From Chinese Postman to Salesman and Beyond: Shortest Tour δ-Covering All Points on All Edges

Fabian Frei
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Ahmed Ghazy
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
Saarland University, Saarbrücken, Germany

Tim A. Hartmann
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Florian Hörsch
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Dániel Marx
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Abstract

A well-studied continuous model of graphs, introduced by Dearing and Francis [Transportation Science, 1974], considers each edge as a continuous unit-length interval of points. For $\delta \geq 0$, we introduce the problem δ-Tour, where the objective is to find the shortest tour that comes within a distance of δ of every point on every edge. It can be observed that 0-Tour is essentially equivalent to the Chinese Postman Problem, which is solvable in polynomial time. In contrast, 1/2-Tour is essentially equivalent to the graphic Traveling Salesman Problem (TSP), which is NP-hard but admits a constant-factor approximation in polynomial time. We investigate δ-Tour for other values of δ, noting that the problem's behavior and the insights required to understand it differ significantly across various δ regimes. On the one hand, we first examine the approximability of the problem for every fixed $\delta > 0$:

(1) For every fixed $0 < \delta < 3/2$, the problem δ-Tour admits a constant-factor approximation and is APX-hard, while for every fixed $\delta \geq 3/2$, the problem admits an $O(\log n)$-approximation in polynomial time and has no polynomial-time $o(\log n)$-approximation, unless P = NP.

Our techniques also yield a new APX-hardness result for graphic TSP on cubic bipartite graphs. When parameterizing by the length of a shortest tour, it is relatively easy to show that 3/2 is the threshold of fixed-parameter tractability:

(2) For every fixed $0 < \delta < 3/2$, the problem δ-Tour is fixed-parameter tractable (FPT) when parameterized by the length of a shortest tour, while it is W[2]-hard for every fixed $\delta \geq 3/2$.

On the other hand, if δ is considered to be part of the input, then an interesting nontrivial phenomenon appears when δ is a constant fraction of the number of vertices:

(3) If δ is part of the input, then the problem can be solved in time $f(k)n^{O(k)}$, where $k = \lceil n/\delta \rceil$; however, assuming the Exponential-Time Hypothesis (ETH), there is no algorithm that solves the problem and runs in time $f(k)n^{o(k/\log k)}$.

2012 ACM Subject Classification Mathematics of computing → Approximation algorithms; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Chinese Postman Problem, Traveling Salesman Problem, Continuous Graphs, Approximation Algorithms, Inapproximability, Parameterized Complexity

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.31

Related Version *Full Version*: https://arxiv.org/abs/2410.10613

35th International Symposium on Algorithms and Computation (ISAAC 2024).
Editors: Julián Mestre and Anthony Wirth; Article No. 31; pp. 31:1–31:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

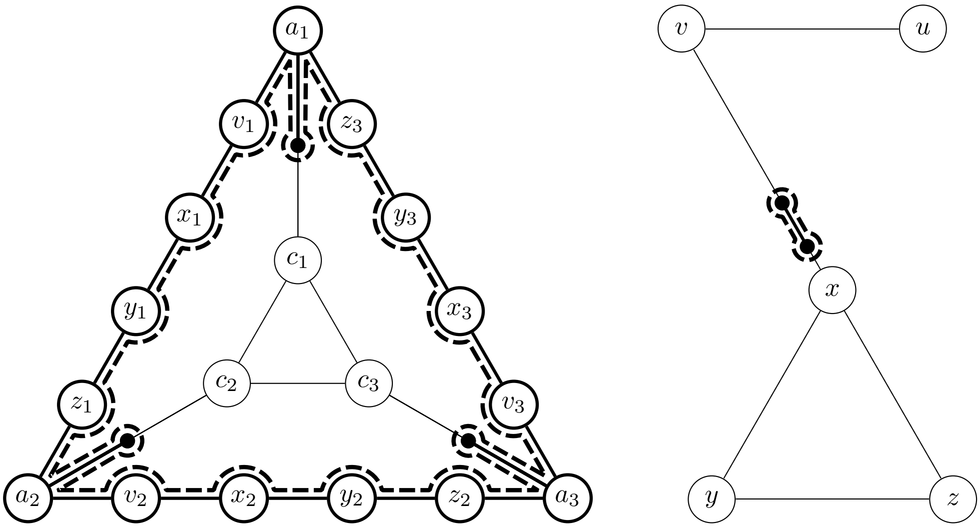
1 Introduction

We consider a well-studied continuous model of graphs introduced by Dearing and Francis [4]. Each edge is seen as a continuous unit interval of points with its vertices as endpoints. For any given graph G, this yields a compact metric space $(P(G), \mathrm{d})$ with a point set $P(G)$ and a distance function $\mathrm{d}\colon P(G)^2 \to \mathbb{R}_{\geq 0}$.

A prototypical problem in this setting is δ-Covering, introduced by Shier [28] for any positive real δ. The task is to find in G a minimum set S of points that δ-*covers* the entire graph, in the sense that each point in $P(G)$ has distance at most δ to some point in S. This problem, which is also often referred to as the continuous p-Center problem has been extensively studied; we cite only a few examples: [14, 2, 24]. Observe that the problem differs from typical discrete graph problems in two ways: the solution has to δ-cover every point of every edge (not just the vertices) and the solution may (and for optimality sometimes must) use points inside edges. How does the complexity of this problem depend on the distance δ? First, the problem is polynomial-time solvable when δ is a unit fraction, i.e., a rational with numerator 1, and NP-hard for all other rational and irrational δ [10, 13]. One can show that VertexCover is reducible to 2/3-Covering and DominatingSet is reducible to 3/2-Covering. Thus δ-Covering behaves very differently for different values of δ and can express problems of different nature and complexity: for example, while vertex cover is fixed-parameter tractable (FPT) when parameterized by the solution size, dominating set is W[2]-hard. This is reflected also in the complexity of δ-Covering: at the threshold of $\delta = 3/2$, the parameterized complexity of the problem, parameterized by the size of the solution, jumps from FPT to W[2]-hard [13]. Similarly, δ-Covering allows a constant factor approximation for $\delta < 3/2$ and becomes log-APX-hard for $\delta \geq 3/2$ [11]. The problem dual to δ-Covering is δ-Dispersion, as studied for example by Shier and Tamir [28, 29]. The task is to place a maximum number of points in the input graph such that they pairwise have distance at least δ from each other. For this problem, $\delta = 2$ marks the threshold where the parameterized complexity for the solution size as the parameter jumps from FPT to W[1]-hard [12]. Furthermore, the problem is polynomial-time solvable when δ is a rational with numerator 1 or 2, and NP-hard for all other rational and irrational δ [9, 12]. With δ-Covering being the continuous version of VertexCover and DominatingSet, and δ-Dispersion being a continuous version of IndependentSet, we now turn to the natural continuous variant of another famous problem.

We study the graphic Traveling Salesman Problem (TSP) with a positive real covering range δ in the continuous model, which we call δ-Tour. A δ-tour T is a tour that may make U-turns at arbitrary points of the graph, even inside edges, and is δ-covering, that is, every point in the graph is within distance δ from a point T passes by. The task in our problem δ-Tour is to find a shortest δ-tour. See Figure 1 for two examples of δ-tours that cannot be described as graph-theoretic closed walks. Note that computing a shortest 0-tour is equivalent to computing a shortest Chinese Postman tour (a closed walk going through every edge), which is known to be polynomial-time solvable [26, Chapter 29]. Moreover, one can observe that if every vertex of the input graph has degree at least two, then there is a shortest 1/2-tour that visits every vertex and, conversely, any tour visiting every vertex is a 1/2-tour. Thus, finding a shortest 1/2-tour is essentially equivalent to solving a TSP problem on a graph, with some additional careful handling of degree-1 vertices.

