Approximating the Generalized Minimum Manhattan Network Problem*

Aparna Das[1], Krzysztof Fleszar[2], Stephen Kobourov[1], Joachim Spoerhase[2], Sankar Veeramoni[1], and Alexander Wolff[2]

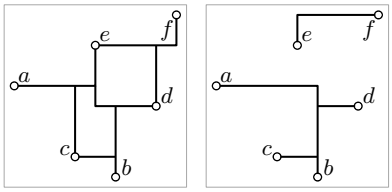
[1] Department of Computer Science, University of Arizona, Tucson, AZ, U.S.A.
[2] Lehrstuhl I, Institut für Informatik, Universität Würzburg, Germany

Abstract. We consider the *generalized minimum Manhattan network problem* (GMMN). The input to this problem is a set R of n pairs of terminals, which are points in $\mathbb{R}^2$. The goal is to find a minimum-length rectilinear network that connects every pair in R by a *Manhattan path*, that is, a path of axis-parallel line segments whose total length equals the pair's Manhattan distance. This problem is a natural generalization of the extensively studied *minimum Manhattan network problem* (MMN) in which R consists of all possible pairs of terminals. Another important special case is the well-known *rectilinear Steiner arborescence problem* (RSA). As a generalization of these problems, GMMN is NP-hard. No approximation algorithms are known for general GMMN.

We obtain an $O(\log n)$-approximation algorithm for GMMN. Our solution is based on a stabbing technique, a novel way of attacking Manhattan network problems. Some parts of our algorithm generalize to higher dimensions, yielding a simple $O(\log^{d+1} n)$-approximation for the problem in arbitrary fixed dimension d. As a corollary, we obtain an exponential improvement upon the previously best $O(n^{\varepsilon})$-ratio for MMN in d dimensions [ESA'11]. En route, we show that an existing $O(\log n)$-approximation for 2D-RSA generalizes to higher dimensions.

1 Introduction

Given a set of terminals, which are points in $\mathbb{R}^2$, the *minimum Manhattan network problem* (MMN) asks for a minimum-length rectilinear network that connects every pair of terminals by a Manhattan path (*M-path*, for short), i.e., a path consisting of axis-parallel segments whose total length equals the pair's M-distance. Put differently, every pair is to be connected by a shortest path in the L_1-norm (M-path). See Fig. 1a for an example.



(a) an MMN for $\{a, b, c, d, e, f\}$ (b) a GMMN for $\{(a,b), (c,d), (e,f)\}$

Fig. 1: MMN versus GMMN.

In the *generalized minimum Manhattan network problem* (GMMN), we are given a set R of n unordered terminal *pairs*, and the goal

* Slides are available at http://www1.informatik.uni-wuerzburg.de/pub/wolff/slides/gmmn.pdf.
This work was supported by the ESF EuroGIGA project GraDR (DFG grant Wo 758/5-1).

is to find a minimum-length rectilinear network such that every pair in R is *M-connected*, that is, connected by an M-path. GMMN is a generalization of MMN since R may contain all possible pairs of terminals. Figure 1b depicts such a network.

We remark that, in this paper, we define n to be the number of terminal *pairs* of a GMMN instance, previous works on MMN defined n to be the number of *terminals*. Moreover, we identify each terminal pair with a rectangle, namely the bounding box of this pair. This is a natural convention as every M-path for this terminal pair lies within the bounding box.

MMN naturally arises in VLSI circuit layout [7], where a set of terminals (such as gates or transistors) needs to be interconnected by rectilinear paths (wires). Minimizing the cost of the network (which means minimizing the total wire length) is desirable in terms of energy consumption and signal interference. The additional requirement that the terminal pairs are connected by *shortest* rectilinear paths aims at decreasing the interconnection delay (see Cong et al. [4] for a discussion in the context of rectilinear Steiner arborescences, which have the same additional requirement; see definition below). Manhattan networks also arise in the area of geometric spanner networks. Specifically, a minimum Manhattan network can be thought of as the cheapest spanner under the L_1-norm for a given set of points (allowing Steiner points). Spanners, in turn, have numerous applications in network design, distributed algorithms, and approximation algorithms, see, e.g., the book [13] and the survey [8]. Spillner [18] points out an application of higher-dimensional MMN in the area of computational biology.

MMN requires a Manhattan path between every terminal pair. This assumption is, however, not always reasonable. For example, in VLSI design a wire connection is necessary only for an, often comparatively small, subset of terminal pairs, which may allow for substantially cheaper circuit layouts. In this scenario, GMMN appears to be a more realistic model than MMN.

Previous Work and Related Problems. MMN was introduced by Gudmundsson et al. [7] who gave 4- and 8-approximation algorithms for MMN running in $O(n^3)$ and $O(n \log n)$ time, respectively. The currently best known approximation algorithms for MMN have ratio 2; they were obtained independently by Chepoi et al. [7] using an LP-based method, by Nouioua [8] using a primal-dual scheme, and by Guo et al. [9] using a greedy approach. The complexity of MMN was settled only recently by Chin et al. [3]; they proved the problem NP-hard. It is not known whether MMN is APX-hard. Gudmundsson et al. [6] consider a variant of MMN where the goal is to minimize the number of (Steiner) nodes and edges. Using divide-and-conquer they show that there is always a Manhattan network with $O(n \log n)$ nodes and edges. Knauer and Spillner [10] show that MMN is fixed-parameter tractable. More specifically, they show that there is an exact algorithm for MMN taking $O^*(2^{14h})$ time, where h is the number of horizontal lines that contain all terminals and the O^*-notation neglects factors polynomial in n.

Recently, there has been an increased interest in (G)MMN for higher dimensions. Muñoz et al. [12] proved that 3D-MMN is NP-hard to approximate within a factor of 1.00002. They also gave a constant-factor approximation algorithm for a (rather restricted) special case of 3D-MMN. Das et al. [5] described the first approximation algorithm for MMN in arbitrary, fixed dimension. Their algorithm recursively computes

a grid and attaches the terminals within a grid cell to grid vertices using RSA as a subroutine. Its ratio is $O(n^\varepsilon)$ for any $\varepsilon > 0$.

GMMN was defined by Chepoi et al. [7] who posed the question whether it admits an $O(1)$-approximation. Suprisingly, only special cases of GMMN such as MMN have been considered so far – despite the fact that the problem is very natural and relevant for practical applications.

Another special case of GMMN that has received significant attention in the past is the *rectilinear Steiner arborescence problem* (RSA). Here, one is given a set of n terminals in the first quadrant, and the goal is to find a minimum-length rectilinear network that M-connects every terminal to the origin o. Hence, RSA is the special case of GMMN where o is considered a (new) terminal and the set of terminal pairs contains, for each terminal $t \neq o$, only the pair (o, t). Note that RSA is very different from MMN. Although every RSA solution is connected (via the origin), terminals are not necessarily *M-connected* to each other. RSA was introduced by Nastansky et al. [14]. RSA is NP-hard [17]. Rao et al. [9] gave a 2-approximation algorithm for RSA. They also provided a conceptually simpler $O(\log n)$-approximation algorithm based on rectilinear Steiner trees. We generalize this algorithm to dimensions $d > 2$ (see Appendix E). Lu et al. [11] and, independently, Zachariasen [19] described polynomial-time approximation schemes (PTAS) for RSA, both based on Arora's technique [3]. Zachariasen pointed out that his PTAS can be generalized to the all-quadrant version of RSA but that it seems difficult to extend the approach to higher dimensions.

