that $\mathcal{T}^R$ only contains the internal nodes of $\mathcal{T}$. Consider that *level*-2 nodes still have their child nodes. Thus, we do not remove the pairs on *level*-2 in the example.

Algorithm 3: TravelHinge(ϕ): Travel the hinge sets in the tree $\mathcal{T}_\phi^R$

```
1 Visit(Λ_φ^(−)) ;
2 if φ is at level-l (l < m − 1) then
3 |   foreach child-pair ϕ of φ in T_φ^R do
4 |   |   TravelHinge(ϕ) ;
5 else
6 |   Visit(Λ_φ)
7 Visit(Λ_φ^(+)) ;
```

Euclidean distances to the former hinge set of their parent. Overall, the ordering $\prec_h$ of the hinge sets can be defined by a DFS traversing of $\mathcal{T}^R$. When we reach a pair ϕ at *level-l* ($l < m-1$) for the first time①, we visit its former hinge set $\Lambda_\phi^{(-)}$. Next, we recursively traverse its child-pairs in the order we just defined. Then we return to the pair and visit its latter hinge set $\Lambda_\phi^{(+)}$. When we reach a pair ϕ at *level-*$(m-1)$, we visit $\Lambda_\phi^{(-)}, \Lambda_\phi, \Lambda_\phi^{(+)}$ in order and return.② We denote the procedure by TravelHinge(ϕ) and the pseudocode can be found in Algorithm 3.

3.3.2 Long Range Connection

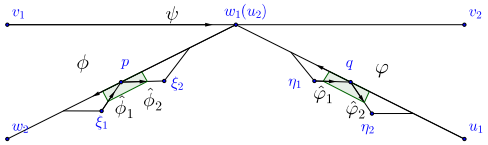
We call the edges connecting two non-adjacent hinge sets *long range connections*.

Definition 3.12 (Long range connection)： A *long range connection* is an edge connecting two points in two non-adjacent hinge sets.

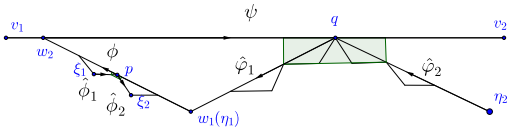
If there is no long range connection, the total order of the hinge sets corresponds to the ordering of the shortest path from μ_1 to μ_2 in the final construction. It means that each hinge set has at least one point on the shortest path between μ_1 and μ_2 and the order of these points is consistent with $\prec_h$. However, there indeed exist long range connections among normal points. In order to achieve the above purpose, we should break the edges connecting two non-adjacent hinge sets. Fortunately, the long range connections in $\mathcal{P}_m^{\mathrm{n}}$ have relatively simple form. We claim that after introducing some auxiliary points (in Section 3.4), we can cut the long range connections without introducing any new long

① *level-*$(m-1)$ is the second to last level of $\mathcal{T}$ and the last level of $\mathcal{T}^R$.

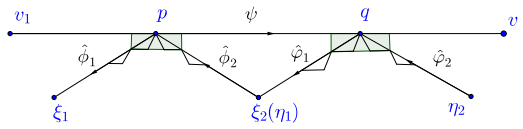
② Note that two adjacent sibling pairs share the same hinge set. So the same hinge set may be visited twice, and the two visits are adjacent in the total order. So it does not affect the order between two distinct hinge sets.



(a) $\psi = (v_1, v_2)$, $\phi = (w_1, w_2)$, $\varphi = (u_1, u_2)$, $\hat{\varphi}_1 = (\eta_1, q)$, $\hat{\varphi}_2 = (q, \eta_2)$, $\hat{\phi}_1 = (\xi_1, p)$, $\hat{\phi}_2 = (p, \xi_2)$, w_1 may equal to u_2

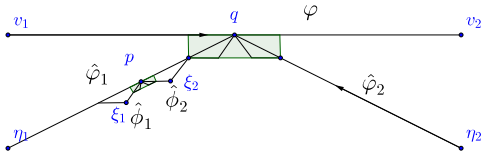


(b) $\psi = \varphi = (v_1, v_2)$, $\phi = (w_1, w_2)$, $\hat{\varphi}_1 = (q, \eta_1)$, $\hat{\varphi}_2 = (\eta_2, q)$, $\hat{\phi}_1 = (\xi_1, p)$, $\hat{\phi}_2 = (p, \xi_2)$, w_1 may equal to η_1



(c)

$\psi = \phi = \varphi = (v_1, v_2)$, $\hat{\varphi}_1 = (q, \eta_1)$, $\hat{\varphi}_2 = (\eta_2, q)$, $\hat{\phi}_1 = (p, \xi_1)$, $\hat{\phi}_2 = (\xi_2, p)$, ξ_2 may equal to η_1



(d) $\varphi = (v_1, v_2)$, $\hat{\varphi}_1 = (q, \eta_1)$, $\hat{\varphi}_2 = (\eta_2, q)$, $\hat{\phi}_1 = (\xi_1, p)$, $\hat{\phi}_2 = (p, \xi_2)$

Figure 3.16 The possible relative positions of p and q. Although, in the figure, p and q are partition points, it does not induce any new case when p or q is an apex point according to our divided condition in the proof.

range connections. Hence, only adjacent hinge sets in the above order $\prec_h$ have edges in the Yao-Yao graph.

Now, we examine the long range connections in $\mathrm{YY}_{2k+1}(\mathcal{P}_m^n)$. First, we show that we only need to consider the long range connections between the points in $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ for any two sibling pairs ϕ and φ. Recall that $\mathcal{T}_\phi$ denotes the subtree rooted at ϕ (including ϕ). If there exist two points $p \in \mathcal{T}_\phi - \mathcal{T}_\phi \cap \mathcal{T}_\varphi$ and $q \in \mathcal{T}_\varphi - \mathcal{T}_\phi \cap \mathcal{T}_\varphi$ such that pq is a long range connection, we say there is a long range connection between $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$.

Claim 3.13: Suppose that 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.

Proof Consider two non-adjacent hinge sets Λ_p and Λ_q (if two hinge sets are adjacent, the edges between them are hinge connections) and $\lambda_1 \in \Lambda_p$ and $\lambda_2 \in \Lambda_q$. We prove that we can find two sibling pairs ϕ and φ such that $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ contains λ_1 and λ_2 respectively. Then, we consider the possible cases about the two non-adjacent hinge sets.

First, we consider the case that each of the two hinge sets is centered on a point of some internal-pair. Denote the two center points by p and q and the two hinge sets by Λ_p

and Λ_q. p belongs to one or two adjacent internal-pairs. W.l.o.g., suppose they are $\hat{\phi}_1$ and $\hat{\phi}_2$ ($\hat{\phi}_2 = \emptyset$ if p only belongs to one internal-pair). Meanwhile, q belongs to one or two adjacent internal-pairs. Suppose they are $\hat{\varphi}_1$ and $\hat{\varphi}_2$. W.l.o.g., suppose the level of $\hat{\varphi}_1$ and $\hat{\varphi}_2$ is no more than the level of $\hat{\phi}_1$ and $\hat{\phi}_2$. Then, we distinguish two cases. In the first one, none of $\hat{\varphi}_1$ and $\hat{\varphi}_2$ is an ancestor of $\hat{\phi}_1$ and $\hat{\phi}_2$. Otherwise, it is the second case.

Consider the first case. Suppose the closest common ancestor of $\hat{\varphi}_1$, $\hat{\varphi}_2$, $\hat{\phi}_1$ and $\hat{\phi}_2$ is pair ψ in $\mathcal{T}$. If p and q do not belong to $\mathcal{A}_\psi \cup \mathcal{B}_\psi$, there are two different child-pairs ϕ and φ of ψ (see Figure 3.16(a)), such that Λ_p belongs to $\mathcal{T}_\phi$ and Λ_q belongs to $\mathcal{T}_\varphi$. Since points between $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ have no long range connection according to the assumption, points between Λ_p and Λ_q have no long range connection. Then consider the case that q belongs to $\mathcal{A}_\psi \cup \mathcal{B}_\psi$ (see Figure 3.16(b)). Note that Λ_q is a subset of $\mathcal{T}_{\hat{\varphi}_1} \cup \mathcal{T}_{\hat{\varphi}_2}$. Because there is no long range connections for the points between $\mathcal{T}_{\hat{\varphi}_1}, \mathcal{T}_{\hat{\varphi}_2}$ and $\mathcal{T}_\phi$, Λ_p and Λ_q have no long range connection. Finally, if both p and q belong to $\mathcal{A}_\psi \cup \mathcal{B}_\psi$ (see Figure 3.16(c)), since there is no long range connection for points between $\mathcal{T}_{\hat{\phi}_1}, \mathcal{T}_{\hat{\phi}_2}$ and $\mathcal{T}_{\hat{\varphi}_1}, \mathcal{T}_{\hat{\varphi}_2}$, Λ_p and Λ_q have no long range connection.

Consider the second case. See Figure 3.16(d). W.l.o.g., suppose $\hat{\phi}_1$ and $\hat{\phi}_2$ are in the subtree of $\mathcal{T}_{\hat{\varphi}_1}$. Λ_q is a subset of $\mathcal{T}_{\hat{\varphi}_1} \cup \mathcal{T}_{\hat{\varphi}_2}$. Moreover, according to the assumption, the points between $\mathcal{T}_{\hat{\varphi}_1}$ and $\mathcal{T}_{\hat{\varphi}_2}$ have no long rang connections. Since $\hat{\phi}_1$ and $\hat{\phi}_2$ are in the subtree of $\mathcal{T}_{\hat{\varphi}_1}$, we know that the points in $\Lambda_q \cap \mathcal{T}_{\hat{\varphi}_2}$ have no long range connections to Λ_p. Then we consider the long range connection between $\Lambda_q \cap \mathcal{T}_{\hat{\varphi}_1}$ and Λ_p. Actually, it is reduced to the first case, thus they have no long range connection.

Above all, we have discussed the case that each of the two hinge sets is centered on a point of some internal-pair. Next, suppose that at least one of the hinge sets is a third type hinge set which contains only leaf-pairs at *level-m*. If both of them are the third type hinge sets, denoted by Λ_ϕ and Λ_φ, there must exist two sibling pairs $\hat{\phi}$ and $\hat{\varphi}$ such that $\Lambda_\phi \in \mathcal{T}_{\hat{\phi}}$ and $\Lambda_\varphi \in \mathcal{T}_{\hat{\varphi}}$. According to the hypothesis that there is no long range connection between the points in $\mathcal{T}_{\hat{\phi}}$ and $\mathcal{T}_{\hat{\varphi}}$, there is no long range connection between Λ_ϕ and Λ_φ. Finally, consider the case that there is only one third type hinge set, denoted by Λ_ϕ and the other is centered on q, denoted by Λ_q. Suppose q is the shared point of $\hat{\varphi}_2$ and $\hat{\varphi}_1$. We distinguish two cases according to whether $\phi \in \mathcal{T}_{\hat{\varphi}_1} \cup \mathcal{T}_{\hat{\varphi}_2}$ or not. As we have discussed above, we can prove that there is no long range connection between Λ_q and Λ_ϕ. □

According to Claim 3.13, next, we discuss the possible long range connections between $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ for two sibling pairs ϕ and φ. Suppose p belongs to $\mathcal{T}_\phi$ and q belongs

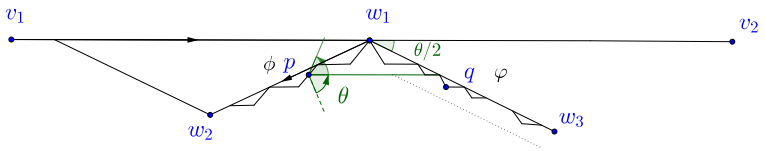


Figure 3.17 The points of $\mathcal{T}_\varphi$ locate in at most two cones of p.

to $\mathcal{T}_\varphi$. In the following, we prove that if the directed edge $\overrightarrow{pq}$ is an edge in $\mathrm{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$, then $\phi \prec \varphi$. Moreover, note that the points of $\mathcal{T}_\varphi$ locate in at most two cones of p. See Figure 3.17 for an illustration. $p \in \mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ locates in two cones of p according to the angular relation. We prove that for each point p, only one of the two cones may contain a long range connection. Intuitively, these properties (Observation 3.14, Lemma 3.15 and 3.16) result from Property A.4 which does not hold for even Yao-Yao graphs.

We prove the properties formally below. Consider two sibling pairs $\phi \prec \varphi$. First, suppose φ is a leaf-pair (see Observation 3.14). Second, we consider that φ is an internal-pair (see Lemma 3.15 and 3.16).

Observation 3.14: Consider two sibling pairs ϕ and φ such that $\phi \prec \varphi$. If φ is a leaf-pair, there is no long range connection between $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ (i.e., φ itself).

Proof See Figure 3.18 for an illustration. Suppose $\varphi = (w_3, w_1)$. First, we consider the case that ϕ and φ share one point. Let $\phi = (w_1, w_2)$. Then, $\mathcal{T}_\phi \cap \mathcal{T}_\varphi = w_1$. We should prove that for any $p \in \mathcal{T}_\phi - w_1$ (i.e., $p = p^{(1)}$ in Figure 3.18), there is no edge pw_3 in Yao-Yao graph. Let (η_1, η_2) be the pair in Gg_ϕ closest to w_1. There is no edge $\overrightarrow{w_3p}$ in the Yao-step since η_2 and p are in the same cone of w_3 and $|\eta w_3| < |pw_3|$. Note that $\eta_2 w_3$ is a hinge connection. If directed edge $\overrightarrow{pw_3}$ is accepted in the Yao-step, $\overrightarrow{pw_3}$ cannot be accepted in the reverse-Yao step since $\overrightarrow{\eta_2 w_3}$ exists in the Yao-step, and η_2 and p are in the same cone of w_3 and $|\eta_2 w_3| < |pw_3|$. Thus, there is no long range connection between $\mathcal{T}_\phi$ and φ.

Second, we consider the that ϕ and φ do not overlap, i.e., Then, $\mathcal{T}_\phi \cap \mathcal{T}_\varphi = \emptyset$. W.l.o.g., let $\phi = (w_2, w_4)$ and $p \in \phi$ (i.e., $p = p^{(2)}$). Similar to the first case, we can prove that there is no long range connection pw_3. Then we consider the point w_1. There is edge from w_1 to p in the Yao-step since η_1 and p are in the same cone of w_1 and $|w_1\eta_1| < |w_1p|$. Besides, if directed edge $\overrightarrow{pw_1}$ is accepted in the Yao-step, $\overrightarrow{pw_1}$ cannot be accepted in the reverse-Yao step since $\overrightarrow{\eta_1 w_1}$ exists in the Yao-step and $|\eta_1 w_1| < |pw_1|$. □

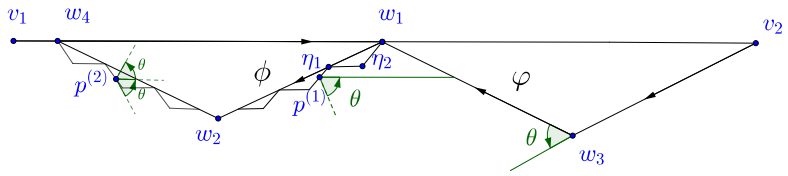


Figure 3.18 ϕ and φ are sibling pairs such that $\phi \prec \varphi$. φ is a leaf-pair. There is no long range connection between $\mathcal{T}_\phi$ and the points of $\mathcal{T}_\varphi$ (i.e., φ itself).

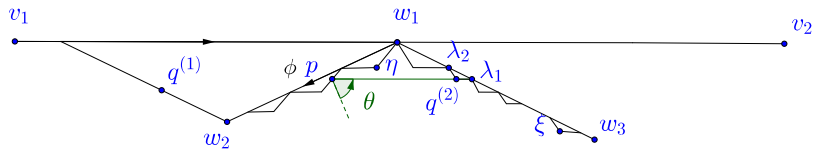
Given a pair (v_1, v_2) with child-pair set Φ, consider two sibling pairs ϕ and φ in Φ where $\phi = (w_1, w_2)$. For convenience, let $\angle u_1u_2$ be the polar angle of vector u_1u_2. Let $\angle(u_1u_2, v_1v_2)$ be $\angle v_1v_2 - \angle u_1u_2$, i.e., the angle from u_1u_2 to v_1v_2 in the counterclockwise direction.

Recall that there are two kinds of normal points according to the definition of gadget: partition points and apex points. According to the type of point w_1 and the relative position between $\phi = (w_1, w_2)$ and (v_1, v_2), there are four cases: (1) w_1 is a partition point and ϕ is on the right side of v_1v_2, (2) w_1 is an apex point and ϕ is on the left side of v_1v_2, (3) w_1 is a partition point and ϕ is on the left side of v_1v_2, (4) w_1 is an apex point and ϕ is on the right side of v_1v_2. See Figure 3.19 and 3.21 for illustrations.

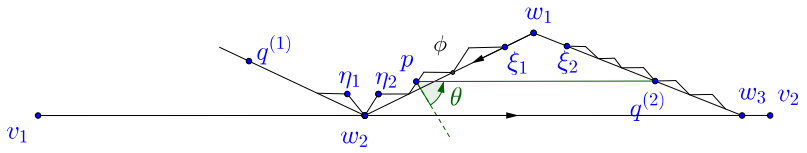
We prove the possible long rang connections between the points of $\mathcal{T}_\phi$ and $\mathcal{T}_\varphi$ case by case. Lemma 3.15 covers case (1) and case (2) which satisfy the condition $\angle(v_1v_2, w_2w_1) = \theta/2$. Lemma 3.16 covers case (3) and case (4) which satisfy the condition $\angle(v_1v_2, w_2w_1) = -\theta/2$.

Lemma 3.15: Given a pair (v_1, v_2) with child-pair set Φ, consider two sibling pairs ϕ and φ in Φ where $\phi = (w_1, w_2)$. ϕ and φ are at *level-l* for $l \leq m - 1$. Suppose point p belongs to $\mathcal{T}_\phi$ and q belongs to $\mathcal{T}_\varphi$. If $\angle(v_1v_2, w_2w_1) = \theta/2$ and there is a directed edge from p to q in $\mathsf{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$, then $\angle(v_1v_2, pq) = 0$ (i.e., pq is parallel to v_1v_2), and q is a point in the gadget Gg_φ generated by φ.

