A Unified PTAS for Prize Collecting TSP and Steiner Tree Problem in Doubling Metrics

T-H. Hubert Chan[1]

Department of Computer Science, The University of Hong Kong, Hong Kong, China
hubert@cs.hku.hk

Haotian Jiang[2]

Department of Physics, Tsinghua University, Beijing, China
jht14@mails.tsinghua.edu.cn

Shaofeng H.-C. Jiang

The Weizmann Institute of Science, Rehovot, Israel
shaofeng.jiang@weizmann.ac.il
https://orcid.org/0000-0001-7972-827X

Abstract

We present a unified (randomized) polynomial-time approximation scheme (PTAS) for the prize collecting traveling salesman problem (PCTSP) and the prize collecting Steiner tree problem (PCSTP) in doubling metrics. Given a metric space and a penalty function on a subset of points known as terminals, a solution is a subgraph on points in the metric space, whose cost is the weight of its edges plus the penalty due to terminals not covered by the subgraph. Under our unified framework, the solution subgraph needs to be Eulerian for PCTSP, while it needs to be a tree for PCSTP. Before our work, even a QPTAS for the problems in doubling metrics is not known.

Our unified PTAS is based on the previous dynamic programming frameworks proposed in [Talwar STOC 2004] and [Bartal, Gottlieb, Krauthgamer STOC 2012]. However, since it is unknown which part of the optimal cost is due to edge lengths and which part is due to penalties of uncovered terminals, we need to develop new techniques to apply previous divide-and-conquer strategies and sparse instance decompositions.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Doubling Dimension, Traveling Salesman Problem, Polynomial Time Approximation Scheme, Steiner Tree Problem, Prize Collecting

Digital Object Identifier 10.4230/LIPIcs.ESA.2018.15

Related Version A full version of the paper can be found at [11], `https://arxiv.org/abs/1710.07774`.

[1] This work was partially supported by the Hong Kong RGC under the grant 17217716.

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

1 Introduction

We study prize collecting versions of two important optimization problems: the prize collecting traveling salesman problem ($\mathsf{PC}^{\mathsf{TSP}}$) and the prize collecting Steiner tree problem ($\mathsf{PC}^{\mathsf{STP}}$). In both problems, we are given a metric space and a set of points called *terminals*, and a non-negative penalty function on the terminals. A solution for either problem is a connected subgraph with vertex set from the metric. In addition, it needs to be an Eulerian (multi-)graph[3]for $\mathsf{PC}^{\mathsf{TSP}}$ and a tree for $\mathsf{PC}^{\mathsf{STP}}$. The cost of a solution is the sum of the weights of edges in the solution plus the sum of penalties due to terminals not visited by the solution.

Prize Collecting Problems in General Metrics. The prize collecting setting was first considered by Balas [4], who proposed the prize collecting TSP. However, the version that Balas considered is actually more general, in the sense that each terminal is also associated with a reward, and the goal is to find a tour that minimizes the tour length plus the penalties, and collects at least a certain amount of rewards. The setting that we consider was suggested by Bienstock et al. [8], and they used LP rounding to give a 2.5-approximation algorithm for the $\mathsf{PC}^{\mathsf{TSP}}$ and a 3-approximation for the $\mathsf{PC}^{\mathsf{STP}}$. Later on, a unified primal-dual approach for several network design problems was proposed [17]; this approach improves the approximation ratios for both $\mathsf{PC}^{\mathsf{TSP}}$ and $\mathsf{PC}^{\mathsf{STP}}$ to 2 in general metrics. The 2-approximation had remained the state of the art for more than a decade, until Archer et al. [1] finally broke the 2 barrier for both problems. Subsequently, in a note [16], Goemans combined their argument with other algorithms, and gave a 1.915-approximation for the $\mathsf{PC}^{\mathsf{TSP}}$, which is the state of the art.

Prize Collecting Problems in Bounded Dimensional Euclidean Spaces. $\mathsf{PC}^{\mathsf{TSP}}$ and $\mathsf{PC}^{\mathsf{STP}}$ are APX-hard in general metrics, because even the special cases, the TSP and the Steiner tree problem, are APX-hard. Although the seminal result by Arora [2] showed that both TSP and STP have PTAS's in bounded dimensional Euclidean spaces, the prize collecting setting was not discussed. However, we do believe that their approach may be directly applied to get PTAS's for the prize collecting versions of both problems, with a slight modification to the dynamic programming algorithms. Later, A PTAS for the Steiner Forest Problem (which generalizes the STP) was discovered by Borradaile et al. [9]. Based on this result, Bateni et al. [7] studied the Prize Collecting Steiner Forest Problem, and gave a PTAS for the special case when the penalties are multiplicative, but this does not readily imply a PTAS for the $\mathsf{PC}^{\mathsf{TSP}}$ or the $\mathsf{PC}^{\mathsf{STP}}$.

Prize Collecting Problems in Special Graphs. Planar graphs is an important class of graphs. Both problems are considered in planar graphs, and a PTAS is presented by Bateni et al. [6] for $\mathsf{PC}^{\mathsf{TSP}}$ and $\mathsf{PC}^{\mathsf{STP}}$. Moreover, they noted that both problems are solvable in polynomial time in bounded treewidth graphs, and their PTAS relies on a reduction to the bounded treewidth cases. They also showed that the Prize Collecting Steiner Forest Problem, which is a generalization of the $\mathsf{PC}^{\mathsf{STP}}$, is significantly harder, and it is APX-hard in planar graphs and Euclidean instances. As for the minor forbidden graphs, which generalizes planar graphs, there are PTAS's for various optimization problems, such as TSP by Demaine et al. [14]. However, the PTAS's for prize collecting problems, to the best of our knowledge, are unknown.

---

[3] An undirected connected multi-graph is Eulerian, if every vertex has even degree.