Our Contribution. Our main result is the first approximation algorithm for GMMN. Its ratio is $O(\log n)$ (see Section 3). Our algorithm is based on two ideas. First, we use a simple (yet powerful) divide-and-conquer scheme to obtain a performance $O(\log^2 n)$. To bring down the ratio to $O(\log n)$ we develop a new *stabbing technique*, which is a novel way to approach Manhattan network problems and constitutes the main technical contribution of this paper.

We also consider higher dimensions. More specifically, we generalize an existing $O(\log n)$-approximation algorithm for RSA to arbitrary dimensions (see Appendix E). Combining this with our divide-and-conquer scheme yields an $O(\log^{d+1} n)$-approximation algorithm for d-dimensional GMMN (see Section 4). For the special case of d-dimensional MMN, this constitutes an exponential improvement upon the $O(n^\varepsilon)$-approximation algorithm of Das et al. [5]. Another advantage of our algorithm is that it is significantly simpler and easier to analyze than that algorithm.

Our result is a first step towards answering the open question of Chepoi et al. [7]. We give indications that it may be difficult to obtain an $O(1)$-approximation since the problem can be viewed as a geometric rectangle covering problem (see Appendix A). We also argue (see Appendix B) why existing techniques for MMN seem to fail, which underlines the relevance of our techniques.

2 Divide-And-Conquer Scheme

As a warm-up, we start with a simple $O(\log^2 n)$-approximation algorithm illustrating our divide-and-conquer scheme. This is the basis for (a) an improved $O(\log n)$-appro-

ximation algorithm that uses our stabbing technique (see Section 3) and (b) a divide-and-conquer scheme for GMMN in arbitrary dimensions (Section 4). We prove the following.

Theorem 1. *GMMN admits an $O(\log^2 n)$-approximation algorithm running in $O(n \log^3 n)$ time.*

Our algorithm consists of a *main algorithm* that recursively subdivides the input instance into instances of so-called *x-separated* GMMN; see Section 2.1. We prove that the instances of x-separated GMMN can be solved independently by paying a factor of $O(\log n)$ in the overall approximation ratio. Then we solve each x-separated GMMN instance within factor $O(\log n)$; see Section 2.2. This yields an overall approximation ratio of $O(\log^2 n)$. Our analysis is tight; see Appendix G. Our presentation follows this natural top-down approach; as a consequence, we will make some forward references to results that we prove later.

2.1 Main Algorithm

Our algorithm is based on divide and conquer. Let R be the set of terminal pairs that are to be M-connected. Recall, that we identify each terminal pair with its bounding box. As a consequence of this, we consider R a set of rectangles. Let m_x be the median in the multiset of the x-coordinates of terminals where a terminal occurs as often as the number of pairs it is involved in. We identify m_x with the vertical line at $x = m_x$.

Now we partition R into three subsets R_{left}, R_{mid}, and R_{right}. R_{left} consists of all rectangles that lie *completely* to the left of the vertical line m_x. Similarly, R_{right} consists of all rectangle that lie *completely* to the right of m_x. R_{mid} consists of all rectangles that intersect m_x.

We consider the sets R_{left}, R_{mid}, and R_{right} as separate instances of GMMN. We apply the main algorithm recursively to R_{left} to get a rectilinear network that M-connects terminal pairs in R_{left} and do the same for R_{right}.

It remains to M-connect the pairs in R_{mid}. We call a GMMN instance (such as R_{mid}) *x-separated* if there is a vertical line (in our case m_x) that intersects every rectangle. We exploit this property to design a simple $O(\log n)$-approximation algorithm for x-separated GMMN; see Section 2.2. In Section 3, we improve upon this and describe an $O(1)$-approximation algorithm for x-separated GMMN.

In the following lemma we analyze the performance of the main algorithm, in terms of $\rho_x(n)$, our approximation ratio for x-separated instances with n terminal pairs.

Lemma 1. *If x-separated GMMN admits a $\rho_x(n)$-approximation, GMMN admits a $(\rho_x(n) \cdot \log n)$-approximation.*

Proof. We determine an upper bound $\rho(n)$ on the main algorithm's approximation ratio for instances with n terminal pairs. Let N^{opt} be an optimum solution to an instance R of size n and let OPT be the cost of N^{opt}. Let $N^{\text{opt}}_{\text{left}}$ and $N^{\text{opt}}_{\text{right}}$ be the parts of N^{opt} to the left and to the right of m_x, respectively. (We split horizontal segments that cross m_x and ignore vertical segments on m_x.)

Due to the choice of m_x, at most n terminals lie to the left of m_x. Therefore, R_{left} contains at most $n/2$ terminal pairs. Since $N^{\text{opt}}_{\text{left}}$ is a feasible solution to R_{left}, we conclude (by induction) that the cost of the solution to R_{left} computed by our algorithm is bounded by $\rho(n/2) \cdot \|N^{\text{opt}}_{\text{left}}\|$, where $\|\cdot\|$ measures the length of a network. Analogously, the cost of the solution computed for R_{right} is bounded by $\rho(n/2) \cdot \|N^{\text{opt}}_{\text{right}}\|$. Since N^{opt} is also a feasible solution to the x-separated instance R_{mid}, we can compute a solution of cost $\rho_x(n) \cdot \text{OPT}$ for R_{mid}.

As the networks $N^{\text{opt}}_{\text{left}}$ and $N^{\text{opt}}_{\text{right}}$ are separated by line m_x, they are edge disjoint and hence $\|N^{\text{opt}}_{\text{left}}\| + \|N^{\text{opt}}_{\text{right}}\| \leq \text{OPT}$. Therefore, we can bound the total cost of our algorithm's solution N to R by

$$\rho(n/2) \cdot (\|N^{\text{opt}}_{\text{left}}\| + \|N^{\text{opt}}_{\text{right}}\|) + \rho_x(n) \cdot \text{OPT} \leq (\rho(n/2) + \rho_x(n)) \cdot \text{OPT}\,.$$

This yields the recurrence $\rho(n) = \rho(n/2) + \rho_x(n)$, which resolves to $\rho(n) \leq \log n \cdot \rho_x(n)$. □

Lemma 1 together with the results of Section 2.2 allow us to prove Theorem 1.

Proof (Proof of Theorem 1). By Lemma 1, our main algorithm has performance $\rho_x(n) \cdot \log n$, where $\rho_x(n)$ denotes the ratio of an approximation algorithm for x-separated GMMN. In Lemma 2 (Section 2.2), we will show that there is an algorithm for x-separated GMMN with ratio $\rho_x(n) = O(\log n)$. Thus overall, the main algorithm yields an $O(\log^2 n)$-approximation for GMMN. See Appendix F for the running time analysis.

2.2 Approximating x-Separated and xy-Separated Instances

We describe a simple algorithm for approximating x-separated GMMN with a ratio of $O(\log n)$. Let R be an x-separated instance, that is, all rectangles in R intersect a common vertical line.

The algorithm works as follows. Analogously to the main algorithm we subdivide the x-separated input instance, but this time using the line $y = m_y$, where m_y is the median of the multiset of y-coordinates of terminals in R. This yields sets R_{top}, R'_{mid}, and R_{bottom}, defined analogously to the sets R_{left}, R_{mid}, and R_{right} of the main algorithm, using m_y instead of m_x. We apply our x-separated algorithm to R_{top} and then to R_{bottom} to solve them recursively. The instance R'_{mid} is a *y-separated* sub-instance with all its rectangles intersecting the line m_y. Moreover, R'_{mid} (as a subset of R) is already x-separated, thus we call R'_{mid} an *xy-separated* instance. Below, we describe a specialized algorithm to approximate xy-separated instances within a constant factor. Assuming this for now, we prove the following.

