Polynomial-Time Approximation Schemes for Bounded-Capacity Vehicle Routing and Clustering Problems in Metrics with Bounded Highway Dimension

Amariah Becker∗,§, Philip N. Klein†,§, David Saulpic‡,§

Abstract

The concept of bounded highway dimension was developed to capture observed properties of the metrics of road networks. We show that a metric with bounded highway dimension and a distinguished point (the depot) can be embedded into a graph of bounded treewidth in such a way that the distance between $u$ and $v$ is preserved up to an additive error of $\epsilon$ times the distance from $u$ or $v$ to the depot. We show that this theorem yields a PTAS for BOUNDED-CAPACITY VEHICLE ROUTING in metrics of bounded highway dimension. In this problem, the input specifies a depot and a set of client locations; the output is a set of depot-to-depot tours, where each tour is responsible for visiting at most $Q$ client locations. Our PTAS can be extended to handle penalties for unvisited clients.

We extend this embedding in a case where there is a set $S$ of distinguished points. The treewidth depends on $|S|$, and the distance between $u$ and $v$ is preserved up to an additive error of $\epsilon$ times the distance from $u$ and $v$ to $S$.

This embedding implies a PTAS for MULTIPLE DEPOT BOUNDED-CAPACITY VEHICLE ROUTING, where the tours can go from one depot to another. The embedding also implies that, for fixed $k$, there is a PTAS for $k$-CENTER in metrics of bounded highway dimension. In this problem, the goal is to minimize $d$ such that there exist $k$ points (the centers) such that every point is within distance $d$ of some center. Similarly, for fixed $k$, there is a PTAS for $k$-MEDIAN in metrics of bounded highway dimension. The goal of this problem is to minimize the sum of distances to the $k$ centers.

∗Department of Computer Science, Brown University. Email: amariah.becker@brown.edu
†Department of Computer Science, Brown University. Email: klein@brown.edu
‡Département d’Informatique, École Normale Supérieure, Paris. Work done while visiting Brown University. Email: david.saulpic@ens.fr
§Research supported by National Science Foundation grant CCF-1409520.
1 Introduction

The notion of highway dimension was introduced by Abraham et al. \[3\] to explain the efficiency of some shortest-path heuristics. The motivation of this parameter comes from the work of Bast et al. \[11, 12\] who observed that, on a road network, a shortest path from a compact region to points that are far enough must go through one of a small number of access nodes. They experimentally showed that the US road network has this property. Abraham et al. \[3, 1, 2\] proved results on the efficiency of shortest-path heuristics on metrics with bounded highway dimension.

Since some road networks tend to induce metrics with small highway dimension, it makes sense to consider other optimization problems in such metrics. Feldmann et al. \[20\] inaugurated this line of research. First, they proposed a slightly modified definition of highway dimension, which we use in this paper.

**Definition 1.** The highway dimension of a graph $G$ is the smallest integer $\eta$ such that, for some universal constant $c > 4$, for every $r \in \mathbb{R}^+$ and $v \in V$, there is a set of at most $\eta$ vertices in $B_\epsilon(v)$ such that every shortest path in $B_\epsilon(v)$ of length at least $r$ intersects this set.

(In the original definition, $c = 4$. Requiring $c > 4$ seems to make a big difference in tractability of the problem, as we discuss in Section 1.1.)

Second, they proved an important result, which builds on another definition.

**Definition 2.** The aspect ratio of a metric space is the ratio of the maximum point-to-point distance to the minimum point-to-point distance.

Their result is that, under these definitions, for any $\epsilon > 0$, a metric with bounded highway dimension can be probabilistically embedded with stretch $1 + \epsilon$ into graphs having treewidth that is bounded by a polylogarithm in the aspect ratio of the original metric. Feldmann et al. used this result to obtain quasipolynomial-time approximation schemes for optimization problems such as Traveling Salesman, Steiner Tree, and Facility Location. That is, for each of these problems, for every $\epsilon > 0$, there is an $n^{\text{poly}(\log n)}$ algorithm to find a solution whose value is within a $1 + \epsilon$ factor of optimal.

Our goal in this paper is to achieve true polynomial-time approximation schemes for some other optimization problems that arise in road networks, $k$-Center and Bounded-Capacity Vehicle Routing. There are two obstacles to using the embedding of Feldmann et al.: it is probabilistic (so point-to-point distance approximation error is only bounded in expectation), and the treewidth depends on the aspect ratio, which can be unbounded in our applications. We therefore present new embedding results that are deterministic and are not dependent on aspect ratio (though our new results do not dominate the previous embedding).

First, for metrics of bounded diameter, we describe the construction of a deterministic embedding into graphs of bounded treewidth that preserves distances up to a small additive error.

**Lemma 1.** There is a function $f(x, y)$ such that, for any $\epsilon > 0$ and $\eta > 0$, for any metric space $G$ with highway dimension at most $\eta$ and diameter $\Delta$, there is a graph $H$ with treewidth at most $f(\epsilon, \eta)$ and an embedding $\phi(\cdot)$ of $G$ into $H$ such that, for all points $u$ and $v$,

$$d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq d_G(u, v) + 4\epsilon\Delta$$

Furthermore, there is a polynomial-time algorithm to construct $H$ and the embedding.

We use Lemma 1 in another embedding result. For this result, the $u$-to-$v$ distance approximation error is not uniform but depends on $u$ and $v$. 
Theorem 1. For every \( \varepsilon > 0 \) and metric space \( G \) of highway dimension \( \eta \) with a designated point \( s \), there exists a graph \( H \) and an embedding \( \phi(\cdot) \) of \( G \) into \( H \) such that

- \( H \) has bounded treewidth, and
- for all points \( u \) and \( v \), \( d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq d_G(u, v) + O(\varepsilon)(d_G(s, u) + d_G(s, v)) \)

This theorem can be applied to obtain polynomial-time approximation schemes. Consider Capacitated Vehicle Routing, a problem defined as follows. An instance consists of a number \( Q \), a metric space, a subset \( Z \) of points, called clients, and a distinguished point, called the depot. A solution is a set of tours, where each tour starts and ends at the depot, each client is assigned to one of the tours that passes through it, and each tour is assigned at most \( Q \) clients. The number \( Q \) is the capacity. If a client \( v \) is assigned to a tour, we say that the tour visits \( v \).

For arbitrary metrics, the problem is APX-hard, even when \( Q > 0 \) is fixed [9]. When \( Q \) is unbounded, it is NP-hard to approximate within a factor of 1.5 even when the metric is that of a tree [22]. We are interested in finding solutions that are within a factor \( 1 + \varepsilon \) of optimal for any given \( \varepsilon \).

Though the problem naturally arises in road networks, theoretical work on approximation schemes has so far mostly addressed Euclidean metrics. Recently a quasipolynomial-time approximation scheme was given [13] for planar graphs.

Since highway dimension has been defined to describe road networks, it is natural to look for an approximate solution in this setting. Using Theorem 1 we can prove the following:

Theorem 2. For any \( 1/4 \geq \varepsilon > 0 \), \( \eta > 0 \) and any \( Q > 0 \), there is a polynomial-time algorithm that, given an instance of Capacitated Vehicle Routing in which the capacity is \( Q \) and the metric has highway dimension at most \( \eta \), finds a solution whose cost is \( 1 + O(\varepsilon) \) times optimum.

Note that the running time is bounded by a polynomial whose degree depends on \( \varepsilon \), \( \eta \), and \( Q \).

To prove that Theorem 1 suffices in addressing Capacitated Vehicle Routing, we use the following lower bound, due to Haimovich and Rinnoy Kan [25]:

Lemma 2. For Capacitated Vehicle Routing with capacity \( Q \) and client set \( Z \),

\[
\text{cost}(\text{OPT}) \geq \frac{2}{Q} \sum \{d(c, s) : c \in Z\}
\]

The PTAS consists of applying Theorem 1 with the designated point \( s \) being the depot, to obtain a bounded-treewidth graph \( H \) and an embedding from the original metric into \( H \), and then using dynamic programming to find an optimal solution to the corresponding instance of Capacitated Vehicle Routing in \( H \). The optimal solution in the original instance corresponds to a solution in \( H \) of slightly higher cost. Using Lemma 2 we show that the increase in cost is \( Q \) times \( O(\varepsilon) \) times the optimal value. The PTAS therefore finds a solution whose cost is \( 1 + O(\varepsilon) \) times the optimal value.

Our approach can be modified to handle a generalization in which an instance also specifies a penalty for each client; the solution is allowed to omit some clients and the goal is to find a solution that minimizes the sum of costs plus penalties. We call this Capacitated Vehicle Routing with Penalties.

To handle a further generalization, in which there are a fixed number \( k \) of depots, we provide a generalization of Theorem 1, our embedding theorem.
**Theorem 3.** For $\varepsilon > 0$, metric space $G$ of highway dimension $\eta$ and a set $S$ of points of $G$, there exists a graph $H$ and an embedding $\phi(\cdot)$ of $G$ into $H$ such that

- $H$ has treewidth $f(\eta, |S|, \varepsilon)$ for some fixed function $f$, and
- for all points $u$ and $v$, $d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq (1+O(\varepsilon))d_G(u, v)+O(\varepsilon)\min(d_G(S, u), d_G(S, v))$

Using this theorem, we obtain a generalization of Theorem 2 that permits multiple depots: in this case, the tours can start and end at different depots. The embedding can also be used for other problems.