Proof As we have discussed above, there are two cases under the conditions. Consider case (1). See Figure 3.19(a). First, we prove that $\angle(v_1v_2, pq)$ should belong to $(-\theta/2, 0]$. If q belongs to $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.19(a)), w_2 and q are in the same cone of p. There is no edge from p to q since $|pw_2| < |pq|$ and the edge pq is rejected in the Yao-step. Then consider that q belongs to $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e., $q = q^{(2)}$ in Figure 3.19(a)). According to Observation 3.14, we safely assume that φ is an internal-pair. Denote the point in Gg_ϕ closest to w_1 by η. If $\angle(v_1v_2, pq) > 0$, η and q are in the



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



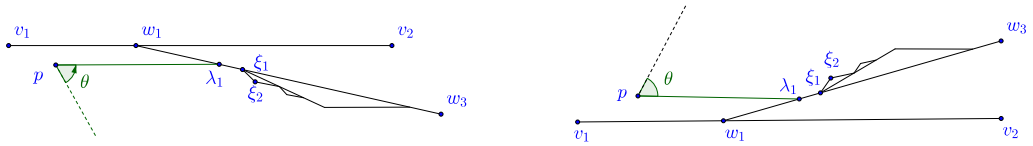
(b) Case 2: w_1 is the apex point and ϕ is on the left side of v_1v_2

Figure 3.19 The two cases about $\angle(v_1v_2, w_2w_1) = \theta/2$. Here $\phi = (w_1, w_2)$ and $p \in \mathcal{T}_\phi$.

same cone of p since $w_1\eta$ has the maximum length among its sibling pairs according to Corollary 3.9. Thus, there is no edge from p to q in the Yao-step since $|p\eta| < |pq|$. Thus, $\angle(v_1v_2, pq) \in (-\theta/2, 0]$. Then, we prove that $\angle(v_1v_2, pq) = 0$. Suppose the projection point of p to pair φ is λ_1 (the λ_1 must exist according to the projection process) and $q^{(2)}$ is an apex point of the piece $\lambda_1\lambda_2$. Note that $\theta < \pi/3$ for $k \geq 3$ and the maximum length among child internal-pairs of φ is at most twice longer than the minimum one (see Property 3.6). It is not difficult to check that the point closest to p in cone $C_p(-\theta, 0]$ is $q^{(2)}$. Thus, $q = q^{(2)}$ and pq is parallel to v_1v_2.

Note that there is a degenerated case in which the projection λ_1 is an end point of an empty piece. Thus, we do not generate the corresponding apex point q. See Figure 3.20(a) for an illustration. (ξ_2, ξ_1) is the pair closest to λ_1. Note that for $k \geq 3$, the angle $\angle p\lambda_1\xi_2 > \pi/2$ and (ξ_1, ξ_2) is a leaf. Thus, in this degenerated case, the point closest to p in cone $C_p(-\theta, 0]$ is λ_1. $p\lambda_1$ is also parallel to v_1v_2. Thus, the lemma is still true. We can process the degenerated case in the same framework in the following and do not distinguish the degenerated case particularly.

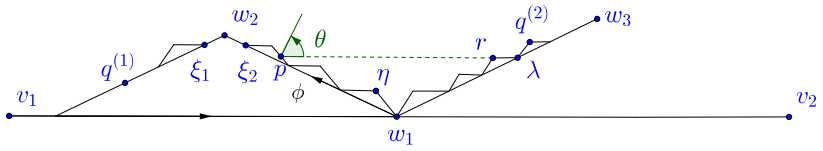
Consider case (2). See Figure 3.19(b). Suppose η_2 is the apex point of the near-empty piece of ϕ incident on w_2. Note that $w_2\eta_2$ has the maximum length among its sibling pairs. If q belongs to $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.19(b)), η_2 and q are in the same cone of p and $|\eta_2 p| < |qp|$. Thus, there is no edge from p to q in the Yao-step. Then consider that q belongs to $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e., $q = q^{(2)}$ in Figure 3.19(b)). According to Observation 3.14, we assume that φ is an internal-pair. If $\varphi \succ \phi$, the polar angle of pq should belong to $(-\theta/2, 0]$. If not, w_1 and q are in the same cone. Thus, there is no



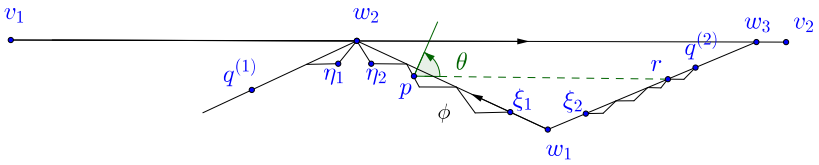
(a) The degenerated case of case 1 in Lemma 3.15

(b) The degenerated case of case 3 in Lemma 3.16

Figure 3.20 The degenerated cases in which the projection point of p is an isolated partition point, i.e., λ_1 in the figure is an isolated partition point which is incident on a short piece.



(a) Case 3: w_1 is the partition point and ϕ is on the left side of v_1v_2



(b) Case 4: w_1 is the apex point and ϕ is on the right side of v_1v_2

Figure 3.21 The two cases about $\angle(v_1v_2, w_2w_1) = -\theta/2$. Here $\phi = (w_1, w_2)$ and $p \in \mathcal{T}_\phi$.

edge from p to q in the Yao-step since $|pq| > |pw_1|$. Then we prove that pq is parallel to v_1v_2. The point closest to p in the cone $C_p(-\theta, 0]$ is the projection point of p (p must exist because of the projection). Thus, pq is parallel to v_1v_2. □

Lemma 3.16: Given a pair (v_1, v_2) with child-pair set Φ, consider two sibling pairs ϕ and φ in Φ where $\phi = (w_1, w_2)$. Suppose ϕ and φ are at *level-l* for $l \leq m-1$. Suppose point p belongs to $\mathcal{T}_\phi$, and q belongs to $\mathcal{T}_\varphi$. If $\angle(v_1v_2, w_2w_1) = -\theta/2$ and there is a directed edge from p to q in $\mathrm{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$, then $\angle(v_1v_2, pq) \in (0, \theta/2)$. Moreover, there exists a point r in $\mathcal{T}_\varphi$ such that pr is parallel to v_1v_2 and $|pr| < |pq|$. Moreover, r is a point in the gadget Gg_φ generated by φ.

Proof As we have discussed above, case (3) and (4) satisfy the condition $\angle(v_1v_2, w_2w_1) = -\theta/2$. Suppose q belongs to $\mathcal{T}_\varphi$. Consider case (3). See Figure 3.21(a). Suppose $w_2\xi_1$ and $w_2\xi_2$ are the two empty pieces incident on w_2. If q is in $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e.,

$q = q^{(1)}$ in Figure 3.21(a)), ξ_1 and q are in the same cone of p. If q is not ξ_1, there is no edge from p to q even in the Yao-step since $|p\xi_1| < |pq|$. If q is ξ_1, $p\xi_1$ would not be accepted by ξ_1 in the reverse-Yao step, since there is an edge from ξ_2 to ξ_1 and $|\xi_1\xi_2| < |p\xi_1|$. Then consider that q (i.e., $q = q^{(2)}$ in Figure 3.21(a)) is in $\mathcal{T}_\varphi$ and $\varphi \succ \phi$. According to Observation 3.14, we safely assume that φ is an internal-pair. Thus, $\angle(v_1v_2, pq) \in [-\theta/2, \theta/2)$. If $\angle(v_1v_2, pq) \in [-\theta/2, 0]$, pq is not a directed edge in Yao-step since w_1 and q are in the same cone and $|w_1p| < |pq|$. Finally, consider the projection point λ (λ exists because of the projection) of p to pair φ. r is the apex point related to λ and on the segment $p\lambda$. It is not difficult to check that $|pr| < |pq|$ since $\theta/2 \leq \pi/2$ and the maximum length among the non-empty pieces of φ is at most twice longer the minimum one (according to refinement). Similar to case 1 in Lemma 3.15, these is a degenerated case that λ is the end point of an empty piece. See Figure 3.20(b). In this case, it is not difficult to check $|p\lambda| < |pq|$.

Consider case (4). See Figure 3.21(b). Suppose η_1 and η_2 are the apex points of the near-empty pieces incident on w_2. If q is in $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.21(b)), η_1 and q are in the same cone of p. If q is not η_1, there is no edge from p to q in the Yao-step since $|p\eta_1| < |pq|$. If q is η_1, $p\eta_1$ would not be accepted by η_1 in the reverse-Yao step since there is an edge from η_2 to η_1 and $|\eta_1\eta_2| < |p\eta_1|$. Then consider q is in $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e. $q = q^{(2)}$ in Figure 3.21(b)). Based on Observation 3.14, we assume that φ is an internal-pair. The polar angle of pq should belong to $(0, \theta/2)$. If not, w_1 and q are in the same cone. Thus, there is no edge from p to q since $|pq| > |pw_1|$. Finally, consider the projection point r of p (r must exist because of the projection) to pair φ. $|pr| < |pq|$ and pr is parallel to v_1v_2 since $\theta/2 \leq \pi/2$. □

In the next section, we discuss how to cut such long range connections. Roughly speaking, under the condition of Lemma 3.15, we can cut the long range connection pq through adding two auxiliary points close to q. Under the condition of Lemma 3.16, we can cut the long range connection pq through adding two auxiliary points close to r.

Based on Lemma 3.15 and 3.16, we have the following corollary.

Corollary 3.17: 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 $\mathcal{T}_\varphi$. If directed edge $\overrightarrow{pq}$ is in $\mathsf{YY}_{2k+1}(\mathcal{P}^{\mathrm{n}}_m)$, then $\phi \prec \varphi$.

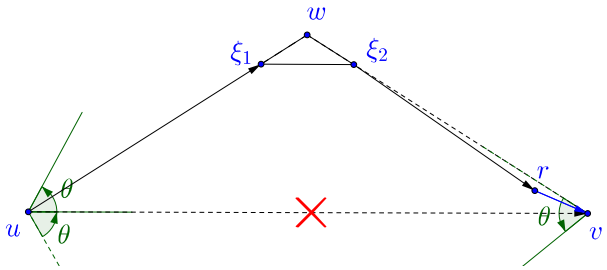


Figure 3.22 A simple example to explain how an auxiliary point cuts a long range connection.

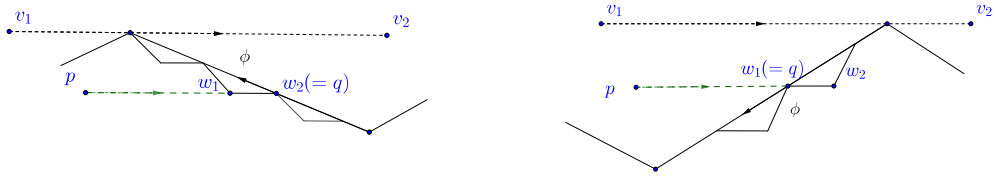
3.4 The Positions of Auxiliary Points

We discuss how to use the auxiliary points to cut the long range connections in the Yao-Yao graph $\mathrm{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$. According to Claim 3.13, it is sufficient by cutting all long range connections between siblings. Denote the set of auxiliary points by $\mathcal{P}_m^{\mathrm{a}}$. Let $\mathcal{P}_m = \mathcal{P}_m^{\mathrm{n}} \cup \mathcal{P}_m^{\mathrm{a}}$.

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 3.22. Note that the polar angles of a cone in the Yao-Yao graph belong to a 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 according to the definition of Yao-Yao graphs, the point v rejects the edge uv in the reverse-Yao step since rv exists in the Yao-step, and point r and u are in the same cone of v and $|rv| < |vu|$. Then, consider ur and $r\xi_1$. The directed edge ur is not in Yao graph since ξ_1 and r are in the same cone of u and $|\xi_1 u| < |ur|$. The directed edge ru is not in Yao graph since ξ_1 and u are in the same cone of r and $|\xi_1 r| < |ur|$. Besides, directed edge $\xi_1 r$ is not in the Yao graph since r and ξ_2 are in the same cone of ξ_1 and $|\xi_1\xi_2| < |\xi_1 r|$. Finally, directed edge $r\xi_1$ is not accepted by ξ_1 in the reverse-Yao step since there is an edge $\xi_2\xi_1$ in the same cone of r and $|\xi_2\xi_1| < |r\xi_1|$. Overall, the shortest path between uv becomes $u\xi_1\xi_2 rv$.

The positions of the auxiliary points: Inspired by the example in Figure 3.22, we call the normal point closest to an auxiliary point the *center* of the auxiliary point. Then, we find *candidate centers* to add auxiliary points.

Lemma 3.18 (Candidate center): Given a pair (v_1, v_2) with its child-pair set Φ, consider two sibling pairs $\varphi, \phi \in \Phi$ and $\varphi \prec \phi$. Suppose p is a point in $\mathcal{T}_\varphi$ and its projected point



(a) Case 1: the projected point q is w_2.

(b) Case 2: the projected point q is w_1.

Figure 3.23 The illustration for candidate center of p.

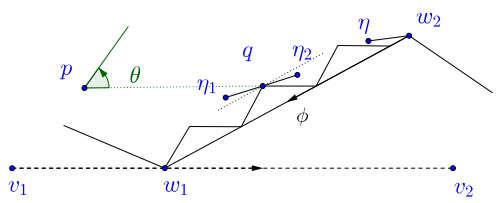
(denoted by q) on the segment of ϕ along direction $\overrightarrow{v_1v_2}$. Then, there exists a nonempty subset $\mathcal{S} \subseteq \mathsf{Gg}_\phi$ such that for any $u \in \mathcal{S}$, pu is parallel to v_1v_2. See Figure 3.23 for an illustration.

We call the point $u := \arg\min_{u\in\mathcal{S}} |pu|$ a *candidate center* of ϕ. Note that the candidate center may not be the projected point q.

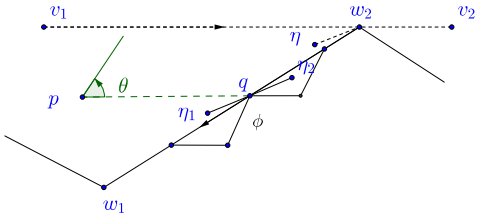
Proof The correctness directly results from the projection process. In the first case (see Figure 3.23(a)) where the apex points of Gg_ϕ and p are in the same side of segment of ϕ, we will generate a pair (w_1, w_2) such that $q = w_2$ and pw_1 and pw_2 are parallel to v_1v_2. Thus, $\mathcal{S} = \{w_1, w_2\}$ and we call the point w_1 a candidate center of ϕ. Note that w_1 is not the projected point of p. In the second case (see Figure 3.23(b)) where the apex points of Gg_ϕ and p are on the different sides of segment of ϕ, we will generate a pair (w_1, w_2) such that $q = w_1$ and pw_1 and pw_2 are parallel to v_1v_2. Thus, $\mathcal{S} = \{w_1, w_2\}$ and we call the point w_1 a candidate center of ϕ. Beside, q may be an isolated partition point or a point in ϕ, then $\mathcal{S}$ and the candidate center is point q itself. □

Definition 3.19 (Candidate center set of ϕ): Consider a pair ϕ with parent pair (v_1, v_2). Let Φ be the set of child-pairs of (v_1, v_2). In the projection process, we project all points $p \in \bigcup_{\varphi\prec\phi,\varphi\in\Phi} \mathcal{T}_\varphi$ to the segment of ϕ along the direction v_1v_2. Each such point p whose projected point falls inside the segment of ϕ corresponds to a candidate center of ϕ defined in Lemma 3.18. We call the set consisting of all these candidate centers the *candidate center set* of ϕ. Note that candidate center set is a subset of Gg_ϕ.

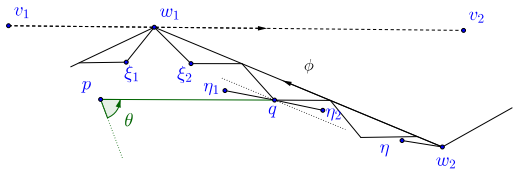
We add some auxiliary points centered on these candidate centers to break long range connection. For convenience, we define some parameters first. Let Δ be the minimum distance between any two normal points and n be the number of the normal points. Recall that we partition the root pair μ_1, μ_2 into d_0 equidistant pieces. Let γ be a very small



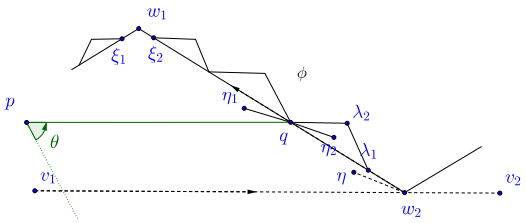
(a) Case 1a: w_1 is the partition point and ϕ is on the left side of v_1v_2



(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 3.24 The auxiliary points for each point. Here $\phi = (w_2, w_1)$ and $q \in \mathsf{Gg}_\phi$. 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. This is just a diagram to explain the relative positions between $\{\eta_1, \eta_2\}$ and q.

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

We traverse $\mathcal{T}$ in the DFS preorder. Each time we reach a pair ϕ, we find all candidate centers in Gg_ϕ and add auxiliary points centered on them.[①] 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 q depends on κ. We use the polar coordinate to describe the relative location of an auxiliary point to its center.

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 3.24(a) and 3.24(b)):
  - 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 $\angle(w_2w_1, q\eta_1) = \angle(w_2w_1, \eta_2 q) = -\gamma$ and $|q\eta_1| = |\eta_2 q| = \sigma^\kappa\chi^{-1}$.

① Note that the candidate centers belong to Gg_ϕ, may not belong to ϕ itself. Besides, here we do not need to distinguish whether the candidate center related to a long range connection or not. It may reduce the number of auxiliary points but do not influent the correctness.

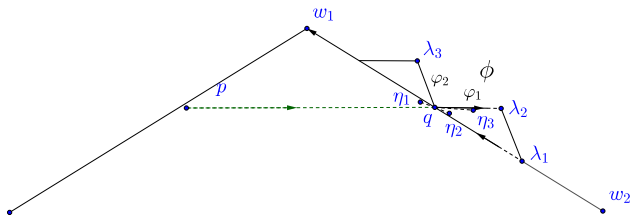


Figure 3.25 The positions of auxiliary points (η_1, η_2, η_3) centered on normal point q.

- $\angle(v_1v_2, w_1w_2) = -\theta/2$ (see Figure 3.24(c) and 3.24(d)):
  - 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 3.24(c) or 3.24(d)), 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 $\mathcal{P}_m$.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 $|q\eta_1| = |\eta_2q| = \sigma^\kappa\chi^{-1}$.

First, we list some useful properties of the auxiliary points below.

Property 3.20: Properties of auxiliary points:

P1 The maximum length between an auxiliary point and its center is at most $d_0^{-1}\Delta$.

