A Quasipolynomial Time Approximation Scheme for Euclidean Capacitated Vehicle Routing

Aparna Das · Claire Mathieu

Received: 9 September 2011 / Accepted: 24 May 2014 / Published online: 12 June 2014 © Springer Science+Business Media New York 2014

Abstract In the capacitated vehicle routing problem we are given the locations of customers and depots, along with a vehicle of capacity $k$. The objective is to find a minimum length collection of tours covering all customers such that each tour starts and ends at a depot and visits at most $k$ customers. The problem is a generalization of the traveling salesman problem. We present a quasipolynomial time approximation scheme for the Euclidean setting of the problem when all points lie in $\mathbb{R}^d$ for constant dimension $d$.

Keywords Geometric algorithm · Approximation algorithms · Vehicle routing · Combinatorial optimization

1 Introduction

In 1959 Dantzig and Ramser [10] introduced the truck dispatching problem where the goal was to find short delivery routes to customers using a truck of limited capacity that refills at depots. At the time the authors stated that “no practical applications have been made as yet” for the problem, but went on to describe a linear programming algorithm whose “calculations may be readily performed by hand or automatic digital computing machine” [10].

This makes the truck dispatching problem the first mathematically formulated vehicle routing problem (VRP). Since its introduction numerous VRPs with a variety of
different constraints have been formulated. VRPs are widely studied by researchers in Operations Research and Computer Science and several books (see [13,22] and [12], among others) have been written on these problems. All VRPs involve finding delivery routes using vehicles of limited capacity and typically the objective is to minimize the total cost of the routes.

**Capacitated vehicle routing problem.** We study the most basic form of the vehicle routing problem, the capacitated version (CVRP), which is also referred to as the $k$-tours problem in the computer science literature [2,6]. The input consists of an integer $k$ denoting the capacity of the vehicle, a set $C$ of $n$ customer locations and a set $D$ of depot locations. The objective is to find a collection of tours that covers all customers in $C$, such that each tour starts and ends at some depot in $D$ and visits at most $k$ customers, and such that the total length of the collection of tours is minimized. We study the Euclidean version of the problem where all locations (customers and depots) lie on the Euclidean plane and distances are given by the Euclidean metric.

**Approximation algorithms.** CVRP is NP-hard [15] but admits constant factor approximations. All known constant factor approximations for metric CVRP are based on solving the traveling salesman problem (TSP) as a subroutine. For the single depot setting of CVRP, Haimovich and Rinnooy Kan [14] gave a $(1 + \alpha)$ approximation where $\alpha$ is the best approximation factor for TSP. Li and Simchi-Levi [18] extended this to obtain a $2(1 + \alpha)$ factor approximation for the multiple depot setting. Both algorithms start with a tour of all the customers and partition it into smaller tours such the total number of customers in each tour is at most $k$.

There is unlikely to be better than constant factor approximations for metric CVRP, as even the single depot problem is known to be APX-complete for all $k \geq 3$ [5]. However these hardness results do not extend to the Euclidean setting of the problem.

As CVRP is closely related to TSP, it has been conjectured that, like TSP, Euclidean CVRP also admits a PTAS [2,3,19]. Indeed, in the case of very large capacity, $k = \Omega(n)$, Arora’s PTAS for TSP extends directly to a PTAS for CVRP. For small capacity $k = O(\log n / \log \log n)$, and a single depot, Asano et al. [6] presented a PTAS extending [14]; and, for slightly larger capacity, $k \leq 2^{\log^\delta n}$ (where $\delta$ is a function of $\epsilon$), Adamszek et al. presented a PTAS using the algorithm of this paper as a black box [1]. For small capacity and multiple depot setting, Cardon et al. [8] extend the methods of Asano et al. [6] and give a PTAS when the number of depots is also small i.e $|D| = O(\log n / \log \log n)$.

**Our results.** We present quasipolynomial time approximation schemes (QPTAS) for CVRP which work for the entire range of $k$. The following theorems summarize our main results.

**Theorem 1** (Main Theorem—Single Depot) The Single-Depot Scheme is a randomized quasipolynomial time approximation scheme for two dimensional Euclidean CVRP with a single depot. Given $\epsilon > 0$, it outputs a solution with expected length $(1 + O(\epsilon))OPT$ in time $n^{\log^{O(1/\epsilon)} n}$. The Single-Depot Scheme can be derandomized.

**Theorem 2** (Main Theorem—Multiple Depot) The Multi-Depot Scheme is a randomized quasipolynomial time approximation scheme for two dimensional Euclidean
CVRP with multiple depots. Given $\epsilon > 0$, it outputs a solution with expected length $(1 + O(\epsilon))OPT$, in time $n^{\log^{O(1/\epsilon)} n}$. The Multi-Depot Scheme can be derandomized.

We first describe our algorithm for the single depot setting and extend it to handle multiple depots. We present our results for the two dimensional setting and describe how to extend them to higher dimension $d$, for constant $d$.

**Where previous approaches fail.** Similar to the PTAS for Euclidean TSP from [2], our algorithms place a bounding box around the input, apply a randomized recursive dissection, and search for a solution which has few crossings of box boundaries. Unfortunately, as noted by Arora [3],

we seem to need a result stating that there is a near-optimum solution which enters or leaves each area a small number of times. This does not appear to be true. […] The difficulty lies in deciding upon a small interface between adjacent boxes, since a large number of tours may cross the edge between them. It seems that the interface has to specify something about each of them, which uses up too many bits.

Indeed, to combine solutions in adjacent boxes it seems necessary to remember the number of points covered by each tour segment and that is too much information to remember.

**Overview of our approach.** To get around this we remember only the approximate number of points covered by tour segments. Naively, this may lead to tours covering more than $k$ customer points. To make the tours feasible we design a simple randomized technique that drops points from tour segments. Our technique ensures that the dropped points can be covered at low cost with additional tours found using a constant factor approximation [14,15] (See Figs. 1, 7). Thanks to dropping points, we may assume that the number of points on each tour segment is a power of $(1 + \epsilon / \log n)$, so there are only $O(\log n \log k)$ possibilities. The quasipolynomial running time of our dynamic program follows.

**Related Techniques.** We refer the reader to [2–4,7,17,19,21] for similar techniques for the design of PTAS or QPTAS for various NP-Hard geometric problems.

### 2 Preliminaries

We present two lower bounds from [14,15]. Intuitively Rad measures the average distance from customers to their closest depot. Rad is a lower bound for all CVRPs.

**Definition 1 (Radius, Rad)** Let $I$ be an instance of CVRP with customers $C$, depots $D$, and vehicle capacity $k$. The radius of customer $i$ is defined as the distance of the closest depot to $i$, $r_i = \min_{d \in D} \text{dist}(i, d)$. Rad is defined as,

$$\text{Rad} = \frac{2}{k} \sum_{i \in C} r_i$$

**Lemma 1 (Rad Lower Bound.)** Let $I$ be an instance of CVRP and $OPT$ denote the cost of the optimal solution of $I$. Then, $\text{Rad} \leq OPT$. 

© Springer
Proof Let $\Pi$ be the set of tours in OPT. The length of a tour $\pi \in \Pi$ is at least $2 \max_{i \in \pi} r_i$. Replacing max with the average we get:

$$\text{OPT} \geq \sum_{\pi \in \Pi} 2 \cdot \max_{i \in \pi} r_i = 2 \cdot \frac{\sum_{i \in \pi} r_i}{k} = \frac{2}{k} \sum_{i \in C} r_i.$$  

⊓⊔

The following lower bound applies only to the single depot version of CVRP.

Lemma 2 (TSP Lower bound) Let $I$ be an instance of single depot CVRP with customers $C$ and depot $d$, and let $\text{OPT}$ denote the cost of the optimal solution of $I$ and $\text{TSP}(I)$ be the traveling salesman tour of $C \cup d$. Then, $\text{OPT} \geq \text{TSP}(I)$.

Proof From OPT, build a solution to $\text{TSP}$ using the triangular inequality. 

⊓⊔

To see that Lemma 2 does not hold in the multiple depot setting: consider an input with two depots which are very far apart such that each depot is surrounded by a cloud of near by customers.

3 Algorithm for Single Depot CVRP

We start with an overview of each step of our algorithm with details appearing in the subsections. Our algorithm pre-processes the input just as in [2]. We place a bounding box around the input points, perform a random dissection of the box, and place portals on the dissection grid lines. Then a quasipolynomial time dynamic program is used to find a structured solution, consisting of tours which are portal respecting, do not cross box boundaries too many times, and cover approximately $k$ points—see Definition 5. The approximation allows us to compute a structured solution in quasi-polynomial time.

The structured solution has near optimal cost but is not feasible: it may include some tours that slightly exceed the capacity $k$. A feasible set of tours is obtained in two phases. First we drop some points from the infeasible tours of the structured solution, making them into what we call black tours. Second, we use the 3-approximation of [14] on the dropped (red) points to get red tours. We output the feasible solution consisting of the union of the black tours and the red tours. See Fig. 1. The approximation scheme is outlined below in algorithm Single-Depot Scheme.

3.1 Preprocessing [2]

Perturbation. (as in [2]) Let $A \leq \text{OPT}/2$ denote the maximum distance between any two input points. Define a bounding box as a smallest box that contains all the input points and whose side length $L \leq 2A$ is a power of 2. Place a grid of granularity $\delta = A\epsilon/(2n)$ inside the bounding box. Move every input point (customers and depots) to the center of the grid box it lies in, to obtain a multiset of points. Scale distances by $8/\delta$ so that all coordinates become integral and the minimum non-zero distance is
Algorithm Single-Depot Scheme

Input: Customers $C$, a depot in $\mathbb{R}^2$, and integer $k$

1: Perturb instance, perform random dissection, and place portals – Subsection 3.1.
2: Compute a structured solution (defined in Section 3.2) with the dynamic program from Section 4.
3: Use the DP's history to construct the structured tours and assign types to points using the randomized type assignment from Subsection 3.3.
4: Color a point black if it has type 0 and otherwise red. Drop all red points from the structured tours.
5: Use the 3-approximation Single-Depot Tour Partitioning algorithm to get solution for the red points.