For any fixed integer $k > 0$, consider the $k$-CENTER problem. Given a metric space, the goal is to find a set of $k$ points (the centers) so as to minimize the maximum distance, $d$, of a point to the nearest center. This problem might arise, for example, in selecting locations for $k$ firehouses.

Theorem 3 implies an efficient PTAS for this problem in metric spaces with bounded highway dimension. The algorithm is as follows. First, using a 2-approximation algorithm [29, 23], find a set $S$ of $k$ centers. Use this set $S$ in Theorem 2 to obtain an embedding into a graph $H$ of bounded-treewidth, and find a approximately optimal solution in $H$; we show in Section 6.2 that this solution is also an approximately optimal solution to the original instance.

There are polynomial-time approximation schemes [21, 30] for $k$-CENTER in bounded-treewidth graphs. The running time of each of those approximation schemes is bounded by a polynomial of fixed degree, independent of the treewidth. In this way, we obtain the following theorem.

**Theorem 4.** There is a constant $c$ such that, for any $\eta > 0$, $k > 0$ and $\varepsilon > 0$, there is an $f(\eta, k, \varepsilon)n^c$ algorithm that, given an instance of $k$-CENTER in which the metric has highway dimension at most $\eta$, finds a solution whose cost is at most $1 + \varepsilon$ times optimum.

Note that the running time is bounded by a polynomial in $n$ whose degree does not depend on $\eta$, $k$, or $\varepsilon$.

By contrast, when the number $k$ of centers is unbounded, for any $\delta > 0$, it is NP-hard [23, 29] to obtain a $(2 - \delta)$-approximation for $k$-CENTER, even in unweighted planar graphs [33]. The exact optimization problem is W[2]-hard [19] for parameter $k$, indicating that there is probably not an exact algorithm whose running time has the form $O(f(k)n^c)$.

We can similarly obtain an approximation for the $k$-MEDIAN problem. This problem differs from $k$-CENTER in that the goal is to minimize not the maximum distance to a center, but the sum of these distances.

### 1.1 Related Work

The definition of highway dimension we use is the one given by Feldmann et al. [20]. However, alternate definitions exist. We summarize here the differences between them that are discussed in Feldmann et al. The original definition comes from Abraham et al. [3], in 2010. Their work uses $c = 4$, but interestingly they remark that “one could use constants bigger than 4”. Nonetheless, Feldmann et al. [20] shows that changing the constant is not innocuous: for any constant $c$, there is a graph with $n$ vertices that has highway dimension 1 with respect to $c$ and highway dimension $\Omega(n)$ with respect to any $c' > c$.

Another definition of highway dimension comes from a 2011 paper of Abraham et al. [1]. Their definition differs from Definition 1 in that they use $c = 2$ and all shortest paths of length in $(r, 2r]$ that intersect the ball $B_{2r}(v)$ (not just the ones that stay inside the ball). This is a generalization of Definition 1 for $c = 4$: a path of length at most $2r$ that intersects the ball $B_{2r}(v)$ is also entirely contained in the ball $B_{4r}(v)$. As is, the results of Feldmann et al. and, consequently, the ones presented in this paper cannot be generalized to this definition.
The final definition of highway dimension that we consider was also introduced by Abraham et al. [2] in a journal paper in 2016. This definition is stricter than the one of 2010 (and therefore the one of our paper), Feldmann et al. show that if a metric has a highway dimension $h$ according to the 2016 definition, it has a highway dimension $O(h^2)$ according to the 2010 definition.

The $k$-Center problem arises in several problems on transportation networks and clustering, and has therefore been widely studied. Hochbaum and Shmoys [29] and Gonzalez [23] showed that it is NP-hard to approximate $k$-Center within a factor $2 - \varepsilon$ in arbitrary metrics, and provided a polynomial-time 2-approximation algorithm.

Feldmann [19] showed hardness result for $k$-Center. He proved that in general graphs and for $\varepsilon > 0$, computing a $(2 - \varepsilon)$ approximate is W[2]-hard for parameter $k$, and shows other results based on the definition of highway dimension of 2011 [1]. Specifically, he showed that it is NP-hard to approximate $k$-Center within a factor $2 - \varepsilon$ in $n$-point metric spaces of highway dimension $h = O(\log^2 n)$. He also proved a lower bound, for the dependence on $h$, on the complexity of finding a $2 - \varepsilon$-approximation under the Exponential-Time Hypothesis (ETH), and gives a $3/2$-approximation algorithm that runs in time $2^{O(kh \log h)}n^{O(1)}$.

For $k$-Center, there have been a constant-factor approximation algorithms given for general metric spaces (e.g. [34]) and approximation schemes for Euclidean spaces [6, 10, 14, 17, 32, 18, 28, 27] and planar graphs [14].

The other problem we consider, Capacitated Vehicle Routing, is a generalization of TSP (for TSP $Q = n$). Haimovich and Rinnoy Kan [25] proved the lower bound of Lemma 2. Building from a solution to TSP, they also show how to achieve a constant-factor approximation, where the constant depends on the approximation ratio for TSP. Since Capacitated Vehicle Routing in general graphs is APX-hard for every fixed $Q \geq 3$ [8, 9], much work has focused on the Euclidean plane. Haimovich and Rinnoy Kan [25] gave a polynomial-time approximation scheme (PTAS) for the Euclidean plane for the case when the capacity $Q$ is constant. Asano et al. [9] showed how to improve this algorithm to get a PTAS when $Q = O(\log n / \log \log n)$. For general capacities, Das and Mathieu [16] gave a quasi-polynomial-time approximation scheme. Building on this work, Adamaszek, Czumaj, and Lingas [4] gave a PTAS that for any $\varepsilon > 0$ can handle $Q$ up to $2^{\log^d n}$ where $\delta$ depends on $\varepsilon$.

Little is known for higher dimensions or other metrics. Kachay gave a PTAS in $R^d$ that requires $Q$ to be $O(\log^{1/d} \log n)$ [31], and Hamaguchi and Katoh [26] and Asano, Katoh, and Kawashima [7] focused on constant-factor approximation algorithms for the case where the graph is a tree. Becker, Klein and Saulpic [13] gave the first approximation-scheme for a non-Euclidean metric: they describe a quasi-polynomial-time approximation scheme in planar graphs, but only when the capacity $Q$ is polylogarithmic in the graph size. They introduce the idea of an error that depends on the distance to the depot, which we also use in the embedding presented in our work here.

1.2 Paper Outline

Section 2 provides preliminary definitions and presents useful results from Feldmann et al. [20]. In Section 3 we give the first embedding result and prove Lemma 1. Section 4 explains the second embedding result (Theorem 1) and presents how to use it to achieve a PTAS for Capacitated Vehicle Routing, proving Theorem 2. Section 5 describes the dynamic program used for Capacitated Vehicle Routing, and finally Section 6 gives the third embedding result (Theorem 3) and applies it to several problems.
2 Preliminaries

We use \( OPT \) to denote the optimum solution for an optimization problem. For minimization problems, an \( \alpha \)-approximation algorithm returns a solution with cost at most \( \alpha \cdot \text{cost}(OPT) \). An approximation scheme is a family of \((1+\varepsilon)\)-approximation algorithms indexed by \( \varepsilon > 0 \). A polynomial-time approximation scheme (PTAS) is an approximation scheme that for each \( \varepsilon \) runs in polynomial time.

For an undirected graph \( G = (V,E) \), we use \( d_G(u,v) \) (or \( d(u,v) \)) when \( G \) is unambiguous) to denote the shortest-path distance between \( u \) and \( v \). For any vertex subsets \( W \subseteq V \) and vertex \( v \in V \) we let \( d(v,W) \) denote \( \min_{w \in W} d(v,w) \), and we let \( \text{diam}(W) \) denote \( \max_{u,v \in W} d(u,v) \). For \( r \in \mathbb{R}^+ \) and \( v \in V \), let \( B_v(r) = \{ u \in V \mid d(u,v) \leq r \} \) denote the ball with center \( v \) and radius \( r \).

An embedding of a graph \( G = (V,E) \) is a mapping \( \phi \) from a guest graph \( G \) to a host graph \( H = (V,E_H) \). If for all \( u,v \in V \), \( d_G(u,v) \leq d_H(\phi(u),\phi(v)) \leq \alpha d_G(u,v) \) for some \( \alpha \geq 1 \), we call \( \alpha \) the stretch of the embedding. For notational simplicity, we identify the vertices of \( H \) with points of \( G \) and therefore omit \( \phi \). For many problems, solving optimally in the host graph of an embedding with stretch \( \alpha \) provides an \( \alpha \)-approximation for the problem in the guest graph.

A tree decomposition of a graph \( G \) is a tree \( T_G \) whose nodes are bags of vertices that satisfy the following three criteria:

1. Every \( v \in V \) appears in at least one bag.
2. For every edge \((u,v) \in E\) there is some bag containing both \( u \) and \( v \).
3. For every \( v \in V \), the bags containing \( v \) form a connected subtree.

The width of \( T_G \) is the size of the largest bag minus one. The treewidth of \( G \) is the minimum width among all tree decompositions of \( G \). It is a measure of how treelike a graph is. Observe that the treewidth of a tree is one. Tree decompositions are useful for dynamic programming algorithms, but often give runtimes that are exponential in the treewidth. Therefore many problems that are \( NP \)-hard in general can be solved efficiently in graphs of bounded treewidth.