Lemma 2. *x-separated GMMN admits an $O(\log n)$-approximation.*

Proof. Let $\rho_x(n)$ be the ratio of our algorithm for approximating x-separated GMMN instances and let $\rho_{xy}(n)$ be the ratio of our algorithm for approximating xy-separated GMMN instances. In Lemma 3, we show that $\rho_{xy}(n) = O(1)$.

Following the proof of Lemma 1 (exchanging x- and y-coordinates and using R_{top}, R'_{mid}, R_{bottom} in place of $R_{\text{left}}, R_{\text{mid}}, R_{\text{right}}$), yields $\rho_x(n) = \log n \cdot \rho_{xy}(n) = O(\log n)$. □

It remains to show that xy-separated GMMN can be approximated within a constant ratio. Let R be an instance of xy-separated GMMN. We assume, w.l.o.g., that it is the x- and the y-axes that intersect all rectangles in R, that is, all rectangles contain the origin o. To solve R, we compute an RSA network that M-connects the set of terminals in R to o. Clearly, we obtain a feasible GMMN solution to R. In Appendix C we prove that this is a constant-factor approximation.

Lemma 3. *xy-separated GMMN admits a constant-factor approximation.*

3 An $O(\log n)$-Approximation Algorithm via Stabbing

In this section, we present an $O(\log n)$-approximation algorithm for GMMN, which is the main result of our paper. Our algorithm relies on an $O(1)$-approximation algorithm for x-separated instances and is based on a novel stabbing technique that computes a cheap set of horizontal line segments that stabs all rectangles. Our algorithm connects these line segments with a suitable RSA solution to ensure feasibility and approximation ratio. We show the following (noting that our analysis is tight up to a constant factor; see Appendix G).

Theorem 2. *For any $\varepsilon > 0$, GMMN admits a $((6+\varepsilon) \cdot \log n)$-approximation algorithm running in $O(n^{1/\varepsilon} \log^2 n)$ time.*

Proof. Using our new subroutine for the x-separated case given in Lemma 7 below, along with Lemma 1 yields the result. See Appendix F for the run-time analysis.

We begin with an overview of our improved algorithm for x-separated GMMN. Let R be the set of terminal pairs of an x-separated instance of GMMN. We assume, w.l.o.g., that each terminal pair $(t, t') \in R$ is separated by the y-axis, that is $x(t) \leq 0 \leq x(t')$ or $x(t') \leq 0 \leq x(t)$. Let N^{opt} be an optimum solution to R. Let $\mathrm{OPT}_{\mathrm{ver}}$ and $\mathrm{OPT}_{\mathrm{hor}}$ be the total costs of the vertical and horizontal segments in N^{opt}, respectively. Hence, $\mathrm{OPT} = \mathrm{OPT}_{\mathrm{ver}} + \mathrm{OPT}_{\mathrm{hor}}$. We first compute a set S of horizontal line segments of total cost $O(\mathrm{OPT}_{\mathrm{hor}})$ such that each rectangle in R is *stabbed* by some line segment in S; see Sections 3.1 and 3.2. Then we M-connect the terminals to the y-axis so that the resulting network, along with S, forms a feasible solution to R of cost $O(\mathrm{OPT})$; see Section 3.3.

3.1 Stabbing the Right Part

We say that a horizontal line segment h *stabs* an axis-aligned rectangle r if the intersection of r and h equals the intersection of r and the supporting line of h. A set of horizontal line segments is a *stabbing* of a set of axis-aligned rectangles if each rectangle is stabbed by some line segment. For any geometric object, let its *right part* be its intersection with the closed half plane to the right of the y-axis. For a *set* of objects, let its right part be the set of the right parts of the objects. Let R^+ be the right part of R, let N^+ be the right part of N^{opt}, and let N^+_{hor} be the set of horizontal line segments in N^+. In this section, we show how to construct a stabbing of R^+ of cost at most $2 \cdot \|N^+_{\mathrm{hor}}\|$.

For $x' \geq 0$, let $\ell_{x'}$ be the vertical line at $x = x'$. Our algorithm performs a left-to-right sweep starting with ℓ_0. For $x \geq 0$, let $\mathcal{I}_x = \{r \cap \ell_x \mid r \in R^+\}$ be the "traces" of the rectangles in R^+ on ℓ_x. The elements of $\mathcal{I}_x$ are vertical line segments; we refer to them as *intervals*. A set P_x of points on ℓ_x constitutes a *piercing* for $\mathcal{I}_x$, if every interval in $\mathcal{I}_x$ contains a point in P_x.

Our algorithm continuously moves the line ℓ_x from left to right starting with $x = 0$. In doing so, we maintain an inclusion-wise minimal piercing P_x of $\mathcal{I}_x$ in the following way: At $x = 0$, we start with an arbitrary minimal piercing P_0. (Note that we can even compute an optimum piercing.) We update P_x whenever $\mathcal{I}_x$ changes. Observe that with increasing x, the set $\mathcal{I}_x$ can only inclusion-wise decrease as all rectangles in R^+ touch the y-axis. Therefore, it suffices to update the piercing P_x only at *event points*; x is an event point if and only if x is the x-coordinate of a right edge of a rectangle in R^+. Let x' and x'' be consecutive event points. Let x be such that $x' < x \leq x''$. Note that $P_{x'}$ is a piercing for $\mathcal{I}_x$ since $\mathcal{I}_x \subset \mathcal{I}_{x'}$. The piercing $P_{x'}$ is, however, not necessarily *minimal* w.r.t. $\mathcal{I}_x$. When the sweep line passes x', we therefore have to drop some of the points in $P_{x'}$ in order to obtain a new minimal piercing. This can be done by iteratively removing points from $P_{x'}$ such that the resulting set still pierces $\mathcal{I}_x$. We stop at the last event point (afterwards, $\mathcal{I}_x = \emptyset$) and output the *traces* of the piercing points in P_x for $x \geq 0$ as our stabbing.

Note that with increasing x, our algorithm only *removes* points from P_x but never add points. Thus, the traces of P_x form horizontal line segments that touch the y-axis. These line segments form a stabbing of R^+; see the thick solid line segments in Fig. 2a. The following lemma is crucial to prove the overall cost of the stabbing.

Lemma 4. *For any $x \geq 0$, it holds that $|P_x| \leq 2 \cdot |\ell_x \cap N^+_{\mathrm{hor}}|$.*

Proof. Since P_x is a minimal piercing, there exists, for every $p \in P_x$, a *witness* $I_p \in \mathcal{I}_x$ that is pierced by p but not by $P_x \setminus \{p\}$. Otherwise we could remove p from P_x, contradicting the minimality of P_x.