Output: Union of red tours and black tours.

Fig. 1 A solution computed by the Single-Depot Scheme for an instance with 26 customers (circles), one depot (the star), and $k = 7$. The solid circles are the black points and the empty circles are the red points. The solid tours are computed by the DP in step 2 and each one covers $\leq k$ black points. The dotted tour covers the red points and is computed in step 5 using the 3-approximation.

At least 4. For the analysis, note that up to adding detours from perturbed points to points in the original input, any solution for the perturbed instance yields a solution to the original instance using additional length at most $2n \cdot 2\delta \leq \epsilon \text{OPT}$.

Henceforth we use OPT to denote the optimal CVRP solution of the perturbed instance. We build a randomized dissection of the perturbed instance as described below.

Randomized Dissection. (as in [2]) A dissection recursively partitions a box into 4 smaller boxes of equal size using one horizontal and one vertical dissection line. The bounding box has level 0 as well as its bounding dissection lines, the 4 boxes created by the first dissection have level 1 as well as the two dissection lines that created those boxes, and so on. After scaling the side length of the bounding box is $L \leq 2A \cdot 8/\delta = O(n/\epsilon)$, thus the level of the smallest boxes is $\ell_{\text{max}} = O(\log L) = O(\log(n/\epsilon))$. See
Fig. 2. A randomized dissection is obtained by choosing random integers \(a, b \in [0, L)\), and shifting the \(x\) coordinates of all horizontal dissection lines by \(a\) and all vertical dissection lines by \(b\) and reducing modulo \(L\).

**Property 1** [2] Fix a dissection line \(l\) and level \(\ell \leq \ell_{\text{max}}\). For a randomized dissection, 
\[
\Pr[\text{level}(l) = \ell] = \frac{2^\ell}{L}.
\]

**Portals.** (as in [2]) Let \(m = O(\log L/\epsilon)\) be a power of 2. We place \(2^\ell m\) equidistant portal points on each level \(\ell\) dissection line for all \(\ell \leq \ell_{\text{max}}\). Since a level \(\ell\) line forms the boundary of \(2^\ell\) level \(\ell\) boxes, there are at most \(4^m\) portals along the boundary of any dissection box \(b\). As \(m\) and \(L\) are powers of 2, portals at lower level boxes will also be portals in higher level boxes. See Fig. 2.

**Definition 2** *(Portal respecting and light)* A tour is *portal respecting* if it crosses dissection lines only at portals and *light* if it crosses each side of a dissection box at most \(r = O(1/\epsilon)\) times.

**Theorem 3** [2] (Arora’s Structure Theorem) Let \(OPT_{TSP}\) denote the cost of an optimal solution for an instance of Euclidean TSP and let \(\Delta\) be a randomized dissection. With probability \(\geq 1/2\) there exists a tour that is portal respecting and light with respect to \(\Delta\), and that has cost \((1 + O(\epsilon))OPT_{TSP}\).

Arora’s Structure Theorem 3 implies that one can focus on computing a CVRP solution consisting of only portal respecting and light tours. We will make this argument formal in the proof of Theorem 4.

### 3.2 Structured Solution

We define \(\tau = O(\log L \log k)\) *thresholds* in the range \([1, k]\). Instead of remembering the exact number of points covered by a tour segment we round it to one of the
predefined thresholds. More specifically, initially all points are assigned to be type 0. At each level $\ell$, to “round” a tour segment covering $x$ type 0 points, we first choose exactly $x - t$ type 0 points on the segment, such that $t$ is the largest predefined threshold which is at most $x$. Then the type of these chosen points are set to $\ell$ to indicate that the points should be dropped from the segment at level $\ell$. In the end all points with types between 1 and $\ell_{\text{max}}$ are dropped from the structured tours. We formally define thresholds, types, and rounded segments below.

**Definition 3** *(Thresholds, rounded segments, rounded box)*

- Let $\tau = \lceil \log(1 + \epsilon / \log L_1(L)) \rceil + 1/\epsilon$, where $L$ is the side length of the bounding box. For each $i \in \{1, 2, \ldots, \tau\}$ let threshold $t_i = i$ if $i \leq 1/\epsilon$, and $t_i = t_{i-1}(1 + \epsilon / \log L)$ otherwise.
- The type of a point is an integer between 0 and $\ell_{\text{max}}$. If a point has type 0 it is active in all levels. If a point has a type $y > 0$, it is active in all levels $> y$.
- For any box $b$, a segment of $b$ is a piece of a tour that enters and exits $b$ at most once. A segment of $b$ is rounded if it covers exactly $t_i$ active points at level $(b)$ for some $i$, unrounded otherwise. See Fig. 3. A box is rounded if it contains only rounded segments, unrounded otherwise.

Rounding the number of points on tour segments means that, some tours may end up covering more than $k$ points. To ensure that such tours can be made feasible at small cost, we will only round tour segments that lie in dissection boxes with many (at least $\gamma$) segments. In such cases the cost of adding additional tours from the depot to the dropped points in the box can be charged to OPT. Definition 4 below formally defines how we relax capacity constraints.

**Definition 4** *(Relaxed)* A set of tours satisfies the relaxed conditions if they cover all the customers and there exists an assignment of types for the points such that:

1. Each tour visits the depot and covers at most $k$ type 0 points.
2. Let $b$ be a dissection box and let $s$ be a segment in $b$ that covers $t$ active points at level$(b)$. Then $s$ has at most $t(1 + \epsilon / \log L)$ active points at level$(b) + 1$.
3. Let $\gamma = \lceil \log^4 L / \epsilon^4 \rceil$ be the rounding size. Then any box $b$ with more than $\gamma$ segments is a rounded box containing only rounded segments.

By the first condition of Definition 4 a relaxed tour can cover at most $k$ points of type 0 but may also cover some additional points of type $i > 0$. Note that type 0 points are never dropped and hence are included in the final black tours. The second condition implies that at most $O(\epsilon / \log L)k$ points can be dropped at each level from a tour segment. Thus a total of at most $O(\epsilon)k$ points are dropped to make a relaxed tour feasible. Thanks to the third condition, tour segments are only rounded in boxes containing more than $\gamma$ segments.

We now define structured tours.

**Definition 5** *(Structured)* A set of tours $\Pi'$ is structured with respect to a fixed random dissection $\Delta$, if $\Pi'$ consists of tours which are portal respecting and light, and there exists a type assignment for the customer points satisfying the relaxed conditions.
Fig. 3 Types and rounded segments. The figure shows two boxes at levels \( \ell + 1 \), one box at level \( \ell \) and four types of points. The points of type \( \ell + 1 \) (white) and type \( = \ell + 1 \) (stripped) are inactive in all boxes shown. Points with type \( \ell \) (dotted) are active at level \( \ell + 1 \) and the points of type \( < \ell \) (solid) are active in all shown boxes. If the thresholds are 5, 9, the segment is rounded at level \( \ell \) covering 9 active points. At level \( \ell + 1 \), the segment is rounded in the left box covering 5 active points and unrounded in the right box covering 6 active points.

We define an extended objective function \( F \) that in addition to charging for the length of the tours will also charge for the number of tour crossings at each level.

Definition 6 (Extended Objective Function) Fix a dissection \( \Delta \) and let \( \Pi \) be a set of tours. For every level \( \ell \) let \( c(\pi, \ell) \) be the number of times a tour \( \pi \in \Pi \) crosses the boundaries of level \( \ell \) boxes, and let \( d_\ell = L/2^\ell \) denote the length of a level \( \ell \) dissection box. The extended objective function is:

\[
F(\Pi) = \sum_{\pi \in \Pi} \text{length}(\pi) + \frac{\epsilon}{\log L} \sum_{\text{level } \ell} \sum_{\pi \in \Pi} c(\pi, \ell) \cdot d_\ell.
\] (1)

Let \( \Pi \) a feasible solution. Thus the tours in \( \Pi \) are not required to be portal respecting and light but they do satisfy the capacity constraints. Our structure theorem proves that we can obtain a set of structured tours \( \Pi' \) such that the extended cost of \( \Pi' \) is close to the total length of \( \Pi \).

Theorem 4 (Structure Theorem) Let \( \Pi \) be a set of feasible tours. Let \( \Pi' \) be the portal respecting and light tours obtained by applying Arora’s structure theorem to each tour of \( \Pi \). Then there exists a type assignment satisfying the relaxed conditions to make \( \Pi' \) structured, and such that, in expectation over random shifts of the dissection, \( F(\Pi') \leq (1 + O(\epsilon))\text{length}(\Pi) \).

The proof of Theorem 4 appears in Sect. 5.

Section 4 presents the dynamic program (DP) that proves the following theorem.
Fig. 4 Randomized type assignment. \( b \) is a level \( \ell \) box with 8 active points (solid circles) and two inactive points (white circles). If the largest threshold less than 8 is \( t = 5 \) then \( y = 3 \) points are labeled type \( \ell \) and marked to be dropped. The \( y = 3 \) chosen points are \( p \) and the next two active points.

**Theorem 5** (Dynamic Program) Given an instance \( I \) of single depot CVRP, let OPT denote the length of the optimal solution and OPT\( ^S \) denote the length of the structured solution minimizing objective function \( F \). The dynamic program of Sect. 4 computes \( \text{OPT}^S(I) \) in time \( n \log^{O(1/\epsilon)} n \).

### 3.3 Assigning Types

The Type Assignment algorithm illustrated in Fig. 4 below rounds tour segment \( S \) as follows. It picks an interval of points on \( S \) and sets the type of all these points to indicate that they should be dropped. If \( x \) is the number of active points covered by segment \( S \) and \( t \) is the largest threshold such that \( t_i \leq x < t_{i+1} \), the Type Assignment algorithm labels \( y = x - t_i \) active points in \( S \) as type \( \ell \).