P2 Any point $q \in \mathcal{P}_m^{\mathrm{n}}$ can become a center for auxiliary points at most twice. Here, for each time that we indeed add some auxiliary points for a candidate center p, we say that p becomes a center once.

P3 There are at most three auxiliary points centered on a normal point.

P4 Suppose q is a candidate center because of the projection of p and we add the auxiliary point η centered on q. If there is an auxiliary point ξ centered on p, then $|\xi p| \le \sigma^{-1}|\eta q|$. Hence, the perpendicular distance from η to the line pq is larger than $|\xi p|$.

P5 If auxiliary points η_1, η_2 and η_3 are centered on q and $|q\eta_1| = |q\eta_2|$, then $|q\eta_1| \le \sigma^{-1}|q\eta_3|$, $\angle\eta_2q\eta_3 = (\theta/2 - 2\gamma)$, and $\angle q\eta_3\eta_2 < \gamma$.

① It is slightly different from the first case. We add two auxiliary points with distance slightly larger than $\sigma^\kappa\chi^{-1}$ to its center when p and q are in the same hinge set. 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. See the details in the proof of Lemma 3.21.

Figure 3.26 $|\xi p| \le \sigma^{-1}|\eta q|$ for the auxiliary point of p. Moreover, the perpendicular distance from η to pq (i.e., $|\eta\lambda|$ in the figure) is larger than $|\xi p|$.

Proof [P1] Note that the largest κ is at most n since there are at most n pairs in the tree. The maximum length between the auxiliary point and its center is at most $\sigma^n \chi^{-1} = d_0^{-1}\Delta$.

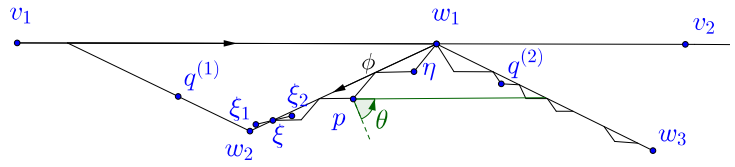
[P2] Note that each point $q \in \mathcal{P}_m^{\mathrm{n}}$ belongs to at most three gadgets, one pair ϕ such that $q \in \mathsf{Gg}_\phi$ and two sibling pairs φ_1 and φ_2 such that $q \in \varphi_1 \cap \varphi_2$. See Figure 3.25 for an example. We visit ϕ first and then φ_1 and φ_2 in order. Note that q is the shared point of φ_1 and φ_2. According to the way to add auxiliary point, when we visit φ_2, q corresponds to the point "w_1" (in Figure 3.24) in the rules. Thus, we do not add auxiliary points for q. Hence, there are only two times that q can become a center for auxiliary points. The first time happens when we visit ϕ and the second time happens when we visit φ_1.

[P3] Followed by the proof of [P2], in the first time, we add two auxiliary points η_1 and η_2 centered on q. In the second time, we add one auxiliary point η_3 centered on q. Thus, there are at most three auxiliary points centered on a normal point.

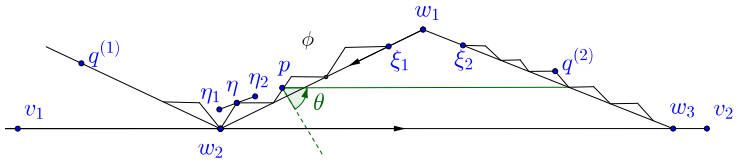
[P4] Suppose p belongs to subtree $\mathcal{T}_\varphi$ and q belongs to subtree $\mathcal{T}_\phi$ and $\varphi \prec \phi$. Thus, the auxiliary points are added for p earlier than q. It means that $|\xi p| \le \sigma^{-1}|\eta q|$. Note that the acute angle between ηq and pq is $(\theta/2 - \gamma)$ and $\sigma > \sin^{-1}(\theta/2 - \gamma)$. Thus, the perpendicular distance from η to the line pq is larger than $|\xi p|$. See Figure 3.26.

[P5] According to the proof of [P3] (see Figure 3.25) we add η_1 and η_2 earlier than η_3. According to the construction, we can add these three auxiliary points for q. Checking the four cases in construction, we can get $\angle(pq, q\eta_3) = -\gamma$ and $\angle(pq, q\eta_2) = -\theta/2 + \gamma$. Thus, $\angle\eta_2 q\eta_3 = \theta/2 - 2\gamma$. Moreover, note that $\sigma > \sin(\theta/2 - \gamma)/\sin\gamma$ and $|\eta_2 q| < \sigma^{-1}|\eta_3 q|$. According to the law of sines, we get $\angle q\eta_3\eta_2 < \gamma$. □

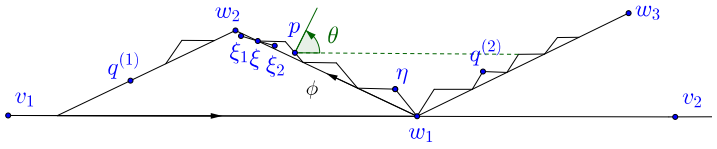
Extended hinge set: We extend the concept of hinge sets to the *extended hinge set* to include auxiliary points. The extended hinge set consists of the normal points in the hinge set and the auxiliary points centered on these normal points. Besides, if p belongs to $\mathcal{T}_\phi$, then the auxiliary points centered on p belong to *extended* $\mathcal{T}_\phi$. Then Claim 3.13 is still true for $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ with the same proof. It means that we only need to consider the long range connections between the descendants of any two sibling pairs.



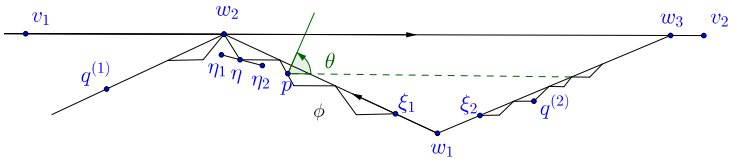
(a) Case 1: w_1 is the partition point and ϕ is on the right side of v_1v_2



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



(c) Case 3: w_1 is the partition point and ϕ is on the left side of v_1v_2



(d) Case 4: w_1 is the apex point and ϕ is on the right side of v_1v_2

Figure 3.27 Here $\phi = (w_1, w_2)$ and p belongs to the extended $\mathcal{T}_\phi$ which includes auxiliary points.

Moreover, we can get similar properties as Lemma 3.15 and 3.16 for the auxiliary points. Suppose ϕ and φ are two sibling pairs. If $p \in \mathcal{T}_\phi$ and $q \in \mathcal{T}_\varphi$ and there is a long range connection $\overrightarrow{pq}$ in YY_{2k+1}, then $\phi \prec \varphi$. Meanwhile, the points in $\mathcal{T}_\varphi$ locate in two cones of p. But only one of the two cones may contain a long range connection. We describe the property formally as follows.

Lemma 3.21: Given a pair (v_1, v_2) at *level-l* for $l < m-1$, with child-pair set Φ, consider two sibling pairs ϕ and φ in Φ where $\phi = (w_1, w_2)$. p is a point in extended $\mathcal{T}_\phi$ and q is a point in extended $\mathcal{T}_\phi$. Suppose there is a directed edge $\overrightarrow{pq}$ in $\mathrm{YY}_{2k+1}(\mathcal{P}_m)$.

- If $\angle(v_1v_2, w_2w_1) = \theta/2$, then $\angle(v_1v_2, pq) \in (-\theta, 0]$.
- If $\angle(v_1v_2, w_2w_1) = -\theta/2$, then $\angle(v_1v_2, pq) \in (0, \theta]$.

Proof The proof follows the same procedure as the proof of Lemma 3.15 and 3.16. We also distinguish into two cases. Given a pair (v_1, v_2) and its child-pair set Φ, consider two

sibling pairs ϕ and φ in Φ where $\phi = (w_1, w_2)$. The first case is that $\angle(v_1v_2, w_2w_1) = \theta/2$. The second case is that $\angle(v_1v_2, w_2w_1) = -\theta/2$. Consider a point p in extended $\mathcal{T}_\phi$. p can be a normal point or an auxiliary point.

Consider that $\angle(v_1v_2, w_2w_1) = \theta/2$. First, suppose w_1 is the partition point and ϕ is on the right side of $\overrightarrow{v_1v_2}$. See Figure 3.27(a). Suppose q belongs to $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.27(a)). Denote the partition point in $\mathcal{A}_\phi$ closest to w_2 by ξ. Because of the projection of points in $\mathcal{T}_\varphi$, ξ has two auxiliary points, denoted by ξ_1 and ξ_2. Thus, $\overrightarrow{pq}$ is not an edge in the Yao-step since ξ_1 and q are in the same cone of p. Then consider that q belongs to $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e., $q = q^{(2)}$ in Figure 3.27(a)). Denote the point in $\mathcal{B}_\phi$ closest to w_1 by η. According to the fact that $w_1\eta$ is the maximum length pair among the child-pairs of ϕ (see Corollary 3.9), η and q are in the same cone of p when $\angle(v_1v_2, pq) > 0$. Thus, there is no long range connection for p in cone $C_p(0, \theta]$.

Second, suppose w_1 is the apex point and ϕ is on the left side of $\overrightarrow{v_1v_2}$. See Figure 3.27(b). Denote the closest point in $\mathcal{B}_\phi$ to w_2 by η. Suppose q belongs to $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.27(b)). Because of the projection of points in $\mathcal{T}_\varphi$, η has two auxiliary points, denoted by η_1 and η_2. There is no edge $\overrightarrow{pq}$ in the Yao-step since η_1 and q are in the same cone p and $|\eta_1 p| < |qp|$. Then consider that q belongs to $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e., $q = q^{(2)}$ in Figure 3.27(b)). If q in the cone $C_p(0, \theta]$, q and w_1 are in the same cone of p and $|pw_1| < |pq|$. Thus, in the Yao-step, there is no edge from p to q in the cone $C_p(0, \theta]$. Thus, we prove the first part of the lemma.

Next, we consider the case $\angle(v_1v_2, w_2w_1) = -\theta/2$. First, we consider the case in which w_1 is the partition point and ϕ is on the left side of $\overrightarrow{v_1v_2}$. See Figure 3.27(c). Denote the partition point in $\mathcal{A}_\phi$ closest to w_2 by ξ. According to the construction for auxiliary point (case 2a), we add two auxiliary points ξ_1 and ξ_2 such that $|\xi\xi_1| = |\xi_2\xi| = \sigma^\kappa\chi^{-1} + \epsilon_0$. If q is in $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.27(c)), ξ_1 and q are in the same cone of p. Because the distance $|\xi\xi_1|$ ($> \sigma^\kappa\chi^{-1}$) is slightly longer than the distances from other auxiliary points of Gg_ϕ to their centers. Thus, $\overrightarrow{pq}$ is not an edge in the Yao-step since $|\xi_1 p| < |pq|$ and ξ_1 and q are in the same cone of p. Then consider that q (i.e., $q = q^{(2)}$) in Figure 3.27(c)) is in $\mathcal{T}_\varphi$ and $\varphi \succ \phi$. If $\angle(v_1v_2, pq) \in [-\theta/2, 0]$, $\overrightarrow{pq}$ is not a directed edge in Yao-step since w_1 and q are in the same cone and $|w_1 p| < |pq|$.

Finally, we consider the case that w_1 is the apex point and ϕ is on the right side of $\overrightarrow{v_1v_2}$. Suppose η is the apex point in $\mathcal{B}_\phi$ closest to w_2. η_1 and η_2 are auxiliary points of η. If q is in $\mathcal{T}_\varphi$ and $\varphi \prec \phi$ (i.e., $q = q^{(1)}$ in Figure 3.27(d)), η_1 and q are in the same cone

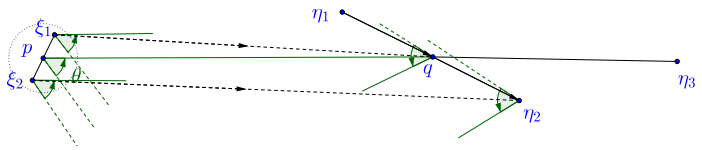


Figure 3.28 The case that $q \in \mathcal{A}_\phi \cup \mathcal{B}_\phi$. The figure is an enlarged view of Figure 3.25. q may have three auxiliary points $\{\eta_1, \eta_2, \eta_3\}$. ξ_1 and ξ_2 are two possible positions of the auxiliary point ξ centered on p.

of p according to the construction for auxiliary point (case 2b). Thus, $\overrightarrow{pq}$ is not an edge in the Yao-step since $|p\eta_1| < |pq|$. Then consider q is in $\mathcal{T}_\varphi$ and $\varphi \succ \phi$ (i.e. $q = q^{(2)}$ in Figure 3.27(d)). If the polar angle of pq belongs to $(-\theta, 0]$, w_1 and q are in the same cone. Thus, there is no edge from p to q since $|pq| > |pw_1|$. Thus, we prove the second part of the lemma.

Overall, we have proved the lemma. □

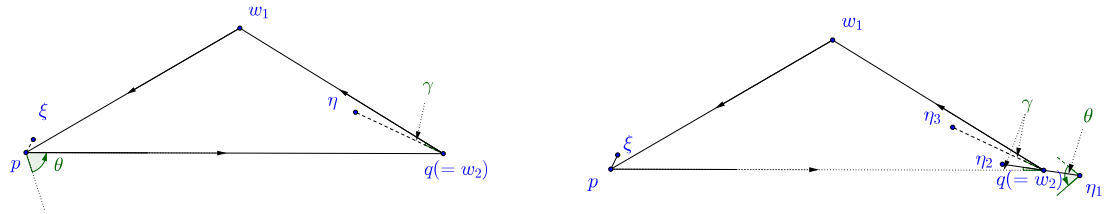
Then, we prove that after adding the auxiliary points, there is no long range connection.

Lemma 3.22: There is no long range connection in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$.

Proof Consider a pair (v_1, v_2) and the set Φ of its child-pairs. Suppose $\phi, \varphi \in \Phi$ and $\varphi \prec \phi$. p is a point in $\mathcal{T}_\varphi$. Denote an auxiliary point centered on p, if any, by ξ. Let $u \in \{p, \xi\}$. There exists a point q closest to p such that $q \in \mathcal{T}_\phi$ and pq is parallel to v_1v_2 based on the projection process. If not, i.e., $\mathcal{T}_\phi$ only locates in one cone of p, according to Lemma 3.21, there is no long range connection between u and points in extended $\mathcal{T}_\phi$.

According to Lemma 3.21, first, there is no directed edge from a point in (extended) $\mathcal{T}_\phi$ to (extended) $\mathcal{T}_\varphi$. Next, we prove there is no long range connection from $\mathcal{T}_\varphi$ to $\mathcal{T}_\phi$. Since p is an arbitrary point in $\mathcal{T}_\varphi$, we prove that there is no long range connection between u and the points in $\mathcal{T}_\phi$, (recall $u \in \{p, \xi\}$). According to whether q is in $\mathcal{A}_\phi \cup \mathcal{B}_\phi$ or ϕ, there are two cases.

q belongs to $\mathcal{A}_\phi \cup \mathcal{B}_\phi$: q has two auxiliary points η_1 and η_2 because of the projection $\overrightarrow{pq}$ and q is a candidate center. Note that q may have a third auxiliary point η_3. But p and η_3 are on the two different sides of $\eta_1\eta_2$ and $|\eta_3 q| > |\eta_1 q| = |\eta_2 q|$ because of Property 3.20[P5]. Therefore, there is no directed edge $p\eta_3$ in the Yao-step because η_2 and η_3 are in the same cone of p and $|\eta_2 p| < |\eta_3 p|$. According to Property 3.20[P4], $|\xi p|$ is much less than $|\eta_1 q|$ or $|\eta_2 q|$ and the perpendicular distance from η_1 and η_2 to the line



(a) Only one auxiliary point centered on q. (b) Three auxiliary points centered on q.

Figure 3.29 q is a point of pair ϕ, i.e., $q = w_2$.

pq is longer than $|\xi p|$. Suppose $u \in \{p, \xi\}$. See Figure 3.28 which is an enlarged view of Figure 3.25, in which ξ_1 and ξ_2 are two possible positions of ξ. According to Lemma 3.21, $u\eta_1$ does not exist in the Yao-step since η_1 and one point of ϕ (denoted by w_1, refer to Figure 3.25) are in the same cone of u and $|w_1u| < |\eta_1u|$. If there is an edge $u\eta_2$ in the Yao-step, the edge $u\eta_2$ cannot be accepted by η_2 in the reverse-Yao step since $q\eta_2$ exists, and point q and u are in the same cone of q and $|q\eta_2| < |u\eta_2|$. If there is an edge uq in the Yao-step, the edge uq cannot be accepted by q in the reverse-Yao step since η_1 and u are in the same cone and $|\eta_1 q| < |uq|$. Therefore, there is no long range connection related to p and its auxiliary points.

q belongs to ϕ: See Figure 3.29. Note that in this case, q is point w_2 of ϕ. According to Property 3.20[P3], any point has at most three auxiliary points. Since q is a projection point of p, q has at least one auxiliary point. Thus, there are two possible situations. One is that there is only one auxiliary point centered on q (see Figure 3.29(a)). It means that in the first time that q was able to be a candidate center, there is no auxiliary point added centered on it (see the proof of Property 3.20[P2]). Denote the auxiliary point of q by η. Let $u \in \{p, \xi\}$ where ξ is an auxiliary point centered on p. According to the Property 3.20[P4], η and w_1 are in the same cone of u and $|uw_1| < |u\eta|$. Therefore, there is no edge $u\eta$ in the Yao-step. Moreover, uq cannot be accepted in the reverse-Yao step. Because the edge ηq exists in the Yao-step. u and η are in the same cone of q and $|\eta q| < |uq|$. Combining with Lemma 3.21, there is no long range connection from u to $\mathcal{T}_\phi$. The second case is that there are three auxiliary points of q (see Figure 3.29(b)). Denote the auxiliary points of q by $\{\eta_1, \eta_2, \eta_3\}$. According to Property 3.20[P5], we know $\angle\eta_2 q\eta_3 = (\theta/2 - 2\gamma)$. Again, denote $u \in \{p, \xi\}$. There is no edge $u\eta_3$ in the Yao-step since w_1 and η_3 are in the same cone of u and $|uw_1| < |u\eta_3|$. There is no edge $u\eta_1$ in the reverse-Yao step, since there is an edge $q\eta_1$ in the Yao-step and $|q\eta_1| < |u\eta_1|$. Similarly,

there is no edge uq since there is an edge $\eta_2 q$ in the Yao-step and $|\eta_2 q| < |uq|$. Next, note that $|q\eta_2| \leq \sigma^{-1}|q\eta_3|$. According to Property 3.20[P5], we know η_3 and u are in the same cone of η_2. Thus, there is no edge from $u\eta_2$.