Let \( Y \subseteq X \) be a subset of elements in a metric space \((X,d)\). \( Y \) is a \( \delta \)-covering of \( X \) if for all \( x \in X \), \( d(x,Y) \leq \delta \). \( Y \) is a \( \beta \)-packing of \( X \) if for all \( y_1,y_2 \in Y \) with \( y_1 \neq y_2 \), \( d(y_1,y_2) \geq \beta \). \( Y \) is an \( \varepsilon \)-net if it is both an \( \varepsilon \)-covering and an \( \varepsilon \)-packing.

2.1 Concepts and results from Feldmann

We rely on two main concepts from Feldmann et al. \cite{feldmann2020approximation}: the town decomposition and the set of approximate core hubs. We summarize the relevant details in this section.

Instead of working directly with Definition 1, we use the concept of a shortest-path cover, which is a more convenient tool.

Recall that \( c \) is a constant greater than 4.

Definition 3. For a graph \( G \) with vertex set \( V \) and \( r \in \mathbb{R}^+ \), a shortest-path cover \( \text{SPC}(r) \subseteq V \) is a set of hubs such that every shortest path of length in \((r,cr/2]\) contains at least one hub from this set. Such a cover is called locally \( s \)-sparse for scale \( r \) if every ball of diameter \( cr \) contains at most \( s \) vertices from \( \text{SPC}(r) \).

For a graph of highway dimension \( \eta \), Abraham et al. \cite{abraham2019approximation} showed how to find a \( \eta \log \eta \)-sparse shortest-path cover in polynomial time. This result justifies using shortest-path covers instead of directly using highway dimension.
2.1.1 Town Decomposition

A shortest-path cover for scale $r$ naturally defines a clustering of the vertices into towns \[20\]. A town at scale $r$ is defined by at least one $v \in V$ such that $d(v, \text{SPC}(r)) > 2r$ and is composed of \{u $\in V | d(u, v) \leq r$\}. Lemma 3 describes key properties of towns proved in Feldmann et al.

**Lemma 3** (Lemma 3.2 in \[20\]). If $T$ is a town at scale $r_i$, then

1. $\text{diam}(T) \leq r_i$ and
2. $d(T, V \setminus T) > r_i$

Feldmann et al. define a recursive decomposition of the graph using the concept of towns. Fix a set of scales $r_i = (c/4)^i$. We say that a town $T$ at scale $r_i$ is on level $i$. Consider the set $T = \{T \subseteq V | T$ is a town on level $i \in \mathbb{N}\}$ of towns for all levels. This set forms a laminar family and therefore has a tree structure. Moreover, it has the following properties:

**Lemma 4** (Lemma 3.5 in \[20\]). For every town $T$ in a town decomposition $\mathcal{T}$,

1. $T$ has either 0 children or at least 2 children, and
2. if $T$ is a town at level $i$ and has child town $T'$ at level $j$, then $j < i$.

The set $\mathcal{T}$ is called the town decomposition of $G$, with respect to the shortest-path cover, and is a key concept used in this paper.

2.1.2 Approximate Core Hubs

For $\varepsilon > 0$, Feldmann et al. also define for each town $T$ a set $X_T$ of approximate core hubs which is a subset of $T \cap \bigcup_i \text{SPC}(r_i)$ with the properties described in Lemma 5. One key property of these sets is their doubling dimension. The doubling dimension of a metric is the smallest $\theta$ such that for every $r$, every ball of radius $2r$ can be covered by at most $2^\theta$ balls of radius $r$.

**Lemma 5** (Theorem 4.2 and Lemma 5.1 in \[20\]). Let $G$ be a metric of highway dimension $\eta$, and $\mathcal{T}$ be a town decomposition with respect to an $s$-sparse shortest path cover. For any town $T \in \mathcal{T}$,

1. if $T_1$ and $T_2$ are different child towns of $T$, and $u \in T_1$ and $v \in T_2$, then there is some $h \in X_T$ such that $d(P[u, v], h) \leq \varepsilon d(u, v)$, where $P[u, v]$ is the shortest $u$-to-$v$ path, and
2. the doubling dimension of $X_T$ is $\theta = O(\log(\eta s \log(1/\varepsilon)))$.

3 Embedding for Graphs of Bounded Diameter

In this section we show how to construct the embedding described by Lemma 1 for the case when the graph has bounded diameter. This embedding gives only a small additive error, and will prove to be a useful tool for the following sections.

**Lemma 1**. There is a function $f(x, y)$ such that, for any $\varepsilon > 0$ and $\eta > 0$, for any metric space $G$ with highway dimension at most $\eta$ and diameter $\Delta$, there is a graph $H$ with treewidth at most $f(\varepsilon, \eta)$ and an embedding $\phi(\cdot)$ of $G$ into $H$ such that, for all points $u$ and $v$,

$$d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq d_G(u, v) + 4\varepsilon \Delta$$

Furthermore, there is a polynomial-time algorithm to construct $H$ and the embedding.
Figure 1: (a) An example town decomposition. $T_1$ has diameter at most $\varepsilon \Delta$ and $T_2$ has diameter greater than $\varepsilon \Delta$. (b) Two cases of town embeddings. $T_1$ is embedded as a star with center $v_{T_1}$. The embedding of $T_2$ connects all vertices in $T_2$ to all hubs in $\hat{X}_{T_2}$ (depicted as squares). (c) Hub $\hat{h} \in \hat{X}_T$ is close to hub $h \in X_T$ which itself is close to the shortest $u$-to-$v$ path.

We first present an algorithm to compute the host graph $H$ and a tree decomposition of $H$. This algorithm relies on the town decomposition $T$ of $G$, described in Section 2.

The host graph $H$ is constructed as follows. First, consider a town $T$ that has diameter $d \leq \varepsilon \Delta$ but has no ancestor towns of diameter $\varepsilon \Delta$ or smaller. We call such a town a maximal town of diameter at most $\varepsilon \Delta$. The town $T$ is embedded into a star: choose an arbitrary vertex $v_T$ in $T$, and for each $u \in T$, include an edge in $H$ between $u$ and $v_T$ with length $d_G(u, v_T)$ equal to their distance in $G$ (see Figures 1a and 1b).

Now consider a town $T$ of diameter $d > \varepsilon \Delta$. The set of approximate core hubs $X_T$ can be used as portals to preserve distances between vertices lying in different child towns of $T$. Specifically, by Lemma 5, for every pair of vertices $(u, v)$ in different child towns of $T$, $X_T$ contains a vertex that is close to the shortest path between $u$ and $v$. In order to approximate the shortest paths, it is therefore sufficient to consider a set of points close to $X_T$. Let $\hat{X}_T$ be an $\varepsilon d_T$-net of $X_T$. For each $\hat{h} \in \hat{X}_T$ and $v \in T$, include an edge in $H$ connecting $v$ to $\hat{h}$ with length $d_H(v, \hat{h}) = d_G(v, \hat{h})$ equal to the $v$-to-$\hat{h}$ distance in $G$ (see Figures 1a and 1b).

The tree decomposition $D$ mimics the town decomposition tree: for each town $T$ of diameter greater than $\varepsilon \Delta$, there is a bag $b_T$. This bag is connected in $D$ to all of the bags of child towns of $T$ and contains all of the vertices of the net assigned to $T$ and of the nets assigned to $T'$s ancestors in the town decomposition. Formally, if $A_T$ denotes the set of all towns that contain $T$, $b_T = \bigcup_{T' \in A_T} \hat{X}_{T'}$. Note that if $T'$ is the parent of $T$ in the town decomposition, $b_T = \hat{X}_T \cup b_{T'}$. Now for each maximal town $T$ of diameter at most $\varepsilon \Delta$ with parent town $T'$, the tree decomposition contains a bag $b_T^0$ connected to a bag $b_T^*$ for each vertex $u \in T$. We define $b_T^0 = \{v_T\} \cup b_{T'}$ and $b_T^* = \{u\} \cup b_T^0$.

Following Feldmann et al. [20], the above construction can be shown to be polynomial-time constructible. The following three lemmas therefore prove Lemma 1.
Lemma 6. D is a valid tree decomposition of H.

Proof. For D to be a valid tree decomposition of H, it has to satisfy the three criteria listed in the preliminaries.

As every vertex v in is in some maximal town T of diameter at most εΔ, there is a leaf b_{εT} of D that contains v. Moreover, this leaf contains all of the vertices adjacent to v in H: if an edge connects u and v, then either u or v is the center of the star for T, or u is in the net of some town that contains v. In both cases the construction of D ensure that u is in b_{εT}. Finally, let T be a town such that b_{εT} is the highest bag in the tree decomposition that contains v. As the towns at a given level of the tree decomposition form a partition of the vertices, this town is unique. Since the town decomposition has a laminar structure, v cannot appear in a bag that is not a descendant of b_{εT}. Furthermore, by definition of the bags, v appears in all descendants of b_{εT}, proving the third property. 

Lemma 7. H has a treewidth $O((\frac{1}{\varepsilon})^\theta \log \frac{1}{\varepsilon})$, where θ is a bound on the doubling dimension of the sets $X_T$.

Proof. Since the size of the bags is clearly bounded by the depth times the maximal cardinality of $X_T$, it is enough to prove that, for each town T, $X_T$ is bounded by $(\frac{1}{\varepsilon})^\theta$, and that the tree decomposition has a depth $O(\log \frac{1}{\varepsilon})$.