Our Results. It turns out that finding a shortest δ-tour is NP-hard for all $\delta > 0$; hence, we present approximation algorithms. As is standard, an α-*approximation algorithm* is one that runs in polynomial time and finds a solution of value within a factor α of the optimum.



(a) A graph and a δ-tour for $\delta = 1$. The tour δ-covers the inner part of this graph by peeking into three edges up to the midpoint. These three peek points are highlighted as the thick dots. The depicted tour (the thick dashed line) has length 18, which is shortest.

(b) The depicted shortest δ-tour for $\delta = 5/3$ of length $2 \cdot 1/6$ travels between the two points on edge vx at distances $1/3$ and $1/6$ from x.

Figure 1 Two examples of a δ-tour in a graph. On the right, see the special case of a tour fully contained in an edge.

As our main approximation result, for every fixed $\delta \in (0, 3/2)$, we give constant-factor approximation algorithms for finding a shortest δ-tour. We list our results in Table 1 and plot the approximation ratio against δ in Figure 3. As the complementing lower bound, we prove APX-hardness for every fixed $\delta \in (0, 3/2)$. Theorem 1.1 summarizes the general behavior; more details follow.

▶ **Theorem 1.1** (Constant-Factor Approximation). *For every fixed $\delta \in (0, 3/2)$, the problem δ-Tour admits a constant-factor approximation algorithm and is* APX*-hard.*

The problem behaves very differently in the various regimes of δ, even within the range of $(0, 3/2)$, and we exploit connections to different problems for different values of δ:

Case $\delta \in (0, 1/6]$. There is a close relation between our problem and the ChinesePostman-Problem in this range, which gives a good approximation ratio. When δ approaches 0, our approximation ratio approaches 1. See Theorem 3.3.

Case $\delta \in (1/6, 33/40)$. When $1/6 < \delta < 1/2$, the problem can be reduced to solving TSP on metric instances, for which we can use Christofides' 3/2-approximation algorithm [3] to obtain the same approximation ratio for our problem. See Theorem 3.4.
A simplification of this approach for $\delta = 1/2$ allows us to use the better 7/5-approximation for Graphic TSP due to Sebő and Vygen [27]. See Theorem 3.5.
Finally, for $1/2 < \delta < 33/40$, it turns out that a 1/2-tour is a good approximation of a δ-tour. See Theorem 3.6.

Case $\delta \in [33/40, 3/2)$. The problem here is closely related to a variation of the Vertex-Cover problem, some results on which we exploit in our approximation algorithms [1, 21]. See Theorems thm:approx:ub:one:threehalves and 3.9.

The APX-hardness results are most challenging for small values of δ, where we first prove a lower bound for a family of cycle-covering type of problems, which we call $(\alpha, \beta, \gamma, \kappa)$-CycleSubpartition; see Section 3.2. Our reduction developed for δ-Tour further directly implies a new result for graphic TSP, namely APX-hardness on cubic bipartite graphs. To the best of our knowledge, even for graphic TSP restricted to cubic graphs without the added restriction to bipartitness, there is only one APX-hardness result that unfortunately happens to be flawed [19, Thm. 5.4]. In particular, the proposed tour reconfiguration argument appears to split the original TSP tour into two disjoint ones. The issue seems to affect results in a series of other papers [7, 8, 17, 18, 16, 15]. Fortunately, our separate approach closes the gap and yields an even stronger hardness result.

▶ **Theorem 1.2.** TSP *is* APX*-hard even on cubic bipartite graphs.*

Once δ reaches $3/2$, the problem δ-TOUR suddenly changes character: it becomes similar to DOMINATINGSET, where only a logarithmic-factor approximation is possible, unless P = NP; see Theorem 3.11 and Theorem 3.14.

▶ **Theorem 1.3** (Logarithmic Approximation). *For every fixed $\delta \geq 3/2$, the problem δ-TOUR admits an $O(\log n)$-approximation algorithm and has no $o(\log n)$-approximation algorithm unless* P = NP.

The above approximation ratio in fact depends on δ, which the big-O notation hides. Thus, if δ is not fixed and is rather given as an input, this approximation guarantee can be arbitrarily bad. We show that a polylogarithmic-factor approximation is fortunately still possible in that setting.

▶ **Theorem 1.4** (Polylogarithmic Approximation). *There is a polynomial-time algorithm that, given $\delta > 0$ and a graph G of order n, computes a $64(\log n)^3$-approximation of a shortest δ-*tour *of G.*

Furthermore, we study the problem parameterized by the solution size, which is the length of the δ-tour. As mentioned above, when $\delta \geq 3/2$, then δ-COVERING becomes similar to DOMINATINGSET and is W[2]-hard. Therefore, it is not very surprising that $\delta = 3/2$ marks the threshold for the parameterized complexity of δ-tour as well; see Section 3.4.

▶ **Theorem 1.5** (Natural Parameterization). *Computing a shortest length δ-tour, parameterized by the length of the tour, is* FPT *for every fixed $0 < \delta < 3/2$, and* W[2]*-hard for every fixed $\delta \geq 3/2$.*

It is much more surprising what happens when δ is really large, comparable to the number of vertices. For this to make sense, we have to again consider the problem of computing a shortest δ-tour when δ is part of the input. In this regime, the problem becomes somewhat similar to covering the whole graph with $k = \lceil \frac{n}{\delta} \rceil$ balls of radius δ, suggesting the problem to be solvable in large part by guessing k centers in $n^{O(k)}$ time. Indeed, we give an algorithm for computing a shortest δ-tour in this runtime, and show the exponent to be essentially optimal.

▶ **Theorem 1.6** (XP Algorithm for Parameter n/δ). *There is an algorithm, which, given a connected n-vertex graph G, computes a shortest δ-*tour *of G in $f(k) \cdot n^{O(k)}$ time where $k = \lceil n/\delta \rceil$.*

▶ **Theorem 1.7** (Hardness for Parameter n/δ). *There are constants $\alpha > 0$ and k_0 such that, unless* ETH *fails, for every $k \geq k_0$, there is no algorithm that, given an n-vertex graph, computes a shortest δ-*tour *in $O(n^{\alpha k/\log k})$ time where $k = \lceil n/\delta \rceil$. Moreover, the problem is* W[1]*-hard parameterized by k.*

Section 2 begins with formal notions including a thorough definition of a δ-tour. Then Section 3 gives an extended overview of our results.

2 Formal Definitions

General Definitions. For a positive integer n, we denote the set $\{1, \ldots, n\}$ by $[n]$. All graphs in this article are undirected, unweighted and do not contain parallel edges or loops. Let G be a graph. For a subset of vertices $V' \subseteq V(G)$, we denote by $G[V']$ the subgraph induced by V'. The neighborhood of a vertex u is $N_G(u) := \{v \in V(G) \mid uv \in E(G)\}$. We write uv for an edge $\{u, v\} \in E(G)$. We denote by ln the natural logarithm and by log the binary logarithm.