Overall, we prove that there is no long range connection in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$. □

3.5 The Length Between μ_1 and μ_2 in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$

In this section, we prove that the length of the shortest path between the initial points μ_1 and μ_2 in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ diverges as m approaches infinity.

First, recall that we have extended the concept of hinge sets to *extended hinge sets* which consist of the normal points in the hinge set and the auxiliary points of these normal points. Consider two extended hinge sets Λ and Λ'. Define the shortest path between Λ and Λ' to be the shortest path in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ between any two points p and q such that $p \in \Lambda$ and $q \in \Lambda'$. Consider any pair $\phi = (w_1, w_2)$ at *level*-$(m-1)$. We give a lower bound on the shortest path distance between its former extended hinge set and latter extended hinge set.

Lemma 3.23: Consider any pair (w_1, w_2) at *level*-$(m-1)$. Denote its former extended hinge set by $\Lambda_\phi^{(-)}$, and latter extended hinge set by $\Lambda_\phi^{(+)}$. The shortest path distance between $\Lambda_\phi^{(-)}$ and $\Lambda_\phi^{(+)}$ is at least $(1 - 6d_0^{-1})|w_1 w_2|$.

Proof Let $|w_1 w_2| = \delta$. See Figure 3.30. Note that $\Lambda_\phi^{(-)}$ and $\Lambda_\phi^{(+)}$ (the two hinge sets centered on w_1 and w_2) are not overlapping. Denote the near-empty piece incident on w_1 by $w_1\eta_1$ and the empty piece incident on w_2 by $w_2\xi_1$. $\xi_1\xi_2$ is the leaf-pair closest to w_2. $\xi_2\eta_2$ is perpendicular to $w_1 w_2$. The shortest Euclidean distance between the two hinge sets is no less than $|\eta_1\eta_2|$. According to Property 3.10, $|w_1\eta_1| \leq 2d_0^{-1}\delta$, $|w_2\xi_1| \leq d_0^{-1}\delta$ and $\xi_1\eta_2 \leq 0.5d_0^{-1}\delta$. Thus, $|\eta_1\eta_2| > (1 - 3.5d_0^{-1})\delta$.

Then consider the auxiliary points. Note that according to the Property 3.20[P1], the maximum distance between an auxiliary point and its center is $d_0^{-1}\Delta$, where Δ is the minimum distance between any two normal points. Since $\Delta \leq \delta$, according to triangle inequality, the auxiliary points can reduce the distance between the two hinge sets by at most $2d_0^{-1}\delta$. Overall, the shortest path between $\Lambda_\phi^{(-)}$ and $\Lambda_\phi^{(+)}$ is at least $(1 - 6d_0^{-1})|w_1 w_2|$.□

According to Lemma 3.22, there is no long range connection in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$. Thus, the shortest path between μ_1 and μ_2 should pass through all hinge sets in order. Thus, for each pair ϕ at *level*-$(m-1)$, there is a path between $\Lambda_\phi^{(-)}$ and $\Lambda_\phi^{(+)}$.

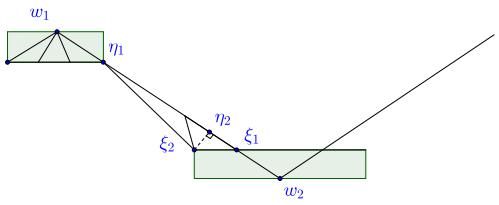


Figure 3.30 The shortest Euclidean distance between two hinge sets centered on points of a pair $\phi = (w_1, w_2)$ at *level*-$(m-1)$.

Let the shortest path between $\Lambda_\phi^{(-)}$ and $\Lambda_\phi^{(+)}$ be Δ_ϕ. Then, we prove that the sum of lengths of Δ_ϕ over all pairs at *level*-$(m-1)$ diverges as m approaches infinity. Thus, the length of the shortest path between μ_1 and μ_2 diverges too.

Lemma 3.24: 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.

Proof We give a lower bound of the sum of lengths $|w_1 w_2|$ over all pairs (w_1, w_2) at *level*-$(m-1)$. Recall that the length of a pair is the length of the segment between the two points of the pair. Consider any pair $\phi = (v_1, v_2)$ with length δ. According to Property 3.10, the sum of lengths of half-empty, near-empty and empty pieces is no more than $6d_0^{-1}\delta$. Thus, the pieces which generate internal-pairs in next level have length at least $(1-6d_0^{-1})\delta$. For each piece, it generates two child-pairs. The sum of lengths of the two pairs is $\cos^{-1}(\theta/2)$ times larger than the piece itself. Overall, the sum of the lengths of the pairs in generated next levels is at least $(1-6d_0^{-1})\delta\cos^{-1}(\theta/2)$. Let $\rho = (1-6d_0^{-1})\cos^{-1}(\theta/2)$. Thus, after $(m-1)$ rounds, the length of the pairs at *level*-$(m-1)$ is at least $\rho^{m-1}|\mu_1\mu_2|$. According to Lemma 3.23, the shortest path from μ_1 to μ_2 is at least $(1-6d_0^{-1})\rho^{m-1}|\mu_1\mu_2|$. When $d_0 > 6(1-\cos(\theta/2))^{-1}$, the shortest path between μ_1 and μ_2 in $\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ diverges as m approaches infinity. □

Finally, combining with the results that YY_3 [48] and YY_5 [51] may not be spanners, we have proved Theorem A.4.

Theorem A.4 *For any $k \geq 1$, there exists a class of 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.*

Chapter 4 Conclusion and Future Work

In this dissertation, we study two important computational geometric problems. First, we study the weighted unit disk set cover problem (WUDC). We provide the first PTAS. As applications, we give a polynomial-time $(4.475 + \epsilon)$-approximation for MWCDS for any fixed constant $\epsilon > 0$. And we provide a PTAS for MLC when all sensors and targets lie in the Euclidean plane and all sensors have the same covering radius. Second, we study the Yao-Yao graphs. We prove that for any integer $k \geq 1$, there exist odd Yao-Yao graph YY_{2k+1} instances which are not spanners. We believe our result and insight are useful to tackle other problems. There are serveral topics we will try in the future.

- Obtaining PTAS for the weighted disk cover problem with arbitrary disks is still a central open problem in this domain. An interesting intermediate step would be to consider the special case where the ratio between the longest radius and the shortest radius is bounded.
- We would like to find the PTAS for the minimum weight connected dominating set in unit disk graphs. It is useful in the computation of routing for mobile ad hoc networks. For the unweighted version, there exists PTAS[9,78]. For the weighted version, current best result is a $(4.475 + \epsilon)$-approximation algorithm.
- It is possible to generalize our PTAS for WUDC from 2-dimension to higher dimensions.
- Actually, we can define other sparse t-spanner candidates inspired from Yao-Yao graphs, such as Θ-Θ graph through using Θ-step twice. We want to know whether these graphs are spanners or not.
- We would like to apply our theoretical results in real world applications. Actually, lots of optimization problems such as facility location, route planning in spatio-temporal data are related to set cover and t-spanner problems.

List of Figures

List of Equations

References

[1] Hunt H B, III, Marathe M V, et al. Nc-approximation schemes for np- and pspace-hard problems for geometric graphs[J]. Journal of Algorithms, 1997.

[2] Bauer M, Damian M. An infinite class of sparse-Yao spanners[C]//Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms. [S.l.]: SIAM, 2013: 184-196.

[3] Li J, Zhan W. Almost all even Yao-Yao graphs are spanners[J]. The 24rd Annual European Symposium on Algorithms, 2016.

[4] Li X Y, Wan P J, Wang Y, et al. Sparse power efficient topology for wireless networks[C]//System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on. [S.l.]: IEEE, 2002: 3839-3848.

[5] Ambühl C, Erlebach T, Mihalák M, et al. Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs[M/OL]//Díaz J, Jansen K, Rolim J, et al. Lecture Notes in Computer Science: volume 4110 Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Springer Berlin Heidelberg, 2006: 3-14. http://dx.doi.org/10.1007/11830924_3

[6] Erlebach T, Mihalák M. A (4+ϵ)-approximation for the minimum-weight dominating set problem in unit disk graphs[M/OL]//Bampis E, Jansen K. Lecture Notes in Computer Science: volume 5893 Approximation and Online Algorithms. Springer Berlin Heidelberg, 2010: 135-146. http://dx.doi.org/10.1007/978-3-642-12450-1_13

[7] Erlebach T, van Leeuwen E. Ptas for weighted set cover on unit squares[M/OL]//Serna M, Shaltiel R, Jansen K, et al. Lecture Notes in Computer Science: volume 6302 Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Springer Berlin Heidelberg, 2010: 166-177. http://dx.doi.org/10.1007/978-3-642-15369-3_13

[8] Erlebach T, Grant T, Kammer F. Maximising lifetime for fault-tolerant target coverage in sensor networks[C]//Proceedings of the twenty-third annual ACM symposium on Parallelism in algorithms and architectures. [S.l.]: ACM, 2011: 187-196.

[9] Du D Z, Wan P J. Connected dominating set: Theory and applications: volume 77[M]. [S.l.]: Springer Science & Business Media, 2012

[10] van Leeuwen E J. Optimization and approximation on systems of geometric objects[D]. [S.l.]: University of Amsterdam, 2009.

[11] Feige U. A threshold of ln n for approximating set cover[J]. Journal of the ACM (JACM), 1998, 45(4): 634-652.

[12] Dinur I, Steurer D. Analytical approach to parallel repetition[C]//Proceedings of the 46th Annual ACM Symposium on Theory of Computing. [S.l.]: ACM, 2014: 624-633.

[13] Hochbaum D S, Maass W. Fast approximation algorithms for a nonconvex covering problem [J/OL]. Journal of Algorithms, 1987, 8(3): 305 - 323. http://www.sciencedirect.com/science/article/pii/0196677487900125.

[14] Clark B N, Colbourn C J, Johnson D S. Unit disk graphs[J/OL]. Discrete Math., 1991, 86(1-3): 165-177. http://dx.doi.org/10.1016/0012-365X(90)90358-O.

[15] Chan T M, Grant E. Exact algorithms and apx-hardness results for geometric packing and covering problems[J/OL]. Computational Geometry, 2014, 47(2, Part A): 112 - 124. http://www.sciencedirect.com/science/article/pii/S0925772112000740.

[16] Har-Peled S, Lee M. Weighted geometric set cover problems revisited[J/OL]. JoCG, 2012, 3(1): 65-85. http://jocg.org/index.php/jocg/article/view/77.

[17] Brönnimann H, Goodrich M T. Almost optimal set covers in finite vc-dimension[J/OL]. Discrete & Computational Geometry, 1995, 14(1): 463-479. http://dx.doi.org/10.1007/BF02570718.

[18] Clarkson K L, Varadarajan K. Improved approximation algorithms for geometric set cover [J/OL]. Discrete & Computational Geometry, 2007, 37(1): 43-58. http://dx.doi.org/10.1007/s00454-006-1273-8.

[19] Even G, Rawitz D, Shahar S M. Hitting sets when the vc-dimension is small[J]. Information Processing Letters, 2005, 95(2): 358-362.

[20] Varadarajan K. Epsilon nets and union complexity[C/OL]//SCG '09: Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry. New York, NY, USA: ACM, 2009: 11-16. http://doi.acm.org/10.1145/1542362.1542366.

[21] Varadarajan K. Weighted geometric set cover via quasi-uniform sampling[C]//Proceedings of the forty-second ACM symposium on Theory of computing. [S.l.]: ACM, 2010: 641-648.

[22] Chan T M, Grant E, Könemann J, et al. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling[C/OL]//SODA '12: Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM, 2012: 1576-1585. http://dl.acm.org/citation.cfm?id=2095116.2095241.

[23] Mustafa N H, Ray S. PTAS for geometric hitting set problems via local search[C]//Proceedings of the twenty-fifth annual symposium on Computational geometry. [S.l.]: ACM, 2009: 17-22.

[24] Gibson M, Pirwani I A. Algorithms for dominating set in disk graphs: breaking the log n barrier [M]//ESA. [S.l.]: Springer, 2010: 243-254

[25] Bansal N, Pruhs K. The geometry of scheduling[J]. SIAM Journal on Computing, 2014, 43(5): 1684-1698.

[26] Pyrga E, Ray S. New existence proofs ε-nets[C]//Proceedings of the twenty-fourth annual symposium on Computational geometry. [S.l.]: ACM, 2008: 199-207.

[27] Har-Peled S, Kaplan H, Sharir M, et al. Epsilon-nets for halfspaces revisited[J]. arXiv preprint arXiv:1410.3154, 2014.

[28] Bus N, Garg S, Mustafa N H, et al. Tighter estimates for epsilon-nets for disks[J]. arXiv preprint arXiv:1501.03246, 2015.

[29] Hochbaum D S, Maass W. Approximation schemes for covering and packing problems in image processing and vlsi[J]. Journal of the ACM (JACM), 1985, 32(1): 130-136.

[30] Huang Y, Gao X, Zhang Z, et al. A better constant-factor approximation for weighted dominating set in unit disk graph[J/OL]. Journal of Combinatorial Optimization, 2009, 18(2): 179-194. http://dx.doi.org/10.1007/s10878-008-9146-0.

[31] Dai D, Yu C. A 5 + ϵ-approximation algorithm for minimum weighted dominating set in unit disk graph[J]. Theoretical Computer Science, 2009, 410(8): 756-765.

[32] Zou F, Wang Y, Xu X H, et al. New approximations for minimum-weighted dominating sets and minimum-weighted connected dominating sets on unit disk graphs[J]. Theoretical Computer Science, 2011, 412(3): 198-208.

[33] Ding L, Wu W, Willson J, et al. Constant-approximation for target coverage problem in wireless sensor networks[C]//INFOCOM. [S.l.]: IEEE, 2012: 1584-1592.

[34] Zhang Z, Willson J, Lu Z, et al. Approximating maximum lifetime k-coverage through minimizing weighted k-cover in homogeneous wireless sensor networks[J]. IEEE/ACM Transactions on Networking, 2016, 24(6): 3620-3633.

[35] Adamaszek A, Wiese A. Approximation schemes for maximum weight independent set of rectangles[C]//Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on. [S.l.]: IEEE, 2013: 400-409.

[36] Mustafa N H, Raman R, Ray S. Qptas for geometric set-cover problems via optimal separators [J]. arXiv preprint arXiv:1403.0835, 2014.

[37] Marx D. On the optimality of planar and geometric approximation schemes[C]//Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on. [S.l.]: IEEE, 2007: 338-348.

[38] Gabriel K R, Sokal R R. A new statistical approach to geographic variation analysis[J]. Systematic Biology, 1969, 18(3): 259-278.

[39] Toussaint G T. The relative neighbourhood graph of a finite planar set[J]. Pattern recognition, 1980, 12(4): 261-268.

[40] Aurenhammer F. Voronoi diagrams – survey of a fundamental geometric data structure[J]. ACM Computing Surveys (CSUR), 1991, 23(3): 345-405.

[41] Yao A C C. On constructing minimum spanning trees in k-dimensional spaces and related problems[J]. SIAM Journal on Computing, 1982, 11(4): 721-736.

[42] Sack J R, Urrutia J. Handbook of computational geometry[M]. [S.l.]: Elsevier, 1999

[43] Chew P. There is a planar graph almost as good as the complete graph[C]//Proceedings of the second annual symposium on Computational geometry. [S.l.]: ACM, 1986: 169-177.

[44] Eppstein D. Spanning trees and spanners[J]. Handbook of computational geometry, 1999: 425-461.

[45] Li X Y. Wireless ad hoc and sensor networks: Theory and applications[M]. [S.l.]: Cambridge, 2008

[46] Narasimhan G, Smid M. Geometric spanner networks[M]. [S.l.]: Cambridge University Press, 2007

[47] Flinchbaugh B E, Jones L K. Strong connectivity in directional nearest-neighbor graphs[J]. SIAM Journal on Algebraic Discrete Methods, 1981, 2(4): 461-463.

[48] El Molla N M. Yao spanners for wireless ad hoc networks[D]. [S.l.]: Villanova University, 2009.

[49] Bose P, Damian M, Douïeb K, et al. $\pi/2$-angle Yao graphs are spanners[J]. International Journal of Computational Geometry & Applications, 2012, 22(01): 61-82.

[50] Damian M, Nelavalli N. Improved bounds on the stretch factor of Y_4[J]. Computational Geometry, 2017, 62: 14-24.

[51] Barba L, Bose P, Damian M, et al. New and improved spanning ratios for Yao graphs[J]. JoCG, 2015, 6(2): 19-53.

[52] Damian M, Raudonis K. Yao graphs span Θ-graphs[M]//Combinatorial Optimization and Applications. [S.l.]: Springer, 2010: 181-194

[53] Li X Y, Wan P J, Wang Y. Power efficient and sparse spanner for wireless ad hoc networks[C]// Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on. [S.l.]: IEEE, 2001: 564-567.

[54] Bose P, Maheshwari A, Narasimhan G, et al. Approximating geometric bottleneck shortest paths [J]. Computational Geometry, 2004, 29(3): 233-249.

[55] Damian M, Molla N, Pinciu V. Spanner properties of $\pi/2$-angle Yao graphs[C]//Proc. of the 25th European Workshop on Computational Geometry. [S.l.]: Citeseer, 2009: 21-24.

[56] Jia L, Rajaraman R, Scheideler C. On local algorithms for topology control and routing in ad hoc networks[C]//Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures. [S.l.]: ACM, 2003: 220-229.

[57] Kanj I A, Xia G. On certain geometric properties of the Yao-Yao graphs[M]//Combinatorial Optimization and Applications. [S.l.]: Springer, 2012: 223-233

[58] Wang Y, Li X Y, Frieder O. Distributed spanners with bounded degree for wireless ad hoc networks[J]. International Journal of Foundations of Computer Science, 2003, 14(02): 183-200.

[59] Damian M. A simple Yao-Yao-based spanner of bounded degree[J]. arXiv preprint arXiv:0802.4325, 2008.

[60] Barba L, Bose P, De Carufel J L, et al. On the stretch factor of the Θ_4-graph[C]//Workshop on Algorithms and Data Structures. [S.l.]: Springer, 2013: 109-120.

[61] Bose P, Morin P, van Renssen A, et al. The Θ_5-graph is a spanner[J]. Computational Geometry, 2015, 48(2): 108-119.