By Lemma 5, the doubling dimension of $X_T$ is bounded by θ. $X_T$ is a subset of $X_T$ so it inherits this bound. Furthermore, the aspect ratio of $X_T$ is at most $\frac{1}{\varepsilon}$: the longest distance between members of $X_T$ is bounded by the diameter d of the town, and the smallest distance is at least $\varepsilon d$ by definition of a net. The cardinality of a set with doubling dimension d and aspect ratio $\gamma$ is bounded by $2^{x[\log_2 \gamma]}$ (see [22] for a proof), therefore $|X_T|$ is bounded by $(\frac{1}{\varepsilon})^\theta$.

We prove now that the tree decomposition has a depth $O(\log \frac{1}{\varepsilon})$. Let T be a town of diameter $d_T > \varepsilon \Delta$ and let $r_T$ be the scale of that town. By Lemma 3, $d_T \leq r_T$, and since $r_T = (\frac{4}{\varepsilon})^i$ and $d_T > \varepsilon \Delta$, we can conclude that $i > \log \frac{ε}{\varepsilon} \Delta$. As the diameter of the graph is Δ, the biggest town has a diameter at most Δ. It follows that $r_T \leq \Delta$ and therefore $i \leq \log \frac{4}{\varepsilon} \Delta$. The depth of $b_{εT}$ in the tree decomposition is therefore bounded by $\log \frac{4}{\varepsilon} \Delta = \log \frac{1}{\varepsilon}$. Furthermore, the tree decomposition of a town of diameter at most $\varepsilon \Delta$ has depth 2. The overall depth is therefore $O(\log \frac{1}{\varepsilon})$, concluding the proof.

Lemma 8. For all vertices u and v, $d_G(u, v) \leq d_H(u, v) \leq d_G(u, v) + 4\varepsilon \Delta$

Proof. Let u and v be vertices in V, and let T be the town that contains both u and v such that u and v are in different child towns of T.

If T has diameter $d_T \leq \varepsilon \Delta$, then let $T'$ be the maximal town of diameter at most $\varepsilon \Delta$ that is an ancestor of T (possibly T itself). By construction, $T'$ was embedded into a star centered at some vertex $v_{T'} \in T'$, so $d_H(u, v) \leq d_H(u, v_{T'}) + d_H(v_{T'}, v) \leq d_G(u, v_{T'}) + d_G(v_{T'}, v) \leq 2\varepsilon \Delta$.

Otherwise if T has diameter $d_T > \varepsilon \Delta$, then by Lemma 5 there is some $h \in X_T$ such that $d_G(P[u, v], h) \leq \varepsilon d(u, v)$. Since $X_T$ is an $\varepsilon d_T$ cover of $X_T$, there is some $\hat{h} \in X_T$ such that $d(h, \hat{h}) \leq \varepsilon d_T$. The host graph H includes edges (u, $\hat{h}$) and $(h, v)$, so $d_H(u, v) \leq d_H(u, \hat{h}) + d_H(\hat{h}, v) \leq d_G(u, h) + d_G(h, v) + 2\varepsilon d(u, v) + 2\varepsilon d_T \leq d_G(u, v) + 4\varepsilon \Delta$ (see Figure 1c).

Finally, since edge lengths in H are given by distances in G, $d_G(u, v) \leq d_H(u, v)$ for all u, v $\in V$. 

The next sections present some applications of the above embedding.
4 Capacitated Vehicle Routing

4.1 PTAS for Bounded Highway Dimension

The Capacitated Vehicle Routing problem for some graph $G$, client set $Z \subseteq V$, depot vertex $s \in V$ and capacity $Q > 0$ seeks a set of tours of minimal total length that collectively visit all clients, such that each tour contains $s$ and visits at most $Q$ clients.

In this section, we prove Theorem 1 and apply it to Capacitated Vehicle Routing, for graphs of bounded highway dimension $\eta$ and fixed capacity $Q$. Specifically, we show that the embedding given in Theorem 1 is such that an optimal solution in the host graph $H$ gives a $(1 + \varepsilon)$ solution in $G$. Furthermore, the embedding ensures that $H$ has small treewidth, allowing Capacitated Vehicle Routing to be solved exactly in polynomial time using dynamic programming. Putting these together gives Theorem 2.

We restate for convenience the two theorems of this section.

**Theorem 1.** For every $\varepsilon > 0$ and metric space $G$ of highway dimension $\eta$ with a designated point $s$, there exists a graph $H$ and an embedding $\phi(\cdot)$ of $G$ into $H$ such that

- $H$ has bounded treewidth, and
- for all points $u$ and $v$, $d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq d_G(u, v) + O(\varepsilon)(d_G(s, u) + d_G(s, v))$

**Theorem 2.** For any $1/4 \geq \varepsilon > 0$, $\eta > 0$ and any $Q > 0$, there is a polynomial-time algorithm that, given an instance of Capacitated Vehicle Routing in which the capacity is $Q$ and the metric has highway dimension at most $\eta$, finds a solution whose cost is $1 + O(\varepsilon)$ times optimum.

Given an embedding with the properties described in Theorem 1, all that remains in proving Theorem 2 is showing how to solve Capacitated Vehicle Routing optimally on the host graph $H$ and proving that such an optimal solution has a corresponding near optimal solution in $G$. We do so in the following two lemmas.

**Lemma 9.** Given a graph with bounded treewidth $\omega$ and a capacity $Q > 0$, Capacitated Vehicle Routing can be solved optimally in $n^{O(\omega Q)}$ time.

**Proof.** See Section 5

**Lemma 10.** For an embedding with the properties given by Theorem 2, the cost of an optimal solution in the host graph $H$ is within a $(1 + O(\varepsilon))$-factor of the cost of the optimal solution in the guest graph $G$.

**Proof.** Let $OPT_H$ be the optimal solution in the host graph $H$ and $OPT_G$ be the optimal solution in $G$. A solution is described by the order in which the clients and the depot are visited: $(u, v) \in S$ indicates that the solution $S$ visits the client $v$ immediately after visiting $u$. We want to prove that $cost_G(OPT_H) \leq (1 + O(\varepsilon))cost_G(OPT_G)$.

First, since $d_G \leq d_H$, $cost_G \leq cost_H$. Second, the solution $OPT_G$ is also a solution in the host graph $H$, since the vertices of $G$ and $H$ are the same. So, by definition of $OPT_H$, $cost_H(OPT_H) \leq cost_H(OPT_G)$. It is therefore sufficient to prove that $cost_H(OPT_G) \leq (1 + O(\varepsilon))cost_G(OPT_G)$. 
By definition of cost, \( \text{cost}_H(\text{OPT}_G) = \sum_{(u,v) \in \text{OPT}_G} d_H(u,v) \). Applying Theorem 1 gives

\[
\text{cost}_H(\text{OPT}_G) \leq \sum_{(u,v) \in \text{OPT}_G} d_G(u,v) + O(\varepsilon)(d_G(s,u) + d_G(s,v))
\]

The right side of the inequality can be rewritten as

\[
\sum_{(u,v) \in \text{OPT}_G} d_G(u,v) + O(\varepsilon) \sum_{(u,v) \in \text{OPT}_G} d_G(s,u) + d_G(s,v) = \text{cost}_G(\text{OPT}_G) + O(\varepsilon) \sum_{v \in Z} 2d_G(s,v) \leq O(\varepsilon) \frac{2}{\varepsilon} \text{cost}_G(\text{OPT}_G) \quad (*)
\]

To get the inequalities \((*)\), it is enough to remark that \(\text{OPT}_G\) visits every client exactly once and then to apply Lemma 2. As \(Q\) is constant, the whole inequality becomes

\[
\text{cost}_H(\text{OPT}_G) \leq \text{cost}_G(\text{OPT}_G) + O(\varepsilon) \text{cost}_G(\text{OPT}_G) = (1 + O(\varepsilon)) \text{cost}_G(\text{OPT}_G)
\]

The rest of this section is devoted to proving Theorem 1.

4.2 Embedding

We present the construction of an embedding that has the properties of Theorem 1. Our construction relies on a slight modification of the town decomposition used by Feldmann et al. [20]: it is necessary to add the depot to the shortest-path cover \(\text{SPC}(r_i)\) for every scale \(r_i\) before computing the decomposition. This modification is safe. Although the shortest-path covers may no longer be inclusion-wise minimal (as they are in Feldmann et al.), adding a single vertex does not affect correctness or the proof of their lemmas. This modification is helpful to bound the distance between a town \(T\) and the depot \(s\): as \(s\) is in the shortest-path cover at each level, \(s \notin T\) and therefore Lemma 3 gives the bound \(d(T,s) \geq \text{diam}(T)\). This bound turns out to be very helpful in the construction of the host graph.

The root town in the composition, denoted \(T_0\), is the town that contains the entire graph. We say that a town \(T\) that is a child of the root town is a top-level town, which means that the only town that properly contains \(T\) is \(T_0\).

We use Lemma 1 to construct an embedding for each top-level town. It remains to connect these embeddings: we cannot approximate \(X_{T_0}\) with a net as we did in Lemma 1 because the diameter of \(G\) may be arbitrarily large.