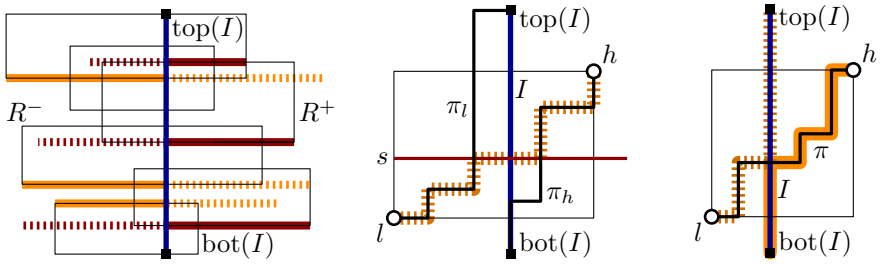
Now we show that an arbitrary point q on ℓ_x is contained in the witnesses of at most two points in P_x. Assume, for the sake of contradiction, that q is contained in the witnesses of points $p, p', p'' \in P_x$ with strictly increasing y-coordinates. Suppose that q lies above p'. But then the witness I_p of p, which contains p and q, must also contain p', contradicting the definition of I_p. The case q below p' is symmetric.

Observe that $\ell_x \cap N^+_{\mathrm{hor}}$ is a piercing of $\mathcal{I}_x$ and, hence, of the $|P_x|$ many witnesses. Since every point in $\ell_x \cap N^+_{\mathrm{hor}}$ pierces at most two witnesses, the lemma follows.

Next, we analyze the overall cost of the stabbing.

Lemma 5. *Given a set R of rectangles intersecting the y-axis, we can compute a set of horizontal line segments of cost at most $2 \cdot \mathrm{OPT}_{\mathrm{hor}}$ that stabs R^+.*

Proof. Observe that $\|N^+_{\mathrm{hor}}\| = \int |\ell_x \cap N^+_{\mathrm{hor}}|\, dx$. The cost of our stabbing is $\int |P_x|\, dx$. By Lemma 4, this can be bounded by $\int |P_x|\, dx \leq \int 2 \cdot |\ell_x \cap N^+_{\mathrm{hor}}|\, dx = 2 \cdot \|N^+_{\mathrm{hor}}\|$.



(a) The dark (light) segments S^+ (S^-) stab R^+ (R^-). The dotted segments are mirror images of $S^+ \cup S^-$.

(b) $N = A_{\text{up}} \cup A_{\text{down}} \cup S$ is feasible for R.

(c) $N^{\text{opt}} \cup \{I\}$ is feasible for RSA instances $(L, \text{top}(I))$, $(H, \text{bot}(I))$.

Fig. 2: The improved algorithm for x-separated GMMN.

3.2 Stabbing the Right and Left Parts

We now detail how we construct a stabbing of R. To this end we apply Lemma 5 to compute a stabbing S^- of cost at most $2 \cdot \|N^-_{\text{hor}}\|$ for the left part R^- of R and a stabbing S^+ of cost at most $2 \cdot \|N^+_{\text{hor}}\|$ for the right part R^+. Note that $S^- \cup S^+$ is not necessarily a stabbing of R since there can be rectangles that are not *completely* stabbed by one segment (even if we start with the same piercing on the y-axis in the sweeps to the left and to the right). To overcome this difficulty, we mirror S^- and S^+ to the respective other side of the y-axis; see Fig. 2a. Let S denote the union of $S^- \cup S^+$ and the mirror image of $S^- \cup S^+$.

Lemma 6. *Given a set R of rectangles intersecting the y-axis, we can compute a set of horizontal line segments of cost at most $4 \cdot \text{OPT}_{\text{hor}}$ that stabs R.*

Proof. Let S be the set of horizontal line segments described above. The total cost of S is at most $4(\|N^-_{\text{hor}}\| + \|N^+_{\text{hor}}\|) = 4 \cdot \text{OPT}_{\text{hor}}$. The set S stabs R since, for every rectangle $r \in R$, the larger among its two (left and right) parts is stabbed by some segment s and the smaller part is stabbed by the mirror image s' of s. Hence, r is stabbed by the line segment $s \cup s'$.

3.3 Connecting Terminals and Stabbing

We assume that the union of the rectangles in R is connected. Otherwise we apply our algorithm separately to each subset of R that induces a connected component of $\bigcup R$. Let I be the line segment that is the intersection of the y-axis with $\bigcup R$. Let $\text{top}(I)$ and $\text{bot}(I)$ be the top and bottom endpoints of I, respectively. Let $L \subseteq T$ be the set containing every terminal t with $(t, t') \in R$ and $y(t) \leq y(t')$ for some $t' \in T$. Symmetrically, let $H \subseteq T$ be the set containing every terminal t with $(t, t') \in R$ and $y(t) > y(t')$ for some $t' \in T$. Note that, in general, L and H are not disjoint.

Using a PTAS for RSA [11,19], we compute a near-optimal RSA network A_{up} connecting the terminals in L to $\text{top}(I)$ and a near-optimal RSA network A_{down} connecting

the terminals in H to $\mathrm{bot}(I)$. Then we return the network $N = A_{\mathrm{up}} \cup A_{\mathrm{down}} \cup S$, where S is the stabbing computed by the algorithm in Section 3.2.

We prove in the following lemma that the resulting network is a feasible solution to R, with cost at most constant times OPT.

Lemma 7. *x-separated GMMN admits, for any $\varepsilon > 0$, a $(6+\varepsilon)$-approximation.*

Proof. First we argue that the solution is feasible. Let $(l, h) \in R$. W.l.o.g., $l \in L$ and $h \in H$. (If $l, h \in L$, then $y(l) = y(h)$ and the line segment that stabs (l, h) already M-connects l and h.) Hence, A_{up} contains a path π_l from l to $\mathrm{top}(I)$, see Fig. 2b. This path starts inside the rectangle (l, h). Before leaving (l, h), the path intersects a line segment s in S that stabs (l, h). The segment s is also intersected by the path π_h in A_{down} that connects h to $\mathrm{bot}(I)$. Hence, walking along π_l, s, and π_h brings us in a monotone fashion from l to h.

Now, let us analyze the cost of N. Clearly, the projection of (the vertical line segments of) N^{opt} onto the y-axis yields the line segment I. Hence, $|I| \leq \mathrm{OPT}_{\mathrm{ver}}$. Observe that $N^{\mathrm{opt}} \cup \{I\}$ constitutes a solution to the RSA instance $(L, \mathrm{top}(I))$ connecting all terminals in L to $\mathrm{top}(I)$ and to the RSA instance $(H, \mathrm{bot}(I))$ connecting all terminals in H to $\mathrm{bot}(I)$. This holds since, for each terminal pair, its M-path π in N^{opt} crosses the y-axis in I; see Fig. 2c. Since A_{up} and A_{down} are near-optimal solutions to these RSA instances, we obtain, for any $\delta > 0$, that $\|A_{\mathrm{up}}\| \leq (1+\delta) \cdot \|N^{\mathrm{opt}} \cup I\| \leq (1+\delta) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}})$ and, analogously, that $\|A_{\mathrm{down}}\| \leq (1+\delta) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}})$.

By Lemma 6, we have $\|S\| \leq 4 \cdot \mathrm{OPT}_{\mathrm{hor}}$. Assuming $\delta \leq 1$, this yields

$$\begin{aligned}\|N\| &= \|A_{\mathrm{up}}\| + \|A_{\mathrm{down}}\| + \|S\| \leq (2+2\delta) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}}) + 4 \cdot \mathrm{OPT}_{\mathrm{hor}} \\ &\leq (2+2\delta) \cdot \mathrm{OPT} + 4 \cdot (\mathrm{OPT}_{\mathrm{ver}} + \mathrm{OPT}_{\mathrm{hor}}) = (6+2\delta) \cdot \mathrm{OPT}.\end{aligned}$$

Setting $\delta = \varepsilon/2$ yields the desired approximation factor.