Generalizing Euclidean Dimension. Going beyond Euclidean spaces, doubling dimension [3, 13, 18] is a popular notion of dimensionality. It captures the bounded local growth of Euclidean spaces, and does not require any specific Euclidean properties such as vector representation or dot product. A metric space has doubling dimension at most k, if every ball can be covered by at most 2^k balls of half the radius. This notion generalizes the Euclidean dimension, in that every subset of $\mathbb{R}^d$ equipped with ℓ_2 has doubling dimension $O(d)$. Although doubling metrics are more general than Euclidean spaces, recent results show that many optimization problems have similar approximation guarantees for both spaces: there exist PTAS's for the TSP [5], a certain version of the TSP with neighborhoods [12], and the Steiner forest problem [10], in doubling metrics.

Our Contributions. In this paper, we extend this line of research, and give a unified PTAS framework for both $\mathsf{PC}^{\mathsf{TSP}}$ and $\mathsf{PC}^{\mathsf{STP}}$. We use PC^{X} when the description applies to either problem. Our main result is Theorem 1.

▶ **Theorem 1.** *For any $0 < \epsilon < 1$, there exists an algorithm that, for any PC^{X} instance with n terminal points in a metric space with doubling dimension at most k, runs in time*

$$n^{O(1)^{O(k)}} \cdot \exp(\sqrt{\log n} \cdot O(\frac{k}{\epsilon})^{O(k)}),$$

and returns a solution that is a $(1 + \epsilon)$-approximation with constant probability.

Technical Issues. As a first trial, one might try to adapt the sparsity framework used in previous PTAS's for the TSP and Steiner forest problems [5, 12, 10] in doubling metrics. The framework typically uses a polynomial-time estimator H on any ball B, which gives a constant approximation for PC^{X} on some appropriately defined sub-instance around B. Intuitively, the estimator works because the local behavior of a (nearly) optimal solution can be well estimated by looking at the sub-instance locally. In particular, the following properties are needed in this framework:

- If $\mathsf{H}(B)$ is large, then the optimal solution for the sub-instance induced on B is large; moreover, any (nearly) optimal solution for the global instance would have a large part of its cost due to B.
- If $\mathsf{H}(B)$ is small, then for any (nearly) optimal solution F for the global instance, the cost of F contributed by the sub-instance due to B should be small.

While the first property is somehow straightforward, the following example shows that the second property is non-trivial to achieve in PC^{X}.

Example Instance: Figure 1. The example is defined on the real line. The terminals are grouped into two clusters. The left cluster contains $2m$ terminals, and the right cluster contains m terminals. Within each cluster, the distance between adjacent terminals is 1. The two clusters are at distance l apart. The penalty for each terminal is t. The parameters are chosen such that $l \gg mt$ and $t \gg m$. Observe that for PC^{X}, the optimal solution is to visit all the terminals in the left cluster with total edge weights $O(m)$ and incur the penalty mt for the terminals in the right cluster. The reason is that it will be too costly to add an edge to connect terminals from different clusters, and it is better to visit the cluster with more terminals and suffer the penalty for the cluster with fewer terminals.

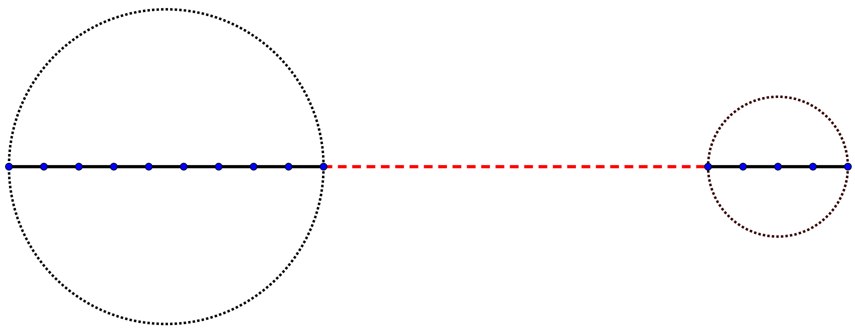

Figure 1 Example instance for PC^{X}.

Local Estimator Fails on the Example Instance. Suppose the estimator is applied around a ball B centered at some terminal in the right cluster with radius r. Then, any constant-approximate solution for the sub-instance needs to connect all $\Theta(r)$ terminals in the ball, since the penalty for any single terminal is too large. This costs $\Theta(r)$. However, in the optimal solution, no terminal in the right cluster is visited and all penalties are taken, which has cost $\Omega(tr)$. Hence, the estimator fails to serve as an upper bound for the contribution by ball B to the cost of an optimal solution.

The conclusion is that the optimal solution of a local sub-instance can differ a lot from how an optimal global solution behaves for that sub-instance.

Our Insight: Trading between Weight and Penalty. Our example in Figure 1 shows that what points a local optimal solution visits in a sub-instance can be very different from the points in the sub-instance visited by a global optimal solution. Our intuition is that the optimal cost of a sub-instance should reflect part of the cost in a global optimal solution due to the sub-instance. In other words, if a sub-instance has large optimal cost, then any global solution either (1) has a large weight within the sub-instance, or (2) suffers a large penalty due to unvisited terminals in the sub-instance. This insight leads to the following key ingredients to our solution.

1. **Inferring Local Behavior from Estimator.** In Lemma 5, we show that the value returned by the local estimator (which consists of both the weight and the penalty) on a ball B gives an upper bound on the weight $w(F|_B)$ of any (near) optimal solution F inside ball B. We emphasize that this estimator is an upper bound for the *weight* $w(F|_B)$ *only*, and is *not* an upper bound for both the weight and penalty of the optimal solution inside the ball. In the example in Figure 1, a global optimal solution does not visit the right cluster at all, and hence, the local estimator on the right cluster does give an upper bound on the *weight* part of the global solution due to the right cluster. This turns out to be sufficient because the sparsity of a solution is defined with respect to only the weight part (and not the penalty part).
   Hence, the local estimator can be used in the sparsity decomposition framework [5, 12, 10] to identify a *critical* instance W_1 (i.e., the local estimator reaches some threshold, but still not too large) around some ball B. Since the instance W_1 is sparse enough, an approximate solution F_1 can be obtained by the dynamic program framework. Then, one can recursively solve for an approximate solution F_2 for the remaining instance W_2. However, we need to carefully define W_2 and combine the solutions F_1 and F_2, because, as we remarked before, even if the approximate algorithm returns F_1 for the instance W_1, a near optimal global solution might not visit any terminals in W_1.