[62] Bonichon N, Gavoille C, Hanusse N, et al. Connections between Θ-graphs, Delaunay triangulations, and orthogonal surfaces[C]//International Workshop on Graph-Theoretic Concepts in Computer Science. [S.l.]: Springer, 2010: 266-278.

[63] Ruppert J, Seidel R. Approximating the d-dimensional complete euclidean graph[C]//Proceedings of the 3rd Canadian Conference on Computational Geometry (CCCG 1991). [S.l.: s.n.], 1991: 207-210.

[64] Bose P, van Renssen A, Verdonschot S. On the spanning ratio of theta-graphs[C]//Workshop on Algorithms and Data Structures. [S.l.]: Springer, 2013: 182-194.

[65] Grünewald M, Lukovszki T, Schindelhauer C, et al. Distributed maintenance of resource efficient wireless network topologies[C]//European Conference on Parallel Processing. [S.l.]: Springer, 2002: 935-946.

[66] Schindelhauer C, Volbert K, Ziegler M. Spanners, weak spanners, and power spanners for wireless networks[C]//International Symposium on Algorithms and Computation. [S.l.]: Springer, 2004: 805-821.

[67] Schindelhauer C, Volbert K, Ziegler M. Geometric spanners with applications in wireless networks[J]. Computational Geometry, 2007, 36(3): 197-214.

[68] Eppstein D. Beta-skeletons have unbounded dilation[J]. Computational Geometry, 2002, 23(1): 43-52.

[69] Du D Z, l Ko K, Hu X. Design and analysis of approximation algorithms[M]. [S.l.]: Springer, 2011

[70] Sharir M. Davenport-schinzel sequences and their geometric applications[M/OL]//Earnshaw R. NATO ASI Series: volume 40 Theoretical Foundations of Computer Graphics and CAD. Springer Berlin Heidelberg, 1988: 253-278. http://dx.doi.org/10.1007/978-3-642-83539-1_9

[71] Kedem K, Livne R, Pach J, et al. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles[J/OL]. Discrete & Computational Geometry, 1986, 1(1): 59-71. http://dx.doi.org/10.1007/BF02187683.

[72] Zou F, Li X, Gao S, et al. Node-weighted steiner tree approximation in unit disk graphs [J/OL]. Journal of Combinatorial Optimization, 2009, 18(4): 342-349. http://dx.doi.org/10.1007/s10878-009-9229-6.

[73] Byrka J, Grandoni F, Rothvoß T, et al. An improved lp-based approximation for steiner tree[C]// Proceedings of the forty-second ACM symposium on Theory of computing. [S.l.]: ACM, 2010: 583-592.

[74] Cardei M, Thai M, Li Y, et al. Energy-efficient target coverage in wireless sensor networks[C]// INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedin gs IEEE: volume 3. 2005: 1976-1984 vol. 3.

[75] Berman P, Calinescu G, Shah C, et al. Efficient energy management in sensor networks[C]//Ad Hoc and Sensor Networks. Nova Science Publishers. [S.l.]: Nova Science Publisher, 2005.

[76] Garg N, Könemann J. Faster and simpler algorithms for multicommodity flow and other fractional packing problems[J/OL]. SIAM J. Comput., 2007, 37(2): 630-652. http://dx.doi.org/10.1137/S0097539704446232.

[77] Du H, Pardalos P, Wu W, et al. Maximum lifetime connected coverage with two active-phase sensors[J/OL]. Journal of Global Optimization, 2013, 56(2): 559-568. http://dx.doi.org/10.1007/s10898-012-9871-x.

[78] Cheng X, Huang X, Li D, et al. Polynomial-time approximation scheme for minimum connected dominating set in ad hoc wireless networks[J]. Networks, 2003, 42.

Acknowledgements

首先要感谢我的导师李建老师，博士期间每个点滴进步，都离不开李老师帮助。五年前，因为一种特殊的缘分，我得以到清华大学交叉信息院攻读博士学位。本科学天文的我，当时对理论计算机领域可以说是一无所知，凭着一股一知半解带来的冲动和自以为知其然的热情，愣头愣脑的就来了交叉信息院的夏令营。经过笔试面试的双重打击后，承蒙李老师不弃，得以在“茶园”进行理论计算机研究，这里感谢李老师的知遇之恩。余热未减的我其实不知道这五年将面临怎样的 hard-模式，感谢李老师的精心指导，让许多的不可能变成了可能。一路走来，从理论基础的培养、专业知识的学习，到博士选题、科研问题的攻克，再到论文写作、最终的科学汇报，李老师总是能给出有效的意见和建议。李老师在各个环节悉心指导，亲身鼓励让我受益良多。五年的博士学习，让我看问题更加本质，能在更短的时间里找到问题的症结，进而提出有效的办法。此外，李老师不仅博闻强识，而且勤奋刻苦。作为人生榜样，他的言行时常感召着我，要从各个领域里汲取营养，并且努力践行才能有所进步。从学五载，希望不负君期，将来能有所作为。

其次要感谢实验室全体老师和师兄弟们，是你们让实验室生活更加丰富多彩，让研究生活更加从容不迫。感谢我的师兄黄棱霄，同样他也是我工作的合作者。一个优秀的师兄，总能帮助人少走很多弯路。棱霄处理问题及其高效而专注，随时随地能进入工作状态。从他身上我学到做科研绝不是“焚香沐浴”，做好万全准备才开始两手一摊思考问题。往往要时刻保持兴奋状态，在无数次碰撞中才能最终爆发灵感。感谢我的合作者占玮，我们虽然接触不多，但你对问题的细致剖析，对后续工作起到至关重要的作用。感谢同窗曹玮和傅昊，我们共同经历了太多，记不清多少次一起学习，一起讨论问题。祝两位毕业顺利，前程似锦。感谢我的两位小伙伴巩慧超和淦创。虽然我们学术交流不多，但是有你们的日子，让我在学术以外的生活更加丰富多彩。而这也使得我总是有饱满的热情，去面对每一天的生活。感谢我的入党介绍人房智轩和徐源同志，谢谢你们引领我走向共产主义道路。最后感谢实验室的其他兄弟姐妹，感谢金铠、刘宇、许梦雯、王栋、李志泽、吴旋、张楚珩、郭建波、施宇等。我这里就不一一点名了，多有不周，还望海涵。

还要感谢我博士生活的两位室友吕曰洲和戴元熙。两位都极其聪明刻苦，在自己的专业方向也极为优秀。是你们让我了解了很多其他领域的工作和思维模式，让每个夜晚都富有诗意。感谢你们包容我不那么规律的作息时间，希望两位都早日找到人生归宿。

感谢我的论文评审堵丁柱、丁虎、Mirela Damian、Thomas Erlebach、李国良、王海涛、张昭教授，谢谢你们的评审意见。你们的专业评审和悉心指导让我看到了一个科研工作者的专业素养和前辈对晚辈的悉心关怀。感谢参加我预答辩及答辩的陈卫、李国良、孙晓明、唐平中、吴文斐老师，感谢你们在百忙之中抽出时间，参加一个博士候选人的“成人礼”。感谢我的答辩秘书段然老师，您承担了太多繁琐的组织工作，谢谢您为我的无私付出。

最后，我由衷感谢我的父母和两位姐姐，从小到大，你们包容了我太多的无知和任性，而我为你们付出的太少了。希望今后的日子里，我能像个男子汉一样承担起生活的责任，你们的健康幸福是我奋斗的源泉和动力。

Declaration

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____ 日　期：_____

Appendix A　中文摘要

在本论文中，我们研究了两个重要的计算几何问题。首先，我们研究了最小带权单位圆覆盖问题。最小带权单位圆覆盖问题是否存在多项式时间近似方案（PTAS）是学术界一个长期以来的开放问题。在之前的很多研究中[5-10] 都有提及该开放问题，并强调了该问题在理论和实践中的重要意义。在本论文中，我们通过给出该问题的第一个 PTAS 算法正面解决了该问题。其次我们研究了几何 t-跨度图问题。该图保持了任意两点间在图上的最短距离是这两点间欧几里德距离的至多 t 倍的性质。几何 t-跨度图在通讯网络中有重要的应用。Yao-Yao 图因其构造简单，并且每个点的度数有限，而成为一个很好的几何 t-跨度图候选体。关于 Yao-Yao 图（YY_k 图）是否具有 t-跨度属性，是一个长期困扰人们的开放问题。Bauer 和 Damian[2] 证明了所有 $\mathrm{YY}_{6k}(k >= 6)$ 都是 t-跨度图。Li 和 Zhan[3] 推广了他们的结论，证明了所有的 $\mathrm{YY}_{2k}(k >= 42)$ 都是 t-跨度图。但是这些已知的技术都不能拓展到奇数 Yao-Yao 图上。我们的工作第一次给出了结论——对于任意 $k >= 1$, 存在点集 $\mathcal{P}$，使得定点集上的奇数 Yao-Yao 图（YY_{2k+1}）不是 t-跨度图。

最小带权单位圆覆盖

集合覆盖问题是理论计算机科学和组合优化中的核心问题之一。集合覆盖问题的输入包括两个集合 U 和 $\mathcal{S}$。其中集合 $\mathcal{S}$ 的元素是集合 U 的子集。$\mathcal{S}$ 中的每个元素都有一个非负的权重 w_S。集合覆盖的目标是找到最小总重量的子集合 $C \subseteq \mathcal{S}$ 使得 $\bigcup C$ 涵盖 U 的所有元素。人们对一般性的集合覆盖问题的可近似性认识已经较为彻底了：通过贪心算法，可以给出该问题的 H_n-近似（$H_n = \sum_{i=1}^{n} 1/i$）。另一方面，给定任何常数 $\epsilon > 0$，获得一个 $(1+\epsilon)\ln n$-近似的解是 NP-hard 的[11-12]。在几何集合覆盖问题中，U 是欧几里德空间 $\mathbb{R}^d$ 中的一组点，$\mathcal{S}$ 由几何对象（例如，圆盘，正方形，三角形等）组成。在几何环境下，由于 $\mathcal{S}$ 的特殊结构，我们可以期待能够获得好于对数近似比的近似算法。然而对于大多数几何对象，即使对于非常简单的对象类，例如单位圆，几何集合覆盖问题仍然是 NP 难的。[13-14]（更多示例参见[15-16]。因为问题本身的重要性，而且还与其他重要的概念和问题有着密切的联系，例如 VC-维[17-19]，ϵ-网（ϵ-net），并集复杂度（union 复杂度）[20-22]，平面图分割（planar separators）[23-24]，甚至机器调度问题[25]。几何集覆盖的近似算法，过去二十年来一直在被广泛关注，深入研究。

在本工作中，我们研究了一类最简单的几何对象——单位圆的几何集合覆盖

问题。该问题的严格定义如下：

Definition A.1 (最小带权单位圆覆盖 (WUDC))： 给定一个包含 n 个单位圆的集合 $\mathcal{D} = \{\mathsf{D}_1, \ldots, \mathsf{D}_n\}$ 和欧几里得平面 $\mathbb{R}^2$ 上的 m 个点的点集 $\mathcal{P} = \{P_1, \ldots, P_m\}$ 。其中每个单位圆 D_i 带权重 $w(\mathsf{D}_i)$。问题的目标是选择一个单位圆的子集使的这些子集的并集能覆盖 $\mathcal{P}$ 中的所有点并且这些圆的的总权重和最小。

WUDC 推广了下面单位圆图上的最小带权支配集问题。

Definition A.2 (单位圆图上的最小带权支配集问题 (MWDS))： 给定单位圆图 $G(V, E)$。其中 V 是欧几里得平面 $\mathbb{R}^2$ 上的带权点集。对任意的点 $u, v \in V$, 边 (u, v) 属于 E 当且仅当 $\|u - v\| \leq 1$。S 是 V 的一个子集，并且满足对于任意节点 $v \notin S$，存在一个节点 $u \in S$ 满足 $(u, v) \in E$，我们称 S 为一个支配集。单位圆图上的最小带权支配集问题的目标是求出图 G 上最小带权支配集。

通过如下归约，我们可以证明 WUDC 是 MWDS 问题的推广。给定图 $G(V, E)$ 的一个支配集实例，通过在每个属于 V 的点 v 上放置一个与 v 点权重相同的，圆心位于 v 的单位圆，我们可以构造一个 WUDC 问题的实例。因此，在本论文中，我们重点在 WUDC 问题上描述我们的算法和结果。

相关工作及我们的贡献

对于一个求最小值的最优化问题，一个多项式时间的近似方案 (PTAS) 指的是一个算法 $\mathcal{A}$, 输入一个实例和一个大于零的常数 ϵ，返回一个解 SOL 使得 $\mathsf{SOL} \leq (1 + \epsilon)\mathsf{OPT}$，其中 OPT 代表问题的最优值。而算法 $\mathcal{A}$ 的运行时间对于 n 是多项式大小的。

WUDC 是 NP 难的问题，即使当单位圆的权重都相等的情况下（即, $w(\mathsf{D}_i) = 1$[14]），仍然是 NP 难的。对于等权的单位圆图支配集问题，Hunt 等人[1] 获得了第一个 PTAS 解法。对于更一般的圆图（半径不固定的圆图），注意到三维空间的半空间存在 $O(1/\epsilon)$ 大小的 ϵ-net[26] (又见[27]), 通过集合覆盖问题和 ϵ-网之间的关系[17-19], 可以得到一个常数近似的算法。在文献[23] 中，作者分析了该常数最好可以达到 20。一项最近的成果表明[28]，该常数可以提高到 13。此外，通过局部搜索，可以找到等权单位圆支配集的 PTAS 解法[23-24]。

对于一般带权 WUDC 问题，研究历史则更久远。Ambühl 等人[5] 得到了第一个常数近似算法，其算法的常数为 72。应用文献[29] 中的平移技术（shifting technique），Huang 等人得到了一个常数为 $(6+\epsilon)$ 的近似算法。近年来，常数算法的近似度不断被提高，其中一些研究者获得了 $(5+\epsilon)$ 近似[31] 和 $(4+\epsilon)$ 近似的算法[6,32-33]. 在本论

文之前最好的结果为 3.63 近似。[①] 最近，张等人还给出了一种 $(3+\epsilon)$ 近似解法[34]。利用*近似平均采样法*（quasi-uniform sampling method) [21-22], 可以通过另一个途径实现 WUDC (甚至一般圆图) 的常数近似解法。然而，其近似度依赖于其他一些技术的近似度，例如线性规划舍入中的近似度和并集复杂度中的常数。而这些常数一般很少有精确的结果。最近，基于 Adamaszek 和 Wiese 的分离框架[35]，Mustafa 等人[36]，获得了对于 $\mathbb{R}^2$ 中的带权圆（事实上，是 $\mathbb{R}^3$ 中的加权半空间）QPTAS（准多项式时间近似方案），因此排除了 WUDC 的是 APX 难的可能性。

另一个与之密切相关的工作带权单位正方形覆盖，Erlebach 和 Van Leeuwen[7] 提出了第一个 PTAS 算法。该工作有开创性意义，即使对于任何平面物体上的加权几何图形覆盖（除了多时间可解的情况[15-16]），这是也是第一个 PTAS 算法。虽然看起来他们的结果与带权 WUDC 的 PTAS 非常接近，如他们的论文所承认的，他们的技术不足以应付单位圆问题，并且"完全不同的技术可能是必需的"。

鉴于上述所有结果，看来我们应该期待 WUDC 应该存在一个 PTAS 算法，但它仍然是一个悬而未决的问题。（明确提到这是一个开放性问题的论文有很多，例如:[5-10]）。本论文的一个主要贡献是通过提供 WUDC 的第一个 PTAS，肯定地解决这个问题。

Theorem A.1： WUDC 问题存在一个多项式时间逼近方案。运行时间是 $n^{O(1/\epsilon^9)}$。

由于 WUDC 比 MWDS 更普遍，我们可以得到以下推论。

Corollary A.1： 带权单位圆图中最小权重支配集问题有一个多项式时间逼近格式。

我们注意到，根据 Marx 的"否定"结果，运行时间 $n^{\text{poly}(1/\epsilon)}$ 几乎是最优的，他证明即使对未加权单位圆支配集问题，如果存在 EPTAS （即，高效的 PTAS，运行时间为 $f(1/\epsilon)\text{poly}(n)$），将与指数时间假设相矛盾。最后，我们还证明了我们的 WUDC 的 PTAS 可以用来获得改进的近似算法对于无线传感器网络中的两个重要问题: 单位圆图中的连通支配集问题和最大生命周期覆盖问题。

带权单位圆图覆盖 PTAS

技术概览

通过标准移位技术[69]，只要证明当所有圆位于一个大小不变的正方形时，（我们称之为块，常数取决于 $1/\epsilon$），能为 WUDC 提供一个 PTAS 就足够了。这个想法在 Huang 等人的论文中被形式化如下[30]。

① 该算法被记载在 Du 和 Wan 撰写的图书中[9]，算法由 Willson 等人提供。

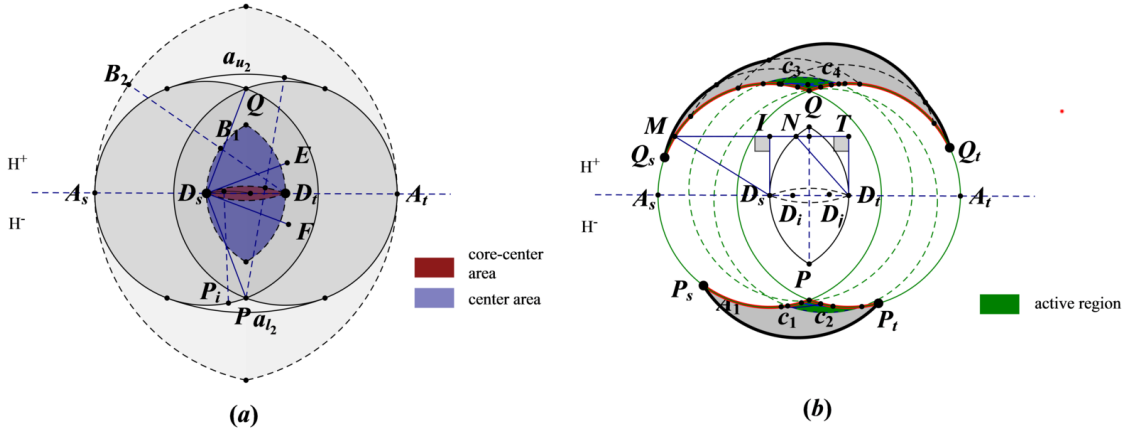


Figure A.1　小构件。D_s 和 D_t 是圆心位于小方块 Γ 中圆心距离最远的两个圆盘，它们的中心分别是 D_s 和 D_t. 在左边的图中，蓝色区域是中心区域，定义为 $\mathfrak{C} = \mathsf{D}(D_s, r_{st}) \cap \mathsf{D}(D_t, r_{st})$，其中 $r_{st} = |D_s D_t|$。棕色区域是核心区域，定义为 $\mathfrak{C}_o = \mathsf{D}(P, 1) \cap \mathsf{D}(Q, 1)$. 在右边的图中，绿色区域是活动区域定义为 $\left(\bigcup_{i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^+$ 和 $\left(\bigcup_{i \in \mathfrak{C}_o} \mathsf{D}_i - (\mathsf{D}_s \cup \mathsf{D}_t)\right) \cap H^-$.