To cope with that issue, we define inductively the hub sets \(X^0_k, X^1_k, \ldots\) such that \(X^0_k\) is a net of \(X_{T_0} \cap B_s(2^k)\). Let \(X^0_k\) be an \(\varepsilon\)-net of \(X_{T_0} \cap B_s(1)\) that contains the depot, \(s\), and for \(k \geq 0\) let \(X^{k+1}_0\) be an \(\varepsilon 2^{k+1}\)-net of the set \((X_{T_0} \cap (B_s(2^{k+1}) - B_s(2^k))) \cup X^k_0\) that contains the depot. This construction ensures that \(X^{k+1}_0 \cap B_s(2^k) \subseteq X^k_0\), which will be helpful in Section 4.4 to find a tree decomposition of the host graph.

For a set of vertices \(X \subseteq V\), we define \(l(X) = \lceil \log_2(\max_{v \in X} d(s,v)) \rceil\) (See Figure 2a).

For every child town \(T\) of \(T_0\), the host graph connects every vertex \(v\) of \(T\) to every hub \(h\) in \(X^{l(T)}_0, \ldots, X^{l(T)+\log_2(1/\varepsilon)}_0\) with an edge of length \(d_G(v,h)\) (See Figure 2b).
4.3 Proof of Error Bound

In this section we prove the following bound on the error incurred by the embedding.

**Lemma 11.** For all vertices \( u \) and \( v \),
\[
\begin{align*}
    d_G(u, v) &\leq d_H(u, v) \leq d_G(u, v) + O(\varepsilon)(d_G(s, u) + d_G(s, v))
\end{align*}
\]

**Proof.** Consider two vertices \( u \) and \( v \). Let \( T_u \) and \( T_v \) denote the top-level towns that contain \( u \) and \( v \), respectively. There are two cases to consider.

If \( T_u = T_v \), Lemma 3 gives
\[
    d_G(u, v) \leq \text{diam}(T_u) \leq d_G(T_u, V \setminus T_u),
\]
and therefore
\[
    \text{diam}(T_u) \leq \min\{d_G(s, u), d_G(s, v)\}.
\]
Because \( T_u = T_v \) is a top-level town, its embedding is given by Lemma 1, which directly gives the desired bound.

Otherwise \( T_u \neq T_v \). Without loss of generality, assume that \( d_G(u, s) \geq d_G(v, s) \). We show that there exists some \( X_0^k \) connected to \( u \) with a vertex \( h \in X_0^k \) close to \( P[u, v] \).

By definition of the approximate core hubs, there exists \( h \in X_T^0 \) such that \( d(h, P[u, v]) \leq \varepsilon d(u, v) \). Moreover, since \( h \in B_s(2^{l(T_u+2)}) \):
\[
\begin{align*}
    d(s, h) &\leq d(s, u) + d(u, h) \\
    &\leq d(s, u) + (1 + \varepsilon)d(u, v) \\
    &\leq d(s, u) + (1 + \varepsilon)(d(s, u) + d(s, v)) \quad \text{by the triangle inequality} \\
    &\leq d(s, u) + (1 + \varepsilon) \cdot 2d(s, u) \quad \text{since } d(u, s) \geq d(v, s) \\
    &\leq (2 + \varepsilon)2^{l(T_u)} \\
    &\leq 2^{l(T_u)+2}
\end{align*}
\]
Therefore \( h \in X_{T_0} \cap B_k(2^{l(T_u)+2}) \), so there is an \( \hat{h} \in X_{l(T_u)+2}^0 \) such that \( d(\hat{h}, h) \leq \varepsilon 2^{l(T_u)+2} \). Finally, since \( \log_{\frac{1}{\varepsilon}} \geq 2 \), \( u \) is connected to \( \hat{h} \) in the host graph.

Depending on \( v \), there remain two cases: either \( v \) is connected to \( \hat{h} \) (see Figure 3a) or not (Figure 3b). First, if \( v \) is connected to \( \hat{h} \) in the host graph, \( d_H(v, \hat{h}) = d_G(v, \hat{h}) \) (and the same holds for \( u \)). The triangle inequality gives therefore,

\[
d_H(u, v) \leq d_G(u, \hat{h}) + d_G(v, \hat{h}) \leq \frac{d_G(u, h) + d_G(v, h)}{1+2\varepsilon} \leq \frac{2d_G(\hat{h}, h')}{1+2\varepsilon} \leq 2\varepsilon^2 l(T_u)+2 = O(\varepsilon) d(s, u)
\]

Since \( d_G(u, v) \leq d_G(s, u) + d_G(s, v) \), we can conclude that,

\[
d_H(u, v) \leq d_G(u, v) + O(\varepsilon)(d_G(s, u) + d_G(s, v))
\]

Otherwise, \( v \) is not connected to \( \hat{h} \). That means that \( l(T_u) + 2 > l(T_v) + \log_{\frac{1}{\varepsilon}} \), and implies that \( d_G(s, u) \geq O(\frac{1}{\varepsilon}) d_G(s, v) \). Since the host graph connects the source \( s \) to all the vertices, \( d_H(u, v) \leq d_G(s, u) + d_G(s, v) \leq d_G(u, v) + 2d_G(s, v) \leq d_G(u, v) + O(\varepsilon)(d_G(s, u) + d_G(s, v)) \). \( \square \)

![Figure 3](image1.png)

Figure 3: The shortest path between \( u \) and \( v \) in \( G \) is indicated by the curved, directed lines. The path in the host graph is represented by the straight lines.

### 4.4 Tree Decomposition

We present here the construction of a tree decomposition \( D \) of the host graph with a bounded width.

For each \( k \) let \( B_k = \bigcup_{i=k}^{k+\log_{\frac{1}{\varepsilon}}} X_i^0 \). For a top-level town \( T \), the tree decomposition \( D \) connects the decomposition \( D_T \) given by Lemma [1] to the bag \( B_{l(T)} \). Moreover, we add all vertices that appear in \( B_{l(T)} \) to all bags in the tree \( D_T \). Finally, for every \( k \) we connect \( B_k \) to both \( B_{k-1} \) and \( B_{k+1} \) in \( D \). (See Figure 2b.)
We prove that Capacitated Vehicle Routing with Penalties.

As stated in the introduction, the host graph is close to an optimal solution in the guest graph.

Proof. Since the host graph is close to an optimal solution in the guest graph, it has to satisfy the three properties listed in Section 2.

First, because the top-level towns are a partition of the vertices, each vertex appears in some tree decomposition $D_T$. The union of all bags is therefore $V(H)$.

Next, let $(u, v)$ be an edge of $H$. There are two cases to consider: if $u$ and $v$ are in the same top-level town, Lemma 1 ensures that $u$ and $v$ appear together in some bag. Otherwise, as the top-level towns are disjoint, one of $u$ or $v$ is a hub connected to the other. Without loss of generality assume that $v$ is a hub of $X^k_0$ for some $k \in \{l(T_u), ..., l(T_v) + \log_2 \frac{1}{\varepsilon}\}$. In this case, $v \in B_i(T_u)$, so $v$ is added to all the bags of $D_{T_u}$, and in particular is in some bag that contains $u$.

Finally, let $v$ be a vertex that appears in two different bags. If the two bags are in the tree decomposition of the same top-level town $T$, Lemma 2 ensures that the bags are connected in $D_T$ and thus also in $D$. Otherwise, as the top-level towns are disjoint, $v$ must be a hub. Consider all nets $X^k_i$ containing $v$. Any bag $B_c$ containing such a net also contains $v$. Let $I = \{k|v \in X^k_0\}$. We prove that $I$ is an interval, and therefore that the bags $B_\ell$ are connected. Let $i = \min(I)$ and $j = \max(I)$. As $v \in X^0_0$, it must be that $v \in B_s(2^i) \subseteq B_s(2^{i+1}) \subseteq ... \subseteq B_s(2^j)$. Repeatedly applying the property $X^k_0 \cap B_s(2^{k-1}) \subseteq X^k_{0-1}$ proves that for all $k \in \{i, i + 1, ..., j\}$, $v \in X^k_0$. Therefore $I$ is an interval, and the bags $B_\ell$ such that $v \in B_\ell$ are connected. Finally, since the vertices of $B_\ell$ are added to every adjacent $D_T$, the bags containing $v$ form a connected subtree of $D$. \hfill \Box

Lemma 13. For all $k$, $|X^k_0| \leq (\frac{2}{\varepsilon})^\theta$.

Proof. Since $X^k_0$ is a subset of $X^k_0$, it inherits doubling dimension $\theta$ (see Lemma 5). Since $X^k_0$ is a $\varepsilon 2^k$-net, the smallest distance between two hubs in $X^k_0$ is at least $\varepsilon 2^k$. Moreover, since $X^k_0 \subseteq B_s(2^k)$, the longest distance between two hubs is at most $2 \cdot 2^k$, therefore, $X^k_0$ has an aspect ratio of at most $\frac{2}{\varepsilon}$. The bound used in Lemma 7 on the cardinality of a set using its aspect ratio and its doubling dimension concludes the proof. \hfill \Box

Lemma 14. The tree decomposition $D$ has bounded width.