2. **Adaptive Recursion.** In all previous applications of the sparsity decomposition framework, after a critical ball B around some center u is identified, the original instance is decomposed into sub-instances W_1 and W_2 that can be solved independently.
   An issue in applying this framework is that after obtaining solutions F_1 and F_2 for the sub-instances, in the case that F_1 and F_2 are far away from each other as in our example in Figure 1 where it is too costly to connect them directly, it is not clear immediately which of F_1 and F_2 should be the weight part of the global solution and which would become the penalty part.
   We use a novel idea of the adaptive recursion, in which W_2 depends on the solution F_1 returned for W_1. The high level idea is that in defining the instance W_2, we add an extra terminal point at u, which becomes a representative for solution F_1. The penalty of u in W_2 is the sum of the penalties of terminals in W_1 minus the cost $c(F_1)$ of solution F_1. After a solution F_2 for W_2 is returned, if F_2 does not visit the terminal u, then edges in F_1 are discarded, otherwise the edges in F_1 and F_2 are combined to return a global solution. We can see that in either case, the sum $c(F_1) + c(F_2)$ of the costs of the two solutions reflect the cost of the global solution. In the first case, F_2 does not visit u and hence, $c(F_2)$ contains the penalty due to u, which is the penalties of unvisited terminals in W_1 minus $c(F_1)$. Therefore, when $c(F_1)$ is added back, the sum simply contains the original penalties of unvisited terminals in W_1.
   In the second case, F_2 does visit u and does not incur a penalty due to u. Therefore, $c(F_1) + c(F_2)$ does reflect the cost of the global solution after combining F_1 and F_2.

Revisiting the Sparsity Structural Lemma. Many PTAS's in the literature for TSP-like problems in doubling metrics rely on the sparsity structural lemma [5, Lemma 3.1]. Intuitively, it says that if a solution is sparse, then there exists a structurally light solution that is $(1 + \epsilon)$-approximate. Hence, one can restrict the search space to structurally light solutions, which can be explored by a dynamic program algorithm. Because of the significance of this lemma, we believe that it is worthwhile to give it a more formal inspection, and in particular, resolve some significant technical issues as follows.

- *Issue with Conditioning on the Randomness of Hierarchical Decomposition.* Given a hierarchical decomposition and a solution T, the first step is to reroute the solution such that every cluster is only visited through some designated points known as *portals*. The randomness in the hierarchical decomposition is used to argue that the expected increase in cost to make the solution portal-respecting is small.
  However, typically the randomness in the hierarchical decomposition is still needed in subsequent arguments. Hence, if one analyzes the portal-respecting procedure as a conceptually separate step, then subsequent uses of the randomness of the hierarchical decomposition need to condition on the event that the portal-respecting step does not increase the cost too much. Moreover, edges added in the portal-respecting step are actually random objects depending on the hierarchical decomposition, and hence, will in fact cross some clusters with probability 1. Unfortunately, even in the original paper by Talwar [20] on the QPTAS for TSP in doubling metrics, these issues were not addressed properly.
- *Issues with Patching Procedure.* A patching procedure is typically used to reduce the number of times a cluster is crossed. In the literature, after reducing the number of crossings, the triangle inequality is used to implicitly add some shortcutting edges outside the cluster. However, it is never argued whether these new shortcutting edges are still portal-respecting. It is plausible that making them portal-respecting might introduce new crossings.

From the above discussion, it is evident that one should consider the portal-respecting step and the patching procedure together, because they both rely on the randomness of the hierarchical decomposition. To make our arguments formal, we need a more precise notation to describe portals, and we actually revisit the whole randomized hierarchical decomposition to make all relevant definitions precise. We analyze the portal-respecting step and the patching procedure together through a sophisticated accounting argument so that the patching cost is eventually charged back to the original solution (as opposed to stopping at the transformed portal-respecting solution).

Moreover, we give a unified patching lemma that works for both $\mathsf{PC}^{\mathsf{TSP}}$ and $\mathsf{PC}^{\mathsf{STP}}$. Even though our proofs use similar ideas as previous works, the charging argument is significantly different. Specifically, our argument does not rely on the small MST lemma [20, Lemma 6], which was also used in [5].

Paper Organization. Section 2 gives the formal notation and describes the outline of the sparsity decomposition framework to solve PC^{X}. Section 3 gives the properties of the local sparsity estimator. Section 4 gives the technical details of the sparsity decomposition and shows that approximate solutions in sub-instances can be combined to give a good approximation to the global instance. Some proofs, together with other sections, are omitted due to space limit, and they can be found in the full version [11].

2 Preliminaries

We consider a metric space $M = (X, d)$ (see [15, 19] for more details on metric spaces), where we refer to an element $x \in X$ as a point or a vertex. For $x \in X$ and $\rho \geq 0$, a *ball* $B(x, \rho)$ is the set $\{y \in X \mid d(x, y) \leq \rho\}$. The *diameter* $\mathsf{Diam}(Z)$ of a set $Z \subset X$ is the maximum distance between points in Z. For $S, T \subset X$, we denote $d(S, T) := \min\{d(x, y) : x \in S, y \in T\}$, and for $u \in X$, $d(u, T) := d(\{u\}, T)$. Given a positive integer m, we denote $[m] := \{1, 2, \ldots, m\}$.