#### Algorithm Type Assignment

| Input: Segment \( S \) at level \( \ell \) with \( x \) active points |
|---|
| 1: Choose an active point \( p \) uniformly among the active points of \( S \). |
| 2: Let \( t \) be the largest threshold \( \leq x \). |
| 3: Choose an interval of \( x - t \) active points from \( S \) as follows: Choose \( p \) and the next \( (x - t) - 1 \) active points after \( p \) on \( S \), wrapping around to the start of \( S \) if necessary. |
| 4: Label all \( x - t \) points in the interval to type \( \ell \). |

**Output:** Segment \( S \) with \( t \) active points.

### 3.4 A Constant Factor Approximation [14]

To get a solution for the red dropped points, we use the tour partitioning algorithm of Haimovich and Rinnooy Kan. The algorithm computes a TSP of the red points and partitions it into tours that cover at most \( k \) points.

**Theorem 6** [6, 14] For any single depot instance \( I \) of metric CVRP the output of the Single-Depot Tour Partitioning algorithm has expected length at most \( \text{Rad}(I) + 2 \cdot \text{TSP}(I) \leq 3 \text{OPT} \).
Algorithm Single-Depot Tour Partitioning [14]

Input: \( n \) Customers, a depot, and integer \( k \)

1: Compute a tour \( \pi \) of the \( n \) customers and the depot using a 2-approximation of TSP.
2: Choose a point \( p \) uniformly at random from \( \pi \).
3: Go around \( \pi \) starting at \( p \), and every time \( k \) points are visited, take a detour to the depot.

Output the resulting set of \( \lfloor n/k \rfloor + 1 \) tours.

4 The Dynamic Program

The table specification. For each dissection box \( b \), a configuration \( C \), either rounded or unrounded, describes the tour segments that cross the boundaries of \( b \).

- Rounded Configuration: a list of numbers \( r_{p,q,t,d} \), for every pair of portals \( p, q \) of \( b \), every threshold \( t \in \{t_1, \ldots, t_t\} \), and depot indicator \( d \in \{0, 1\} \). A particular \( r_{p,q,t,d} \) is the number of rounded tour segments in \( b \) that use portals \( p \) and \( q \) to enter and exit \( b \), cover exactly \( t \) active points, and visit the depot as indicated by \( d \).

- Unrounded Configuration: a list of \( \gamma \) tuples of the form \((p, q, u, d)\). The \( j \)-th tuple \((p, q, u, d)_j\) describes an unrounded tour segment in \( b \) that uses portals \( p \) and \( q \) to enter and exit \( b \), covers exactly \( u \) points, and visits the depot as indicated by \( d \).

For each dissection box \( b \) and configuration \( C \) of \( b \), the table entry \( L_b[C] \) is the minimum cost to place tour segments in \( b \) which are compatible with \( C \) and structured as defined by Definition 5, with cost computed by objective \( F \).

The algorithm returns the minimum table entry over all configurations of the root level box.

The recurrence. The table entries are computed bottom-up. The base case is for a leaf box \( b \). Consider the case where \( b \) doesn’t contain the depot. If \( C \) is an unrounded configuration, then it is feasible if the segments cover all points of \( b \). If \( C \) is a rounded configuration whose segments together cover \( t \) points, then \( C \) is feasible if the segments cover all but at most \( t(\epsilon/\log L) \) points of \( b \). Its cost can be computed easily as all points are located in the center of a leaf box. Now consider the case where \( b \) contains the depot. Then all points of \( b \) which are not described in \( C \) can be covered at zero cost as the depot and all points are in the center of leaf box \( b \).

Let \( b \) be a box at level \( \ell \) and let \( b_1, b_2, b_3, b_4 \) be the children of \( b \) at level \( \ell + 1 \). As every tour is structured, a tour segment (or a tour) in \( b \) crosses the boundaries of \( b_1, b_2, b_3, b_4 \) inside \( b \), at most \( 4r \) times, so it is the concatenation of at most \( 4r + 1 \) portal-to-portal pieces. Each piece can be described by a tuple \((p, p', x, d)\), where \( p, p' \) are portals, \( d \) is the depot indicator flag and \( x \) is either a threshold \( t_i \) for some \( i < \tau \), or \( x \) is a number \( j \leq \gamma \) indicating that the piece is the \( j \)-th unrounded tour in a child box of \( b \). Each tour segment (or tour) in \( b \) is described by a concatenation profile \( \Phi \) which is a list of the at most \( 4r + 1 \) tuples it is constructed from. Let \( x_\Phi \) denote the sum of active points picked up by all pieces in \( \Phi \). A profile \( \Phi \) that describes a tour has \( p_1 = p_{s+1} \), and is feasible if it visits the depot and \( x_\Phi \leq k \). If \( C \) is an unrounded configuration, then a profile \( \Phi \) that describes a segment is marked as having exactly \( x_\Phi \) active points, else it is rounded and marked as having \( t_i \) active points, \( t_i \leq x_\Phi < t_{i+1} \).
Let \( n_i \) denote the number of tour segments in \( b \) with concatenation profile \( \Phi_i \) for \( i \leq \varphi \), the number of possible concatenation profiles. An interface vector \( I = (n_i)_{i \leq \varphi} \) is a list of \( \varphi \) entries. Let \( C_0, C_1, C_2, C_3, C_4 \) be configurations for box \( b \) and its four children boxes. They are compatible if there exists an interface vector \( I \) such that gluing \( C_1, C_2, C_3, C_4 \) according to \( I \) yields \( C_0 \). This is tested by the dynamic program by exhaustive search. Let \( c_b(I) \) be the total number of tour segments in \( b \) as specified by \( I \). The cost of \( C_0 \) according to objective function \( F = (\epsilon / \log L) \cdot 2c_b(I) \) plus the sum of the costs of \( L_b(C_i) \) for each child box \( i \leq 4 \). Minimizing over all compatible tuples yields \( L_b(C_0) \).

**The runtime.** There are \( O(n) \) non empty boxes. The number of possible configurations for a box \( b \) is the number of possible rounded configurations plus the number of unrounded configurations. A rounded configuration is a list of entries. There are \( O(\log^2 L) \) possible pairs of portals \((p, q)\), \( \tau = O(\log^2 L) \) thresholds, and \( 2 \) values for \( d \), so the rounded configuration is a list of \( O(\log^3 L) \) numbers, and there are \( n^{O(\log^4 L)} \) possible rounded configurations. Each unrounded configuration is a list of \( \gamma = O(\log^4 L) \) tuples \((p, q, u, d)\) and there are \( \log^2 L \cdot n \cdot 2 \) possibilities for each tuple, for a total of \( n^{O(\log^4 L)} \) possible unrounded configurations. Thus the DP table has overall size \( n^{O(\log^4 L)} = n^{O(\log^4 \cdot n)} \).

Each concatenation profile \( \Phi \) has a list of \( O(r) = O(1/\epsilon) \) tuples \((p, p', x, d)\). There are \( O(\log L) \) choices for portal \( p \), \( O(\log L) \) for portal \( p' \), \( 2 \) for \( d \), and \( \gamma + \tau = O(\log^6 L) \) for \( x \), so there are \( \varphi = (\log^6 L)^{O(r)} = (\log L)^{O(1/\epsilon)} \) possible concatenation profiles. There are \( n^\varphi \) possible interface vectors \( I \) for a box \( b \), a quasipolynomial number of possibilities. There are \( n^{O(\log^4 L)} \) possible values for each \( C_i \). Checking consistency for a particular interface vector \( I \) and configurations \( \{C_i\}_{i \leq 4} \) can be done in polynomial time in the size of their descriptions. Thus to compute \( L_b[C_0] \) takes time polynomial in \( n^{\log^{O(1/\epsilon)} L} \), which is \( n^{\log^{O(1/\epsilon)} L} \) as \( L = O(n) \).

**Remark.** The DP only verifies that for the set of tours it returns there exists a type-assignment satisfying the relaxed Definition 4. The DP does not actually label points with specific types. Instead it records the number of active points the tour segments should have. Once the value of \( \text{OPT}^3 \) is found, we can trace through the DP history and make type assignments.

## 5 Proof of the Structure Theorem (Theorem 4)

Theorem 4 follows from Lemmas 4 and 5 below. For each tour \( \pi \in \Pi \), let \( \pi^L \) denote the portal respecting and light version obtained using Arora’s Algorithm.

**Lemma 3** (Can be derived from [2]) Let \( \ell \) be a level and \( d_{\ell} \) the side length of a level \( \ell \) dissection box,

\[
E[c(\pi^L, \ell) \cdot d_{\ell}] \leq O(2 + \epsilon) \cdot \text{length}(\pi)
\]

**Lemma 4** Let \( \Pi \) be a any set of tours \( \pi \) and \( \Pi^L \) be the set of portal respecting and light tours \( \pi^L \) obtained when Arora’s structure theorem (Theorem 3) is applied to each tour in \( \Pi \). Then in expectation, \( F(\Pi^L) \leq (1 + O(\epsilon))\text{length}(\Pi) \).
Proof $F(\Pi^L) = \sum_{\pi^L \in \Pi^L} F(\pi^L)$. By definition of $F$,

$$F(\pi^L) = \text{length}(\pi^L) + \frac{\epsilon}{\log L} \sum_{\text{level } \ell} c(\pi^L, \ell) \cdot d_\ell$$  

where $d_\ell$ is the side length of a level $\ell$ box. We apply Arora’s Structure Theorem 3 to the first term, and Lemma 3 to each summand in the second term.

$$F(\pi^L) \leq (1 + O(\epsilon)) \cdot \text{length}(\pi) + \frac{\epsilon}{\log L} \sum_{\text{level } \ell} O(2 + \epsilon) \cdot \text{length}(\pi)$$

As there are $\ell_{\text{max}} = O(\log L)$ levels in the dissection, we get $F(\pi^L) \leq (1 + O(\epsilon))\text{length}(\pi)$, and summing over $\pi$ concludes the proof. $\square$

Lemma 5 constructs a type assignment for $\Pi'$ that satisfies the relaxed conditions.

Lemma 5 Let $\Pi$ denote any set of feasible tours. Then there exists a type assignment such that $\Pi$ satisfies the relaxed conditions of Definition 4.

**Algorithm Relaxed Type Assignment**

- **Input:** A set of feasible tours $\Pi$.
- **Output:** $\Pi$ and the type assignment for points.