Problem Related Definitions. For a graph G, we define a metric space whose point set $P(G)$ contains, somewhat informally speaking, all points on the continuum of each edge, which has unit length. We use the word *vertex* for the elements in $V(G)$, whereas we use the word *point* to denote elements in $P(G)$. Note however, that each vertex of G is also a point of G.

The set $P(G)$ is the set of points $p(u, v, \lambda)$ for every edge $uv \in E(G)$ and every $\lambda \in [0, 1]$ where $p(u, v, \lambda) = p(v, u, 1 - \lambda)$; $p(u, v, 0)$ coincides with u and $p(u, v, 1)$ coincides with v. The *distance* of points p, q on the same edge uv, say $p = p(u, v, \lambda_p)$ and $q = p(u, v, \lambda_q)$, is $\mathrm{d}(p, q) = |\lambda_q - \lambda_p|$. The *edge segment* $P(pq)$ of p and q then is the subset of points $\{p(u, v, \mu) \mid \min\{\lambda_p, \lambda_q\} \leq \mu \leq \max\{\lambda_p, \lambda_q\}\}$. A *pq-walk* T between points $p_0 := p$ and $p_z := q$ is a finite sequence of points $p_0 p_1 \ldots p_z$ where every two consecutive points lie on the same edge, that is, formally, for every $i \in [z]$ there are an edge $u_i v_i \in E(G)$ and $\lambda_i, \mu_i \in [0, 1]$ such that $p_{i-1} = p(u_i, v_i, \lambda_i)$ and $p_i = p(u_i, v_i, \mu_i)$. When p and q are not specified, we may simply write *walk* instead of pq-walk. The *length* $\ell(T)$ of a walk T is $\sum_{i \in [z]} \mathrm{d}(p_{i-1}, p_i)$. A pq-walk T, whose length is minimum among all pq-walks, is called *shortest*. The points in the sequence defining a walk are called its *stopping points*. The point set of T is $P(T) = \bigcup_{i \in [z]} P(p_{i-1}, p_i)$. For some $p \in P(T)$, we say that T *passes* p. The *distance between two points* $p, q \in P(G)$, denoted as $\mathrm{d}(p, q)$, is the length of a shortest pq-walk, and ∞ if no such walk exists. Further, let $\mathrm{d}(p, Q) = \min\{\mathrm{d}(p, q) \mid q \in Q\}$ for $p \in P(G)$ and $Q \subseteq P(G)$.

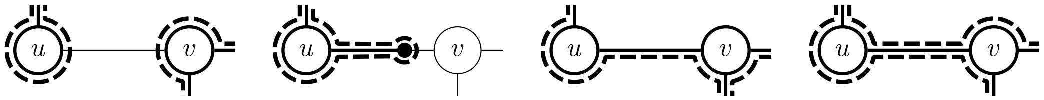
A *tour* T is a $p_0 p_z$-walk with $p_0 = p_z$. For a real $\delta > 0$, a *δ-tour* is a tour where $\mathrm{d}(p, P(T)) \leq \delta$ for every point $p \in P(G)$. We study the following minimization problem.

Optimization Problem δ-Tour, where $\delta \geq 0$.

Instance	A connected simple graph G.
Solution	Any δ-tour T.
Goal	Minimize the length $\ell(T)$.

Further, we use the following notions for a tour $T = p_0 p_1 \ldots p_z$. A *tour segment* of T is a walk given by a contiguous subsequence of $p_0 p_1 \ldots p_z$. The tour T *stops* at a point $p \in P(G)$ if $p \in \{p_0, p_1, \ldots, p_z\}$ and *traverses* an edge uv if uv or vu is a tour segment of T. The *discrete length* of a tour is z, that is, the length of the finite sequence of points representing it. We denote the discrete length of a tour T by $\alpha(T)$.

A point $p \in P(G)$ is *integral* if it coincides with a vertex. Similarly, $p = p(u, v, \lambda)$ is *half-integral* if $\lambda \in \{0, \frac{1}{2}, 1\}$.



(a) Staying out. **(b)** Peeking in. **(c)** Traversing once. **(d)** Traversing twice.

Figure 2 The four ways a nice δ-tour defined by at least 3 points can interact with an edge uv.

The *extension* of a tour $T = p_0p_1 \dots p_z$, denoted as $\lceil T \rceil$, is the integral tour where, for every edge $uv \in E(G)$ and every $\lambda < 1$, every tour segment $up(u, v, \lambda)u$ is replaced by uvu. Fruther, the *truncation* of a tour T, denoted as $\lfloor T \rfloor$, is the integral tour where for every edge $uv \in E(G), \lambda < 1$, every tour segment $up(u, v, \lambda)u$ in T is replaced by u. We note that $P(\lfloor T \rfloor) \subseteq P(T) \subseteq P(\lceil T \rceil)$.

3 Overview of Results

Section 3.1 provides key technical insights. We present our approximation algorithms in Section 3.2 and our hardness results in Section 3.3. Finally we turn to the parameterized complexity results in Section 3.4. All details are provided in the full version of this paper.

3.1 Structural Results

Because TSP in the continuous model of graphs is studied in this paper for the first time, we need to lay a substantial amount of groundwork. Due to the continuous nature of the problem, it is not clear a priori how to check if a solution is a valid δ-tour, or whether it is possible to compute a shortest δ-tour by a brute force search over a finite set of plausible tours. We clarify these issues in this section. While some of the arguments are intuitively easy to accept, the formal proofs are delicate with many corner cases to consider; the reader might want to skip these proofs at first reading.

Sometimes, a δ-tour has to make U-turns inside edges to be shortest; see Figure 1a. Indeed, it can be checked that an optimal 1-tour for the graph in Figure 1a must look exactly as depicted. Except for a single case, it is unnecessary for a tour to make more than one U-turn inside an edge. Indeed, the only case where a shortest tour is forced to make two U-turns in an edge is when the tour remains entirely within a single edge; see Figure 1b for an example. Note also that there exist degenerate cases in which a shortest δ-tour consists of a single point.

However, unless a tour is completely contained in a single edge, we can see that there are only four reasonable ways for a δ-tour to interact with the interior of any given edge, illustrated in Figure 2:

(a) completely avoiding the interior,
(b) peeking into the edge from one side,
(c) fully traversing the edge exactly once from one vertex to the other, or
(d) traversing the edge exactly twice.

We call a tour that restricts itself to this reasonable behavior a *nice* tour. The following result allows us to restrict our search space to nice tours.

▶ **Lemma 3.1** (Nice Shortest Tours). *Let G be a graph and δ a constant. Further, let a δ-tour T in G be given. Then, in polynomial time, we can compute a δ-tour T' in G with $\ell(T') \leq \ell(T)$ and such that all stopping points of T' are stopping points of T and T' is either nice or has at most two stopping points.*

Table 1 Approximation upper bounds (UB) and lower bound (LB) for δ-Tour.

$\boldsymbol{\delta}$	$(0, 1/6]$	$(1/6, 1/2)$	$1/2$	$(1/2, 33/40)$	$[33/40, 1)$	$[1, 3/2)$	$[3/2, \infty)$
UB	$1/(1-2\delta)$	1.5	1.4	$1.4/(2-2\delta)$	4	$3/(3-2\delta)$	$\min\{2\delta, 64\log^2 n\}\log n$
	Thm. 3.3	Thm. 3.4	Thm. 3.5	Thm. 3.6	Thm. 3.9	Thm. 3.8	Thms. 1.4 and 3.11
LB	APX-hard			APX-hard			$\Omega(\log n)$
	Thm. 3.12			Thm. 3.13			Thm. 3.14

Despite the continuous nature of δ-Tour, we show that we can actually study the problem in a discrete setting instead. More precisely, we prove that there is a nice shortest δ-tour T defined by points whose edge positions λ come from a small explicitly defined set. To show this, the idea is that there are only three scenarios for the edge position of a non-vertex stopping point p of T.