A set $S \subseteq X$ is a ρ-packing, if any two distinct points in S are at a distance more than ρ away from each other. A set S is a ρ-cover for $Z \subseteq X$, if for any $z \in Z$, there exists $x \in S$ such that $d(x, z) \leq \rho$. A set S is a ρ-net for Z, if S is a ρ-packing and a ρ-cover for Z. We assume the access to an oracle that takes a series of balls $\{B_i\}_i$ where each B_i is identified by the center and radius, and returns a point $x \in X$ such that $\forall i, x \notin S_i$[4]. A greedy algorithm can construct a ρ-net efficiently given the access to this oracle.

We consider metric spaces with *doubling dimension* [3, 18] at most k; this means that for all $x \in X$, for all $\rho > 0$, every ball $B(x, 2\rho)$ can be covered by the union of at most 2^k balls of the form $B(z, \rho)$, where $z \in X$. The following fact captures a standard property of doubling metrics.

▶ **Fact 2** (Packing in Doubling Metrics [18]). *Suppose in a metric space with doubling dimension at most k, a ρ-packing S has diameter at most R. Then, $|S| \leq (\frac{2R}{\rho})^k$.*

Edges. An edge[5] e is an unordered pair $e = \{x, y\} \in \binom{X}{2}$ whose weight $w(e) = d(x, y)$ is induced by the metric space (X, d). Given a set F of edges, its vertex set $V(F) := \cup_{e \in F}\, e \subset X$ is the vertices covered (or *visited*) by the edges in F. If $T \subset X$ is a set of vertices, we use the shorthand $T \setminus F := T \setminus V(F)$ to denote the vertices in T that are not covered by F.

[4] Such an oracle is trivial to construct for finite metric spaces. It may also be efficiently constructed for many special infinite metric spaces, such as bounded dimensional Euclidean spaces.

[5] To have a complete description, we also need the notion of self-loop, which is simply a singleton $\{x\}$.

Problem Definition. We give a unifying framework for the prize collecting traveling salesman problem ($\mathsf{PC}^{\mathsf{TSP}}$) and the prize collecting Steiner tree problem ($\mathsf{PC}^{\mathsf{STP}}$), and we use PC^{X} when the description applies to both problems. An instance $W = (T, \pi)$ of PC^{X} consists of a set $T \subset X$ of *terminals* (where $|W| := |T| = n$) and a penalty function $\pi : T \to \mathbb{R}_+$. The goal is to find a (multi-)set $F \subset \binom{X}{2}$ of edges with minimum cost[6] $c_W(F) := w(F) + \pi(T \setminus F)$, such that the following additional conditions are satisfied for each specific problem:

- For $\mathsf{PC}^{\mathsf{TSP}}$, the edges in the multi-set F form a circuit on $V(F)$; for $|V(F)| = 1$, F contains only a single self-loop (with zero weight).
- For $\mathsf{PC}^{\mathsf{STP}}$, the edges F form a connected graph on $V(F)$, where we also allow the degenerate case when F is a singleton containing a self-loop. The vertices in $V(F) \setminus T$ are known as *Steiner* points.

Simplifying Assumptions and Rescaling Instance. Fix some constant $\epsilon > 0$. Since we consider asymptotic running time to obtain $(1+\epsilon)$-approximation for PC^{X}, we consider sufficiently large $n > \frac{1}{\epsilon}$. Since F can contain a self-loop, an optimal solution covers at least one terminal u. Moreover, there is some terminal v (which could be the same as u) such that the solution covers v and does not cover any terminal v' with $d(u, v') > d(u, v)$. Since we aim for polynomial time algorithms, we can afford to enumerate the $O(n^2)$ choices for u and v.

For some choice of u and v, suppose $R := d(u, v)$. Then, R is a lower bound on the cost of an optimal solution. Moreover, the optimal solution F has weight $w(F)$ at most nR, and hence, we do not need to consider points at distances larger than nR from u. Since F contains at most $2n$ edges (because of Steiner points in $\mathsf{PC}^{\mathsf{STP}}$), if we consider an $\frac{\epsilon R}{32n^2}$-net S for X and replace every point in F with its closest net-point in S, the cost increases by at most $\epsilon \cdot \mathsf{OPT}$. Hence, after rescaling, we can assume that inter-point distance is at least 1 and we consider distances up to $O(\frac{n^3}{\epsilon}) = \text{poly}(n)$. By the packing property of doubling dimension (Fact 2), we can hence assume $|X| \le O(\frac{n}{\epsilon})^{O(k)} \le O(n)^{O(k)}$.

Hierarchical Nets. As in [5], we consider some parameter $s = (\log n)^{\frac{c}{k}} \ge 4$, where $0 < c < 1$ is a universal constant that is sufficiently small. Set $L := O(\log_s n) = O(\frac{k \log n}{\log \log n})$. A greedy algorithm can construct $N_L \subseteq N_{L-1} \subseteq \cdots \subseteq N_1 \subseteq N_0 = N_{-1} = \cdots = X$ such that for each i, N_i is an s^i-net for N_{i-1}, where we say *distance scale* s^i is of *height* i.

Net-Respecting Solution. As defined in [5], a graph F is net-respecting with respect to $\{N_i\}_{i \in [L]}$ and $\epsilon > 0$ if for every edge $\{x, y\}$ in F, both x and y belong to N_i, where $s^i \le \epsilon \cdot d(x, y) < s^{i+1}$. By [5, Lemma 1.6], any graph F may be converted to a net-respecting F' visiting all points that F visits, and $w(F') \le (1 + O(\epsilon)) \cdot w(F)$.

Given an instance W of a problem, let $\mathsf{OPT}(W)$ be an optimal solution; when the context is clear, we also use $\mathsf{OPT}(W)$ to denote the cost $c(\mathsf{OPT}(W))$, which includes both its weight and the incurred penalty; similarly, $\mathsf{OPT}^{nr}(W)$ refers to an optimal net-respecting solution.