1: Label all customer points as type 0
2: for level $\ell$ from $\ell_{\text{max}}$ to 0 do
3: for each level $\ell$ box $b$ with more than $\gamma$ segments do
4: for each segment $s$ in $b$, round $s$ as follows do
5: Let $x$ be the number of active points on segment $s$, and $t_i$ the largest threshold s.t $t_i \leq x$.
6: Pick any $x - t_i$ active points on $s$ and label them as type $\ell$.
7: end for
8: end for
9: end for

Proof (Proof of Lemma 5) Given a feasible solution $\Pi$ the Relaxed Type Assignment algorithm given below summarizes how we assign types such that $\Pi$ satisfies the relaxed conditions of Definition 4.

We now show that the output of the Relaxed Type Assignment algorithm is a relaxed solution satisfying Definition 4. Clearly, the output satisfies the first and third conditions. Now consider the second condition. For a segment that is rounded at level $\ell$, prior to rounding all points on the segment either have type 0 or a type strictly greater than $\ell$. Thus the segment has the same number of active points at level $\ell$ and at level $\ell + 1$. Let $x$ be the number of active points prior to rounding where $t_i \leq x \leq t_{i+1}$. After rounding, the segment has $x - t_i$ points labeled with type $\ell$. This leaves $t_i$ active points at level $\ell$ and $x$ active points at level $\ell + 1$. As $t_i(1 + \epsilon / \log L) = t_{i+1} > x$, the second condition of Definition 4 is satisfied. $\square$
6 Proof of Main Theorem 1

The output of the Single-Depot Scheme has cost equal to the length of the black tours plus the length of red tours.

Since the black tours are obtained by dropping points from the tours computed by the DP their length is at most the length of the DP tours. By Theorem 5 the DP outputs $\text{OPT}^S$. By Theorem 4 applied to the optimal set of tours for $I$, there exists a structured set of tours $\Pi'$ such that in expectation, $F(\Pi') \leq (1 + O(\epsilon))\text{OPT}$. Then:

$$\text{OPT}^S = \text{length}(\Pi^S) \leq F(\Pi^S) \leq F(\Pi') \leq (1 + O(\epsilon))\text{OPT}.$$ 

Let $R$ denote the points colored red by the Single-Depot Scheme. The red tours are found using the 3-approximation Single-Depot Tour Partitioning algorithm of [14], and by Theorem 6 they have length at most $\text{Rad}(R) + 2TSP(R \cup \text{depot})$. The following lemmas show that each of these terms is at most $O(\epsilon)\text{OPT}$, which proves that in expectation over the random shifts of the dissection and the random type assignment the length of the red tours output by the Single-Depot Scheme is $O(\epsilon)\text{OPT}$. The proof of the lemmas appear in the subsections below.

**Lemma 6** Let $R$ be the set of points colored red by the Single-Depot Scheme. In expectation over the random type assignment $\text{Rad}(R) = O(\epsilon)\text{OPT}$.

**Lemma 7** Let $R$ be the set of points colored red by the Single-Depot Scheme. In expectation over the random dissection and type assignment $\text{TSP}(R \cup \text{depot}) = O(\epsilon)\text{OPT}$.

Thus the solution output by the Single-Depot Scheme has total length $(1 + O(\epsilon))\text{OPT}$.

The DP dominates the running time. The derandomization of the Algorithm is discussed in Sect. 8.

6.1 Proof of Lemma 6

For each level $\ell$ let $R_{\ell}$ denote the points labeled as type $\ell$. The red points $R$ is the union of $R_{\ell}$ over all levels. Since a customer point $i$ may be labeled at any level the probability that $i$ is red is,

$$\Pr[i \in R] \leq \sum_{\ell > 0} \Pr[i \in R_{\ell}] \tag{3}$$

We compute the probability that a point $i$ is in $R_{\ell}$. If $i \in R_{\ell}$ it was in the interval chosen by the Type Assignment algorithm while rounding some level $\ell$ segment $S$. Let us prove that the probability that $i$ is chosen in the interval, and hence included in $R_{\ell}$, is $O(\epsilon/\log L)$.

Let $x$ be the number of active points covered by $S$ prior to rounding, and let $t_i$ be the largest threshold such that $t_i \leq x < t_{i+1}$. To round $S$ the Type Assignment algorithm
labels $y = x - t_i$ active points of $S$ as type $\ell$. The $y$ points are selected as an interval by uniformly picking an active point $p$ and then selecting the $y - 1$ active points lying after $p$, wrapping around $S$ if necessary.

There are a total of $x$ different intervals, each one starting at a different active point in $S$, and each active point $i$ lies in $y$ intervals. As the Type Assignment algorithm picks uniformly among these intervals, the probability that $i$ lies in the selected interval is $y/x$. By Definition 3 $t_{i+1} \leq t_i(1 + \epsilon/\log L)$, so we have that,

$$Pr[i \text{ is in the interval}] = \frac{y}{x} = \frac{x - t_i}{x} < \frac{t_{i+1} - t_i}{t_i} \leq \frac{\epsilon}{\log L}.$$  

As there are $\ell_{\text{max}} = O(\log L)$ levels, Eq. 3 reduces to $Pr[i \in R] = O(\epsilon)$. By the definition of Rad (Definition 1) where $C$ is the set of customers and $r_i$ is the radius of $i \in C$ we have,

$$E[\text{Rad}(R)] = \frac{2}{k} \sum_{i \in C} r_i \cdot Pr[i \in R] \leq \frac{2}{k} \sum_{i \in C} r_i \cdot O(\epsilon) = O(\epsilon)\text{Rad}.$$  

The lemma follows as Rad is a lower bound on OPT by Lemma 1.

6.2 Proof of Lemma 7

For each level $\ell \in [0, \ell_{\text{max}}]$ let $R_\ell$ denote points labeled $\ell$, and note that $R = \cup_{\ell \geq 0} R_\ell$. For any $\ell$, we show that $E[\text{TSP}(R_\ell \cup \text{depot})] \leq O(\epsilon/\log L)\text{OPT}$. As the tours of $R_\ell$ for each of the $\ell_{\text{max}} = O(\log L)$ levels can be pasted together at the depot to get a tour of $R$ this gives a tour of $R$ of cost $O(\epsilon)\text{OPT}$ and proves the lemma.

Fix a level $\ell$ and let $B_\ell$ be the boxes at level $\ell$ containing points from $R_\ell$. We upper bound the cost of $\text{TSP}(R_\ell \cup \text{depot})$ by thinking of the tour in three parts. For each box $b \in B_\ell$, let $R_b$ denote the points of $R_\ell$ lying inside $b$. Let the inside part be tours of the points in $R_b$ connected to the boundary of boxes $b \in B_\ell$ and let the boundary part include the boundaries of all boxes in $B_\ell$. Given the inside and the boundary parts, to get a tour of $R_\ell$ we only need a tour of $B_\ell$ and the depot. We refer to this as the outside part. Thus we have,

$$\text{TSP}(R_\ell \cup \text{depot}) \leq \sum_{b \in B_\ell} (\text{TSP of } R_b \text{ connected to boundary of } b)$$

$$+ \sum_{b \in B_\ell} (\text{boundary of } b) + \text{TSP}(B_\ell \cup \text{depot})$$

To bound the outside cost $\text{TSP}(B_\ell \cup \text{depot})$, define a complete graph $G = (V, E)$ such that $V$ has a vertex for each box in $B_\ell$ and a vertex to represent the depot. The edge between two box vertices $b, b'$ has cost equal to the length of the shortest path from any portal of $b$ to any portal of $b'$. The cost of an edge between a box vertex $b$ and the depot vertex is equal to the length of the shortest path from any portal of $b$ to
the depot. Observe that a tour of the depot and boxes in \( B_\ell \) has cost at most two times the minimum spanning tree of \( G \), i.e., \( 2 \text{MST}(G) \). Thus we have,

\[
\text{TSP}(R_\ell \cup \text{depot}) \leq \sum_{b \in B_\ell} (\text{TSP of } R_b \text{ connected to boundary of } b) + \sum_{b \in B_\ell} (\text{boundary of } b) + 2\text{MST}(G)
\]  

(4)

Lemmas 8, 9 and 10 which are stated below and proved in the following subsections help us to bound Eq. 4.

**Lemma 8** (Inside cost) In expectation over the random dissection and the random type assignment, \( \sum_{b \in B_\ell} (\text{TSP of } R_b \text{ connected to boundary of } b) \) is at most \( O(\epsilon/\log L) \text{OPT}^S \).

**Lemma 9** (Boundaries cost) In expectation over the random dissection and the random type assignment, \( \sum_{b \in B_\ell} (\text{boundary of } b) \leq O(\epsilon/\log L) \text{F(OPT}^S). \)

**Lemma 10** (Outside cost) In expectation over the random shifted dissection, \( E[2\text{MST}(G)] \leq O(\epsilon/\log L) \text{OPT}^S \).

### 6.2.1 Proof of Lemma 8

For a box \( b \in B_\ell \) the type assignment procedure, the Type Assignment algorithm, chooses \( R_b \) by selecting an interval from each segment of \( b \) and labeling the points in the interval as type \( \ell \).

We break the quantity \( \sum_{b \in B_\ell} (\text{TSP of } R_b \text{ connected to boundary of } b) \) into the sum of two parts and bound each part separately. The first part is the sum of the lengths of intervals over all \( b \in B_\ell \). Let the **connection cost** of \( b \) refer to the cost of connecting the intervals of \( b \) to each other and to the boundary of \( b \). The second part is the sum of the connection costs for all boxes \( b \in B_\ell \). See Fig. 5.

By Property 2 the cost of the first part is \( O(\epsilon/\log L) \text{OPT}^S \) and by Property 3 the cost of the second part is \( O(\epsilon/\log L) \text{F(OPT}^S) \). The proof of the Lemma 8 follows by summing these properties and as \( \text{F(OPT}^S) \geq \text{OPT}^s \).

**Property 2** In expectation over the random type assignment the sum of the lengths of type \( \ell \) intervals over all boxes \( B_\ell \) is \( O(\epsilon/\log L) \text{OPT}^S \).