Proof. Bag $B_i$ is the union of $\log_2 \frac{1}{\varepsilon}$ sets $X^k_0$. Lemma 13 gives $|X^k_0| \leq (\frac{2}{\varepsilon})^\theta$, therefore $|B_i| \leq \log_2(\frac{1}{\varepsilon})(\frac{2}{\varepsilon})^\theta$. Moreover, by Lemma 7 each bag of the $D_T$ decompositions has a cardinality bounded by $O((\frac{2}{\varepsilon})^\theta \log_2 \frac{1}{\varepsilon})$. Therefore, since each bag of the decomposition $D$ is either a bag $B_i$ for some $i$ or is formed by adding a single bag $B_i$ to some bag of a $D_T$ decomposition, its size is bounded. Therefore $D$ has a bounded width. \hfill \Box

4.5 Generalization to Routing with Penalties

As stated in the introduction, the Capacitated Vehicle Routing with Penalties is a natural generalization of Capacitated Vehicle Routing. The embedding proposed previously can be used to solve it. First, the dynamic program for graphs of bounded treewidth can be adapted to solve this problem optimally in such graphs. The only change to make is that instead of visiting a client, the algorithm can chose to pay the penalty. It remains to prove that an optimal solution in the host graph is close to an optimal solution in the guest graph.
Lemma 15. The optimal solution to Capacitated Vehicle Routing with Penalties in the host graph has a cost at most \((1 + \varepsilon)\text{cost}(\text{OPT}_G)\)

Proof. The clients can be divided into two sets \(U\) and \(W\): the optimal solution in \(G\) visits every vertex in \(U\) and pays the penalty for the ones in \(W\). Applying Lemma 2 to the set \(U\), gives the following:

\[
\text{cost}(\text{OPT}_G) \geq \frac{2}{Q} \sum_{v \in U} d(v, s) + \sum_{v \in W} p(v)
\]

With this lower bound, the proof of Lemma 10 can be adapted to handle penalties, giving \(\text{cost}_H(\text{OPT}_G) \leq (1 + O(\varepsilon))\text{cost}_G(\text{OPT}_G)\). The conclusion is similar to the one of Lemma 10:

\[
\text{cost}_G(\text{OPT}_H) \leq (1 + O(\varepsilon))\text{cost}_G(\text{OPT}_G).
\]

5 Dynamic Program for Capacitated Vehicle Routing

In this section, we present a dynamic program running in \(n^{O(\omega Q)}\) to solve Capacitated Vehicle Routing for capacity \(Q\) on graphs with treewidth \(\omega\). Given a tree decomposition, \(D\), choose an arbitrary bag to be the root, and for each bag \(b\) of the decomposition let cluster \(C_b\) be the union of the bags descending from \(b\) in the tree decomposition, minus the elements of \(b\) itself. The bag \(b\) forms a boundary between cluster \(C_b\) and \(V \setminus C_b\).

A configuration in the dynamic program describes how a solution interacts with a cluster: for each vertex \(v\) in the boundary \(b\) of the cluster, and for each possible capacity \(q \leq Q\), the configuration specifies \(I_{v,q}\) and \(O_{v,q}\) which are respectively the number of tours that enter and exit \(C_b\) by vertex \(v\) and that have visited exactly \(q\) clients at the moment they reach \(v\). We refer to this as the flow in and out of \(C_b\) at \(v\). These values are sufficient to recover the intersection of the solution with the cluster: connecting each entering tour with an exiting one, at minimal cost, gives the optimal solution.

To simplify the dynamic program, we first convert \(D\) into a nice tree decomposition with \(O(\omega n)\) bags. This can be done in polynomial time, while preserving the width [15]. In a nice tree decomposition, each non-leaf bag of the decomposition is one of three types:

- An introduce bag \(b\) has one child \(b'\), such that \(b = b' \cup \{v\}\) for some vertex \(v \notin b'\). The vertex \(v\) is introduced at \(b\).
- A forget bag \(b\) has one child \(b'\), such that \(b = b' \setminus \{v\}\) for some vertex \(v \in b'\). The vertex \(v\) is forgotten at \(b\).
- A join bag \(b\) has two children \(b_1\) and \(b_2\) such that \(b = b_1 = b_2\).

Moreover, as observed in [15], the third property of a tree decomposition ensures that each vertex can be forgotten only once.

A tour can be uncrossed to avoid crossing the same vertex in the same direction twice [5]. As there are at most \(n\) different tours, \(I_{v,q} \leq n\) and \(O_{v,q} \leq n\), so there are \(n^{O(\omega Q)}\) possible configurations per bag. Since there are \(O(\omega n)\) bags in the nice tree decomposition, there are a total of \(n^{O(\omega Q)}\) different configurations.

The algorithm runs bottom-up: given a configuration for each child node, it finds all possible compatible configurations for the parent node. Each different type of bag of the nice tree decomposition requires a particular treatment.

For a forget bag, the parent bag \(b\) is equal to its child bag \(b'\) minus some vertex \(u\). For each child bag configuration, the algorithm considers all ways to form a compatible parent bag configuration
by rerouting \( u \)'s flow and, if \( u \) is a client, covering its demand. For each resulting parent bag configuration, the dynamic program stores the cost only if it is less than the current value stored for that configuration. After considering all child bag configurations and ways of forming a parent bag configuration, the values stored in the table are guaranteed to be optimal. Consider some configuration for the child bag. First, if \( u \) is a client, one tour is selected to visit it. There are two cases. If the tour crosses into or out of \( C_y \) after visiting \( u \), the algorithm chooses a capacity \( q \) and a direction (in or out or) and makes the following changes to the flow at \( u \):

\[
I(O)_{u,q} \rightarrow I(O)_{u,q} - 1, \quad I(O)_{u,q+1} \rightarrow I(O)_{u,q+1} + 1
\]

There are \( 2Q \) such choices. Otherwise, the tour segment that visits \( u \) does not cross into \( C_y \). The algorithm chooses \( v_1, v_2 \in b \) and \( q < Q \), makes the following changes to the flow at \( v_1 \) and \( v_2 \):

\[
I_{v_1,q} \rightarrow I_{v_1,q} + 1, \quad O_{v_1,q+1} \rightarrow O_{v_2,q+1} + 1,
\]

and adds \( d(v_1, u) + d(u, v_2) \) to the intermediate configuration cost. There are \( w^2Q \) such choices. The algorithm then reroutes all flow through \( u \) to some vertex in the parent bag, \( b \). The algorithm chooses, for each vertex \( v \) of \( b \) and each capacity, the number of the tours that enter (resp. exit) \( C_y \) though \( u \) directly from (resp. to) \( v \). Each such tour adds a cost of \( d(u, v) \) to the intermediate configuration cost. There are \( O(n^{2wQ}) \) such choices. Thus, for each child configuration there are \( O(w^2Qn^{2wQ}) \) choices, giving an \( n^{O(wQ)} \) overall runtime for each forget bag.

For an introduce bag, the parent bag is equal to its child bag plus some vertex \( u \). Since the child bag forms a boundary between the inside and outside of the cluster, no tour can cross directly into the cluster via \( u \), as it must first cross some vertex of the child bag. Therefore the only compatible parent configurations are those that have no tours crossing at \( u \). So for every parent configuration, if \( I_{u,q} = O_{u,q} = 0 \) for all \( q \), the algorithm stores the cost of the corresponding child configuration, namely the configuration that results by removing \( u \). Otherwise the cost is \( \infty \).

For a join bag, the parent bag has two child bags identical to itself. Lemma 16 presents an oracle that tells, in constant time, the minimal cost needed to form parent configuration \((I^0, O^0)\) given child configurations \((I^1, O^1)\) and \((I^2, O^2)\), with an infinite cost if the configurations are not compatible. The algorithm tries all combinations of configurations: the complexity of this step is \( n^{O(wQ)} \).

Since each vertex will appear exactly once in a forget bag, each client will be visited exactly once. The overall complexity is \( n^{O(wQ)} \), as claimed. The algorithm considers all possible solutions and outputs the minimal one, so the resulting cost is optimal.

**Lemma 16.** For each join bag \( b \), it is possible to compute, in \( O(n^{6wQ}) \) time, a table \( T_b \) such that \( T_b[(I^0, O^0), (I^1, O^1), (I^2, O^2)] \) is the minimal cost to connect child configurations \((I^1, O^1)\) and \((I^2, O^2)\) to form parent configuration \((I^0, O^0)\) of \( b \).

**Proof.** We design a dynamic program to compute this table. The base cases are when \( I^0 = I^1 + I^2 \). If \( O^0 = O^1 + O^2 \) the cost is 0, since the configurations are therefore compatible. Otherwise the cost is \( \infty \), because it is not possible to balance incoming and outgoing flow.

For the recursion step, assume \( I^0 \neq I^1 + I^2 \). Pick the first pair \((u, q)\) such that \( I_{u,q}^1 + I_{u,q}^2 - I_{u,q}^0 = x \neq 0 \). If \( x < 0 \), the incoming flow at \( u \) with capacity \( q \) is bigger in \( b \) than in its child bags. Since this is not possible, the cost is \( \infty \). Otherwise, some flow entering Cluster 1 comes from Cluster 2 (or vice versa). Suppose this flow exits Cluster 2 at vertex \( v \): it means that

\[
T_b[(I^0, O^0), (I^1, O^1), (I^2, O^2)] = T_b[(I^0, O^0), (I^1, O^1), (I^2, O^2)] + d(u, v)
\]
where \( \mathcal{I}^1 = \{ I^1, \mathcal{I}^1_{u,v} - 1 \} \) and \( \mathcal{O}^2 = \{ O^2, O^2_{v,q} - 1 \} \). By this equation, the algorithm connects one segment exiting Cluster 2 at \( v \) with capacity \( q \) to a segment entering Cluster 1 at \( u \). The value of \( T_0((I^1, O^0), (I^1, O^1), (I^2, O^2)) \) can therefore be computed in \( \omega \) steps, by applying the above equality for each vertex \( v \) of the boundary and storing the minimum value. This computation requires \( O(\omega Q) \) operations to find the pair \( (u, q) \), and then \( O(\omega) \) operations to compute the value of the table. The recursion step therefore requires \( O(\omega Q) \) time.