2.1 Overview

We achieve a PTAS for PC^{X} by a unified framework, which is based on the framework of sparse instance decomposition as in [5, 12, 10].

[6] When the context is clear, we drop the subscript in $c_W(\cdot)$.

Sparse Solution [5]. Given an edge set F and a subset $S \subseteq X$, $F|_S := \{e \in F : e \subseteq S\}$ is the edges in F totally contained in S. An edge set F is called q-sparse, if for all $i \in [L]$ and all $u \in N_i$, $w(F|_{B(u,3s^i)}) \leq q \cdot s^i$.

Sparsity Structural Property. An important technical lemma [5, Lemma 3.1] in this framework states that if a (net-respecting) solution F is sparse, then with constant probability, there is some $(1+\epsilon)$-approximate solution $\widehat{F}$ that is *structurally light* with respect to some randomized *hierarchical decomposition*. Then, a bottom-up dynamic program based on the hierarchical decomposition searches for the best solution with the lightness structural property in polynomial time.

▶ Remark. We observe that this technical lemma is used crucially in all previous works on PTAS's for TSP variants in doubling metrics. Hence, we believe that its proof should be verified rigorously. In Section 1, we outlined the technical issue in the original proof [5], and this issue actually appeared as far as in the first paper on TSP for doubling metrics [20]. In the full version, we give a detailed description to complete the proof of this important lemma.

Sparsity Heuristic. As in [5, 12, 10], we estimate the local sparsity of an optimal net-respecting solution with a heuristic. For $i \in [L]$ and $u \in N_i$, given an instance W, the heuristic $\mathsf{H}_u^{(i)}(W)$ is supposed to estimate the sparsity of an optimal net-respecting solution in the ball $B' := B(u, O(s^i))$. We shall see in Section 3 that the heuristic actually gives a constant approximation to some appropriately defined sub-instance W' in the ball B'.

Divide and Conquer. Once we have a sparsity estimator, the original instance can be decomposed into sparse sub-instances, whose approximate solutions can be found efficiently. As we shall see, the partial solutions are combined with the following extension operator. The algorithm outline is described in Figure 2.

▶ **Definition 3** (Solution Extension)**.** Given two partial solutions F and F' of edges, we define the *extension* of F with F' at point u as $F \hookleftarrow_u F' := \begin{cases} F \cup F', & \text{if } u \in V(F) \cap V(F'); \\ F, & \text{otherwise.} \end{cases}$

Analysis of Approximation Ratio. We follow the inductive proof as in [5] to show that with constant probability (where the randomness comes from DP), $\mathsf{ALG}(W)$ in Figure 2 returns a solution with expected length at most $\frac{1+\epsilon}{1-\epsilon} \cdot \mathsf{OPT}^{nr}(W)$, where expectation is over the randomness of decomposition into sparse instances in Step 4.

As we shall see, in $\mathsf{ALG}(W)$, the subroutine DP is called at most poly(n) times (either explicitly in the recursion or in the heuristic $\mathsf{H}^{(i)}$). Hence, with constant probability, all solutions returned by all instances of DP have appropriate approximation guarantees.

Suppose F_1 and F_2 are solutions returned by $\mathsf{DP}(W_1)$ and $\mathsf{ALG}(W_2)$, respectively. We use c_i as a shorthand for c_{W_i}, for $i = 1, 2$, and c as a shorthand for c_W. Since we assume that W_1 is sparse enough and DP behaves correctly, $c_1(F_1) \leq (1+\epsilon) \cdot \mathsf{OPT}(W_1)$. The induction hypothesis states that $\mathbf{E}[c_2(F_2)|W_2] \leq \frac{1+\epsilon}{1-\epsilon} \cdot \mathsf{OPT}^{nr}(W_2)$.

In Step 4, equation (2) guarantees that $\mathbf{E}[\mathsf{OPT}(W_1)] \leq \frac{1}{1-\epsilon} \cdot (\mathsf{OPT}^{nr}(W) - \mathbf{E}[\mathsf{OPT}^{nr}(W_2)])$. By equation (1), $c(F_2 \hookleftarrow_u F_1) \leq c_1(F_1) + c_2(F_2)$. Hence, it follows that

$$\mathbf{E}[\mathsf{ALG}(W)] \leq \mathbf{E}[c_1(F_1) + c_2(F_2)] \leq \frac{1+\epsilon}{1-\epsilon} \cdot \mathsf{OPT}^{nr}(W) = (1 + O(\epsilon)) \cdot \mathsf{OPT}(W),$$

achieving the desired ratio.

Generic Algorithm. We describe a generic framework that applies to PC^{X}. Similar framework is also used in [5, 12, 10] to obtain PTAS's for TSP related problems. Given an instance W, we describe the recursive algorithm $\mathsf{ALG}(W)$ as follows. This description is mostly the same with that in [10], except that the decomposition in Step 4 is more involved.

1. **Base Case.** If $|W| = n$ is smaller than some constant threshold, solve the problem by brute force, recalling that $|X| \le O(\frac{n}{\epsilon})^{O(k)}$.
2. **Sparse Instance.** If for all $i \in [L]$, for all $u \in N_i$, $\mathsf{H}_u^{(i)}(W)$ is at most $q_0 \cdot s^i$, for some appropriate threshold q_0, call the subroutine $\mathsf{DP}(W)$ to return a solution, and terminate.
3. **Identify Critical Instance.** Otherwise, let i be the smallest height such that there exists $u \in N_i$ with *critical* $\mathsf{H}_u^{(i)}(W) > q_0 \cdot s^i$; in this case, choose $u \in N_i$ such that $\mathsf{H}_u^{(i)}(W)$ is maximized.
4. **Divide and Conquer.** Define a sub-instance W_1 from around the critical instance (possibly using randomness). Loosely speaking, W_1 is a sparse enough sub-instance induced in the region around u at distance scale s^i. Since it is sparse enough, we apply the dynamic programming algorithm on W_1 and get solution F_1.
   We define an appropriate sub-instance W_2 *with the information of* F_1. Intuitively, W_2 captures the remaining sub-problem not included in W_1. We emphasize that as opposed to previous work [5, 12, 10], W_2 can depend on F_1 (through the choice of the penalty function). Moreover, we ensure that any solution F_2 of W_2 can be extended to $F_2 \looparrowleft_u F_1$ as a solution for W, and the following holds:

$$c_W(F_2 \looparrowleft_u F_1) \le c_{W_1}(F_1) + c_{W_2}(F_2). \tag{1}$$

   We solve W_2 recursively and suppose the solution is F_2. We note that $\mathsf{H}_u^{(i)}(W_2) \le q_0 \cdot s^i$, and hence the recursion will terminate.
   Moreover, the following property holds:

$$\mathbf{E}[\mathsf{OPT}(W_1)] \le \frac{1}{1-\epsilon} \cdot (\mathsf{OPT}^{nr}(W) - \mathbf{E}[\mathsf{OPT}^{nr}(W_2)]), \tag{2}$$

   where the expectation is over the randomness of the decomposition.
   We return $F := F_2 \looparrowleft_u F_1$ as a solution to W.

Figure 2 Algorithm Outline.

Analysis of Running Time. As mentioned above, if $\mathsf{H}_u^{(i)}(W)$ is found to be critical, then in the decomposed sub-instances W_1 and W_2, $\mathsf{H}_u^{(i)}(W_2)$ should be small. Hence, it follows that there will be at most $|X| \cdot L = \mathrm{poly}(n)$ recursive calls to ALG. Therefore, as far as obtaining polynomial running times, it suffices to analyze the running time of the dynamic program DP. The details are provided in the full version.

3 Sparsity Estimator for PC^{X}

Recall that in the framework outlined in Section 2, given an instance W of PC^{X}, we wish to estimate the weight of $\mathsf{OPT}^{nr}(W)|_{B(u,3s^i)}$ with some heuristic $\mathsf{H}_u^{(i)}(W)$. We consider a more general sub-instance associated with the ball $B(u, ts^i)$ for $t \ge 1$.

Auxiliary Sub-Instance. Given an instance $W = (T, \pi)$, $i \in [L]$, $u \in N_i$ and $t \geq 1$, the sub-instance $W_u^{(i,t)}$ is characterized by terminal set $W \cap B(u, ts^i)$, equipped with penalties given by the same π. Using the classical (deterministic) 2-approximation algorithms by Goemans and Williamson for PC^{X} [17], we obtain a 2-approximation and then make it net-respecting to produce solution $F_u^{(i,t)}$, which has cost $c(F_u^{(i,t)}) \leq 2(1+O(\epsilon)) \cdot \mathsf{OPT}(W_u^{(i,t)})$.

Defining the Heuristic. The heuristic is defined as $\mathsf{H}_u^{(i)}(W) := c(F_u^{(i,4)})$.

In order to show that the heuristic gives a good upper bound on the local sparsity of an optimal net-respecting solution, we need the following structural result in Proposition 4 [10, Lemma 3.2] on the existence of long chain in well-separated terminals in a Steiner tree. As we shall see, the corresponding argument for the case $\mathsf{PC}^{\mathsf{TSP}}$ is trivial.

Given an edge set F, a *chain* in F is specified by a sequence of points $(p_1, p_2, \ldots, p_l)$ such that there is an edge $\{p_i, p_{i+1}\}$ in F between adjacent points, and the degree of an internal point p_i (where $2 \leq i \leq l-1$) in F is exactly 2.

▶ **Proposition 4** (Well-Separated Terminals Contains A Long Chain). *Suppose S and T are sets in a metric space with doubling dimension at most k such that $\mathsf{Diam}(S \cup T) \leq D$, and $d(S,T) \geq \tau D$, where $0 < \tau < 1$. Suppose F is an optimal net-respecting Steiner tree covering the terminals in $S \cup T$. Then, there is a chain in F with weight at least $\frac{\tau^2}{4096k^2} \cdot D$ such that any internal point in the chain is a Steiner point.*

▶ **Lemma 5** (Local Sparsity Estimator). *Let F be an optimal net-respecting solution for an instance W of PC^{X}. Then, for any $i \in [L]$, $u \in N_i$ and $t \geq 1$, we have*
$w(F|_{B(u,ts^i)}) \leq c(F_u^{(i,t+1)}) + O(\frac{skt}{\epsilon})^{O(k)} \cdot s^i$.

Proof. We follow the proof strategy in [10, Lemma 3.3], except that now a feasible solution needs not visit all terminals and can incur penalties instead. We denote $B := B(u, ts^i)$ and $\widehat{B} := B(u, (t+1)s^i)$.

Given an optimal net-respecting solution F for instance W of PC^{X}, we shall construct another net-respecting solution in the following steps.

1. Remove edges in $F|_B$.
2. Add edges $F_u^{(i,t+1)}$ corresponding to some approximate solution to the instance $W_u^{(i,t+1)}$ restricted to the ball $\widehat{B}$.
3. Let $\eta := \Theta(\frac{\epsilon}{(t+1)k^2})$, where the constant in Theta depends on Proposition 4. Let j be the integer such that $s^j \leq \max\{1, \Theta(\frac{\epsilon}{(t+1)k^2}) \cdot s^i\} < s^{j+1}$.
   Add edges in a minimum spanning tree H of $N_j \cap B(u, (t+2)s^i)$ and edges to connect H to $F_u^{(i,t+1)}$.
   Convert each added edge into a net-respecting path if necessary. Observe that the weight of edges added in this step is $O(\frac{stk}{\epsilon})^{O(k)} \cdot s^i$.
4. So far we have accounted for every terminal inside $\widehat{B}$, which is either visited or charged with its penalty according to $c(F_u^{(i,t+1)})$. We will give a more detailed description to ensure that the terminals outside $\widehat{B}$ that are covered by F will still be covered by the new solution.
   For $\mathsf{PC}^{\mathsf{TSP}}$, we will show that this step can be achieved by increasing the weight by at most $O(\frac{stk}{\epsilon})^{O(k)} \cdot s^i$; for $\mathsf{PC}^{\mathsf{STP}}$, this can be achieved by replacing some edges without increasing the weight.