1. It has distance exactly δ to a vertex u. An example is that G is a long path with an end vertex u, and p is the closest point of $P(T)$ to u. Then p has an edge position λ that is the fractional part of δ.
2. It has distance exactly δ to a half-integral point $p(u, v, \frac{1}{2})$. An example is that G is a long ww'-path with a triangle uvw attached to w, and p is a closest point of $P(T)$ to $p(u, v, \frac{1}{2})$. Then p has an edge position λ which is the fractional part of $\delta + \frac{1}{2}$.
3. It has distance exactly 2δ to a vertex u. An example is that G contains a long uv-path P, p is the closest point of $P(T) - \{u\}$ on the path P to u, and T stops at u. Then p has an edge position λ which is the fractional part of 2δ.

Any δ-tour can efficiently be modified into one that is nice and has only such edge positions. Our technical proof uses some theory of linear programming, in particular some results on the vertex cover polytope.

▶ **Lemma 3.2** (Discretization Lemma). *For every $\delta > 0$ and every connected graph G, there is a shortest δ-tour such that each stopping point of the tour can be described as $p(u, v, \lambda)$ with $\lambda \in S_\delta$ where $S_\delta = \left\{0, \delta - \lfloor\delta\rfloor, \frac{1}{2} + \delta - \lfloor\frac{1}{2} + \delta\rfloor, 2\delta - \lfloor 2\delta\rfloor\right\}$.*

As a consequence, we can find a shortest δ-tour by a brute-force algorithm. Using some related arguments, we can check whether a given tour actually is a δ-tour in polynomial time.

3.2 Approximation Algorithms

Here, we overview the approximation algorithms we design for different ranges of δ. Most of our algorithms follow a general paradigm; our approach is to design a collection of *core* approximation algorithms for certain key values of δ and rely on one of the following two ideas to extrapolate the approximation ratios we get to previous and subsequent intervals. The first main idea uses the simple fact that a δ-tour is also $(\delta + x)$-tour where $x > 0$. Having an approximation algorithm for δ-Tour, if we are able to reasonably bound the ratio between the lengths of a shortest δ-tour and a shortest $(\delta + x)$-tour, we obtain an approximation algorithm for $(\delta + x)$-Tour essentially for free. The second main idea is complementary to the first. Namely, we show that we may also extend a given δ-tour to obtain a $(\delta - x)$-tour where $x > 0$. Again, having an approximation algorithm for the δ-Tour, if we have a good bound on the total length of the extensions we add, we obtain an approximation algorithm for $(\delta - x)$-Tour.

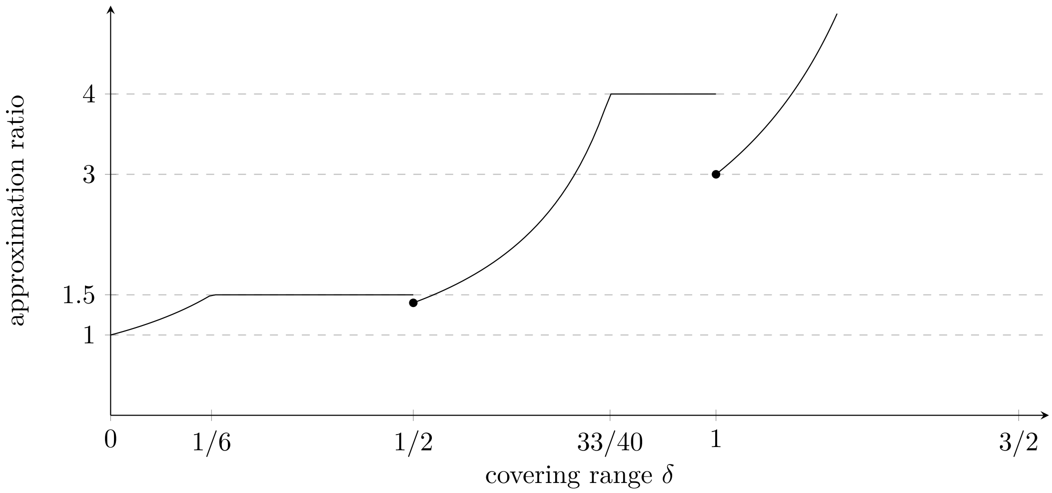


Figure 3 The approximation ratio of our algorithms for δ-Tour plotted against δ.

Approximation for $\delta \in (0, 1/6]$. The main idea is that a shortest Chinese Postman tour, that is, a tour which traverses every edge, is a good approximation of a δ-tour. Let us denote the length of a shortest δ-tour of a given graph by $\mathsf{OPT}_{\delta\text{-tour}}$ and the length of a shortest Chinese Postman tour by $\mathsf{OPT}_{\mathrm{CP}}$. To bound the ratio $\mathsf{OPT}_{\mathrm{CP}}/\mathsf{OPT}_{\delta\text{-tour}}$, we observe that there is a shortest δ-tour that, for every edge uv, either traverses uv or contains the segment of the form $up(u, v, \lambda)u$ for some $\lambda \in \{1 - \delta, 1 - 2\delta\}$. We obtain a Chinese Postman tour by replacing every such segment by a tour segment uvu. This bounds $\mathsf{OPT}_{\mathrm{CP}}/\mathsf{OPT}_{\delta\text{-tour}}$ by $1/(1 - 2\delta)$. Hence, outputting a shortest Chinese Postman tour, which can be computed in polynomial time [5], yields an approximation ratio of $1/(1 - 2\delta)$.

▶ **Theorem 3.3.** *For every $\delta \in (0, 1/6]$, δ-Tour admits a $1/(1-2\delta)$-approximation algorithm.*

Approximation for $\delta \in (1/6, 1/2)$. In this range, we rely on shortest δ-tours that satisfy certain desirable discrete properties. In the following more precise description, we focus on the case $\delta \leq \frac{1}{4}$, the construction needing to be slightly altered if $\frac{1}{4} < \delta \leq \frac{1}{2}$. Here, we prove the existence of a nice shortest δ-tour T such that

(P1) T contains the tour segment $up(u, v, 1 - \delta)u$ for every edge $uv \in E(G)$ incident to a leaf vertex v (that is, $\deg_G(v) = 1$) and

(P2) for every edge uv not incident to a leaf, either T traverses uv or the interaction of T with uv consists of one of the segments $up(u, v, 1 - 2\delta)u$ or $vp(v, u, 1 - 2\delta)v$.