Lemma A.1 (Huang et al.[30])：假设在固定的 $L \times L$ 块中存在一个用于 WUDC 的 ρ-近似算法，运行时间为 $f(n, L)$。则 WUDC 存在一个 $(\rho + O(1/L))$ 近似度的，运行时间为 $O(L \cdot n \cdot f(n, L))$ 的近似算法。特别是，设置 $L = 1/\epsilon$，则 WUDC 存在一个 $O\left(\frac{1}{\epsilon} \cdot n \cdot f(n, \frac{1}{\epsilon})\right)$ 时间的 $(\rho + \epsilon)$ 近似算法。

事实上，之前几乎所有的 WUDC 常数近似算法都是通过对恒定尺寸的单个块开发常数近似值而获得的（这是也是该问题的主要难点）。①

本文的主要贡献在于对前面的工作进行改进[5-6,30-31]，从而得到单个块中的 $(1+\epsilon)$ 近似解法，如下面的引理所示。

Lemma A.2：在大小为 $L \times L$ for $L = 1/\epsilon$ 的固定大小的块中存在 WUDC 的 PTAS。PTAS 的运行时间是 $n^{O(1/\epsilon^9)}$。

在本文中，保证近似误差 $\epsilon > 0$ 是一个固定常数。当我们说一个数量是一个常数，常数可能取决于 ϵ。我们使用 OPT 来表示该块中的最优解（和最优值）。我们用大写字母 $A, B, C, \ldots$ 来表示分数，小写字母 $a, b, c, \ldots$ 表示弧。对于 A 和 B 两个点，我们使用 AB 来表示连接 A 和 B 的线段并用 $|AB|$ 来表示它的长度。我们使用 D_i 来表示圆盘，使用 D_i 来表示它的中心。对于点 A 和实数 $r > 0$，让 $\mathsf{D}(A, r)$ 为以 A 为中心且半径为 r 的圆盘。对于圆盘 D_i，我们使用 $\partial\mathsf{D}_i$ 来表示它的边界。我们称 $\partial\mathsf{D}_i$ 上的一段为一条弧。

① 对于单块中未加权的支配集问题，很容易看出圆的最佳数量是一个常数，这意味着我们可以在多项式时间里计算出最优值。然而，对于加权支配集问题或 WUDC，单个问题中的最优解块可能由 $\Theta(n)$ 个圆组成。

首先，我们猜测对于某些常量 C，OPT 是否包含多于 C 的圆盘。如果 OPT 包含不超过 C 圆盘，我们可以列举所有可能的组合，然后选择覆盖所有点并具有最小权重的组合，从而得到问题的最优解。这是多项式时间内可完成的，总共需要 $O\left(\sum_{i=1}^{C}\binom{n}{i}\right) = O(n^C)$ 的时间，

更具挑战性的情况是 OPT 包含多于 C 个圆盘。在这种情况下，我们猜测（即列举所有可能性）具有 OPT 中 C 个权重最大的圆盘的集合 $\mathcal{G}$。这最多有多项式种（即，$O(n^C)$）可能的猜测。假设我们的猜测是正确的。然后，我们删除 $\mathcal{G}$ 中的所有圆盘以及 $\mathcal{G}$ 覆盖的所有点。让 D_t（带重量 w_t）是 $\mathcal{G}$ 中权重最小的圆盘。我们可以看到 $\mathsf{OPT} \geq Cw_t$。此外，我们也可以安全地忽略所有重量大于 w_t 的圆盘（假设我们的猜测是正确的）。现在，我们的任务是用剩余的圆盘覆盖剩余的点，每个圆盘的重量至多为 w_t。我们使用 $\mathcal{D}' = \mathcal{D} \setminus \mathcal{G}$ 和 $\mathcal{P}' = \mathcal{P} \setminus \mathcal{P}(\mathcal{G})$ 来表示剩余圆盘的集合和剩余点的集合，其中 $\mathcal{P}(\mathcal{G})$ 表示 $\mathcal{G}$ 中的至少一个圆盘所覆盖的点集。

接下来，我们仔细选择包含最多 ϵC 圆盘的集合 $\mathcal{H} \subseteq \mathcal{D}'$。$\mathcal{H}$ 的目的是将整个实例分解为许多（仍然是常量）小块（子结构），这样每个子结构都可以通过动态规划得到最佳解。[①] 一个困难是子结构不是独立的，可以彼此关联（即，圆盘可以出现在多于一个子结构中）。每个子结构都有一个方向（顺时针或逆时针），并且子结构中的所有圆盘都有一个基于方向的偏序。为了将动态编程技术同时应用于所有子结构，我们必须确保不同子结构中圆盘的顺序彼此是一致的。事实上，选择 $\mathcal{H}$ 以确保全局圆盘顺序一致，这篇论文的主要技术挑战。

假设我们有一个适合我们需要的 $\mathcal{H}$（即剩下的实例 $(\mathcal{D}' \setminus \mathcal{H}, \mathcal{P}' \setminus \mathcal{P}(\mathcal{H}))$ 可以在多项式时间内通过动态规划得到最优解）。让 $\mathcal{S}$ 为剩余实例的最优解。我们的最终解是 $\mathsf{SOL} = \mathcal{G} \cup \mathcal{H} \cup \mathcal{S}$。首先，我们可以看到

$$w(\mathcal{S}) \leq w(\mathsf{OPT} - \mathcal{G} - \mathcal{H}) \leq \mathsf{OPT} - w(\mathcal{G}),$$

因为 $\mathsf{OPT} - \mathcal{G} - \mathcal{H}$ 是该实例的一个可行解 $(\mathcal{D}' \setminus \mathcal{H}, \mathcal{P}' \setminus \mathcal{P}(\mathcal{H}))$。因此，我们有

$$\mathsf{SOL} = w(\mathcal{G}) + w(\mathcal{H}) + w(\mathcal{S}) \leq \mathsf{OPT} + \epsilon C w_t \leq (1+\epsilon)\mathsf{OPT},$$

其中因为 $|\mathcal{H}| \leq \epsilon C$，从而第二到最后一个不等式成立最后一个不等式基于事实 $\mathsf{OPT} \geq w(\mathcal{G}) \geq Cw_t$。

构造 $\mathcal{H}$: 现在，我们简要介绍一下如何构建 $\mathcal{H} \subseteq \mathcal{D}'$。首先，我们将块分割成边长为 $\mu = O(\epsilon)$ 的**小方块**，这样任何中心在该小方块中的圆盘可以覆盖整个小方块，并且在同一个小方块中的圆盘圆心足够靠近。$\Xi = \{\Gamma_{ij}\}_{1 \leq i,j \leq K}$ 构成一套小方块，其

① 可以使用类似于[5,16]的动态规划来解决单个子结构。

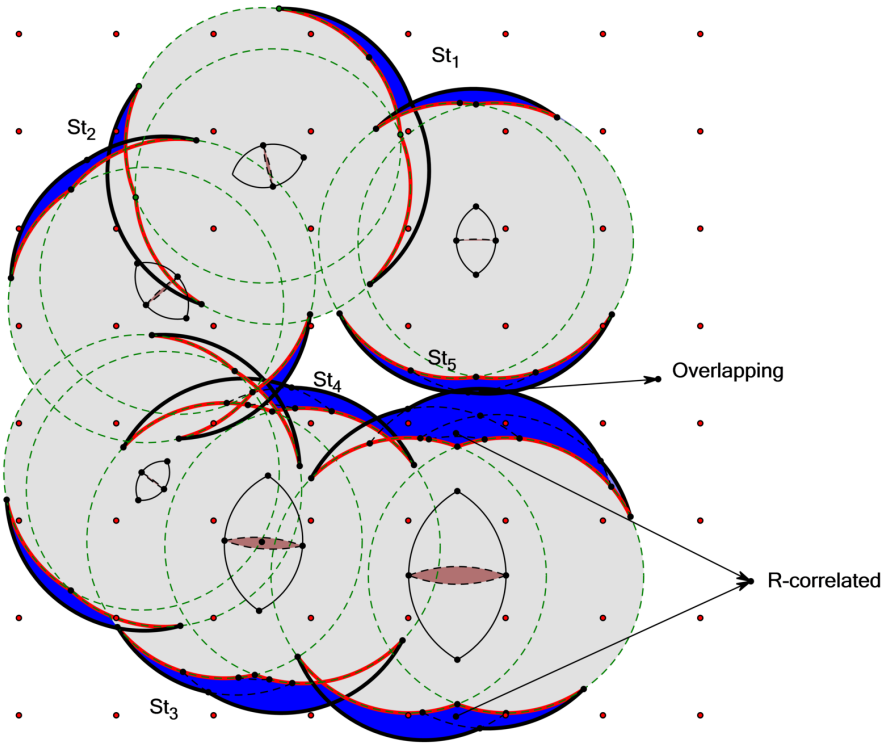


Figure A.2　块中子结构的示意图。红点是小方格的网格点。绿色圆盘是我们选择了的 $\mathcal{H}$ 集合。该图中有五个子结构。

中 $K = L/\mu$。对于一个小方形 Γ，让 $\mathsf{D}_{s_\Gamma} \in \Gamma$ 和 $\mathsf{D}_{t_\Gamma} \in \Gamma$ 表示其中圆心最远的一对圆盘（即 $|D_{s_\square} D_{t_\Gamma}|$ 是最大化的）。我们将这对圆盘 D_{s_Γ} 和 D_{t_Γ} 加入 $\mathcal{H}$ 中，对于每个小方格 $\Gamma \in \Xi$，称 Γ 中这两个圆盘并集合为该小方块的**小构件**。有关示例，请参见图 A.1。然后，我们只需要关注其余未被覆盖的区域 $\mathbb{U}(\mathcal{H})$ 中的点集合。

我们考虑所有圆心都在一个小方块中的圆盘。这些圆盘的未覆盖区域是两个不相交的连通区域。（请参见图 A.1的右侧，两个阴影区域）。我们把这样一个区域和所有相关的弧称为**子构件**（在完整版中会给出正式定义）。事实上，使用动态规划，我们可以解决单个子结构中的圆盘覆盖问题（类似于在[5,16] 中的动态规划）。目前为止，我们离解决该问题已经不远了。因为“直观地”，所有方形的小构件已经覆盖了整个区块的很多区域，我们应该可以使用类似的动态规划来处理所有这样的子结构。但是，情况比我们最初预计的更复杂，因为弧是依赖的。看到图 A.2为一个“不太复杂”的例子。首先，当圆盘居于**核心区**时，如同图A.1）所示，可能存在两个弧（称为**兄弟弧**）属于同一个圆盘。动态规划必须为属于同一个圆却属于两个不同的子结构（称为 R-相关子结构）的两个兄弟弧同时作出决定。其次，为了实施一个动态规划，我们需要所有弧的具有合适顺序。为确保这种顺序的存在，我们需要所有子结构有比较好的关联关系。

特别的，除了小构件外，我们还需要添加到 $\mathcal{H}$ 中恒定数量的额外圆盘。这是通过一系列“切割”操作完成的。切割可以打破一个循环，或者划分一个子结构为两个子结构。为了掌控子结构之间的相互关系，我们定义一个辅助图，称为子结构关系图 $\mathfrak{S}$。其中每个子结构都是一个节点。上述 R-相关定义了一组蓝色边，几何上的重叠关系定义一组红色边。通过分割操作，我们可以使蓝色边形成匹配，并且也可以让红色边形成匹配，并且 $\mathfrak{S}$ 无环的。我们称 $\mathfrak{S}$ 为非循环 2 匹配。$\mathfrak{S}$ 的特殊结构允许我们轻松定义所有弧的排序，再加上其他一些简单的属性，我们可以同时将动态程序从一个子结构推广到所有子结构。

小构件

我们讨论与小方块 Γ 关联的小构件 $\mathsf{Gg}\Gamma$ 的结构。回想一下，正方形小工具 $\mathsf{Gg}\Gamma = \mathsf{D}_s \; cup \mathsf{D}_t$，其中 D_s 和 D_t 是 Γ 中最远的一对圆盘。我们可以看到，对于在 Γ 中的任意圆盘 D_i ，$\partial\mathsf{D}_i$ 有一段或两段弧线不被 $\mathsf{Gg}(\Gamma)$ 覆盖。不失一般性，假设 D_sD_t 是水平的。D_sD_t 将整个平面分成两部分，由 H^+（上半平面）和 H^-（下半平面）表示的半平面。$\partial\mathsf{D}_s$ 和 $\partial\mathsf{D}_t$ 在两点 P 和 Q 处相交。下面，我们定义一些有用的概念。

1. (中心区和核心区) 将 $\mathsf{Gg}(\Gamma)$ 的*中心区*定义为在 Γ 中两个圆盘 $\mathsf{D}(D_sr_{st})$ 和 $\mathsf{D}(D_tr_{st})$ 的交集，其中 $r_{st} = |D_sD_t|$。我们使用 $\mathfrak{C}$ 来表示它。由于 D_s 和 D_t 是最远的一对，我们可以得到 Γ 中的每个其他圆盘的圆心都位于中心区域 $\mathfrak{C}$ 中。我们定义 Gg 的*核心区* (Γ) 为两个分别以 PQ 为中心的单位圆盘的交集。基本上，任何中心区域位于核心区域的单位盘都与小构件的边界有四个交点。让我们用 $\mathfrak{C}_o$ 来表示该区域。
2. (活动区) 考虑如下区域 $\left(\bigcup_{D_i\in\mathfrak{C}_o}\mathsf{D}_i - (\mathsf{D}_s\cup\mathsf{D}_t)\right)\cap H^+$ 及 $\left(\bigcup_{D_i\in\mathfrak{C}_o}\mathsf{D}_i - (\mathsf{D}_s\cup\mathsf{D}_t)\right)\cap H^-$. 我们称他们为和小方块 Γ 关联的*活动区*。我们用 Ar 来表示活动区。

子结构

最初，$\mathcal{H}$ 包含所有小构件。此外 $\mathcal{H}$ 中还包含固定数量的额外圆盘。对于一组圆盘 S，我们使用 $\mathbb{R}(S)$ 来表示在 S 中圆盘的覆盖区域, 即 $\cup_{\mathsf{D}_i\in S}\mathsf{D}_i$。给定一个固定的 $\mathcal{H}$，我们现在描述未覆盖区域的基本结构 $\mathbb{R}(\mathcal{D}') - \mathbb{R}(\mathcal{H})$。[①] 为了便于表示，我们使用 $\mathbb{U}(\mathcal{H})$ 来表示未覆盖的区域 $\mathbb{R}(\mathcal{D}') - \mathbb{R}(\ calH)$。图A.2是一个例子。直观地说，区域由 $\mathcal{H}$ 边界上的几个“条带”组成。现在，我们定义一些概念来描述这些条

① 回想一下 $\mathcal{D}' = \mathcal{D}\setminus\mathcal{G}$，其中 $\mathcal{G}$ 是 OPT 中权重最大的 C 个圆盘。

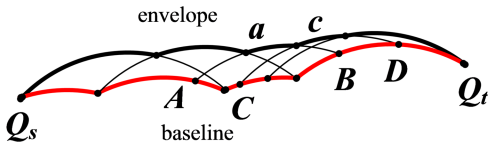


Figure A.3　子结构。基线 b 包含所有红色的弧。Q_s, Q_t 是基线 b 的端点。黑色的弧位于未覆盖区域。

的结构。

1. (基线) 我们使用 $\partial\mathcal{H}$ 来表示 $\mathcal{H}$ 的边界。考虑一个弧 a，其端点 P_1P_2 在 $\partial\mathcal{H}$ 上。如果 P 位于沿 $\partial\mathcal{H}$ 的 P_1 和 P_2 之间的段中, 我们说圆弧 a 在 $\partial\mathcal{H}$ 中覆盖了一个点 P，如果 $\mathcal{D}'$ 覆盖 P，我们说在 $P \in \partial\mathcal{H}$ 可以被覆盖。基线是 $\partial\mathcal{H}$ 上可以被覆盖的连续的最大部分。我们通常使用 b 来表示基线。
2. (子结构) 子结构 St(b, $\mathcal{A}$) 由基线 b 和可以覆盖 b 中任意点的圆弧的集合 $\mathcal{A}$ 组成。每个弧 $a \in \mathcal{A}$ 的两个端点都在 b 上，并且弧的角度 $\angle(a)$ 小于 π。请注意，b 的每个点都被 $\mathcal{A}$ 中的某个弧所覆盖。图A.3演示了一个子结构。

简化问题

子结构可以以各种方式重叠。正如我们在概览中所提到的那样，我们需要在 $\mathcal{H}$ 中包含更多的圆盘，以便使子结构适合动态规划技术。这一步有些复杂，我们决定推迟到结尾描述。相反，我们在本节中介绍在 $\mathcal{H}$ 中包含更多圆盘后，子结构具有的性质。

非自相交性:

在子结构 St 中，假设 $\mathcal{A}$ 中有两个弧段 a 和 c，并带有端点 A, B 和 C, D。如果 $A \prec B \prec C \prec D$，并且 a 和 c 至少覆盖 $calP$ 中有一个相同的点，我们称子结构是自相交。我们将消除所有自相交的子结构，在本节的其余部分，我们假设全部子结构是非自交的。

顺序一致性:

每个子结构的基线有两种可能的方向（顺时针或逆时针），子结构之间有两种类型的关系影响子结构方向的选择。一个是重叠关系，另一个是远距关系。若两个子结构覆盖相同的点，则我们称两个子结构是重叠的。远距关系的定义如下:

Definition A.3 (远距关系)： 考虑两个非重叠的子结构 St_u 和 St_l，若他们包含同一个小构件的不同的活动区域，则我们称这两个子结构存在远距关系。