**Proof** Each type \( \ell \) interval lies on an segment \( S \) which is rounded at level \( \ell \). We prove below that the interval of \( S \) chosen during rounding has expected length \( O(\epsilon/\log L)\text{length}(S) \). Thus by linearity of expectation, the sum of all type \( \ell \) intervals
Let box $b \in B_\ell$ with rounded segments. The white points have type $\ell$. By Property 2 the total length of the white intervals (boxed) is small. By Property 3 the cost of connecting the white intervals to boundary of $b$ is small (dashed lines). The representatives $R'$ of Property 3 are shown as the dotted points.

over all boxes $B_\ell$ is at most $O(\epsilon/\log L)$ times the total length of all segments in $B_\ell$. The proof of the claim follows as the total length all segments in $B_\ell$ is at most $OPT^S$.

Let $S$ be a segment rounded by the Type Assignment algorithm. We now prove that the interval of $S$ chosen during rounding has expected length $O(\epsilon/\log L) length(S)$. Let $x$ be the number of active points on segment $S$. List the active points in the order they appear on segment $S$: $p_1, p_2, \ldots, p_x$. Thus $p_1$ is the first active visited by $S$ after crossing the box boundary and $p_x$ is the last active point visited by $S$ before crossing the box boundary. For $i < x$, let $l_i$ denote the length of segment $S$ between $p_i$ and $p_{i+1}$ and let $l_x$ be the length of $S$ from $p_x$ to boundary plus the length of $S$ from the boundary to $p_1$. Thus we have that,

$$\text{length}(S) = \sum_{i \leq x} l_i$$

Let threshold $t_i$ be such that $t_i \leq x < t_{i+1}$. The Type Assignment algorithm chooses the interval by uniformly selecting a starting active point and then selecting the next $y = x - t_i$ consecutive active points, wrapping around if necessary. Thus if point $i$ is in the interval, it contributes at most $l_i$ to the length of the interval. Using the reasoning in the proof of Lemma 6, $i$ is in the interval with probability $\epsilon/\log L$. Thus we have,

$$E[\text{length}(S)] \leq \sum_{i \leq x} Pr[i \text{ is in the interval}] \cdot l_i \leq \sum_{i \leq x} \frac{\epsilon}{\log L} l_i = \frac{\epsilon}{\log L} \text{length}(S)$$

Property 3 The connection cost for level $\ell$ is $O(\epsilon/\log L) F(OPT^S)$.

Proof The connection cost at level $\ell$ is the sum over all boxes $b \in B_\ell$ of the cost of connecting the intervals of $b$ to each other and to the boundary of $b$. Let $OPT^S_b$ denote the cost of $OPT^S$ inside $b$. Below we show that for $b \in B_\ell$,

$$\text{Connection cost of } b \leq O(\epsilon/\log L) F(OPT^S_b) \quad (5)$$

Summing Eq. 5 over all $b \in B_\ell$ proves the Property.
The connection cost of box \( b \in B_\ell \) is the cost of connecting the type \( \ell \) rounded intervals inside \( b \) together with the boundary of \( b \). Let \( R' \) be a set containing one representative from each type \( \ell \) interval in \( b \) and let \( d_\ell \) denote the side length of \( b \). Since the intervals of \( b \) can be connected using a TSP of \( R' \) and as the boundary of \( b \) is within distance \( d_\ell / 2 \) from any point in \( b \), the connection cost of \( b \) is at most, \[
\text{Connection cost of } b \leq TSP(R') + d_\ell / 2 \quad (6)
\]

We bound TSP\((R')\) using the following known Theorem \([6,14]\).

**Theorem 7** \([6,14]\) If \( U \) is a finite set of points lying in a 2-dimensional box of side length \( \delta \), then \( TSP(U) = O(\delta \sqrt{|U|}) \)

In our context, \( \delta = d_\ell \), and \( U = R' \). Thus by Theorem 7 we have

\[
\text{Connection cost of } b \leq TSP(R') + d_\ell = O(d_\ell \sqrt{|R'|})
\]

As \( b \) is a rounded box it has at least \( \gamma \) segments. We can partition the segments of \( b \) into \( g_b \geq 1 \) groups, where each group has exactly \( \gamma \) segments, and at most one additional group with less than \( \gamma \) segments. As \( R' \) contains at most two representatives from each segment in \( b \) (two representatives are needed for wrapped intervals), \( R' \) has size at most \( 2\gamma(g_b + 1) \leq 4\gamma g_b \). Thus we have that

\[
\text{Connection cost of } b \leq O(d_\ell \sqrt{4\gamma \cdot g_b}) \quad (7)
\]

Now we show that the connection cost of \( b \) computed above can be charged to the crossings we have paid for with objective function \( F \). Each tour segment of \( b \) contributes exactly two crossings at a level \( \ell \) box (once for entering \( b \) and once for leaving \( b \)) and \( b \) contains at least \( \gamma g_b \) segments. Thus we have that

\[
F(OPT^S_b) \geq 2 \frac{\epsilon}{\log L} \cdot d_\ell \cdot \gamma \cdot g_b \quad (8)
\]

Substituting into Eq. 8 for the connection cost of \( b \) derived in Eq. 7 we have,

\[
F(OPT^S_b) \geq 2 \frac{\epsilon}{\log L} \cdot \sqrt{\gamma} g_b \cdot (\text{Connection cost of } b) \quad (9)
\]

\[
\geq O \left( \frac{\log L}{\epsilon} \right) \cdot \sqrt{g_b} \cdot (\text{Connection cost of } b) \quad (10)
\]

where Eq. 10 follows as \( \sqrt{\gamma} = \log^2 L / \epsilon^2 \). As \( g_b \geq 1 \) Eq. 5 is proved. \( \square \)

### 6.2.2 Proof of Lemma 9

The proof follows the proof of Property 3. Let \( |B_\ell| \) denote the number of level \( \ell \) boxes containing type \( \ell \) points. The sum of the boundaries of these boxes is
\[ \sum_{b \in B_\ell} (\text{boundary of } b) = 4d_\ell |B_\ell| \quad (11) \]

As box \( b \in B_\ell \) is rounded it contains at least \( \gamma \) segments, and each of these segments cross the boundary of of \( b \) exactly twice. As objective function \( F \) charges each crossing \((\epsilon / \log L)d_\ell \) we have,

\[ F(\text{OPT}^S) \geq 2 \frac{\epsilon}{\log L} d_\ell \gamma |B_\ell| \quad (12) \]

\[ \geq 2 \frac{\epsilon}{\log L} d_\ell \gamma \sum_{b \in B_\ell} (\text{boundary of } b) \quad (13) \]

\[ \geq O \left( \frac{\log^3 L}{\epsilon^3} \right) \sum_{b \in B_\ell} (\text{boundary of } b) \quad (14) \]

Equation 13 follows by combining Eqs. 12 and 11. Equation 14 follows as \( \gamma = \log^4 L / \epsilon^4 \) and implies that \( \sum_{b \in B_\ell} (\text{boundary of } b) = o(\epsilon / \log L) F(\text{OPT}^S) \).

6.2.3 Proof of Lemma 10

As each \( b \in B_\ell \) contains at least \( \gamma \) segments, \( \text{OPT}^S \) has at least \( \gamma \) tour segments crossing into \( b \). Since each tour in \( \text{OPT}^S \) is structured and in particular light each tour crosses \( b \) at most \( 4r \) times. Thus at least \( \gamma/4r \) tour enter \( b \).

Let \( P \) be a multigraph that denotes the projection of \( \text{OPT}^S \) onto graph \( G \) where every edge in \( \text{OPT}^S \) between two boxes \( b, b' \in B_\ell \) or between the depot and a box \( b \in B_\ell \) is represented by an edge in \( P \). Let the edges of \( P \) have cost equal to the corresponding edges in \( G \). As the cost of edges in \( G \) are less than or equal to distances in \( \text{OPT}^S \) clearly \( \text{cost}(P) \leq \text{OPT}^S \).

As there are has at least \( \gamma/4r \) tours in \( \text{OPT}^S \) crossing each \( b \in B_\ell \), \( P \) has at least \( \gamma/4r \) edges crossing any cut separating the depot vertex and any box vertex. See Fig. 6. Consider the linear program in Eq. 15 on \( G \) with \( V \) denoting the vertices of \( G \) and \( w_e \) the cost of edge \( e \) in \( G \). The linear program finds \( w \) the minimum cost way to fractionally select a set of edges in \( G \) such that each non-trivial subset of \( V \) is crossed at most \( \gamma/4r \) times. Thus the \( \text{cost}(P) \geq w \).

\[ w = \min \sum_{\text{edge } e} w_e x_e \quad \text{s.t.} \quad \left\{ \begin{array}{l} \sum_{e \in \delta(S)} x_e \geq \gamma/4r \ \forall S \subset V, S \neq \emptyset \\ x_e \geq 0 \end{array} \right. \quad (15) \]

Consider linear program in Eq. 16, which is the relaxation of a MST IP on \( G \).

\[ w' = \min \sum_{\text{edge } e} w_e x_e \quad \text{s.t.} \quad \left\{ \begin{array}{l} \sum_{e \in \delta(S)} x_e \geq 1 \ \forall S \subset V, S \neq \emptyset \\ x_e \geq 0 \end{array} \right. \quad (16) \]

Observe that for any solution \( w \) of the linear program in Eq. 15, \( w \cdot 4r/\gamma \) is a solution for linear program 16. If \( w \) is the minimum solution of linear program 15,
Given that $\text{OPT}$ has at least 3 tours entering each box $\text{OPT}$ crosses all non-trivial cuts at least 6 times. This is made explicit in Eq. 15. b The MST crosses all non-trivial cuts at least once as expressed in Eq. 16.

then $w' = w \cdot (4r)/\gamma$ is the minimum solution of linear program 16. The MST relaxation of Eq. 16 is known to have integrality gap at most 2 i.e $w' \geq \frac{1}{2} \cdot \text{MST}(G)$ [23]. Thus we have that

$$\text{OPT}^S \geq \text{cost}(P) \geq w = w' \cdot \frac{\gamma}{4r} \geq \text{MST}(G) \cdot \frac{\gamma}{8r}$$

Thus $(8r/\gamma) \cdot \text{OPT}^S \geq \text{MST}(G)$. As $8r/\gamma = o(\epsilon/\log L)$, the lemma is proved.