4 Generalization to Higher Dimensions

In this section, we describe an $O(\log^{d+1} n)$-approximation algorithm for GMMN in d dimensions and prove the following result (see below for the proof). In Appendix G we show that the analysis of the algorithm is essentially tight (up to one log-factor).

Theorem 3. *In any fixed dimension d, GMMN admits an $O(\log^{d+1} n)$-approximation algorithm running in $O(n^2 \log^{d+1} n)$ time.*

In Section 2 we reduced GMMN to x-separated GMMN and then x-separated GMMN to xy-separated GMMN. Each of the two reductions increased the approximation ratio by $(\log n)$. The special case of xy-separated GMMN was approximated within a constant factor by solving a related RSA problem. This gave an overall $O(\log^2 n)$ approximation for GMMN. We generalize this approach to higher dimensions.

An instance R of d-dimensional GMMN is called *j-separated* for some $j \leq d$ if there exist values $s_1, \ldots, s_j$ such that, for each terminal pair $(t, t') \in R$ and for each dimension $i \leq j$, we have that s_i *separates* the i-th coordinates $x_i(t)$ of t and $x_i(t')$

of t' (meaning that either $x_i(t) \leq s_i \leq x_i(t')$ or $x_i(t') \leq s_i \leq x_i(t)$). Under this terminology, an arbitrary instance of d-dimensional GMMN is always *0-separated*.

The following lemma reduces j-separated GMMN to $(j-1)$-separated GMMN at the expense of a $(\log n)$-factor in the approximation. The proof is similar to the 2D case; see Appendix D.

Lemma 8. *Let $1 \leq j \leq d$. If j-separated GMMN admits a $\rho_j(n)$-approximation, then $(j-1)$-separated GMMN admits a $(\rho_j(n) \cdot \log n)$-approximation.*

Analogously to dimension two we can approximate instances of d-separated GMMN by reducing the problem to RSA. Rao et al. [9] presented an $O(\log |T|)$-approximation algorithm for 2D-RSA, which generalizes to d-dimensional RSA (see Appendix E). This gives the following result (see Appendix D for details).

Lemma 9. *d-separated GMMN admits an $O(\log n)$-approximation for any fixed d.*

We are now ready to give the proof of Theorem 3.

Proof (Proof of Theorem 3). Combining Lemmata 8 and 9 and applying them inductively to arbitrary (that is, 0-separated) GMMN instances yields the claim. See Appendix F for the run-time analysis.

As a byproduct of Theorem 3, we obtain an $O(\log^{d+1} n)$-approximation algorithm for *MMN* where n denotes the number of *terminals*. This holds since any MMN instance with n terminals can be considered an instance of GMMN with $O(n^2)$ terminal pairs.

Corollary 1. *In any fixed dimension d, MMN admits an $O(\log^{d+1} n)$-approximation algorithm running in $O(n^4 \log^{d+1} n)$ time, where n denotes the number of terminals.*

5 Conclusions

In 2D, there is quite a large gap between the currently best approximation ratios for MMN and GMMN. Whereas we have presented an $O(\log n)$-approximation algorithm for GMMN, MMN admits 2-approximations [7,9,8]. In Appendix A we give indications that this gap might not only be a shortcoming of our algorithm. It would be interesting to derive complementing non-approximability results for GMMN or at least to give a formal reduction from a more extensively geometric covering problem. So far, the only such result is APX-hardness of 3D-MMN [12].

Concerning the positive side, for $d \geq 3$, a constant-factor approximation for d-dimensional RSA would shave off a factor of $O(\log n)$ from the current ratio for d-dimensional GMMN. This may be in reach since 2D-RSA admits even a PTAS [11,19]. Alternatively, a constant-factor approximation for $(d-k)$-separated GMMN for some $k \leq d$ would shave off a factor of $O(\log^k n)$ from the current ratio for d-dimensional GMMN.

Acknowledgments. We thank Michael Kaufmann for his hospitality and his enthusiasm during our respective stays in Tübingen. We thank Esther Arkin, Alon Efrat, and Joe Mitchell for discussions. We thank Andreas Spillner for pointing us to the application of Manhattan networks in phylogenetics.

References

1. S. Arora. Approximation schemes for NP-hard geometric optimization problems: A survey. *Math. Program.*, 97(1–2):43–69, 2003.
2. V. Chepoi, K. Nouioua, and Y. Vaxès. A rounding algorithm for approximating minimum Manhattan networks. *Theor. Comput. Sci.*, 390(1):56–69, 2008.
3. F. Chin, Z. Guo, and H. Sun. Minimum Manhattan network is NP-complete. *Discrete Comput. Geom.*, 45:701–722, 2011.
4. J. Cong, K.-S. Leung, and D. Zhou. Performance-driven interconnect design based on distributed RC delay model. In *Proc. 30th IEEE Conf. Design Automation (DAC'93)*, pages 606–611, 1993.
5. A. Das, E. R. Gansner, M. Kaufmann, S. Kobourov, J. Spoerhase, and A. Wolff. Approximating minimum Manhattan networks in higher dimensions. In C. Demetrescu and M. M. Halldórsson, editors, *Proc. 19th Annu. Europ. Symp. on Algorithms (ESA'11)*, volume 6942 of *LNCS*, pages 49–60. Springer, 2011.
6. J. Gudmundsson, O. Klein, C. Knauer, and M. Smid. Small Manhattan networks and algorithmic applications for the Earth Mover's Distance. In *Proc. 23rd Europ. Workshop Comput. Geom. (EuroCG'07)*, pages 174–177, Graz, Austria, 2007.
7. J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Approximating a minimum Manhattan network. *Nordic J. Comput.*, 8:219–232, 2001.
8. J. Gudmundsson, G. Narasimhan, and M. Smid. Applications of geometric spanner networks. In M.-Y. Kao, editor, *Encyclopedia of Algorithms*, pages 1–99. Springer-Verlag, 2008.
9. Z. Guo, H. Sun, and H. Zhu. Greedy construction of 2-approximate minimum Manhattan networks. *Int. J. Comput. Geom. Appl.*, 21(3):331–350, 2011.
10. C. Knauer and A. Spillner. A fixed-parameter algorithm for the minimum Manhattan network problem. *J. Comput. Geom.*, 2(1):189–204, 2011.
11. B. Lu and L. Ruan. Polynomial time approximation scheme for the rectilinear Steiner arborescence problem. *J. Comb. Optim.*, 4(3):357–363, 2000.
12. X. Muñoz, S. Seibert, and W. Unger. The minimal Manhattan network problem in three dimensions. In S. Das and R. Uehara, editors, *Proc. 3rd Int. Workshop Algorithms Comput. (WALCOM'09)*, volume 5431 of *LNCS*, pages 369–380. Springer, 2009.
13. G. Narasimhan and M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007.
14. L. Nastansky, S. M. Selkow, and N. F. Stewart. Cost-minimal trees in directed acyclic graphs. *Zeitschrift Oper. Res.*, 18(1):59–67, 1974.
15. K. Nouioua. *Enveloppes de Pareto et Réseaux de Manhattan: Caractérisations et Algorithmes*. PhD thesis, Université de la Méditerranée, 2005. Available at http://www.lif-sud.univ-mrs.fr/~karim/download/THESE_NOUIOUA.pdf.
16. S. Rao, P. Sadayappan, F. Hwang, and P. Shor. The rectilinear Steiner arborescence problem. *Algorithmica*, 7:277–288, 1992.
17. W. Shi and C. Su. The rectilinear Steiner arborescence problem is NP-complete. *SIAM J. Comput.*, 35(3):729–740, 2005.
18. A. Spillner. Minimum Manhattan networks and split systems. Note, May 2009. Available at www.math-inf.uni-greifswald.de/mathe/images/Spillner/publications/mmn_and_splits.pdf.
19. M. Zachariasen. On the approximation of the rectilinear Steiner arborescence problem in the plane. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.4529, 2000.