我们要求重叠的两个子结构具有想反的方向，具有远距关系的两个子结构也

具有相反方向。为了设计一个能同时应用在所有子结构上的动态规划，我们需要子结构具有全局一致性。

Definition A.4 (全局一致性)： 如果存在一个方向的分配方案，使得重叠的两个子结构具有想反的方向，具有远距关系的两个子结构也具有相反方向，我们称所有的子结构具有全局一致性。

子结构关系图 $\mathfrak{S}$: 我们构造一个图 $\mathfrak{S}$ 来刻画子结构的关系，称之为 *子结构关系图*。$\mathfrak{S}$ 中的每个点代表一个子结构。如果两个子结构是远距相关的，则用一条蓝色边连接，若两个子结构是重叠的，则用一条红边相连。

Definition A.5 (无环 2 匹配)： 我们说子结构关系图 $\mathfrak{S}$ 是非循环 2 匹配，如果 $\mathfrak{S}$ 是非环的并且红色边和蓝色边分别构成匹配。换句话说，$\mathfrak{S}$ 只包含路径，红色边和蓝色边缘交替出现在每个路径中。

介绍了所有相关概念，我们终于可以声明动态规划所需的一组属性。

Lemma A.3： 再选择了 $\mathcal{H}$ 后，我们可以保证如下属性：

P1. (活动区域唯一性) 每个子结构包含最多一个的活动区域。

P2. (子结构非自交) 每个子结构都是不自交的。

P3. (无环 2 匹配) 子结构关系图 $\mathfrak{S}$ 是一个无环 2 匹配，即 $\mathfrak{S}$ 只包含路径。在每条路径中，红色边缘和蓝色边缘交替出现。

P4. (点序一致性) 任何点最多由两个子结构覆盖，点满足点顺序一致性。

我们会在完整版中介绍如何构造 $\mathcal{H}$ 集合。

动态规划

假设我们已经构造了集合 $\mathcal{H}$，使得引理A.3成立。不失一般性，我们可以假设剩下的圆盘可以涵盖所有剩余的点（否则，原始实例是不可行的或我们的猜测是错误的）。事实上，我们的动态规划受到了前任工作的启发。[5-6,16]。

我们需要动态规划指示变量来标记一次状态跳转是否合法。如果 $\mathcal{P}[\{P_k\}_{k\in[m]}] = \mathcal{P}[[\mathcal{P}_k][P_i^b]_{\{i\}}]$, 则 $I_i = 0$. 否则, 否则让 $I_i = 1$。其中我们利用了简化的公式缩写, 对于集合 $\{e_k\}_{k\in[m]}$ 和 $S \subseteq [m]$, 我们标记 $[e_k][e_i']_S = \{e_k\}_{k\in[m]\setminus S} \cup \{e_i'\}_{i\in S}$. 因此 $[P_k][P_i^b]_{\{i\}} = \{P_k\}_{k\in[m]\setminus i} \cup P_i^b$ 和 $[P_k][P_i^t]_{N_\mathrm{p}(\mathsf{D})} = \{P_k\}_{k\in[m]\setminus N_\mathrm{p}(\mathsf{D})} \cup \{P_i^t\}_{i\in N_\mathrm{p}(\mathsf{D})}$. 我们的动态规划形式如下：

$$\mathsf{OPT}\left(\{P_k\}_{k\in[m]}\right) = \min\begin{cases} \min_{i\in[m]}\left\{\mathsf{OPT}\left([P_k][P_i^b]_{\{i\}}\right) + I_i\cdot\infty\right\}, & \text{add no disk} \\ \min_{\mathsf{D}\in\mathcal{R}}\left\{\mathsf{OPT}\left([P_k][P_i^t]_{N_\mathrm{p}(\mathsf{D})}\right) + w_\mathsf{D}\right\}, & \text{add disk } \mathsf{D} \end{cases}$$

WUDC 的 PTAS 算法的应用

单位圆盘图中的加权支配集问题（MWDS）在无线传感器网络领域具有许多应用领域[9]。在本节中，我们显示我们的 WUDC 的 PTAS 可以用于获取这个领域中两个重要问题的更好的近似算法。

单位圆图中的最小带权连通支配集问题

最小带权连通支配集问题 (MWCDS) 的目标是找到一个连通的支配集并且具有最小的权重。Clark 等人[14] 证明了 MWCDS 在单位图中是 NP-难的。Ambühl 等人[5] 获得了 MWCDS 的第一个常数近似算法（常数为 94）。这个常数之后被一系列工作提高[6,30-31]。目前最好的近似比是 7.105[9].

计算 MWCDS 的近似解的方法一种较为通用的办法是先计算最小加权支配集（MWDS），然后使用节点加权斯坦纳树（NWST)[30,72] 连接支配集。最优 MWDS 值不超过最优的 MWCDS 值。在归零所有端点的权量之后，最优的 NWST 值（对于任何一组端点）也不超过最佳的 MWCDS 值。因此，如果 MWDS（或等价的 WUDC）存在 α-近似。和 NWST 的 β-近似，那么就有一个用于 MWCDS 的 $\alpha+\beta$ 因子近似算法。

Zou 等人[72] 证明如果存在一个 ρ 近似的边加权最小斯坦纳树问题，NWST 有一个 2.5ρ 近似算法。当前最小斯坦纳树的最好近似算法的近似比是是 1.39[73]。因此，NWST 存在 3.475 的近似算法。结合我们给出的 WUDC 问题的 PTAS，我们对于 MWCDS 获得以下改进结果。

Theorem A.2： 对于任意常数 $\epsilon > 0$，MWCDS 问题存在一个多项式时间的 $(4.475+\epsilon)$-近似的算法。

单位圆图中的最长生命覆盖问题

最长生命覆盖问题 (MLC) 是无线传感器网络领域的一个经典问题。给定 n 个目标 $t_1 \ldots t_n$ 和 m 个传感器 $s_1 \ldots, s_m$，每个传感器可以覆盖由一些目标构成子集。每个传感器可以使用的最长生命周期为 1。问题的目标是使用这些传感器覆盖一族目标的集合 $S_1 \ldots S_p$，假设覆盖的时间分别为 $\tau_1 \ldots \tau_p$ 其中 $\tau_i \in [0,1]$，使得 $\tau_1 + \ldots + \tau_p$ 最大化。MLC 已知是 NP-难的[74]。Berman 等人通过 Garg-Könemann[76] 技术将 MLC 归约到 最小权量传感器覆盖（MSC）问题。他们证明了如果 MSC 有一个 ρ 近似算法，那么 MLC 对于任何 ϵ 都有一个 $1+\epsilon\rho$ 近似算法。Ding 等人指出[33]，如果所有的传感器和目标位于欧几里德平面上，并且所有传感器的覆盖半径相同，那么 WUDC 的任何近似结果都可以转换成 MLC 的相同的近似解。因此，MLC 当前

最好结果是 $(3+\epsilon)$ 近似值[34]。使用我们的 PTAS，我们获得了第一个用于 MLC 的 PTAS。

Theorem A.3： 当所有传感器和目标位于欧几里德平面上，并且所有传感器具有相同的覆盖半径时，存在用于 MLC 的 PTAS。

让我们再提一个 MLC 的另一个变种，称为最大生命连通覆盖问题。Du 等人最先研究了该问题[77]。问题设置与 MLC 相同，除此之外每个传感器覆盖 S_i 应该是一个连接的子图。当通信半径 R_c 不少于两倍的感测半径 R_s 时，他们获得 $(7.105+\epsilon)$ 近似解。他们通过 WUDC 问题的一个 α 近似的算法以及 NWST 的 β 近似解证明了连通的 MLC 问题，存在 $\alpha+\beta$ 近似解。使用我们的 PTAS，我们可以将近似比提高到 $(4.475+\epsilon)$。

PTAS 算法总结

大部分的技术性技巧源于子结构相互关联的事实，要确保全局一致的顺序并不容易。读者可能想知道，如果我们在一个小方块中选择两个以上的圆盘（但仍然是一个常量），希望未被覆盖的地区变得分离和更易于管理。我们尝试了其他几种方法，例如在小方块中的圆盘的凸包中选择恒定数量的圆盘。但是，这些似乎只会使事情复杂化，而不是简化。

我们相信我们的结果和见解对解决其他涉及单位圆盘或单位圆图问题很有用。另一方面，我们的方法强烈依赖于的单位圆的特殊属性，并且似乎不能推广到具有不同半径的任意圆盘上。使用任意圆盘获得加权圆盘覆盖问题的 PTAS 仍然是这个领域的核心问题。一个有趣的中间步骤是考虑其中的特殊情况，比如最长半径和最短半径之间的比率是有界的。

t-跨度图

假设 $\mathcal{P}$ 是欧几里德平面 $\mathbb{R}^2$ 中的一组点。定义在顶点集合 $\mathcal{P}$ 上的完全欧几里德图是个带权图，其中边集合包含连接所有点对的边，每个边的权重是其两个端点之间的欧几里德距离。存储完整的完全欧几里德图需要二次的空间复杂度，这个代价很大。因此，希望使用稀疏子图来逼近完整的图。这是计算机领域的一门经典而深入研究的课题（参见[4,38-41]）。在本论文 b 中，我们研究了几何 t-跨度图（*geometric t-spanner*），正式定义如下（参见例如[42]）。

Definition A.6： (几何 t-跨度图) 图 G 是完整的欧几里德图的几何 t-跨度图（1）G 是完全欧几里德图的子图；（2）对于 $\mathcal{P}$ 中的任何一对点 p 和 q，在 G 中的 p 和 q 之间的最短路径不会超过 p 和 q 之间的欧几里德距离的 t 倍。

在文献中，因子 t 被称为跨度图的 *拉伸因子*或 *膨胀因子*。如果 G 的最大度数以一个常数 k 为界，那么我们说 G 是一个 *常数有界的跨度图*。几何跨度图的概念最初由 L.P.Chew[43] 提出。可以参阅 Eppstein 的综述文献[44]，了解有关几何跨度图的更详细信息。几何跨度图已经在无线自适应网络和传感器网络中存在许多应用。我们推荐读者参阅 Li[45], Narasimhan 和 Smid[46] 的著作，了解更多细节。

Yao 图是完全欧几里德图的一种近似，由 Flinchbaugh 和 Jones[47] 和 Yao[41] 独立提出。

Definition A.7 (Yao 图 Y_k)： 让 k 是一个固定的整数, 在欧几里德平面 $\mathbb{R}^2$ 中给出一组点 $\mathcal{P}$，Yao 图 $\mathsf{Y}_k(\mathcal{P})$ 定义如下: 假设 $C_u(\gamma_1, \gamma_2]$ 是以 u 为定点的圆锥体, 它由在半开放区间 $(\gamma_1, \gamma_2]$ 中以 u 为定点的射线组成。对于每个点 $u \in \mathcal{P}$，$\mathsf{Y}_k(\mathcal{P})$ 包含将 u 连接到的每个圆锥体 $C_u(j\theta, (j+1)\theta)$ 最邻近点 v 的边, 其中 $\theta = 2\pi/k$，$j \in [0, k-1]$。我们通常将 Yao 图看作无向图。对于*有向 Yao 图*，我们添加有向边 $\overrightarrow{uv}$ 代替无向边 uv。

Molla[48] 证明了 Y_2 和 Y_3 不是跨度图。另一方面，已经证明所有的 Y_k for $k \geq 4$ 均是跨度图。其中 Bose 等人证明 Y_4 是 663-跨度图。Damian 和 Nelavalli[50] 最近将常数改进为 54.6。Barba 等人[51] 证明 Y_5 是 3.74-跨度图。Damian 和 Raudonis[52] 证明了 Y_6 图是一个 17.64-跨度图。Li 等人首先证明了所有 $\mathsf{Y}_k, k > 6$ 都是拉伸系数最多为 $1/(1 - 2\sin(\pi/k))$ 的跨度图。后来 Bose 等人[49,54] 也得到了相同的独立结果。最近，Barba 等人提出了减小拉伸因子的方法, 将 Y_6 的拉伸系数从 17.6 降到 5.8，并将 $k \geq 7$ 且 k 为奇数的拉伸因子改进为 $1/(1 - 2\sin(3\pi/4k))$。

但是，Yao 图可能没有有界度。这在一些无限网络应用中可能是一个严重的限制因为每个节点的网络具有非常有限的能量和通信能力，从而可以只与少数邻居进行交流。为了解决这个问题，Li 等人引入了 Yao-Yao 图（或者称作稀疏 Yao 图）。通过从 $\mathsf{Y}_k(\mathcal{P})$ 中删除一些边，得到一个 Yao-Yao 图 $\mathsf{YY}_k(\mathcal{P})$, 如下所示：

Definition A.8 (Yao-Yao 图 YY_k)：（1）根据定义A.7，构造有向 Yao。（2）对于每个节点 u 以及以 u 为顶点的每个锥，若包含两个或更多入射边，则保留最短入射边并放弃锥中的其他入射边。可以看出 $\mathsf{YY}_k(\mathcal{P})$ 中的最大程度上限为 $2k$。

与 Yao 图不同的是，Yao-Yao 图是否为跨度图尚未得到很好的理解。Li 等人提供了一些经验证据，暗示对于一些足够大的常量 k，YY_k 图可能是 t-跨度图。但是，

他们并没有给出严格的理论证明。目前为止，它仍然是一个开放的问题[2-4,45]。在 Demaine 等人维护的开放问题项目[①]中它也被列为问题 70。

Conjecture A.1 (见文献[2])：存在一个常数 k_0，使得对于任何整数 $k > k_0$，任何 Yao-Yao 图 YY_k 都是一个几何跨度图。

现在，我们简要回顾一下关于 Yao-Yao 图的结果。YY_2 和 YY_3 可能不是跨度图，因为 Y_2 和 Y_3 可能不是跨度图[48]。Damian 和 Molla[48,55] 证明了 YY_4, YY_6 可能不是跨度图。Bauer 等人[51] 证明 YY_5 可能不是跨度图。在积极的方面，Bauer 和 Damian[2] 证明, 对于任何整数 $k \geq 6$，任何 Yao-Yao 图 YY_{6k} 是一个跨度图拉伸系数最高为 11.67，而 $k \geq 8$ 则系数为 4.75。最近，Li 和 Zhan[3] 证明对于任何整数 $k \geq 42$，任何 Yao-Yao 图 YY_{2k} 是一个拉伸因子 $6.03 + O(k^{-1})$ 的跨度图。

从这些积极的结果来看，相信猜想 A.1是相当合理的。然而，我们在这篇论文中表明，令人惊讶的是，猜想 A.1对于奇数度 Yao-Yao 图是错误的。

Theorem A.4：对于任何 $k \geq 1$，都存在一类点集实例 $\{\mathcal{P}_m\}_{m\in\mathbb{Z}^+}$。当 m 接近无穷大时，图 $\mathrm{YY}_{2k+1}(\mathcal{P}_m)$ 的拉伸因子不能被任何常量限制。[②]

相关工作 在一些特殊情况下，可以证明 Yao-Yao 图是跨度图[56-59]。特别的，在 civilized 图中（其中最长边的长度与最短边长度之间的比例由一个常数约束[56-57]），YY_k 图是跨度图。除了 Yao 图和 Yao-Yao 图之外，Θ-图是另一个常见的几何 t-跨度图。Θ-图和 Yao 图之间的区别在于 Θ-图中，圆锥体 C 中 u 的最近邻点是位于 C 中的一个点 $v \neq u$，并且 u 与 v 到 C 的平分线上的正交投影之间的距离时最小的。已经证明，Θ_2 和 Θ_3[48] 除外，对于 $k = 4$[60]，5[61]，6[62]，≥ 7[63-64]，Θ_k 图都是几何跨度图。然而我们也需要注意，和 Yao 图一样，Θ 图的度可能不是有限的。

最近，几何 t-跨度图的一些变体，如弱 t 跨度图（weak t-spanner）和幂 t-跨度图（power t-spanner）也受到很大关注。在弱 t 跨度图中，两点之间的路径可能是任意长，但必须包含在半径长度为两点欧几里德距离 t 倍的圆盘内。对于 $k > 6$，所有的 Yao-Yao 图 YY_k 都是弱 t-跨度图。[65-67]。在幂 t 跨度图中，欧几里德距离 $|\cdot|$ 被替换为 $|\cdot|^\kappa$，其中 $\kappa \geq 2$。Schindelhauer 等人[66-67] 证明了对于 $k > 6$，所有的 Yao-Yao 图 YY_k 对于某些常数 t，是幂 t 跨度图。此外，任何 t-跨度图，同时也是一个弱 t_1 跨度图和幂 t_2 跨度图，其中 t_1, t_2 的值仅取决于 t。然而，反过来不正确[67]。

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

② 这里，m 是递归构造中的一个参数。我们将在完整版中详细解释。粗略地说，m 是递归级别，$\mathcal{P}_m$ 中的点数随 m 增加。

我们的反例得益于分形概念的启发。在构造 β-框架图（skeleton）非常数有界拉伸因子的反例中也有用到分形的概念[68]。这里 β-框架图包含所有边 ab 使得这样没有一个点 c 形成一个角 $\angle acb$ 大于 $\sin^{-1} 1/\beta$ 当 $\beta > 1$ 或 $\pi - \sin^{-1}\beta$ 当 $\beta < 1$。Schindelhauer 等人[67] 使用相同的例子来证明存在图是弱 t 跨度图，但不是 t 跨度图。然而，他们的例子不能作为奇数度 Yao-Yao 图猜想的反例。

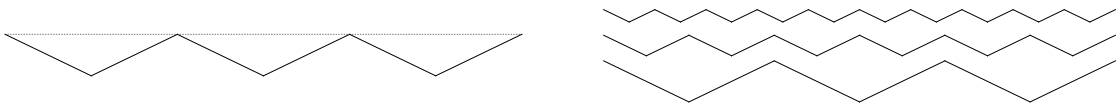
奇数度 Yao-Yao 图不是 t-跨度图

我们首先注意到 YY_3 和 YY_5 的反例都不是弱 t-跨度图[48,51]。然而，Yao-Yao 图 YY_k 对于 $k \geq 7$ 都是弱 t-跨度图[65-67]。因此，为了构造 YY_k，$k \geq 7$ 的反例，以前用在 YY_3 和 YY_5 上的想法就不能使用了。我们将构造一类实例 $\{\mathcal{P}_m\}_{m\in\mathbb{Z}^+}$，$\mathcal{P}_m$ 中的所有点都放置在一个有界区域中。同时，$\mathsf{YY}_{2k+1}(\mathcal{P}_m)$ 中存在两点的最短路径，当 m 接近无穷大时，其长度接近无穷大。