Hence, after the claim in Step 4 is proved, the optimality of F implies the result.

Ensuring Terminals Outside $\widehat{B}$ are accounted for. We achieve this by considering the following steps.

1. Consider a connected component C in $F \setminus (F|_B)$. Recall that the goal is to make sure that all terminals outside $\widehat{B}$ that are visited by C will also be visited in the new solution.
2. Pick some x in $C \cap B$. If no such x exists, this implies that we have the trivial situation $F|_B = \emptyset$. Let $\widehat{C} \subseteq C$ be the maximal connected component containing x that is contained within $\widehat{B}$. Define $S := \widehat{C} \cap B$ (which contains x) and $T := \{y \in \widehat{C} \cap \widehat{B} : \exists v \notin \widehat{B}, \{y, v\} \in F\}$, which corresponds to the points that are connected to the outside $\widehat{B}$. Again, the case that $T = \emptyset$ is trivial.

Case (a): There exists $y \in T$, $d(u, y) \leq (t + \frac{1}{2})s^i$. In this case, this implies there is some $v \notin \widehat{B}$ such that $\{y, v\} \in F$ and $d(y, v) \geq \frac{s^i}{2}$. Since F is net-respecting, this implies that $y \in N_j$ and hence, the component $\widehat{C}$ (and also C) is already connected to H.

Case (b): For all $y \in T$, $d(u, y) > (t + \frac{1}{2})s^i$. We next show that there is a long chain contained in $\widehat{C}$. For $\mathsf{PC}^{\mathsf{TSP}}$, this is trivial, because we know that T contains only y, and $\widehat{C}$ is a chain from $a = x$ to $b = y$ of length at least $d(x, y) \geq \frac{s^i}{2}$.

For $\mathsf{PC}^{\mathsf{STP}}$, by the optimality of F, it follows that $\widehat{C}$ is an optimal net-respecting Steiner tree covering vertices in $S \cup T$. Hence, using Proposition 4, $\widehat{C}$ contains some chain from a to b with length at least $4\eta s^i$ (where the constant in the Theta in the definition of η is chosen such that this holds).

Once we have found this chain from a to b, we remove the edges in this chain. Hence, we can use this extra weight to connect a and b to their corresponding closest points in N_j via a net-respecting path; observe that for $\mathsf{PC}^{\mathsf{TSP}}$, it suffices to connect only $b = y$ to it closest point in N_j.

Finally, observe that for $\mathsf{PC}^{\mathsf{TSP}}$, it is possible to carry out the above procedures such that all vertices with odd degrees are in the minimum spanning tree H. Therefore, extra edges are added to ensure that the degree of every vertex is even to ensure the existence of an Euler circuit. This has extra cost at most $w(H) \leq O(\frac{stk}{\epsilon})^{O(k)} \cdot s^i$. This completes the proof. ◀

▶ **Corollary 6** (Threshold for Critical Instance). *Suppose F is an optimal net-respecting solution for an instance W of PC^{X}, and $q \geq \Theta(\frac{sk}{\epsilon})^{\Theta(k)}$. If for all $i \in [L]$ and $u \in N_i$, $\mathsf{H}_u^{(i)}(W) \leq qs^i$, then F is $2q$-sparse.*

4 Decomposition into Sparse Instances

In Section 3, we define a heuristic $\mathsf{H}_u^{(i)}(W)$ to detect a critical instance around some point $u \in N_i$ at distance scale s^i. We next describe how the instance W of PC^{X} can be decomposed into W_1 and W_2 such that equations (1) and (2) in Section 2.1 are satisfied.

Decomposing a Critical Instance. We define a threshold $q_0 := \Theta(\frac{sk}{\epsilon})^{\Theta(k)}$ according to Corollary 6. As stated in Section 2.1, a critical instance is detected by the heuristic when a smallest $i \in [L]$ is found for which there exists some $u \in N_i$ such that $\mathsf{H}_u^{(i)}(W) = c(F_u^{(i,4)}) > q_0 s^i$. Moreover, in this case, $u \in N_i$ is chosen to maximize $\mathsf{H}_u^{(i)}(W)$. To achieve a running time with an $\exp(O(1)^{k \log(k)})$ dependence on the doubling dimension k, we also apply the technique in [12] to choose the cutting radius carefully.

▶ **Claim 7** (Choosing Radius of Cutting Ball). *Denote $\mathsf{T}(\lambda) := c(F_u^{(i,4+2\lambda)})$. Then, there exists $0 \leq \lambda < k$ such that $\mathsf{T}(\lambda + 1) \leq 30k \cdot \mathsf{T}(\lambda)$.*

Proof. The proof is omitted and can be found in the full version. ◀

Cutting Ball and Sub-Instances. Suppose $\lambda \geq 0$ is picked as in Claim 7, and sample $h \in [0, \frac{1}{2}]$ uniformly at random. Define $B := B(u, (4 + 2\lambda + h)s^i)$. The original instance $W = (T, \pi)$ is decomposed into instances W_1 and W_2 as follows:

- For $W_1 = (T_1, \pi_1)$, the terminal set is $T_1 := (B \cap T) \cup \{u\}$, where for $v \neq u$ $\pi_1(v) := \pi(v)$ and $\pi_1(u) := +\infty$. We denote the cost function associated with W_1 by c_1.
- Suppose F_1 is the (random) solution for instance W_1 (that covers u) returned by the dynamic program for sparse instances (which can be found in the full version). Then, instance $W_2 = (T_2, \pi_2)$ is defined with respect to F_1. The terminal set is $T_2 := (T \setminus B) \cup \{u\}$. For $v \in T_2 \setminus \{u\}$, $\pi_2(v) := \pi(v)$ is the same; however, $\pi_2(u) := \pi(T \cap B) - c_1(F_1) = \pi(T \cap B \cap F_1) - w(F_1)$.