Appendix

A Separating GMMN from MMN

Theorem 2 constitutes a first step towards answering the question of Chepoi et al. [7] whether GMMN admits an $O(1)$-approximation algorithm. While we don't have a lower bound on the approximability of GMMN, we give indications why GMMN appears to be substantially harder than MMN—for which an $O(1)$-approximation is known—and that it appears actually quite difficult to give a positive answer to the question of Chepoi et al. The aim of this discussion is to enable the reader to better assess our result.

GMMN can be viewed as a special rectangle covering problem. Specifically, we are given a set of rectangles and the problem is to find a cheapest set of line segments that "cover" all rectangles in a very specific sense (so that every rectangle contains an M-path connecting two opposite corners). Note that even for structurally simpler rectangle covering problems no $O(1)$-approximation algorithms are known. For example, the best known approximation for the extensively studied problem of covering (piercing) a set of rectangles in the plane by a minimum number of *points* has a ratio of $O(\log \log \mathrm{OPT})$[1], which is not constant. Beating the immediate $O(\log n)$-bound obtained by casting this problem as a set cover problem was a breakthrough [6].

It is also instructive to consider the counterpart of MMN in the context of rectangle piercings: Given a set T of terminals, find a smallest set of points such that *every* rectangle spanned by a terminal pair is pierced by some point. In analogy to MMN, this special case is significantly easier than general rectangle piercing. In fact, it is not hard to see that the set of terminals itself is already a 2-approximation. Denote the terminals $t_1, \ldots, t_n$ in order of increasing x-coordinate. Now, observe that T itself is a piercing (of cardinality n) and that the pairs $(t_1, t_2), (t_3, t_4), \ldots, (t_{n-1}, t_n)$ form a set of disjoint rectangles (assuming that no two terminals have the same x-coordinates), which implies that every feasible piercing needs at least $n/2$ points.

B Applicability of Existing Techniques

It seems compelling to apply Arora's technique [3] that yields PTAS's for a wide variety of geometric optimization problems. An issue that arises here is that a feasible solution to GMMN is not connected in general (in contrast to TSP or the Steiner tree problem) and that a good partition into connected components has therefore to be "guessed" by the algorithm. For the case of Euclidean Steiner forest this issue has been settled only recently by Borradaile et al. [5] who gave a PTAS by cleverly extending Arora's technique. Bateni et al. [4] significantly simplified the algorithm. A second, even more basic problem arises, however, from the monotonicity constraint in (G)MMN. Arora uses a quadtree decomposition and applies a *patching lemma* to modify any given feasible solution to a special type of feasible solutions that cross the separating lines of the decomposition at well-defined points (so-called *ports*) without significantly increasing the cost of the solution. A weaker requirement would be to ensure that the solution crosses only a constant number of times between two ports. This cannot be ensured

for (G)MMN. Imagine an instance where we have an arbitrarily large number of point pairs forming a set of pairwise disjoint, very thin rectangles that cross the separating line between two adjacent ports p_1, p_2. (Remember that we identify each terminal pair with its bounding box.) Then there is no feasible solution such that only a constant number of M-paths cross the separating line between p_1 and p_2. Finally, let us note that there are no works using Arora's technique for the very well-investigated special case MMN.

Many algorithms for MMN rely on the concept of so-called *generating sets* [7,8]. A generating set is a nicely structured subset of terminal pairs with the property that M-connecting all these pairs already guarantees overall feasibility. One feature of MMN that is used in the construction of a generating set is the following simple emptiness property. (The actual construction is quite sophisticated and yields a geometrically strong decomposition of the instance into significantly simpler subinstances; so-called *staircases*.) If a rectangle (p_1, p_3) (terminal pair) is not empty, that is, it contains a point p_2 then we do not need to consider (p_1, p_3) because M-connecting (p_1, p_2) and (p_2, p_3) already implies an M-path from p_1 to p_3. This property does not continue to hold for GMMN. If we consider a large rectangle containing many smaller nested or overlapping rectangles then there is no easy way to reduce this instance.

Another idea that is used is to decompose into (interior-) disjoint rectangles as follows. Sort the terminals $t_1, \ldots, t_n$ by increasing x-coordinate and add the rectangles $(t_1, t_2), \ldots, (t_{n-1}, t_n)$ spanned by consecutive terminals to the generating set (besides the above-mentioned staircases). In the case of GMMN such a decomposition is impossible. For example, if the pairs in the GMMN instance are exactly the pairs $\{(t_i, t_{i+n/2}) : i = 1, \ldots, n/2\}$ then there is no general way to decompose in the above manner into disjoint rectangles.

C Proofs for Two-Dimensional GMMN

Lemma 3. *xy-separated GMMN admits a constant-factor approximation.*

Proof. Let T be the set of terminals in the xy-separated GMMN instance R and N' be an RSA network M-connecting T to o, which we compute using the 2-approximation for RSA of Rao et al. [9].

We first claim that N' is a feasible GMMN solution for R. To see this, note that N' contains, for every terminal pair $(t, t') \in R$, an M-path π from t to o and an M-path π' from o to t'. Concatenating π and π' yields an M-path from t to t' as the bounding box of (t, t') contains o.

Let OPT denote the cost of an optimal GMMN solution for R. We claim there is a solution N for the RSA instance of M-connecting T to o of cost $O(\mathrm{OPT})$.

Let N^{opt} be an optimum solution to R. Let N be the union of N^{opt} and the projections of N^{opt} to the x-axis and to the y-axis. The total length of N is $\|N\| \leq 2 \cdot \mathrm{OPT} = O(\mathrm{OPT})$ since every line segment of N^{opt} is projected either to the x-axis or to the y-axis but not to both. The crucial fact about N is that this network contains, for every terminal t in R, an M-path from t to the *origin* o. In other words, N is a feasible solution to the RSA instance of M-connecting T to o.

To see this, consider an arbitrary terminal pair $(t, t') \in R$. Let Π be an M-path connecting t and t' in N^{opt}; see Fig. 3. Note that, since the bounding box of (t, t') contains o, Π intersects both x- and y-axis. To obtain an M-path from t to o, we follow Π from t to t' until Π crosses one of the axes. From that point on, we follow the *projection* of Π onto this axis. We reach o when Π crosses the other axis; see the dotted path in Fig. 3. Analogously, we obtain an M-path from t' to o.

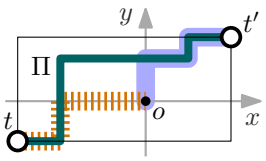


Fig. 3: Network N connects t to o (dashed path) and t' to o (solid path).