We construct an auxiliary graph G' on the above listed points with edge weights corresponding to their distance in G. It turns out that TSP tours in G' are in a one-to-one correspondence with δ-tours in G satisfying properties (P1–P2). More precisely, we prove that an α-approximate TSP tour T' of G' can be efficiently turned into a δ-tour T of G of at most the same length which yields $\ell(T) \leq \ell(T') \leq \alpha \cdot \mathsf{OPT}_{\mathrm{TSP}}$, where $\mathsf{OPT}_{\mathrm{TSP}}$ denotes the length of a shortest TSP tour of G'. Then, noting that a given δ-tour of G satisfying properties (P1) and (P2) can be converted to a TSP tour of G' of at most the same length, we get that T is a δ-tour with $\ell(T) \leq \alpha \cdot \mathsf{OPT}_{\delta\text{-tour}}$. The first part, that is, proving that a TSP tour T' of G' can be turned into a δ-tour of G of the same length, is based on the fact that there is a limited number of ways a reasonable TSP tour interacts with the points corresponding to a certain edge. More precisely, any TSP tour in G' can easily be transformed into one which is not longer and whose interaction with the points in any edge is in direct correspondence with the interaction of a certain δ-tour with this edge in G.

This lets us transfer known positive approximation results for metric TSP to δ-tour. We may use the Christofides algorithm [3], yielding the following theorem.

▶ **Theorem 3.4.** *For every $\delta \in (1/6, 1/2)$, δ-Tour admits a 3/2-approximation algorithm.*

Approximation for $\delta = 1/2$. Even though the idea from the previous section still applies when $\delta = 1/2$, interestingly, we obtain a better approximation ratio observing that the problem further reduces to computing a graphic TSP tour on the non-leaf vertices, which admits a 1.4-approximation algorithm due to Sebő and Vygen [27].

▶ **Theorem 3.5.** 1/2-Tour *admits a 1.4-approximation algorithm.*

Approximation for $\delta \in (1/2, 33/40)$. In this range, we show that computing a 1/2-tour via Theorem 3.5 is a good approximation of a δ-tour. To that end, we characterize δ-tours for $\delta \in [1/2, 1]$, showing that, in particular, the existence of a shortest δ-tour T such that one of the following conditions hold for every edge uv.

(P1) T stops at both u and v, or
(P2) T stops at one of the endpoints, say u, and additionally stops at the point $p(u, v, \lambda)$ for some $\lambda \in [1 - \delta, 1]$, or T stops at the two points $p(u, v, \lambda_1)$ and $p(x, v, \lambda_2)$ for some $x \in N_G(v)$ where $\lambda_1 + \lambda_2 \geq 2 - 2\delta$, or
(P3) T stops at neither u nor v but stops at two points $p(x, v, \lambda_1)$ and $p(y, u, \lambda_2)$ for some $x \in N_G(v)$ and $y \in N_G(u)$ where $\lambda_1 + \lambda_2 \geq 3 - 2\delta$.

Let $\mathsf{OPT}_{1/2}$ and $\mathsf{OPT}_{\delta\text{-tour}}$ be the lengths of a shortest 1/2-tour and δ-tour in G, respectively. To bound the ratio $\mathsf{OPT}_{1/2}/\mathsf{OPT}_{\delta\text{-tour}}$, we observe that a given δ-tour T_δ can be transformed into a 1/2-tour $T_{1/2}$ by an appropriate replacement of every tour segment of T_δ corresponding to one of the cases (P1-3).

It can be shown that these modifications then result in a tour $T_{1/2}$ visiting every non-leaf vertex of G and covering leaves by tour segments of length 1, so $T_{1/2}$ is a 1/2-tour. These modifications increase the tour length by at most a multiplicative factor of $\max\{\frac{1}{2(1-\delta)}, \frac{2}{3-2\delta}\} = 1/(2 - 2\delta)$, so we have the following theorem.

▶ **Theorem 3.6.** *For every $\delta \in (1/2, 33/40)$, δ-Tour admits a $1.4/(2 - 2\delta)$-approximation algorithm.*

Approximation for $\delta \in (33/40, 3/2)$. Here we show that a 1-tour provides constant-factor approximations for δ-tours in this range. For $\delta > 1$, as in the previous range, due to a similar characterization of δ-tours, we can show an α-approximation algorithm for 1-Tour to imply an $\frac{\alpha}{3-2\delta}$-approximation algorithm. For $\delta < 1$, we show that starting from a 1-tour and augmenting it with some tour segments results in a δ-tour of a bounded length. The 3-approximation algorithm for a 1-Tour works as follows. We exploit a connection to the problem of computing a shortest *vertex cover tour*, which is a closed walk in a graph such that the vertices this tour stops at form a vertex cover. This problem, introduced in [1], admits a 3-approximation algorithm using linear programming (LP) techniques [21]. It is easy to see that a vertex cover tour forms a 1-tour; however, a shortest 1-tour can be shorter than a shortest vertex cover tour (e.g., this is the case in Figure 1a). Thus, the 1-tour we get from an arbitrary 3-approximation for vertex cover tour is in general not a 3-approximate 1-tour. Instead, we closely examine the LP formulated by Könemann et al. [21], showing the optimum for this LP to be a lower bound on the length of a 1-tour, which means that the vertex cover tours we get using this approach yield 3-approximate 1-tours.

Given a connected graph G, let $\mathcal{F}(G)$ be the set of subsets F of $V(G)$ such that both $G[F]$ and $G[V(G) - F]$ induce at least one edge. For a set $F \in \mathcal{F}(G)$, let $C_G(F)$ denote the set of edges in G with exactly one endpoint in F. The LP can then be formulated as follows:

$$\begin{array}{lll} \text{Minimize} & \sum_{e \in E(G)} z_e & \\ \text{subject to} & \sum_{e \in C_G(F)} z_e \geq 2 & \text{for all } F \in \mathcal{F}(G) \text{ and} \\ 0 \leq z_e \leq 2 & \text{for all } e \in E(G). & \end{array}$$

Denoting the optimal objective value of the above LP defined for a fixed graph G by $\mathsf{OPT}_{\mathrm{LP}}(G)$, the corollary below follows from [21].

▶ **Theorem 3.7** (Consequence of [21, Thms. 2 and 3]). *Given a connected graph G of order n, in polynomial time, we can compute a vertex cover tour T of G with $\ell(T) \leq 3 \cdot \mathsf{OPT}_{\mathrm{LP}}(G)$.*

It remains to show that $\mathsf{OPT}_{\mathrm{LP}}$ lower-bounds $\mathsf{OPT}_{\text{1-tour}}$, the length of a shortest 1-tour. Let $T_{\text{1-tour}} = p_0 \dots p_k$ $p_k = p_0$ be a nice 1-tour of G. For every edge $uv \in E(G)$, we define $\Lambda_{uv} := \sum_{i \in [k]: P(p_{i-1}p_i) \subseteq P(u,v)} d_G(p_{i-1}, p_i)$, indicating how much the tour $T_{\text{1-tour}}$ spends inside every edge uv. The vector $(\min(2, \Lambda_e))_{e \in E(G)}$ can then be shown to be feasible for the above LP. We observe the length of $T_{\text{1-tour}}$ to be at least $\sum_{e \in E(G)} \Lambda_e$, yielding $\mathsf{OPT}_{\text{1-tour}} \geq \mathsf{OPT}_{\mathrm{LP}}(G)$ and with Theorem 3.7, we obtain the following theorem.