As there are \( O(n^{6\omega Q}) \) states for this DP, the overall complexity is therefore \( O(\omega Q n^{6\omega Q}) = O(n^{6\omega Q}) \), concluding the proof.

### 6 Embedding for Multiple Depots

We present in this section how to extend Theorem 1 and apply it to several problems.

#### 6.1 Theorem

**Theorem 3**

For \( \varepsilon > 0 \), metric space \( G \) of highway dimension \( \eta \) and a set \( S \) of points of \( G \), there exists a graph \( H \) and an embedding \( \phi(\cdot) \) of \( G \) into \( H \) such that

- \( H \) has treewidth \( f(\eta, |S|, \varepsilon) \) for some fixed function \( f \), and
- for all points \( u \) and \( v \), \( d_G(u, v) \leq d_H(\phi(u), \phi(v)) \leq (1 + O(\varepsilon))d_G(u, v) + O(\varepsilon) \min(d_G(S, u), d_G(S, v)) \)

We slightly modify the embedding of Theorem 1 in that purpose. First, the algorithm adds the set \( S \) to the shortest-path cover at every scale. Then it computes the town decomposition with respect to these shortest-path covers, and embeds the top-level towns using Lemma 1. By analogy with Section 4, we define the set \( X_0 \) to be a \( \varepsilon 2^t \)-net of \( \cup_{s \in S} B_2^t(s) \) (and we ensure moreover that the \( X_0 \) are nested). We also modify the definition of \( l(\mathcal{X}) \): for a set \( \mathcal{X} \), \( l(\mathcal{X}) = \lceil \log_2(\max_{v \in \mathcal{X}} d(S, v)) \rceil \).

Following Section 4, the host graph connects every vertex \( v \) of a town \( T \) to every hub \( h \) in \( X_0(l(T)), \ldots, X_0(l(T) + \log_2(1/\varepsilon)) \) with an edge of length \( d_G(v, h) \).

We now prove that this embedding has the properties of Theorem 3. We use \( H \) to denote the host graph produced by the embedding. First, we prove the first point: the treewidth is bounded.

**Proof.** Let \( \theta_S \) be the doubling dimension of the approximate core hubs when the shortest-path cover includes \( S \). The proof of Theorem 4.2 in Feldmann et al. [20] can be extended to show that, if the shortest-path covers are locally \( \eta \log \eta \)-sparse, \( \theta_S = O(\log ((\eta^2 + |S|) \log(1/\varepsilon))) \). The proof of Lemma 13 directly gives that \( |X_0| \leq |S| (\frac{2}{\varepsilon})^{\theta_S} \). Finally, following the proof of Lemma 14, the host graph has a treewidth bounded by a function of \( \eta \), \( |S| \) and \( \varepsilon \).

We now prove the distortion bound.

**Proof.** Let \( u \) and \( v \) be two points of the metric space and \( h \) be the approximate core hub such that \( d_G(u, h) + d_G(h, v) \leq (1 + O(\varepsilon))d_G(u, v) \). Let \( s_u, s_v \) and \( s_h \) denote the points of \( S \) closest to \( u \), \( v \) and \( h \). The proof is divided into three parts, according to the distances between \( l(h), l(T_u) \) and \( l(T_v) \).
We infer from the definition of \( h \) that
\[
\text{Equation 1 finally gives}
\]
\[
d \leq \frac{1}{2} d_G(u, s_h) + d_G(s_h, \hat{h})
\]
In this case, we have
\[
d \leq d_G(h, s_h) \leq d_G(h, s_u) \text{ and using the triangle inequality we obtain}
\]
\[
d_G(h, s_h) \leq d_G(h, u) + d_G(u, s_u). \text{ By definition, } h \text{ is near the shortest path between } u \text{ and } v: \text{ this gives the desired bound for } u \text{ (the same holds for } v). \]

Consider three cases, illustrated in Figure 4. Suppose that both \( l(h) \leq l(T_u) + \log_2(1/\varepsilon) \) and \( l(h) \leq l(T_v) + \log_2(1/\varepsilon) \) (see Figure 4a). Let \( \hat{h} \) be the point in \( X_0^{l(h)} \) closest to \( h \): by definition of a \( \varepsilon \)-net, \( d_G(h, \hat{h}) \leq \varepsilon 2^{l(h)} \leq 2 \varepsilon d_H(h, s_h) \); by construction of the embedding, \( \hat{h} \) is adjacent to \( u \) and \( v \). In this case, we have
\[
d_H(u, v) \leq d_H(u, \hat{h}) + d_H(\hat{h}, v) \leq d_G(u, \hat{h}) + d_G(\hat{h}, v) \leq d_G(h, u) + d_G(v, h) + 2d_G(h, \hat{h})
\]
We infer from the definition of \( h \) and \( \hat{h} \) that \( d_H(u, v) \leq (1 + O(\varepsilon))d_G(u, v) + 4\varepsilon d_H(h, s_h) \) and using Equation 1
\[
d_H(u, v) \leq (1 + O(\varepsilon))d_G(u, v) + O(\varepsilon) \min(d_G(u, s), d_G(v, s))
\]
Then suppose that \( l(h) > l(T_u) + \log_2(1/\varepsilon) \) but \( l(h) \leq l(T_v) + \log_2(1/\varepsilon) \) (see Figure 4b). It means that \( u \) is not adjacent to \( \hat{h} \) but \( v \) is. It means in particular that \( d_G(s_h, h) > \frac{1}{2} d_G(s_u, u) \). The shortest-path between \( u \) and \( v \) is therefore approximated in the host graph by the path \( u, s_u, \hat{h}, v \). The edges along this path have the length as in \( G \), therefore \( d_H(u, v) \leq d_H(u, s_u) + d_H(s_u, \hat{h}) + d_H(\hat{h}, v) \leq d_G(u, s_u) + d_G(s_u, \hat{h}) + d_G(\hat{h}, v) \). We now apply the triangle inequality in \( G \):
\[
d_G(s_u, \hat{h}) \leq d_G(s_u, u) + d_G(\hat{h}, v). \text{ Using the former inequality and previously-derived bounds gives}
\]
\[
d_H(u, v) \leq 2d_G(u, s_u) + d_G(u, \hat{h}) + d_G(\hat{h}, v) \leq 2\varepsilon d_G(h, s_h) + d_G(h, u) + d_G(h, v) + 2d_G(\hat{h}, h)
\]
Recall that \( d_G(h, \hat{h}) \leq 2\varepsilon d_G(h, s_h) \) and \( d_G(u, h) + d_G(h, v) \leq (1 + \varepsilon)d_G(u, v) \). Using this and Equation 1 finally gives
\[
d_H(u, v) \leq (1 + \varepsilon)d_G(u, v) + 4\varepsilon d_H(h, s_h) \leq (1 + O(\varepsilon))d_G(u, v) + O(\varepsilon) \min(d_G(u, s), d_G(v, s))
\]
Finally, suppose that \( l(h) > l(T_u) + \log_2(1/\varepsilon) \) and \( l(h) > l(T_v) + \log_2(1/\varepsilon) \) (see Figure 1c). It means in particular that neither \( u \) nor \( v \) is adjacent to \( \hat{h} \). In this case, the shortest path between \( u \) and \( v \) is approximated in the host graph by the path \( u, s_u, \hat{h}, s_v, v \): using the same arguments as in the former case, we derive that \( d_H(u, v) \leq (1 + O(\varepsilon))d_G(u, v) + O(\varepsilon) \min(d_G(u, S), d_G(v, S)) \).

6.2 Applications

**Multiple-Depot Capacitated Vehicle Routing**

The first application we consider is for *Multiple-Depot Capacitated Vehicle Routing* with a constant number of depots. Let \( S \) denote the set of depots, and recall that \( Z \) is the set of clients.

Generalizing the algorithm from Section 4 relies on generalizing the lower bound given in Lemma 2 to
\[
\frac{1}{Q} \sum\{d(c, S) : c \in Z\},
\]
as proved in [13]. This lower bound allows for an error of \( \varepsilon d(c, S) \) for each client \( c \): the embedding of Theorem 3 can therefore be applied.

**Lemma 17.** For any \( 1/4 \geq \varepsilon > 0, \eta > 0, k \) and any \( Q > 0 \), there is a polynomial-time algorithm that, given an instance of *Multiple-Depots Capacitated Vehicle Routing* in which the capacity is \( Q \), the number of depots is \( k \) and the metric has highway dimension at most \( \eta \), finds a solution whose cost is at most \( 1 + O(\varepsilon) \) times optimum.

The proof that an optimal solution in the host graph gives an approximate solution on the original graph follows directly from Lemma 10, and the DP presented in Section 5 can be extended easily: for a constant number of depots and a constant highway dimension, the embedding gives a constant treewidth.

**k-center**

Another application is to get a fixed-parameter approximation (FPA) for *k-center* in a graph \( G \) with highway dimension \( \eta \), i.e. an algorithm with running time \( f(\eta, k)n^{O(1)} \).