7 Extension to Multiple Depots

This section presents our approximation scheme for CVRP with multiple depots and the proof of Theorem 2. With multiple depots, the solution is no longer necessarily connected thus the distance between the two farthest customers, and the TSP are not lower bounds. Our algorithm and analysis must be updated to no longer rely on these bounds.

7.1 Algorithm and Proof of Main Theorem

The approximation scheme is summarized in the Multi-Depot Scheme.

**Overview of our algorithm.** As the solution may be disconnected, first we partition the instance into sub instances such that it suffices to solve each sub instance independently. Each sub instance may still contain multiple depots but can be solved similarly to the single depot setting. In each sub instance we perform a random dissection and place portals on the dissection lines. As before, we use rounding to remember the approximate number of points covered by each tour segment. Then a quasipolynomial time dynamic program is used to find a structured solution, that is a solution consisting
Fig. 7 A solution computed by the Multi-Depot Scheme for the multiple depot problem with 31 customers (circles), 3 depots (stars) and \( k = 7 \). The solid circles are the black points and the empty circles are the red points. The solid tours are computed by the DP in step 4. Each covers \( \leq k \) black points. The dotted tour covers the red points and is computed in step 7 using the 6-approximation of portal respecting and light tours each covering approximately \( k \) customers. With multiple depots, the dynamic program’s tour configurations describes whether a tour segment visits some depot rather than the depot. But this can be remembered using one bit so the running time does not change.

To get a feasible solution with respect to the vehicle capacity we use the previous randomized procedure to drop points from tours covering more than \( k \) points. A different constant factor approximation, this time the 6-approximation of [18], is used to obtain a solution for the dropped points. The output of the constant approximation can be bounded in terms of the TSP of a slightly different instance (the virtual instance) as well as the Rad of customers. The crux of the analysis is to show that the constant factor approximation on the red points still has negligible cost compared to OPT. Figure 7 shows a solution computed by the Multi-Depot Scheme.

Algorithm Multi-Depot Scheme

Input: Customers \( C \) and depots \( D \) in \( \mathbb{R}^2 \), and integer \( k \)

1: Partition instance into sub instances as described in Section 7.1.1.
2: for each sub instance do
3: Perturb the sub instance, perform random dissection and place portals as described in Section 7.1.2
4: Compute a structured solution (defined in Section 3.2) using the dynamic program from Section 7.2.
5: Use the DP’s history to construct the structured tours and assign types to points using the randomized type assignment procedure from Subsection 3.3.
6: Color points black or red as in the single depot case and drop all red points from the structured tours.
7: Use the 6-approximation the Multi-Depot Tour Partitioning algorithm to get solution for red points.
8: The solution of a sub instance are the red tours on the red points and black tours on the black points
9: end for

Output: Union of the solutions of all sub instances.

7.1.1 Partitioning into Sub Instances

In the single depot setting OPT is at least the distance between the two farthest points. This allowed us to relate OPT to the side length of the bounding box and charge the cost of tour detours and the additional red tours to the length of the bounding box.
This method does not work in the multiple depot setting where the side length of the bounding box could have length much greater than OPT. To overcome this difficulty we use the Create Sub Instances algorithm to partition the instance into sub instances such that the bounding box of each sub instance is proportional to OPT. Lemma 11 shows that it suffices to solve each sub instance independently and combine the solutions to get a solution for the whole instance. A similar technique was used by Borradaile et al. [7] for the Euclidean Steiner Forest problem, where the solution is also not necessarily connected.

**Algorithm Create Sub Instances**

Input: Customers $C$, depots $D$, and integer $k$

1: Let $A$ be the cost of the solution returned by the 6-approximation Multi-Depot Tour Partitioning algorithm

2: Define a graph with edges between customers $c$ and $c'$ if and only if they are within distance $A$

3: Let $Q_1, Q_2, \ldots, Q_x$ be the connected components of the resulting graph.

4: for each component $Q_i$ and each customer $c$ in component $Q_i$ do

5: Include the set of depots within distance $A$ to customer $c$.

6: end for

Output: The resulting components $Q_1, \ldots, Q_x$ as the sub instances.

**Lemma 11** Let $Q_1, Q_2, \ldots, Q_x$ be the sub instances returned by the Create Sub Instances algorithm, and $n_i$ denote the number of customers in $Q_i$. Let $L_i$ be the maximum distance between any two points in $Q_i$. We have that

1. $\sum_i OPT(Q_i) = OPT$
2. $L_i \leq (n_i + 1)6OPT$

**Proof** Consider the first property. For a contradiction suppose that customers $c \in Q$ and $c' \in Q'$ and $Q \neq Q'$, are covered by the same tour in OPT. Since $c$ and $c'$ ended up in different sub instances in algorithm Create Sub Instances, they are also in different components in Line Create Sub Instances. Thus the distance between $c$ and $c'$ must be greater than $A$. Since $A \geq OPT$, this is a contradiction. A similar argument shows that customer $c$ in $Q$ cannot be covered by a depot which is not in $Q$. This implies that each tour of OPT are contained in at most one $Q_i$, and thus each $Q_i$ can be solved independently to find OPT.

Now consider the second property. Let $A \leq 6OPT$ be the cost of the 6-approximation computed in Line Create Sub Instances of the Create Sub Instances algorithm. Fix any $Q_i$ and let points $p, p'$ be the points which are farthest apart in $Q_i$. If $p, p'$ are both customers there is an Euclidean path from $p, p'$ such that $dist(p, p') \leq (n_i - 1)A$. If both $p, p'$ are depots, then $dist(p, p') \leq (n_i + 1)A$. If one point is a depot and the other a customer then $dist(p, p') \leq n_i A$. □

### 7.1.2 Preprocessing

The sub instances $Q_1, \ldots, Q_y$ are preprocessed independently using the same technique as in Sect. 3.1.
**Perturbation.** For any \( Q_i \), define its bounding box as the smallest box whose side length \( L_i \) is a power of 2 that contains all points in \( Q_i \). Let \( n_i \) be the number of customers in \( Q_i \). Let \( A \) denote the cost of the 6-approximation computed in Line Create Sub Instances of the Create Sub Instances algorithm. Thus \( A \leq 6OPT \). Place a grid of granularity \( \delta_i = A\epsilon / (2 \cdot n_i \cdot 6) \) inside the bounding box and move each point to the center of its grid box. Scale distances in \( Q_i \) by \( 4 / \delta_i \) so that all coordinates become integral and the minimum distance between any two grid centers that contain points is least 4.

A solution for the perturbed \( Q_i \) can be extended into a solution for \( Q_i \) by taking detours from the grid centers to the original locations of the points. At most \( 2n_i \) detours will be required for \( Q_i \). The cost of each detour (before scaling) is \( 2\delta_i \), thus the cost of all the detours for \( Q_i \) (before scaling) is at most \( 2n_i \cdot \delta_i \). The cost of detours over all \( Q_1, \ldots, Q_x \) is at most \( \sum_i 2n_i \delta_i \leq A\epsilon / 6 \) which is \( \leq \epsilon OPT \) as \( A \leq 6OPT \) by Theorem 10. Scaling does not change the structure of the optimal solution and we can always rescale to get the cost of the original instance. As the total cost of the perturbation is within the required \( \epsilon \) error parameter, we can work on the perturbed \( Q_i \)'s.

**Randomized Dissection and Portals.** The randomized dissection and placement of portals is done independently for each \( Q_i \). We use the same process from Sect. 3.1 using \( L_i \) in place of \( L \), and \( n_i \) in place of \( n \). Note that by the second part of Lemma 11, after scaling the maximum distance between points and hence the side length of the bounding box is \( L_i = O(n_i^2 / \epsilon) \).

**7.1.3 Extending the Structure Corollary**

Observe that definitions from Sect. 3.2 can be extended to the multiple depot setting and sub instances by replacing the word depot with depots.

Additionally the Structure Theorem 4 also holds for multiple depots - its proof analyzes each tour separately and did not rely on the number of depots. Thus we have the following Theorem whose proof is identical to the proof of Theorem 4.

**Theorem 8** (Structure Corollary) Let \( I \) be an instance (or a sub instance) of CVRP with multiple depots. Let OPT denote the length of the optimal solution of \( I \), and let OPT\( ^S \) denote the length of the structured solution of \( I \) that minimizes objective function \( F \). In expectation over random shifts of the dissection, \( OPT^S \leq (1 + O(\epsilon))OPT \).

By Theorem 8 we can focus on computing the structured solution that minimizes objective \( F \), which we will denote as OPT\( ^S \). In Sect. 7.2 we show that essentially the same dynamic program can be used in the multiple depot setting which proves the following:

**Theorem 9** (Dynamic Program) Given an instance \( I \) of CVRP with a multiple depots the dynamic program of Sect. 7.2 computes OPT\( ^S (I) \) in time \( n^{\log O(1/\epsilon)} n \).

**7.1.4 A Constant Factor Approximation for Multiple Depots**

For any two points \( i, j \in I \) let \( dist(i, j) \) denote the shortest distance between \( i, j \). For each customer \( i \in C \) recall that \( r_i \) denotes distance to the closest depot from \( i \) and
let $D(i)$ denote this depot. The constant factor approximation of Li and Simchi-Levi, which is given in the Multi-Depot Tour Partitioning algorithm, works on the virtual instance, $\tilde{I}$.

**Definition 7 (Virtual Instance)** For any instance $I$ with customers $C$ and depots $D$ the virtual instance $\tilde{I}$ contains all customers in $C$ and a single virtual depot $v$. Distances in $\tilde{I}$ are defined as follows: for each customer $i \in C$ the distance from $i$ to virtual depot $v$ is $r_i$ and for any two customers $i, j \in C$ let their distance be $\min\{r_i + r_j, \text{dist}(i, j)\}$, that is the minimum of going directly from $i$ to $j$ or going through $v$. If the distance between customers $i, j$ in $\tilde{I}$ is less then their distance in $I$, the edge between $i, j$ in $\tilde{I}$ is called a virtual edge.