▶ **Theorem 3.8.** *For any $\delta \in [1, 3/2)$, δ-Tour admits a $3/(3-2\delta)$-approximation algorithm.*

For the remaining range $\delta \in (33/40, 1)$, our algorithm first uses Theorem 3.8 to obtain a 3-approximate 1-tour T that is a vertex cover tour. Then, for every vertex $v \notin V(T)$, we choose an arbitrary neighbor u. Observe that $u \in V(T)$. Then we extend T into a tour T' by replacing an arbitrary occurence of u in T by the segment $up(u, v, 1-\delta)u$ if v is a leaf vertex and by the segment $up(u, v, (1-\delta))u$, otherwise. As T' fulfills the characterizing properties of a δ-tour, T' is a δ-tour. To bound its length, using our characterization, we observe that, given an arbitrary δ-tour T'', each non-leaf vertex v of G can be associated to a segment of T'' of cost at least $4(1-\delta)$ as T'' either stops at v by traversing an edge, incurring a cost of at least 1, or makes two non-vertex stops that can be associated to v with a total cost of at least $2(2-\delta)$. The previous observation can be used to show that the δ-tour we construct achieves an approximation ratio of 4.

▶ **Theorem 3.9.** *For any $\delta \in [33/40, 1)$, δ-Tour admits a 4-approximation algorithm.*

Approximation for $\delta > 3/2$. Here we design polylog(n)-approximation algorithms. We consider two different settings: one where δ is fixed and another where δ is part of the input. We show that each of the two problems can be reduced to an appropriate dominating set problem in an auxiliary graph. Recall that the discretization lemma (Lemma 3.2) shows, at a high-level, that there is a shortest δ-tour T of G whose stopping points on every edge come from a constant-sized set. Let $P_\delta(G)$ be the set of all such points in G.

In order to define our auxiliary graph, we first describe a collection $\mathcal{I}_{G,\delta}$ of edge segments of G. Namely, $\mathcal{I}_{G,\delta}$ is the collection of minimal edge segments each of whose whose endpoints is either a vertex of $V(G)$ or is of distance exactly δ to a point in $P_\delta(G)$ in G. This definition is suitable due to three properties of $\mathcal{I}_{G,\delta}$:

(P1) If T is a δ-tour in G whose stopping points are all contained in $P_\delta(G)$, then every $I \in \mathcal{I}_\delta(G)$ is fully covered by one stopping point of T,
(P2) every point in $P(G)$ is contained in some $I \in \mathcal{I}_{G,\delta}$, and
(P3) the number of segments in $\mathcal{I}_{G,\delta}$ is polynomial in n.

We are now ready to describe the auxiliary graph $\Gamma(G, \delta)$. We let $V(\Gamma(G, \delta))$ consist of $P_\delta(G)$ and a vertex x_I for every $I \in \mathcal{I}_{G,\delta}$. We further let $E(\Gamma(G, \delta))$ contain edges such that $\Gamma(G, \delta)[P_\delta(G)]$ is a clique and let it contain an edge px_I for $p \in P_\delta(G)$ and $I \in \mathcal{I}_{G,\delta}$ whenever p covers all the points in I. The main connection between δ-tours in G and dominating sets in $\Gamma(G, \delta)$ is due to the following lemma, which we algorithmically exploit in both settings, when $\delta \geq 3/2$ is fixed and when δ is part of the input.

▶ **Lemma 3.10.** *Let G be a graph and $\delta > 1$. Further, let T be a tour in G whose stopping points are all in $P_\delta(G)$. Then T is a δ-tour in G if and only if the stopping points of T are a dominating set in $\Gamma(G, \delta)$.*

Approximation for fixed $\delta > 3/2$. By computing a dominating set Y in the auxiliary graph $\Gamma(G, \delta)$ using a standard $\log n$-approximation algorithm and connecting it into a tour of length $O(\delta|Y|)$, we obtain the main result in this setting.

▶ **Theorem 3.11.** *For any $\delta \geq 3/2$, δ-Tour admits a $O(\log n)$-approximation algorithm.*

Approximation for δ as Part of the Input. The approach from the previous section does not yield any non-trivial approximation guarantee in this setting mainly because we get an additional factor of roughly δ when connecting the dominating set into a tour. However, we are able to show that a polylog(n)-approximation is attainable when δ is part of the input. The algorithm for this is based on a reduction to another problem related to dominating sets. Namely, a dominating tree U of a given graph H is a subgraph of H which is a tree and such that $V(U)$ is a dominating set of H. Kutiel [22] proves that given an edge-weighted graph H, we can compute a dominating tree of H of weight at most $\log^3 n$ times the minimum weight of a dominating tree of H.

In order to make use of this result, we now endow $E(\Gamma(G, \delta))$ with a weight function w. For all $p, p' \in P_\delta(G)$, we set $w(pp') = d_G(p, p')$, and all other edges get a very large weight. We now compute an approximate dominating tree U of $\Gamma(G, \delta)$ with respect to w. By the definition of w, we obtain that U does not contain any vertex of $V(\Gamma(G, \delta)) - P_\delta(G)$. It follows that we can obtain a tour T from U that visits all points of $V(U)$ and whose weight is at most $2w(U)$. By Lemma 3.10, we obtain that T is a δ-tour in G.

Finally, in order to determine the quality of T, consider a shortest δ-tour T^* in G. It follows from Lemma 3.10 that the set P_{T^*} of points of $P_\delta(G)$ passed by T^* forms a dominating set of $\Gamma(G, \delta)$. Further, we can easily find a tree in $\Gamma(G, \delta)$ spanning P_{T^*} whose weight is at most the length of T^*. Hence, this tree is a dominating tree in $\Gamma(G, \delta)$, and Theorem 1.4 follows.

3.3 Inapproximability Results

Having presented our approximation algorithms providing a constant-factor approximation for every $\delta > 0$, we now rule out the existence of a PTAS by showing APX-hardness for every $\delta > 0$. The main challenge is to show the hardness for the range $\delta \in (0, 1/2]$. A simple subdivision argument then allows us to extend the hardness result to any $\delta > 0$. Further, we describe a stronger inapproximability result for $\delta \geq 3/2$.

APX-Hardness for Covering Range $\delta \in (0, 1/2]$. As the first step towards the APX-hardness of δ-Tour in the range $\delta \in (0, 1/2]$, we introduce a new family of optimization problems called $(\alpha, \beta, \gamma, \kappa)$-CycleSubpartition, that is also interesting on its own.

Optimization Problem $(\alpha, \beta, \gamma, \kappa)$-CycleSubpartition, where $\alpha, \beta, \gamma, \kappa \in \mathbb{R}$ and $\alpha, \beta, \gamma > 0$.

Instance	A simple graph G.
Solution	Any set $\mathcal{C}$ of pairwise vertex-disjoint cycles in G.
Goal	Minimize $\alpha\lvert\mathcal{C}\rvert + \beta\lvert V(G) - \bigcup_{C\in\mathcal{C}} V(C)\rvert + \gamma\lvert V(G)\rvert + \kappa$.

This bi-objective problem asks us, roughly speaking, for any given graph, to cover as many vertices as possible with a family of as few vertex disjoint cycles as possible. The precise balance between the two opposed optimization goals is tuned by the problem parameters. In particular, α specifies the cost for each cycle in the solution and β for each vertex left uncovered. Disallowing uncovered vertices (or making them prohibitively expensive), yields the classical APX-hard minimization problem CyclePartition [25, Thm 3.1, Prob. (iv)].