  Observe that the instance W_2 depends on F_1 through the choice of the penalty for u.

▶ **Lemma 8** (Combining Solutions of Sub-Instances). *Suppose instance W_1 is defined with cost function c_1 and instance W_2 is defined with respect to F_1 of W_1. Furthermore, suppose $\widehat{F}_2$ is a solution to instance W_2, whose cost function is denoted as c_2. Then, we have the following.*

(i) *Suppose $\widehat{F}_1$ is any solution to W_1 that contains u, and let $F := \widehat{F}_2 \looparrowleft_u \widehat{F}_1$. If $\widehat{F}_2$ covers u, then $F = \widehat{F}_2 \cup \widehat{F}_1$ is a solution to W with cost $c(F) \leq c_1(\widehat{F}_1) + c_2(\widehat{F}_2)$; if F_2 does not cover u, then $F = \widehat{F}_2$ is a solution to W with cost $c(F) \leq c_1(F_1) + c_2(\widehat{F}_2)$. This implies (1) in Section 2.1.*

(ii) *The sub-instance W_2 does not have a critical instance with height less than i, and $\mathsf{H}_u^{(i)}(W_2) = 0$.*

(iii) $\mathsf{H}_u^{(i)}(W_1) \leq O(s)^{O(k)} \cdot q_0 \cdot s^i$.

Proof. The proof is omitted and can be found in the full version. ◀

▶ **Lemma 9** (Combining Costs of Sub-Instances). *Suppose F is an optimal net-respecting solution for instance W of PC^{X}. Then, for any realization of the decomposed sub-instances W_1 and W_2 as described above, there exist (not necessarily net-respecting) solution $\widehat{F}_1$ for W_1 and net-respecting solution $\widehat{F}_2$ for W_2 such that $(1-\epsilon) \cdot \mathbf{E}\left[c_1(\widehat{F}_1)\right] + \mathbf{E}\left[c_2(\widehat{F}_2)\right] \leq c_W(F)$, where the expectation is over the randomness to generate W_1 and W_2. Recall that the randomness to generate W_1 and W_2 involves the random ball B and the randomness used in the dynamic program to generate F_1 to produce instance W_2 and its cost function c_2.*

Proof. The proof is omitted and can be found in the full version. ◀

References

1 Aaron Archer, MohammadHossein Bateni, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for prize-collecting steiner tree and TSP. *SIAM J. Comput.*, 40(2):309–332, 2011.

2 Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, 1998.

3 P. Assouad. Plongements lipschitziens dans $\mathbf{R}^n$. *Bull. Soc. Math. France*, 111(4):429–448, 1983.

4 Egon Balas. The prize collecting traveling salesman problem. *Networks*, 19(6):621–636, 1989.

5 Yair Bartal, Lee-Ad Gottlieb, and Robert Krauthgamer. The traveling salesman problem: Low-dimensionality implies a polynomial time approximation scheme. *SIAM J. Comput.*, 45(4):1563–1581, 2016.

6 M Bateni, Chandra Chekuri, Alina Ene, Mohammad Taghi Hajiaghayi, Nitish Korula, and Dániel Marx. Prize-collecting steiner problems on planar graphs. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1028–1049. Society for Industrial and Applied Mathematics, 2011.

7 MohammadHossein Bateni and MohammadTaghi Hajiaghayi. Euclidean prize-collecting steiner forest. *Algorithmica*, 62(3-4):906–929, 2012.

8 Daniel Bienstock, Michel X Goemans, David Simchi-Levi, and David Williamson. A note on the prize collecting traveling salesman problem. *Mathematical programming*, 59(1):413–420, 1993.

9 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. A polynomial-time approximation scheme for euclidean steiner forest. *ACM Trans. Algorithms*, 11(3):19:1–19:20, 2015.

10 T.-H. Hubert Chan, Shuguang Hu, and Shaofeng H.-C. Jiang. A PTAS for the steiner forest problem in doubling metrics. In *FOCS*, pages 810–819. IEEE Computer Society, 2016.

11 T.-H. Hubert Chan, Haotian Jiang, and Shaofeng H.-C. Jiang. A unified PTAS for prize collecting TSP and steiner tree problem in doubling metrics. *CoRR*, abs/1710.07774, 2017.

12 T.-H. Hubert Chan and Shaofeng H.-C. Jiang. Reducing curse of dimensionality: Improved PTAS for TSP (with neighborhoods) in doubling metrics. In *SODA*, pages 754–765. SIAM, 2016.

13 Kenneth L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete & Computational Geometry*, 22(1):63–93, 1999. `doi:10.1007/PL00009449`.

14 Erik D Demaine, MohammadTaghi Hajiaghayi, and Ken-ichi Kawarabayashi. Contraction decomposition in h-minor-free graphs and algorithmic applications. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 441–450. ACM, 2011.

15 M. M. Deza and M. Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1997.

16 Michel X Goemans. Combining approximation algorithms for the prize-collecting tsp. *arXiv preprint arXiv:0910.0553*, 2009.

17 Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

18 Anupam Gupta, Robert Krauthgamer, and James R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS*, pages 534–543. IEEE Computer Society, 2003.

19 J. Matoušek. *Lectures on discrete geometry*, volume 212 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2002.

20 Kunal Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *STOC*, pages 281–290. ACM, 2004.