我们的例子包含两种类型的点，称为*标准点*和*辅助点*, 分别用 $\mathcal{P}_m^{\mathrm{n}}$ 和 $\mathcal{P}_m^{\mathrm{a}}$ 来表示它们。用 $\mathcal{P}_m = \mathcal{P}_m^{\mathrm{n}} \cup \mathcal{P}_m^{\mathrm{a}}$ 表示所有点的集合。标准点形成基本的骨架，辅助点用于打破连接任何两个相距较远的标准点的边。

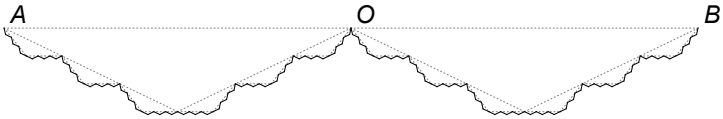
我们受分形概念的启发来构建标准点。分形可以包含在有界区域内，但其长度可能不收敛。在我们的反例中，两个特定标准点的最短路径是分形的多边形路径。这里，多边形路径是指由指定的点的序列并由连接连续点的线段组成。假设两个特定点是 A 和 B，AB 是水平的，并且 $|AB| = 1$。当 $m = 0$ 时，多边形路径就是线段 AB。当 m 增加 1 时，我们将当前多边形路径中的每个线段替换为*类似锯齿状的路径*（见图 A.4(a)）。如果锯齿状路径的每个部分与基部段之间的角度（即被替换线段）是 γ，路径的总长度增加了 $\cos^{-1}\gamma$ 倍。这里一个重要的观察结果是该因素与锯齿的数量无关（见图 A.4(b)）。如果我们直接重复这个过程，由此产生的路径的长度将增加到无穷大 m 当 $\cos^{-1}\gamma > 1$（见图 A.4(c)）接近无穷大。但是，我们需要确保在 Yao-Yao 图中，它确实是从 A 到 B 的最短路径，我们需要克服两个技术难题。

1. 当 m 增加时，多边形路径可能相交。见图 A.4(d)。多边形路径在点 O 附近相交。这相对容易处理: 我们不递归划分那些可能导致自相交的线段。见图A.4(e)。我们不会进一步代替粗线段。我们需要确保这些不再细分的线段的总长度相对较小（以便剩余细分部分的总长度可以随 m 的增大保持而不断增加）。
2. 在递归构造的标准点上定义的 Yao-Yao 图中，可能会存在一些边连接相距很远的点。如何破坏这些相距较远的边是这个问题的主要难点。下面，我们概

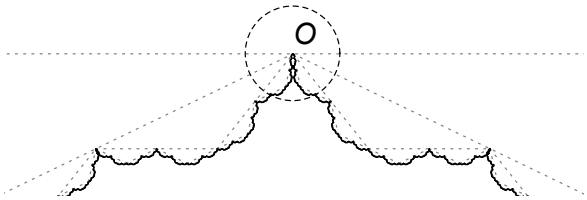


(a) 由锯齿状路径代替水平线段

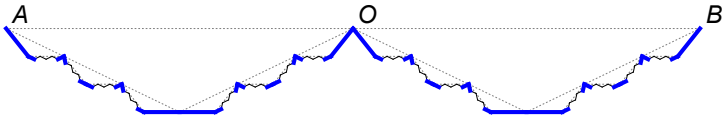
(b) 锯齿状路径的长度与锯齿个数无关



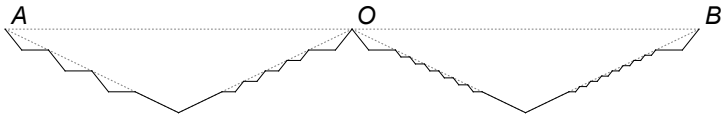
(c) 递归的用锯齿路径代替线段



(d) 图 A.4(c)在 O 点附近的局部放大。



(e) 对于加粗线段，不继续递归替代



(f) 替代的锯齿路径锯齿个数和大小都不一定相同

Figure A.4　反例构造概览。图A.4(a)-A.4(f) 说明了分形及其变种。

述我们克服这个难点的主要技术。

首先，我们不会像通常的分形结构那样，使用相同的锯齿来替换所有的当前分段。对于每个细分线段，我们将选择一个多边形路径，该路径的锯齿数量可能不同，锯齿在路径上的大小也可能不一样。图A.4(f)给出了说明。最后，我们以特定的顺序来构建它们。实际上，我们用一棵 m 层的 *递归树* $\mathcal{T}$ 中组织这些标准点，并用树的前序深度优先遍历顺序来生成它们。我们在完整版中会描述这些细节。

其次，我们将标准点分成一组集合，使得每个正常点属于且恰好属于一组。我们称这样一个集合为*铰链集合*。参考图A.5。然后，我们指定铰链集合的*全序*。称 Yao-Yao 图 $\mathrm{YY}_{2k+1}(\mathcal{P}_m^{\mathrm{n}})$ 中连接同一个铰链集合中的任意两个标准点或两个相邻的铰链集合（对于全序关系）中任意两个标准点为*铰链连接*。称其他边为*长距离连接*。我们将在完整版中描述这些细节。

我们会在完整版中看到的，所有可能的长距离连接都有一个相对简单的形式。然后，我们证明我们可以通过添加一个*辅助点集* $\mathcal{P}_m^a$ 来打破所有长距离连接。每个

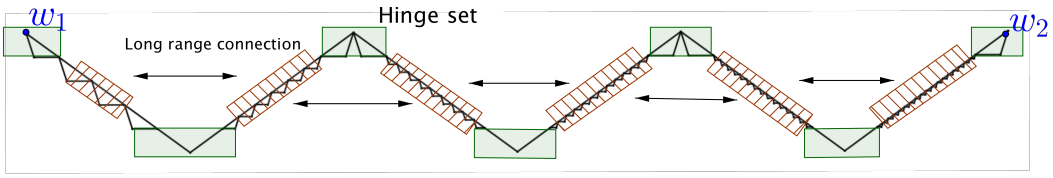


Figure A.5　铰链集合划分。粗略的讲，每个由绿色矩形覆盖的点集合是一个铰链集合。递归的，我们可以进一步解构被阴影矩形的点集合，并把它们划分为铰链集合。铰链连接指的是连接同一个铰链集合中的任意两个标准点或两个相邻的铰链集合（对于全序关系）中任意两个标准点的边。其他的边被称为长距离连接。

辅助点都有一个特定的*中心*，即它最接近的标准点。我们将 $\mathcal{P}_m^{\mathrm{n}}$ 中的任何两个标准点之间的最小距离记为 Δ。辅助点与其中心之间的距离远小于 Δ。我们可以扩展铰链集合的概念为*扩展铰链集合*使其包含铰链集合中的标准点和以这些标准点为中心的辅助点。我们将会看到辅助点打破了所有的长距离连接并且没有引入新的长连接。我们将在完整版中描述这些细节。

最后，根据上面的过程，我们可以看到在 $\mathrm{YY}_{2k+1}(\mathcal{P}_m), m \in \mathbb{Z}^+$ 中标准点 A 和 B 之间的最短路径通过依次所有扩展铰链集合。因此，A 和 B 之间的最短路径的长度 b 随着 m 接近无穷大而发散。我们将在完整版中描述这些细节。

标准点的位置

A.0.0.1　一些基本概念

设 $k \geq 3$ 为一固定正整数。[①]。我们考虑 YY_{2k+1} 并让 $\theta = 2\pi/(2k+1)$。

Definition A.9 (锥边界)：　考虑两个点 u 和 v。如果 $\overrightarrow{uv}$ 的极角是 $j\theta = j \cdot 2\pi/(2k+1)$，对任意整数 $j \in [0, 2k]$，我们称射线 $\overrightarrow{uv}$ a cone boundary for point u.

Property A.4：　考虑 $\mathcal{P}$ 中两点 u 和 v。如果 $\overrightarrow{uv}$ 是 $\mathrm{YY}_{2k+1}(\mathcal{P})$ 一条锥边界，则它的反向射线 $\overrightarrow{vu}$ 不是一条锥边界。

回顾我们的构造过程，这个属性是奇数和偶数 Yao-Yao 图之间的一个关键区别，而我们对奇 Yao-Yao 图的反例将充分利用该属性。

Definition A.10 (边界对)：　一个边界对包含有序的两个点, 记做 (w_1, w_2), 使得 $\mathrm{t}\overrightarrow{w_1w_2}$ 是以 w_1 为起点的一个锥边界。

根据属性A.4，如果 (w_1, w_2) 是一个边界对, 它的逆 (w_2, w_1) 则不是。进一步的，如果一个对 ϕ 是 $(u, \cdot)$ 或 $(\cdot, u)$，我们称 u 属于对 ϕ (即，$u \in \phi$)。

① 注意到 $k = 1, 2$ 已经得到证明 [48]

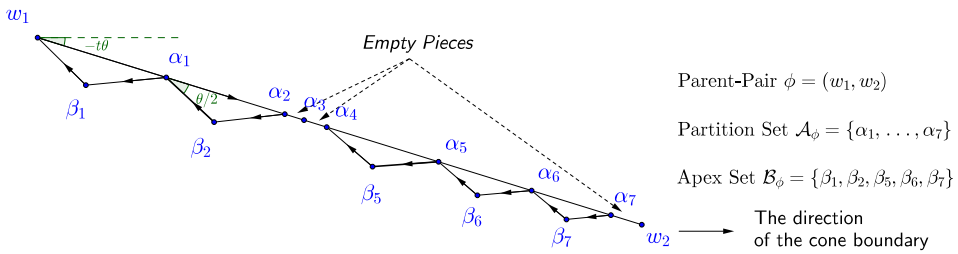


Figure A.6　一个构件的实例。$\phi = (w_1, w_2)$ 是小构件中的一个父亲对。$\mathcal{A}_\phi = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_7\}$ 是集合 partition，$\mathcal{B}_\phi = \{\beta_1, \beta_2, \beta_5, \beta_6, \beta_7\}$ 是集合 apex。存在 8 个小段，其中 $w_1\alpha_1, \alpha_1\alpha_2, \alpha_4\alpha_5, \alpha_5\alpha_6, \alpha_6\alpha_7$ 是非空小段，$\alpha_2\alpha_3, \alpha_3\alpha_4, \alpha_7 w_2$ 是空段。

Gadget　一个小构件 Gg_ϕ 包含三组点。我们一一解释如下。图A.6给出了一个实例。

1. 第一组点事对 $\phi = (w_1, w_2)$. 我们称对 ϕ 为构件 Gg_ϕ 的父对。
2. 第二组是线段 (w_1, w_2) 上的点集合 $\mathcal{A}_\phi$。我们称集合 $\mathcal{A}_\phi$ 为 划分集合，并称点集 $\mathcal{A}_\phi$ 中的点为对 ϕ 的 划分点。集合 $\mathcal{A}_\phi$ 将线段划分为 $|\mathcal{A}_\phi| + 1$ 段，我们称每一段为线段的一个 小段。有两类小段。其一被称为 空段，其二被称作 非空段。一个段是否为空，我们将在构造过程中决定，我们将在完整版中描述。
3. 对于每个非空小段 $\alpha_{i-1}\alpha_i$，我们增加一个点 β_i 使得 $\angle\alpha_{i-1}\beta_i\alpha_i = \pi - \theta$ 和 $|\alpha_{i-1}\beta_i| = |\beta_i\alpha_i|$。所有的 β_i 在 $w_1 w_2$ 的同测。我们称这个点 β_i 为 (w_1, w_2) 的 顶点。让 $\mathcal{B}_\phi$ 为 ϕ 的顶点的集合，称作 顶点集合。$\mathcal{B}_\phi$ 即为第三组点。对于空的小段，我们不增加对应的顶点。

考虑 $\mathsf{Gg}_\phi[\mathcal{A}_\phi, \mathcal{B}_\phi]$, 其中 $\phi = (w_1, w_2)$。对于任何非空段 $\alpha_{i-1}\alpha_i$ 及对应的顶点 β_i，射线 $\overrightarrow{\beta_i\alpha_{i-1}}$ 和 $\overrightarrow{\alpha_i\beta_i}$（注意点的顺序）是锥边界。因此每个点 $\beta_i \in \mathcal{B}_\phi$ 都规约出两个对 (β_i, α_{i-1}) 和 (α_i, β_i)。我们称所有由 $\mathcal{B}_\phi$ 中的点生成的对 (β_i, α_{i-1}) 和 (α_i, β_i) 为 (w_1, w_2) 的 子对, 并且我们称他们互为兄弟对。注意到有些划分点没有关联任何对(例如图A.6中的点 α_3)。我们称其为孤立点。

Definition A.11 (子对的顺序)：　考虑构件 $\mathsf{Gg}_{(w_1,w_2)}$，假设 Φ 是 (w_1, w_2) 衍生的对集合。考虑 Φ 中的两个对 ϕ, φ，定义对的顺序为 $\phi \prec \varphi$, 如果 ϕ 比 φ 更接近 w_1。这里从对 ϕ 到点 w 的距离指的是从 w 到对 ϕ 中两点的最短距离。

A.0.0.2　标准点构造

在这个小节中，我们构造一个 m 层的树。当递归级别增加 1 时，我们需要将每个当前对替换为该对产生的小构件。见图 A.7为例。递归过程可以是自然的表示为树 $\mathcal{T}$。树根据深度优先先序遍历生成，从根开始。我们假设 $\mathcal{T}$ 的根节点是 (μ_1, μ_2)，

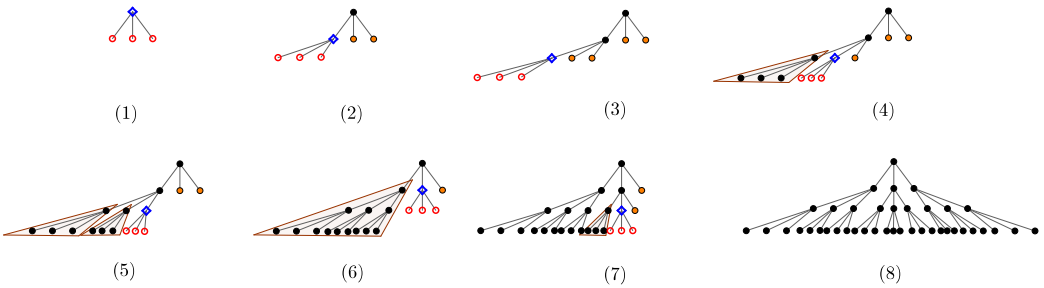


Figure A.7　根据深度优先前序遍历生成树。在每幅子图中 ◇ 表示一个我们正在遍历的点。用 ○表示这一步生成的点。● 表示已经遍历过的点。● 表示已经生成但是还没有遍历的点。由浅棕色三角形覆盖的点是关联投影过程的点。

并且 $\mu_1\mu_2$ 是水平的。我们称一个节点为叶子对如果它是树中的一个叶节点。同时称一个节点为内部对当它不是叶子节点。每当我们访问一个内部对，我们通过两个步骤生成其小构件，称为投影和细分。生成其小构件等同于在 $\mathcal{T}$ 中生成其子对。创建小构件时确定子对是否为叶子。因此，请注意，并非所有的叶子对都在 m 层。我们这个步骤生成的点为标准点，用 $normp$ 表示这些点的集合，其中 m 是树的层级，n 代表“标准”。

根构件　让 d_0 代表一个正的大的常数。考虑对 $\phi = (\mu_1, \mu_2)$。让 $\mathcal{A}_\phi$ 为他的划分集，

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

为方便起见，令 $\alpha_0 = \mu_1, \alpha_{d_0} = \mu_2$。$\mathcal{A}_\phi$ 将线段 $\mu_1\mu_2$ 划分为 d_0 段，每一段有相同侧长度 $|\mu_1\mu_2|/d_0$。所有的段均为非空段。对段 $\alpha_{i-1}\alpha_i$，我们在 $\mu_1\mu_2$ 下增加顶点 β_i。让 $\mathcal{B}_\phi = \{\beta_i\}_{i\in[1,d_0]}$ 为顶点集合。

投影和细分　投影和细化会生成 ϕ 对的分区点。投影的目的是将所有可能的长距连接限制为一个相对简单的形式。细化的目的是使兄弟对具有近似相同的长度，

决定点对是否为空　考虑对 ϕ，其顶点为 β，划分点为 α。[①]我们让这个块入射到顶点 β 为空，其他片段为非空。

① 注意对中的第一个点可以是顶点或分割点。在这里，$\phi = (\alpha\beta)$。或 $\phi = (\beta, \alpha)$，取决于 ϕ 的第一个点是否为顶点。

对于每个非空片段，我们生成一个顶点。正如我们之前讨论的那样，顶点集 $\mathcal{B}_{pair}$ 产生 ϕ 子对的集合 Φ。让最接近 α 的三对和最接近 β 的两个对是叶子对。我们不从叶对进一步扩展树。让其他对成为内部对。

总体而言，在投影和细化过程之后，我们可以为树中的任何对生成小构件。我们用这个来表示这个过程

$$\mathsf{Gg}_{\phi} \leftarrow \mathsf{Proj\text{-}Refn}(\phi). \tag{A-1}$$

最后通过增加辅助点，我们能证明奇数度 Yao-Yao 图非 t 跨度图。

Lemma A.5：　对于 $k \geq 3$ 的 Yao-Yao 图 $\mathrm{YY}_{2k+1}(\mathcal{P}_m)$，$\mu_1$ 和 μ_2 之间的最短距离至少为 ρ^m, 对于一些 $\rho = (1 - O(d_0^{-1})) \cdot \cos^{-1}(\theta/2)$。当 $d_0 > \lceil 6(1 - \cos(\theta/2))^{-1} \rceil$, 这个长度随 m 的增大而趋于无穷。

Resume and Publications

个人简历

1990 年 07 月 15 日出生于浙江省永嘉县。

2009 年 9 月考入北京大学物理学院天文系，2013 年 7 月本科毕业并获得理学学士学位。

2013 年 9 月免试进入清华大学交叉信息院攻读工学博士学位至今。

发表的学术论文

[1] A PTAS for the Weighted Unit Disk Cover Problem. Jian Li, Yifei Jin. The 42nd International Colloquium on Automata, Languages, and Programming (ICALP 2015)

[2] Odd Yao-Yao Graphs are Not Spanners. Yifei Jin, Jian Li, Wei Zhan. In 34th International Symposium on Computational Geometry (SoCG 2018)

[3] SVM via Saddle Point Optimization: New Bounds and Distributed Algorithms. Lingxiao Huang, Yifei Jin and Jian Li. The Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2018).