The two remaining parameters γ and κ may appear artificial since their only immediate effect is to make any solution for a given graph G more expensive by the same cost $\gamma\lvert V(G)\rvert + \kappa$. They will prove meaningful, however, for our main goal of this section. Namely, we first establish APX-hardness for $(\alpha, \beta, \gamma, \kappa)$-CycleSubpartition for cubic graphs in the entire parameter range of $\alpha, \beta, \gamma, \kappa \in \mathbb{R}, \alpha, \beta, \gamma > 0$ and then show that on cubic graphs, for every $\delta \in (0, 1/2]$, we have that δ-Tour coincides with this problem for an appropriate choice of the parameters α, β, γ, and κ. The proof uses a reduction from VertexCover on cubic graphs, which is known to be APX-hard [19, Thm. 5.4].

For some fixed $\alpha, \beta, \gamma, \kappa \in \mathbb{R}$ with $\alpha, \beta, \gamma > 0$, given an instance G of cubic VertexCover, we create a cubic graph H which we view as an instance of $(\alpha, \beta, \gamma, \kappa)$-CycleSubpartition. It is not difficult to obtain a packing of cycles in H with the appropriate properties from a vertex cover in G. The other direction, that is, obtaining a vertex cover in G from a cycle packing in H is significantly more delicate. A collection of careful reconfiguration arguments is needed to transform an arbitrary cycle cover in H into one that is of a certain particular shape and not more expensive. Having a cycle cover of this shape at hand, a corresponding vertex cover in G can easily be found. As mentioned above, we now easily obtain APX-hardness of δ-Tour for any $\delta \in (0, 1/2]$ and even the previously unknown APX-hardness for cubic bipartite graphic TSP.

▶ **Theorem 3.12.** *On cubic bipartite graphs, δ-Tour is APX-hard for $\delta \in (0, 1/2]$.*

TSP is APX-hard even on cubic bipartite graphs.

With Theorem 3.12 at hand, we further easily obtain a hardness result for larger values of δ. Namely, observe that for any nonnegative integer k, any constant δ and any connected graph G, there is a direct correspondence between the δ-tours in G and the $k\delta$-tours in the graph obtained from G by subdividing every edge $k - 1$ times, yielding the following result.

▶ **Theorem 3.13.** *The problem δ-Tour is APX-hard for any real $\delta > 0$.*

Stronger Inapproximability for Covering Range $\delta \geq 3/2$. For $\delta \geq 3/2$, we give a lower bound of roughly $\ln n$ on the approximation ratio, asymptotically matching our upper bound. Like for all our inapproximability results, we prove hardness for the decision version of the problem.

▶ **Theorem 3.14.** *Unless* $\mathrm{P} = \mathrm{NP}$*, for every* $\delta \geq 3/2$*, there exists an absolute constant* α_δ *such that there is no* P*-time algorithm that, given a connected graph* G *and a constant* K*, returns "yes" if* G *admits a* δ-tour *of length at most* K *and "no" if* G *does not admit a* δ-tour *of length at most* $\alpha_\delta \log(|V(G)|)K$*.*

We start from the inapproximability of DominatingSet on split graphs, implicitly given by Dinur and Steurer [6, Corollary 1.5]. Given a split graph G satisfying some nontriviality condition, we can construct a graph G' such that the minimum size of a dominating set of G and the length of a shortest δ-tour in G' are closely related.

3.4 Parameterized Complexity

We examine the problem's parameterized complexity for two parameters: tour length and n/δ.

Parameterization by Tour Length. For $\delta \geq 3/2$, W[2]-hardness follows by a reduction from DominatingSet on split graphs, namely the same as used to show inapproximability for $\delta \geq 3/2$. We complement this result by giving an FPT-algorithm when $\delta < 3/2$. In fact, we give an algorithm which allows δ to be part of the input and that is fixed-parameter tractable for δ and maximum allowed tour length α combined.

▶ **Theorem 3.15.** *There is an algorithm that, given a graph* G *and reals* $\delta \in (0, 3/2)$ *and* $\alpha \geq 0$*, decides whether* G *has a* δ-tour *of length at most* α *in* $f(\alpha, \delta) \cdot n^{O(1)}$ *time, for some computable function* f*.*

Our algorithm is based on a kernelization: we either correctly conclude that G has no δ-tour of length at most α, or output an equivalent instance of size at most $f(\alpha, \delta)$ for a computable function f. The key insight is a bound on the vertex cover size of $f(\alpha, \delta)$ for a computable function, assuming there exists a δ-tour of length at most α. Hence we may compute an approximation C of a minimum vertex cover, and reject the instance if C is too large. We partition the vertices in $V(G) - C$ by their neighborhood in the vertex cover C. Now if a set S of the partition has size larger than $f(\alpha, \delta)$ for a computable function f, then deleting a vertex of S yields an equivalent instance.

XP Algorithm for Large Covering Range. We here give an overview of the proof of Theorem 1.6. The crucial idea is that, if T is a δ-tour, then, while the length of T can be linear in n, there exists a set of points stopped at by this tour that covers all the points in $P(G)$ and whose size is bounded by a function depending only on k where $k = \frac{n}{\delta}$. Intuitively speaking, the remainder of the tour is needed to connect the points in this set, but not to actually cover points in $P(G)$. Therefore, these segments connecting the points in the set can be chosen to be as short as possible. These observations can be subsumed in the following lemma, crucial to the proof of Theorem 1.6.

▶ **Lemma 3.16.** *Let* G *be a connected graph of order* n*,* δ *a positive real,* $k = \lceil \frac{n}{\delta} \rceil$ *and suppose that* $n \geq 12k$*. Further, let* T *be a shortest* δ-tour *in* G*. Then there exists a set* $Z \subseteq P(G)$ *of points stopped at by* T *with* $|Z| \leq 12k$ *such that for every point* $p \in P(G)$*, we have* $\mathrm{d}_G(p, Z) \leq \delta$*.*

With Lemma 3.16 at hand, Theorem 1.6 follows easily. Assuming that the minimum size requirement in Lemma 3.16 is met, due to the discretization lemma (Lemma 3.2), there exists a shortest tour only stopping at points from a set of size $O(n^2)$. We then enumerate all subsets of size at most $12k$ and for each of these sets, compute a shortest tour passing through its elements and check whether it is a δ-tour. It follows from Lemma 3.16 that the shortest tour found during this procedure is a shortest δ-tour. In the proof of Lemma 3.16, we define Z as the union of two sets Z_1 and Z_2. The set Z_1 is an inclusion-wise minimal set of points stopped at by T that covers all points in $P(G)$ whose distance to T is at least $\frac{n}{2k}$. For each $z \in Z_1$, by definition, there exists such a point p_z for which $\mathrm{d}_G(p_z, z') > \delta$ holds for all $z' \in Z - z$. Now for every $z \in Z_1$, we choose a shortest walk from p_z to z. It turns out that these walks do pairwise not share vertices of $V(G)$ and each of them contains $O(\frac{n}{k})$ vertices of $V(G)$. It follows that $|Z_1| = O(k)$. We now walk along $\lfloor T \rfloor$ and, roughly speaking, create Z_2 by adding every $\frac{n}{3k}$-th vertex stopped at by $\lfloor T \rfloor$. It turns out that Z_2 covers all points in $P(G)$ whose distance to T is at most $\frac{n}{2k}$. Further, as the length of $\lfloor T \rfloor$ is bounded by $2n$, we have $|Z_2| = O(k)$.