Finally, as there is a feasible RSA solution N for terminals T of cost $O(\mathrm{OPT})$, the RSA solution N' that we compute will have cost at most $2\|N\| = O(\mathrm{OPT})$. □

D Proofs for d-Dimensional GMMN

Lemma 8. *Let $1 \leq j \leq d$. If j-separated GMMN admits a $\rho_j(n)$-approximation, then $(j-1)$-separated GMMN admits a $(\rho_j(n) \cdot \log n)$-approximation.*

Proof. The separation algorithm and its analysis work analogously to the main algorithm for 2D where we reduced (approximating) 2D-GMMN to (approximating) x-separated 2D-GMMN; see Section 2.1

Let R be a set of $(j-1)$-separated terminal pairs. Let m_x be the median in the multiset of the j-th coordinates of terminals. We divide R into three subsets R_{left}, R_{mid}, and R_{right}. The set R_{left} consists of all terminal pairs (t, t') such that $x_j(t), x_j(t') \leq m_x$ and R_{right} contains all terminal pairs (t, t') with $x_j(t), x_j(t') \geq m_x$. The set R_{mid} contains the remaining terminal pairs, all of which are separated by the hyperplane $x_j = m_x$. We apply our algorithm recursively to R_{left} and R_{right}. The union of the resulting networks is a rectilinear network that M-connects all terminal pairs $R_{\mathrm{left}} \cup R_{\mathrm{right}}$.

In order to M-connect the pairs in R_{mid}, we apply an approximation algorithm for j-separated GMMN of ratio $\rho_j(n)$. Note that the instance R_{mid} is in fact j-separated by construction. The remaining analysis of the resulting algorithm for $(j-1)$-separated GMMN is analogous to the 2D-case (see Section 2.1).

Lemma 9. *d-separated GMMN admits an $O(\log n)$-approximation for any fixed d.*

Proof. First we observe that solving an RSA instance with terminal set T yields a feasible GMMN solution to R since for each pair $(t, t') \in R$ there is an M-path from t to the origin and an M-path from the origin to t'. The union of these paths is an M-path from t to t' since the origin is contained in the bounding box of (t, t').

Below we show that there is a solution of cost $O(\mathrm{OPT})$ to the RSA instance connecting T to the origin. Observing that $|T| \leq 2n$ and using our extension of the $O(\log |T|)$-approximation algorithm of Rao et al. (see Appendix E), we can efficiently compute a feasible GMMN solution of cost $O(\mathrm{OPT} \cdot \log n)$, which implies the claim of the lemma.

Let N^{opt} be an optimal GMMN solution to R and let N be the projection of N^{opt} onto all subspaces that are spanned by some subset of the coordinate axes. Since there are 2^d such subspaces, which is a constant for fixed d, the cost of N is $O(\mathrm{OPT})$.

It remains to show that N M-connects all terminals to the origin, that is, N is a feasible solution to the RSA instance. First, note that $N^{\mathrm{opt}} \subseteq N$ since we project on the d-dimensional space, too. Now consider an arbitrary terminal pair (t, t') in R and an M-path π in N^{opt} that M-connects t and t'. Starting at t, we follow π until we reach the first point p_1 where one of the coordinates becomes zero. W.l.o.g., $x_1(p_1) = 0$. Clearly π contains such a point as the bounding box of (t, t') contains the origin. We observe that p_1 lies in the subspace spanned by the $d-1$ coordinate axes $x_2, \ldots, x_d$. From p_1 on we follow the projection of π onto this subspace until we reach the first point p_2 where another coordinate becomes zero; w.l.o.g., $x_2(p_2) = 0$. Hence, p_2 has at least two coordinates that are zero, that is, p_2 lies in a subspace spanned by only $d-2$ coordinate axes. Iteratively, we continue following the projections of π onto subspaces with a decreasing number of dimensions until every coordinate is zero, that is, we have reached the origin. An analogous argument shows that N also contains an M-path from t' to the origin.

E Solving RSA in Higher Dimensions

In this section, we show that we can approximate d-dimensional RSA with a ratio of $O(\log n)$ even in the all-orthant case where every orthant may contain terminals. In this section, n denotes the number of *terminals*. We generalize the algorithm of Rao et al. [9] who give an $O(\log n)$-approximation algorithm for the one-quadrant version of 2D-RSA.

It is not hard to verify that the $O(\log n)$-approximation algorithm of Rao et al. carries over to higher dimensions in a straightforward manner if all terminals lie in the *same* orthant. We can therefore obtain a feasible solution to the all-orthant version by applying the approximation algorithm to each orthant separately. This worsens the approximation ratio by a factor no larger than 2^d since there are 2^d orthants. Hence, we can quite easily give an $O(\log n)$-approximation algorithm for the all-orthant version since 2^d is a constant for fixed dimension d.

In what follows, we present a tailor-made approximation algorithm for the all-orthant version of d-dimensional RSA that avoids the additional factor of 2^d. Our algorithm is an adaption of the algorithm of Rao et al., and our presentation closely follows their lines, too.

Consider an instance of RSA given by a set T of terminals in $\mathbb{R}^d$ (without restriction of the orthant). Let o denote the origin. The algorithm relies on the following lemma, which we prove below.

Lemma 10. *Given a rectilinear Steiner tree B for terminal set $T \cup \{o\}$, we can find an RSA network A for T of length at most $\lceil \log_2 n \rceil \cdot \|B\|$.*

Every RSA network is also a rectilinear Steiner tree. Since the rectilinear Steiner tree problem (RST) admits a PTAS for any fixed dimension d [2], we can generate a $(1+\varepsilon)$-approximate RST network B that connects T and the origin. By means of Lemma 10, we get a $(1+\varepsilon)\lceil \log_2 n \rceil$-approximation for the RSA instance T.

Theorem 4. *The all-orthant version of d-dimensional RSA admits a $(1+\varepsilon) \cdot \lceil \log_2 n \rceil$ approximation for any $\varepsilon > 0$.*

Our proof of Lemma 10 relies on the following technical lemma, which constitutes the main modification that we make to the algorithm of Rao et al. See Fig. 4 for an illustration.

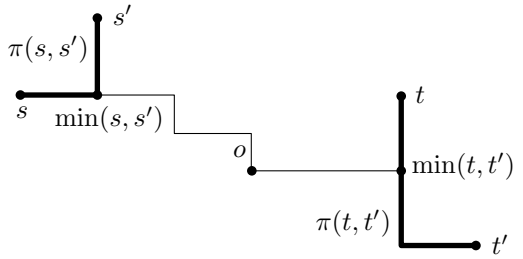


Fig. 4: Illustration of Lemma 11. For each terminal pair t, t', we compute a suitable point $\min(t, t')$ and an M-path $\pi(t, t')$ containing $\min(t, t')$. Adding an arbitrary M-path from $\min(t, t')$ to o also M-connects t and t' to o.

Lemma 11. *Let t, t' be two terminals. Then we can compute in constant time a point $\min(t, t')$ and an M-path $\pi(t, t')$ from t to t' containing $\min(t, t')$ with the following property. The union of $\pi(t, t')$ with an M-path from $\min(t, t')$ to o M-connects t and t' to o.*

Proof. We start with the following simple observation. If s and s' are points and p is a point in the bounding box $B(s, s')$ of s and s', then M-connecting s to p and p to s' also M-connects s to s'.

Observe that the three bounding boxes $B(o, t)$, $B(o, t')$, and $B(t, t')$ have pairwise non-empty intersections. By the Helly property of axis-parallel d-dimensional boxes, there exists a point $\min(t, t')$ that simultaneously lies in all three boxes.