We use $\text{OPT}(\tilde{I})$ to denote the optimal solution of the virtual instance. Note that $\text{OPT}(\tilde{I}) \leq \text{OPT}$ as distances in $\tilde{I}$ are at most the distances in $I$. Also note that distances in $\tilde{I}$ satisfy the triangle inequality.

The Multi-Depot Tour Partitioning algorithm works as follows. Since $\tilde{I}$ is a single depot instance first the 3-approximation Single-Depot Tour Partitioning algorithm is used to get a solution for $\tilde{I}$ and then the solution is converted into a solution for $I$. Converting the solution of $\tilde{I}$ directly to $I$ result in paths rather than tours such that each path starts and ends at different depots. These paths are converted into tours by re-visiting one of two depots. By the triangle inequality revisiting increases the cost of the solution by at most a factor of two. Theorem 10 shows that the Multi-Depot Tour Partitioning algorithm is a 6-approximation. We include a summary of the proof given by Li and Simchi-Levi for completeness.

**Algorithm Multi-Depot Tour Partitioning** [18]

Input: Customers $C$ and depots $D$, and integer $k$

1: Define the virtual instance $\tilde{I}$ and let $\tilde{\Pi}$ be the output of the Single-Depot Tour Partitioning algorithm on $\tilde{I}$.

2: Define $P$ as an empty set which will be populated below with paths that go between two depots

3: for each tour $\tilde{\pi}$ in $\tilde{\Pi}$ do

4: Replace each virtual edge $e = (i, j)$ in $\tilde{\pi}$ between customers $i, j$ with edges $e_1 = (i, D(i))$, $e_2 = (j, D(j))$ where $D(i)$ and $D(j)$ are the depots closest to $i, j$. Add the paths created to set $P$.

5: end for

6: for each path $p \in P$ do

7: Let $i, j$ be the first and last customers in $p$ and $D(i), D(j)$ be their closest depots.

8: if $r_i + \text{dist}(j, D(i)) \leq r_j + \text{dist}(i, D(j))$ then

9: Turn $p$ into a tour $\pi$ that starts and ends at depot $D(i)$.

10: else

11: Turn $p$ into a tour $\pi$ that starts and ends at depot $D(j)$.

12: end if

13: end for

Output the resulting tours.

**Theorem 10** [18] Given an $I$ instance of the metric CVRP problem with multiple depots, the Multi-Depot Tour Partitioning algorithm outputs a solution of expected length at most

$$\text{Rad}(\tilde{I}) + 2 \cdot \text{TSP}(\tilde{I}) \leq 3\text{OPT}(\tilde{I}) \leq 6\text{OPT}.$$
Proof By Theorem 6 the Single-Depot Tour Partitioning algorithm returns a solution of \( \tilde{I} \) of expected length \( \text{Rad}(\tilde{I}) + 2\text{TSP}(\tilde{I}) \leq 3\text{OPT}(\tilde{I}) \). Let \( P \) be the set of paths obtained by replacing all virtual edges in the solution by real edges. The total cost of \( P \) is the same as the cost of the virtual solution since each virtual edge \( e = (i, j) \) of cost \( r_i + r_j \) is replaced by two real edges of cost \( r_i \) and \( r_j \). Thus the length of the paths in \( P \) is at most \( 3\text{OPT}(\tilde{I}) \). By the triangle inequality, converting the paths in \( P \) into tours doubles the cost of \( P \). Thus the length of the output is at most \( 2\text{length}(P) \leq 6\text{OPT}(\tilde{I}) \leq 6\text{OPT} \) as \( \text{OPT}(\tilde{I}) \leq \text{OPT} \). \( \square \)

7.2 Extending the Dynamic Program

A slight modification of the definition of configurations allows the dynamic program of the single depot setting to handle multiple depots. Recall that in the single depot DP a rounded configuration represents segments by numbers \( r_{p,q,t,d} \) and an unrounded configuration represent them by tuples \((p, q, u, d)\). In both cases \( p, q \) are portals, \( t \) and \( u \) represent the number of points on the segment, and \( d \) indicates whether the segment visited the depot. With the multiple depots the modification to the DP is to update the meaning of indicator \( d \). Now \( d = 1 \) will represent that the segment visited a depot, rather than the depot. Thus for each tour segment we now remember whether the segment visits some depot or not.

A configuration profile \( \Phi = ((p_1, p_2, x_1, d_1), (p_2, p_3, x_2, d_2), \ldots, (p_s, p_{s+1}, x_s, d_s)) \) with \( p_1 = p_{s+1} \) represents segments which form a tour. In the single depot setting such a \( \Phi \) is feasible if at least one segment had \( d_i = 1 \) indicating that it has visited the depot. In the multiple depot setting the same condition implies that at least some depot was visited, and so the profile is still feasible. If more than one depot was visited by segments \( \Phi \) we can always shortcut around all but one of the depots in the final solution.

7.3 Proof of Main Theorem 2

Fix any sub instance \( Q_i \). We first show that the Multi-Depot Scheme computes a solution of cost \( (1 + O(\epsilon))\text{OPT}(Q_i) \) following the proof of Theorem 1. The solution of the Multi-Depot Scheme for \( Q_i \) has cost equal to the length of the black tours plus the length of red tours. The black tours have length at most the length of the tours computed by the DP. By Theorem 9 and Theorem 8 the DP’s output has length at most \( (1 + O(\epsilon))\text{OPT}(Q_i) \). Theorem 11 whose proof is given in Sect. 7.4, shows that the red tours of \( Q_i \) have length at most \( O(\epsilon)\text{OPT}(Q - i) \). Thus the Multi-Depot Scheme computes a solution of cost \( (1 + O(\epsilon))\text{OPT}(Q_i) \) for each \( Q_i \).

**Theorem 11** In expectation over the random shifts of the dissection and the random type assignment, the length of the red tours output by the Multi-Depot Scheme is \( O(\epsilon)\text{OPT} \).

The argument above applies for any sub instance \( Q_i \). By the first property of Lemma 11 the union the solutions of \( Q_i \) yields a solution of cost \( (1 + O(\epsilon))\text{OPT} \) for the whole instance.
Running the dynamic program for each $Q_i$ dominates the running time. Let $n_i$ denote the number of customers in $Q_i$. The run time is at most,

$$
\sum_{i}^{y} n_i^{\log^{O(1/\epsilon)} n_i} \leq n^{\log^{O(1/\epsilon)} n}
$$

The derandomization is done by derandomizing the procedure for each sub instance individually as discussed in Sect. 8.

7.4 Proof of Theorem 11

Let $I$ be an instance or sub instance with depots $D$ and customers $C$. Let $R$ be the customers that are marked red by the Multi-Depot Scheme and $\tilde{R}$ be the virtual instance defined on customers $R$ and depots $D$. By Theorem 10 the 6-approximation on $R$ returns a solution of cost $\text{Rad}(\tilde{R}) + 2 \cdot \text{TSP}(\tilde{R})$. Lemmas 12 and 13 show that both quantities are $O(\epsilon)$ $\text{OPT}(I)$ in expectation, which yields the proof of Theorem 11.

**Lemma 12** In expectation over the random type assignment, $\text{Rad}(\tilde{R}) = O(\epsilon) \text{OPT}(I)$

**Proof** By Lemma 6 $\text{Rad}(\tilde{R}) = O(\epsilon) \text{Rad}(\tilde{I})$. By definition $\text{Rad}(\tilde{I}) = \text{Rad}(I)$ since $\text{Rad}$ sums the distance of each customer to its closest depot. The Lemma follows as $\text{Rad}(I) \leq \text{OPT}(I)$ by Lemma 1. $\square$

**Lemma 13** In expectation over the random dissection and type assignment $\text{TSP}(\tilde{R}) = O(\epsilon) \text{OPT}(I)$

**Proof** For each level $\ell \in [0, \ell_{\text{max}}]$ let $R_\ell$ denote the type $\ell$ points of $I$ and note that $R = \bigcup_{\ell > 0} R_\ell$. Let $\tilde{R}_\ell$ denote the virtual instance of the customers in $R_\ell$ and all depots in $D$. We will prove that for any level $\ell$, $E[\text{TSP}(\tilde{R}_\ell)] \leq O(\epsilon / \log L) \text{OPT}$. This implies the lemma as the tours of $\tilde{R}_\ell$ from all $O(\log L)$ levels can be pasted together at virtual depot $v$ to yield a tour of $\tilde{R}$.

We follow the proof of Lemma 7 and upper bound the cost of $\text{TSP}(\tilde{R}_\ell)$ by thinking of the tour of $\tilde{R}_\ell$ in three parts: inside, boundary and outside. The inside and boundary parts are defined just as in the proof Lemma 7, using actual distances rather than distances in the virtual instance. This gives an upper bound on the cost of the virtual TSP as the virtual distances are less than or equal to the actual distances. Given the inside and boundary parts to get a tour of $\tilde{R}_\ell$ we only need a tour that connects boxes $B_\ell$ and the virtual depot. We refer to this as the outside part. We use virtual distances for the outside part.

Define a complete graph $\tilde{G} = (V, E)$, which corresponds to $G$ from the proof of Lemma 7, but $\tilde{G}$ uses virtual distances. $\tilde{G}$ contains a vertex for the virtual depot and a vertex for each box in $B_\ell$. The edge costs in $\tilde{G}$ are defined as follows: the cost of the edge between two box vertices $b, b'$ is equal to the length of the shortest path in $\tilde{I}$ from any portal of $b$ to any portal of $b'$, the cost of an edge between a box vertex $b$ and the virtual depot vertex is equal to the length of the shortest path in $\tilde{I}$ from any
portal of $b$ to the virtual depot. The *outside* cost; i.e. the cost of a tour of the virtual depot and boxes $B_{\ell}$, is at most $2 \text{MST} (\tilde{G})$. Thus we have that:

$$TSP(\tilde{R}_\ell \cup v) = \sum_{b \in B_{\ell}} (\text{TSP of } R_b \text{ connected to boundary of } b)$$

$$+ \sum_{b \in B_{\ell}} (\text{boundary of } b) + 2 \text{MST}(\tilde{G})$$

(17)

The first two terms (i.e the *inside* and *boundary* costs) are defined exactly as in Lemma 7 using the actual distances. Thus they are bounded by Lemmas 8 and 9 to be at most $O(\epsilon / \log L) F(\text{OPT}^S)$. As $F(\text{OPT}^S) \leq (1 + O(\epsilon))\text{OPT}$ by the Structure Theorem 8 the cost of the inside and the boundary is at most $O(\epsilon / \log L)\text{OPT}$.