W[1]-Hardness for Large Covering Range. Given the XP-time algorithm running in time $f(k) \cdot n^{O(k)}$ designed for the regime $\delta = \Omega(n)$, it is natural to ask whether there is an FPT-time algorithm for the same parameter $k := \lceil \frac{n}{\delta} \rceil$. The answer is no. We show W[1]-hardness and the running time of our algorithm for Theorem 1.6 to be close to optimal under the Exponential Time Hypothesis (ETH). The hardness is based on the fact that BinaryCSP on cubic constraint graphs cannot be solved in time $f(k) \cdot n^{o(k/\log k)}$ under ETH; see [20, 23]. (The exact formulation we use is stronger and gives a lower bound for every fixed k.) An instance of BinaryCSP is a graph G with k edges, where the nodes represent variables taking values from a domain $\Sigma = [n]$, and every edge is associated with a constraint relation $C_{i,j} \subseteq \Sigma \times \Sigma$ over the two variables i and j. The instance is *satisfiable* if there is an assignment to the variables $\mathcal{A} : V(G) \to \Sigma$ such that $(\mathcal{A}(i), \mathcal{A}(j)) \in C_{i,j}$ for every constraint relation $C_{i,j}$.

In our reduction, we construct k gadgets corresponding to the constraints, with each gadget having some number of portals. Each gadget has multiple possible states corresponding to the satisfying assignments of its two constrained variables. If two constraints share a variable, then the corresponding gadgets are connected by paths between appropriate portals. These connections ensure that the selected states of the two gadgets agree on the value of the variable. It is now easy to find a tour in the auxiliary graph given a satisfying assignment for the formula. On the other hand, the construction is designed so that all tours in the auxiliary graph are in a certain shape, allowing to obtain an assignment from them.

References

1. Esther M. Arkin, Magnús M. Halldórsson, and Refael Hassin. Approximating the tree and tour covers of a graph. *Inf. Process. Lett.*, 47(6):275–282, 1993. doi:10.1016/0020-0190(93)90072-H.
2. Ramaswamy Chandrasekaran and Arie Tamir. An $o((n \log p)^2)$ algorithm for the continuous p-center problem on a tree. *SIAM J. Algebraic Discret. Methods*, 1(4):370–375, 1980. doi:10.1137/0601043.
3. Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. *Operations Research Forum*, 3(1):20, March 2022. doi:10.1007/s43069-021-00101-z.
4. Perino M. Dearing and Richard Lane Francis. A minimax location problem on a network. *Transportation Science*, 8(4):333–343, 1974.
5. Reinhard Diestel. *Graph Theory.* Springer, 5th edition, 2017.
6. Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *STOC*, pages 624–633. ACM, 2014. doi:10.1145/2591796.2591884.
7. Lars Engebretsen and Marek Karpinski. Approximation hardness of TSP with bounded metrics. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata,*

Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings, volume 2076 of *Lecture Notes in Computer Science*, pages 201–212. Springer, 2001. doi:10.1007/3-540-48224-5_17.

8 Lars Engebretsen and Marek Karpinski. TSP with bounded metrics. *Journal of Computer and System Sciences*, 72(4):509–546, 2006. doi:10.1016/j.jcss.2005.12.001.

9 Alexander Grigoriev, Tim A. Hartmann, Stefan Lendl, and Gerhard J. Woeginger. Dispersing obnoxious facilities on a graph. *Algorithmica*, 83(6):1734–1749, 2021. doi:10.1007/S00453-021-00800-3.

10 Tim A. Hartmann. *Facility location on graphs*. Dissertation, RWTH Aachen University, Aachen, 2022. doi:10.18154/RWTH-2023-01837.

11 Tim A. Hartmann and Tom Janßen. Approximating δ-covering (*to appear*). In *Approximation and Online Algorithms - 22nd International Workshop, WAOA 2024, Egham, United Kingdom, September 5-6, 2024, Proceedings*, Lecture Notes in Computer Science. Springer, 2024.

12 Tim A. Hartmann and Stefan Lendl. Dispersing obnoxious facilities on graphs by rounding distances. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *Proceeding of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *LIPIcs*, pages 55:1–55:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.MFCS.2022.55.

13 Tim A. Hartmann, Stefan Lendl, and Gerhard J. Woeginger. Continuous facility location on graphs. *Math. Program.*, 192(1):207–227, 2022. doi:10.1007/S10107-021-01646-X.

14 Oded Kariv and S. Louis Hakimi. An algorithmic approach to network location problems. I: The p-centers. *SIAM Journal on Applied Mathematics*, 37(3):513–538, 1979.

15 Marek Karpinski. Towards better inapproximability bounds for TSP: A challenge of global dependencies. In Adrian Kosowski and Igor Walukiewicz, editors, *Proceedings of the 20th International Symposium on Fundamentals of Computation Theory (FCT 2015)*, volume 9210 of *Lecture Notes in Computer Science*, pages 3–11. Springer, 2015. doi:10.1007/978-3-319-22177-9_1.

16 Marek Karpinski and Richard Schmied. Improved inapproximability results for the shortest superstring and the bounded metric TSP. https://theory.cs.uni-bonn.de/ftp/reports/cs-reports/2013/85339-CS.pdf. Accessed: 2024-04-01.

17 Marek Karpinski and Richard Schmied. On approximation lower bounds for TSP with bounded metrics. *CoRR*, abs/1201.5821, 2012. arXiv:1201.5821.

18 Marek Karpinski and Richard Schmied. Improved inapproximability results for the shortest superstring and related problems. In Anthony Wirth, editor, *Nineteenth Computing: The Australasian Theory Symposium, CATS 2013, Adelaide, Australia, February 2013*, volume 141 of *CRPIT*, pages 27–36. Australian Computer Society, 2013. URL: http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV141Karpinski.html.

19 Marek Karpinski and Richard Schmied. Approximation hardness of graphic TSP on cubic graphs. *RAIRO – Operations Research*, 49(4):651–668, 2015. doi:10.1051/ro/2014062.

20 Karthik C. S., Dániel Marx, Marcin Pilipczuk, and Uéverton S. Souza. Conditional lower bounds for sparse parameterized 2-CSP: A streamlined proof. *CoRR*, abs/2311.05913, 2023. doi:10.48550/arXiv.2311.05913.

21 Jochen Könemann, Goran Konjevod, Ojas Parekh, and Amitabh Sinha. Improved approximations for tour and tree covers. *Algorithmica*, 38(3):441–449, 2004. doi:10.1007/s00453-003-1071-0.

22 Gilad Kutiel. Hardness results and approximation algorithms for the minimum dominating tree problem. *CoRR*, abs/1802.04498, 2018. arXiv:1802.04498.

23 Dániel Marx. Can you beat treewidth? In *FOCS*, pages 169–179. IEEE Computer Society, 2007. doi:10.1109/FOCS.2007.18.

24 Nimrod Megiddo and Arie Tamir. New results on the complexity of p-center problems. *SIAM J. Comput.*, 12(4):751–758, 1983. doi:10.1137/0212051.

25 Sartaj Sahni and Teofilo F. Gonzalez. P-complete approximation problems. *J. ACM*, 23(3):555–565, 1976. doi:10.1145/321958.321975.

26 A. Schrijver. *Combinatorial Optimization – Polyhedra and Efficiency*. Springer, 2003.

27 András Sebő and Jens Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, pages 1–34, 2014.

28 Douglas R. Shier. A min-max theorem for p-center problems on a tree. *Transportation Science*, 11(3):243–252, 1977. URL: http://www.jstor.org/stable/25767877.

29 Arie Tamir. Obnoxious facility location on graphs. *SIAM J. Discret. Math.*, 4(4):550–567, 1991. doi:10.1137/0404048.