**Theorem 4** There is a constant \( c \) such that, for any \( \eta > 0, k > 0 \) and \( \varepsilon > 0 \), there is an \( f(\eta, k, \varepsilon)n^c \) algorithm that, given an instance of *k-Center* in which the metric has highway dimension at most \( \eta \), finds a solution whose cost is at most \( 1 + \varepsilon \) times optimum.

The algorithm proceeds in two steps: first, computes a constant-factor approximation \( S \) (see [29] or [23] for a 2-approximation). Applying Theorem 3 to \( G \) with the set \( S \) gives a host graph. Finally, the algorithm runs a DP that gives a \((1 + \varepsilon)\)-approximation of the optimal solution in the host graph. We prove that this solution is also a \((1 + \varepsilon)\)-approximation of the optimal solution in the original graph.

**Lemma 18.** A \((1 + \varepsilon)\)-approximation of *k-center* in the host graph given by Theorem 3 is a \((1 + O(\varepsilon))\)-approximation of *k-center* in the original graph.

**Proof.** Let \( OPT_H \) denote the optimal solution in the graph \( H \). For each vertex \( u \), let \( c_u \) denote the closest center to \( u \) in \( OPT_G \). We have the following:

\[
cost_H(OPT_G) = \max_{u \in V(G)} d_H(u, c_u) \leq \max_{u \in V(G)} (1 + O(\varepsilon))d_G(u, c_u) + O(\varepsilon) \min(d_G(u, S), d_G(c_u, S))
\]

This inequality can be rewritten

\[
cost_H(OPT_G) \leq (1 + O(\varepsilon)) \max_{u \in V(G)} d_G(u, c_u) + O(\varepsilon) \max_{u \in V(G)} d_G(u, S)
\]
Since the set $S$ is a $O(1)$-approximate solution in $G$, $\text{cost}_H(\text{OPT}_G) \leq (1 + O(\varepsilon))\text{cost}_G(\text{OPT}_G) + O(\varepsilon)\text{cost}_G(\text{OPT}_G) = (1 + O(\varepsilon))\text{cost}_G(\text{OPT}_G)$. By definition of $\text{OPT}_H$, $\text{cost}_H(\text{OPT}_H) \leq \text{cost}_H(\text{OPT}_G)$ and therefore $\text{cost}_H(\text{OPT}_H) \leq (1 + O(\varepsilon))\text{cost}_G(\text{OPT}_G)$. That is, since the optimal solution in $H$ is an approximate solution in $G$, an approximate solution in $H$ is also an approximate solution in $G$.

The complexity of finding a constant-factor approximation and of constructing the embedding is a polynomial in $n$ with fixed degree. The complexity of the DP given by Schild, Fox-Epstein and Klein [21] for a treewidth $tw$ is $O(n(\log n)^{tw})$ which is $O(n^{O(1)tw^{2tw}})$ following Lemma 1 in Katsikarelis et al. [30]. As the treewidth only depends on the highway dimension $\eta$, $k$ and $\varepsilon$, the FPA claims follows.

$k$-median

The last application presented here is to get a FPA $k$-median. The outline is the same as for $k$-center: first compute a constant-factor approximation $S$ (see [34]), then apply Theorem 3 using the set $S$ and finally compute an approximate solution in the host graph. The dynamic program for $k$-center can be adapted to solve $k$-median with the same complexity. The following lemma is straightforward:

Lemma 19. A $(1 + \varepsilon)$-approximation of $k$-median in the host graph given by Theorem 3 is a $(1 + O(\varepsilon))$-approximation of $k$-median in the original graph.

The proof is indeed the same as for Lemma 18 replacing the max by a sum.

Acknowledgements

Thanks to Andreas Feldmann and Vincent Cohen-Addad for helpful discussions and comments.

References

[1] I. Abraham, D. Delling, A. Fiat, A. V. Goldberg, and R. F. Werneck. Vc-dimension and shortest path algorithms. In International Colloquium on Automata, Languages, and Programming, pages 690–699. Springer, 2011.

[2] I. Abraham, D. Delling, A. Fiat, A. V. Goldberg, and R. F. Werneck. Highway dimension and provably efficient shortest path algorithms. J. ACM, 63(5):41:1–41:26, Dec. 2016.

[3] I. Abraham, A. Fiat, A. V. Goldberg, and R. F. Werneck. Highway dimension, shortest paths, and provably efficient algorithms. In Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms, pages 782–793. Society for Industrial and Applied Mathematics, 2010.

[4] A. Adamaszek, A. Czumaj, and A. Lingas. Ptas for k-tour cover problem on the plane for moderately large values of $k$. Algorithms and Computation, pages 994–1003, 2009.

[5] S. Arora, M. Grigni, D. R. Karger, P. N. Klein, and A. Woloszyn. A polynomial-time approximation scheme for weighted planar graph tsp. In SODA, volume 98, pages 33–41, 1998.
REFERENCES

[6] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-medians and related problems. In Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998, pages 106–113, 1998.

[7] T. Asano, N. Katoh, and K. Kawashima. A new approximation algorithm for the capacitated vehicle routing problem on a tree. Journal of Combinatorial Optimization, 5(2):213–231, 2001.

[8] T. Asano, N. Katoh, H. Tamaki, and T. Tokuyama. Covering points in the plane by k-tours: a polynomial approximation scheme for fixed k. IBM Tokyo Research Laboratory Research Report RT0162, 1996.

[9] T. Asano, N. Katoh, H. Tamaki, and T. Tokuyama. Covering points in the plane by k-tours: towards a polynomial time approximation scheme for general k. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pages 275–283. ACM, 1997.

[10] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In STOC, pages 250–257, 2002.

[11] H. Bast, S. Funke, and D. Matijevic. Ultrafast shortest-path queries via transit nodes. In The Shortest Path Problem, pages 175–192, 2006.

[12] H. Bast, S. Funke, D. Matijevic, P. Sanders, and D. Schultes. In transit to constant time shortest-path queries in road networks. In Proceedings of the Meeting on Algorithm Engineering & Experiments, pages 46–59. Society for Industrial and Applied Mathematics, 2007.

[13] A. Becker, K. Philip N., and D. Saulpíć. A quasi-polynomial-time approximation scheme for vehicle routing on planar and bounded-genus graphs. In European Symposium on Algorithms. Springer, 2017. to appear.

[14] V. Cohen-Addad, P. N. Klein, and C. Mathieu. Local search yields approximation schemes for k-means and k-median in Euclidean and minor-free metrics. In 57th Annual IEEE Symposium on Foundations of Computer Science FOCS, pages 353–364, 2016.

[15] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. Parameterized algorithms, volume 3, chapter 7: Treewidth. Springer, 2015.

[16] A. Das and C. Mathieu. A quasipolynomial time approximation scheme for euclidean capacitated vehicle routing. Algorithmica, 73(1):115–142, 2015.

[17] D. Feldman and M. Langberg. A unified framework for approximating and clustering data. In Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011, pages 569–578, 2011.

[18] D. Feldman, M. Monemizadeh, and C. Sohler. A PTAS for k-means clustering based on weak coresets. In SoCG, pages 11–18, 2007.

[19] A. E. Feldmann. Fixed parameter approximations for k-center problems in low highway dimension graphs. In Proceedings, Part II, of the 42nd International Colloquium on Automata, Languages, and Programming-Volume 9135, pages 588–600. Springer-Verlag New York, Inc., 2015.

[20] A. E. Feldmann, W. S. Fung, J. Könemann, and I. Post. A (1+ε)-embedding of low highway dimension graphs into bounded treewidth graphs. In International Colloquium on Automata, Languages, and Programming, pages 469–480. Springer, 2015.
[21] E. Fox-Epstein, P. N. Klein, and A. Schild. Embedding planar graphs into low-treewidth graphs, with application to efficient approximation schemes for metric problems. Unpublished manuscript, 2017.

[22] B. L. Golden and R. T. Wong. Capacitated arc routing problems. Networks, 11(3):305–315, 1981.

[23] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. Theoretical Computer Science, 38:293–306, 1985.

[24] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on, pages 534–543. IEEE, 2003.

[25] M. Haimovich and A. Rinnooy Kan. Bounds and heuristics for capacitated routing problems. Mathematics of operations Research, 10(4):527–542, 1985.

[26] S. Hamaguchi and N. Katoh. A capacitated vehicle routing problem on a tree. In International Symposium on Algorithms and Computation, pages 399–407. Springer, 1998.

[27] S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. Discrete & Computational Geometry, 37(1):3–19, 2007.

[28] S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004, pages 291–300, 2004.

[29] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. Mathematics of operations research, 10(2):180–184, 1985.

[30] I. Katsikarelis, M. Lampis, and V. T. Paschos. Structural parameters, tight bounds, and approximation for \((k, r)\)-center. arXiv preprint arXiv:1704.08868, 2017.

[31] M. Khachay and R. Dubinin. Ptas for the euclidean capacitated vehicle routing problem in \(r^d\). In Proceedings of the 9th International Conference on Discrete Optimization and Operations Research (DOOR 2016), pages 193–205. Springer, 2016.

[32] A. Kumar, Y. Sabharwal, and S. Sen. Linear-time approximation schemes for clustering problems in any dimensions. J. ACM, 57(2), 2010.

[33] J. Plesník. On the computational complexity of centers located in a graph. Aplikace matematiky, 25(6):445–452, 1980.

[34] D. B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pages 265–274. ACM, 1997.