Lemma 14 proves that $\text{MST}(\tilde{G}) \leq O(\epsilon / \log L)\text{OPT}^S$. By definition of $F$ that is at most $O(\epsilon / \log L)F(\text{OPT}^S)$, which by the Structure Theorem 8 is at most $O(\epsilon / \log L)\text{OPT}$. Thus overall $\text{TSP}(\tilde{R}_\ell \cup v) \leq O(\epsilon / \log L)\text{OPT}$. $\square$

**Lemma 14** *(Outside cost)* In expectation over the random shifted dissection, $E[2 \text{MST}(\tilde{G})] \leq O(\epsilon / \log L)\text{OPT}$.

*Proof* The proof follows by replacing $G$ with $\tilde{G}$ in the proof of Lemma 10. $\square$

## 8 Derandomization

To derandomize our algorithms we need to derandomize the dissection and the Type Assignment algorithm.

The dissection can be derandomized by trying all choices of shifts $a$ and $b$ as was done in [2]. More efficient derandomizations are given by Czumaj and Lingas and by Rao and Smith [9,20].

The Type Assignment algorithm uses randomization to ensure that the cost of the dropped points is small in expectation. The proofs of Lemma 6 and Property 2 imply that the cost of dropped points will be small as long as whenever an interval $Y$ is selected to be dropped from a segment $S$ the following two conditions hold:

1. $\text{Rad}(Y) \leq O(\epsilon / \log L)\text{Rad}(S)$
2. $\text{length}(Y) \leq O(\epsilon / \log L)\text{length}(S)$

The Type Assignment algorithm selects $Y$ by selecting a point uniformly from $S$ and then selecting the next $|Y| - 1$ consecutive points. To derandomize this process we can test the at most $|S|$ intervals of length $|Y|$ in $S$, (each starting from a different point in $S$), and select any interval that satisfies both conditions (1) and (2) above.

## 9 Extension to Higher Dimensions

The approximation schemes extend to dimension $d \geq 2$ while $d$ is a constant independent of the number of points in $C$ and $D$. Our algorithms have two main components: The first is a generalization of Arora’s TSP algorithm that finds the *black* tours and the second is the rounding scheme to handle capacity constraints. Arora’s TSP algorithm extends to $d \geq 2$ dimension for constant $d$, and our generalization of the method can
be extended in the same way. Our rounding scheme has no geometric dependency and does not change in higher dimensions. In the analysis of the cost of the red tours we did use a geometric property that upper bounds the TSP of a set of points in a 2-dimensional box (i.e Theorem 7). However the Theorem 7 generalizes to \( \mathcal{R}^d \) as given below. In fact it was also used in the generalization of Arora algorithm, namely for extending his patching lemma to \( \mathcal{R}^d \).

**Theorem 12** [16] Let \( U \) be a finite set of points in the \( d \)-dimensional cube with side length \( L \). There exists a constant \( c_d \) such that \( \text{tsp}(S) \leq c_d \cdot L \cdot |U|^{d-1/d} \).

As in Arora’s TSP algorithm in \( d \) dimensions, each dissection box is now a \( d \) dimensional cubes with \( 2d \) boundaries (facets) and each box boundary is a \( d - 1 \) dimensional cube. The boundary of a box contains \( m = O(\sqrt{d} \log L / \epsilon)^{d-1} \) portals and each tour is \( r = O((\sqrt{d} / \epsilon)^{d-1}) \) light. The dissection tree will now contain \( O(2^d n) \) non empty boxes. The running time of our algorithms will be \( O(2^d n \cdot n^{O(2d^r)}) = n^{(\log L)^{O(d^r)}} \), since for every box in the dissection the DP will now guess the number of tours of each type that are present in the box, and there are \( m^{O(2d^r)} \) tour types. Thus for the running time of our approximation schemes to remain quasipolynomial we would need \( (d)^{(d-1)} \) to be a constant.

## 10 Conclusion

We presented the first quasipolynomial time approximation scheme for CVRP with single or multiple depots that works for all values of \( k \). Note that the black tours computed by our algorithms gives a solution to the version of the problem with soft capacity constraints, where \( \text{OPT} \) is required to use tours of capacity \( k \) but the algorithm is allowed tours of capacity \( k(1 + \epsilon) \). Additionally it is not hard to see that our dynamic program can be modified to handle a bound \( f \) on the maximum number of vehicles or tours. Thus if \( L \) is the value of the optimal solution that uses at most \( f \) tours each covering at most \( k \) points, then our approach can be extended to output at most \( f \) tours each covering at most \( k(1 + \epsilon) \) points, with total length at most \( L(1 + \epsilon) \).

Our algorithms have quasipolynomial and seriously super-polynomial running time, so they are unlikely to lead to much in the way of practical improvements. However our result make progress toward and provides strong evidence for the long standing conjecture that Euclidean CVRP has a PTAS for all \( k \). The major open question still remaining is to design a PTAS. As noted, PTASs already exist when \( k \) is either large or small compared to number of customers \( n \) [1, 1, 2, 6, 14]. Several authors including [1, 6] have also noted that \( k = \Theta(\sqrt{n}) \) seems to be the difficult case to getting a PTAS for all \( k \).

**Acknowledgments** A preliminary version of this paper appeared in [11] Both authors supported by NSF Grant CCF-0728816.

**References**

1. Adamszek, A., Czumaj, A., Lingas, A.: PTAS for k-tour cover problem on the plane for moderately large values of \( k \). Algorithms and Computation. Springer-Verlag, Berlin (2009)
2. Arora, S.: Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. J. ACM 45(5), 753–782 (1998)
3. Arora, S.: Approximation schemes for NP-hard geometric optimization problems: A survey. Math. Program. 97(1–2), 43–69 (2003)
4. Arora, S., Karakostas, G.: Approximation schemes for minimum latency problems. SIAM J. Comput. 32(5), 1317–1337 (2003)
5. Asano, T., Katoh, N., Tamaki, H., Tokuyama, T.: Covering points in the plane by k-tours: a polynomial approximation scheme for fixed k. Research Report RT0162, IBM Tokyo Research Laboratory (1996)
6. Asano, T., Katoh, N., Tamaki, H., Tokuyama, T.: Covering points in the plane by k-tours: towards a polynomial time approximation scheme for general k. In: STOC ’97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pp. 275–283, ACM, New York (1997)
7. Borradale, G., Klein, P.N., Mathieu, C.: A polynomial-time approximation scheme for Euclidean steiner forest. In: FOCS ’08: Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 115–124, IEEE Computer Society, Washington DC (2008)
8. Cardon, S., Dommers, S., Eksin, C., Sitters, R., Stougie, A., Stougie, L.: A PTAS for the multiple depot vehicle routing problem. SPOR Reports, January 2008. www.win.tue.nl/~bs/spor/. Accessed 9 June 2014
9. Czumaj, A., Lingas, A.: A polynomial time approximation scheme for Euclidean minimum cost k-connectivity. In ICALP ’98: Proceedings of the 25th International Colloquium on Automata, Languages and Programming, pp. 682–694, Springer-Verlag, London (1998)
10. Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. Manag. Sci. 6(1), 80–91 (1959)
11. Das, A., Mathieu, C.: A quasi-polynomial time approximation scheme for Euclidean capacitated vehicle routing. In: SODA ’10: Proceedings of the Twenty First Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia (2009)
12. Fisher, M., Fisher, M.: Chapter 1 vehicle routing. In: Monna, C.L., Ball, M.O., Magnanti, T.L., Nemhauser, G.L. (eds.) Handbooks in Operations Research and Management Science. Network routing, vol. 8, pp. 1–33. Elsevier, Oxford (1995)
13. Golden, B., Raghavan, S., Wasil, E.: The vehicle routing problem: latest advances and new challenges. Operations Research/Computer Science Interfaces Series, vol. 43. Springer, New York (2008)
14. Haimovich, M., Rimnooy Kan, A.H.G.: Bounds and heuristics for capacitated routing problems. Math. Oper. Res. 10(4), 527–542 (1985)
15. Haimovich, M., Rimnooy Kan, A.H.G., Stougie, L.: Analysis of heuristics for vehicle routing problems. In: Vehicle Routing: Methods and Studies. Management Science Systems, vol. 16, pp. 47–61, Elsevier Science B.V., Amsterdam (1988)
16. Karloff, H.: How long can a Euclidean traveling salesman tour be? SIAM J. Discr. Math. 2(1), 91–99 (1989)
17. Kollipoulos, S.G., Rao, S.: A nearly linear-time approximation scheme for the Euclidean k-median problem. SIAM J. Comput. 37(3), 757–782 (2007)
18. Li, C.L., Simchi-Levi, D.: Worst-case analysis of heuristics for multidepot capacitated vehicle routing problems. INFORMS 2(1), 64–73 (1990)
19. Mitchell, J.S.B.: Guillotine subdivisions approximate polygonal subdivisions: a simple polynomial-time approximation scheme for geometric tsp, k-mst, and related problems. SIAM J. Comput. 28(4), 1298–1309 (1999)
20. Rao, S.B., Smith, W.D.: Approximating geometrical graphs via “spanners” and “banyans”. In: STOC ’98: Proceedings of the Thirtyighth Annual ACM Symposium on Theory of Computing, pp. 540–550, ACM, New York (1998)
21. Remy, J., Steger, A.: A quasi-polynomial time approximation scheme for minimum weight triangulation. In: STOC ’06: Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing, pp. 316–325, ACM, New York (2006)
22. Toth, P., Vigo, D. (eds.): The vehicle routing problem. Society for Industrial and Applied Mathematics, Philadelphia (2001)
23. Vazirani, V.: Approximation Algorithms. Springer-Verlag, New York (2001)