M-connecting $\min(t, t')$ with t and t' yields $\pi(t, t')$, and M-connecting $\min(t, t')$ with o yields an M-path between any two of the points t, t', o by a repeated application of the above observation. This completes the proof.

Proof (Proof of Lemma 10). We double the edges of B and construct a Eulerian cycle C that traverses the terminals in $T \cup \{o\}$ in some order $t_0, t_1, \ldots, t_n$. The length of C is at most $2\|B\|$ by construction. Now consider the shortcut cycle $\tilde{C}$ in which we connect consecutive terminals t_i, t_{i+1} by the M-path $\pi(t_i, t_{i+1})$ as defined in Lemma 11; we set $t_{n+1} := t_0$. Clearly $\|\tilde{C}\| \leq \|C\|$. We partition $\tilde{C}$ it into two halves; $C_0 = \{\pi(t_{2i}, t_{2i+1}) \mid 0 \leq i \leq n/2\}$ and $C_1 = \{\pi(t_{2i+1}, t_{2i+2}) \mid 0 \leq i \leq n/2 - 1\}$. For at least one of the two halves, say C_0, we have $\|C_0\| \leq \|B\|$.

We use C_0 as a partial solution and recursively M-connect the points in the set $T' := \{\min(t_{2i}, t_{2i+1}) \mid 0 \leq i \leq n/2\}$, which lie in C_0 (see Lemma 11), to the origin by an arboresence A'. Lemma 11 implies that the resulting network $A = C_0 \cup A'$ is in fact a feasible RSA solution. The length of A is at most $\|C_0\| + \|A'\| \leq \|B\| + \|A'\|$. Note that $|T'| \leq (|T| + 1)/2$.

To summarize, we have described a procedure that, given the rectilinear cycle C traversing terminal set $T \cup \{o\}$, computes a shortcut cycle $\tilde{C}$, its shorter half C_0, and a new point set T' that still has to be M-connected to the origin. We refer to this procedure as *shortcutting*.

To compute the arboresence A', observe that $\tilde{C}$ is a rectilinear cycle that traverses the points in T'. Shortcutting yield a new cycle $\tilde{C}'$ of length at most $\|\tilde{C}\| \leq \|C\|$, a half C_0' no longer than $\|B\|$, which we add to the RSA network, and a new point set T'' of cardinality $|T''| \leq |T'|/2 \leq (|T|+1)/4$, which we recursively M-connect to the origin.

We repeat the shortcutting and recurse. Each iteration halves the number of new points, so the process terminates in $O(\log n)$ iterations with a single point t. Since $\min(o, p) = o$ for any point p (see proof of Lemma 11) and our original terminal set $T \cup \{o\}$ contained o, we must have that $t = o$. This shows that the computed solution is feasible. As each iteration adds length at most $\|B\|$, we have $\|A\| \leq \lceil \log_2 n \rceil \cdot \|B\|$.

F Running Time Analysis

For Theorem 1. We analyze the running time of the algorithm for $d = 2$ from Section 2.

Let $T(n)$ denote the running time of our main algorithm on an instance R with n terminal pairs. The main algorithm can find the median of the x-coordinates of R in $O(n)$ time. After partitioning R into the three instances $R_{\text{left}}, R_{\text{mid}}, R_{\text{right}}$, we require time $2T(n/2)$ to recursively solve R_{right} and R_{left}. This follows as $|R_{\text{left}}|, |R_{\text{right}}|$ are each at most $|R|/2$. Let $T_x(n)$ denote the running time of our x-separated algorithm. As $|R_{\text{mid}}| \leq |R|$, it is solved in time $T_x(n)$. Thus we have

$$T(n) = 2T(n/2) + T_x(n) + O(n) \tag{1}$$

Let $T_{xy}(n)$ denote the running time of our algorithm for xy-separated instances. As our x-separated algorithm works analogously to our main algorithm, we simply replace, in the argument above, R_{left}, R_{right}, and $T_x(n)$ by R_{top}, R_{bottom}, and $T_{xy}(n)$, respectively. This yields

$$T_x(n) = 2T_x(n/2) + T_{xy}(n) + O(n). \tag{2}$$

The running time for our xy-separated algorithm is determined by the running time of the 2-approximation algorithm for RSA. Thus, $T_{xy}(n) = O(n \log n)$ [9]. Solving recurrences 1 and 2, we get $T(n) = O(n \log^3 n)$.

For Theorem 3. Now we analyze the running time of the algorithm for $d > 2$ in Section 4.

Given an instance R of 0-separated d-dimensional GMMN, the algorithm uses d recursive procedures to subdivide the problem into d-separated instances. For $j \in \{0, \ldots, d-1\}$, let $T_j(n)$ denote the running time of the j-th recursive procedure. The j-th recursive procedure takes a j-separated instance R as input and partitions it into two j-separated instances, each of size at most $|R|/2$, and one $(j+1)$-separated instance of size at most $|R|$. The partitioning requires $O(n)$ steps for finding the median of the j-th coordinate value of terminals in R. The two j-separated instances are solved recursively

and the $(j+1)$-separated instance is solved with the $(j+1)$-th recursive procedure. Let $T_d(n)$ denote the running time to solve a d-separated instance.

In Section E we approximated such instances by applying a PTAS for the rectilinear Steiner tree problem in any fixed dimension d [2]. Though the probabilistic running time is nearly linear, the deterministic running time is in $O(n^{O(d)})$. We can improve our running time by computing a *rectilinear minimum spanning tree* in $O(n^2 \log n)$ time [11]. Observe that such a tree is always a 2-approximation of a rectilinear minimum Steiner tree [10]. This worsens our approximation ratio for d-separated instances by only a constant factor and doesn't change its order of magnitude. Thus we have

$$\begin{aligned} T_d(n) &= O(n^2 \log n), \\ T_j(n) &= 2T_j(n/2) + T_{j+1}(n) \qquad \text{for } j \in \{0, \ldots, d-1\}. \end{aligned}$$

The running time of our overall algorithm is given by $T_0(n)$. Solving the recurrences above yields $T_0(n) = O(n^2 \log^{d+1} n)$.

For Theorem 2. Now we analyze the running time of the improved algorithm of Section 3. Since the main algorithm was unchanged, recurrence 1 for $T(n)$ still applies. We now update recurrence 2 for $T_x(n)$, to reflect the improved algorithm. Stabbing x-separated instances can be done with a sweep-line algorithm in $O(n \log n)$ time. The PTAS for RSA requires time $O(n^{1/\varepsilon} \log n)$ for any ε with $0 < \varepsilon \leq 1$. Hence, we have that

$$T_x(n) = O(n^{1/\varepsilon} \log n) \tag{3}$$

Solving recurrences 1 and 3 yields a running time of $T(n) = O(n^{1/\varepsilon} \log^2 n)$ for the improved algorithm.

G Tightness of Analysis

In this section, we provide examples showing that the way we analyzed our algorithms in Sections 2, 4, and 3 is essentially tight.

Observation 1 *There are infinitely many instances where the $O(\log n)$-approximation algorithm for 2D-GMMN described in Section 3 has approximation performance $\Omega(\log n)$.*

Proof. We recursively define an arrangement $A(n)$ of n rectangles each of which represents a terminal pair: the lower left and upper right corner of the rectangle. By $\alpha \cdot A(n)$ we denote the arrangement $A(n)$ but uniformly scaled in both dimensions so that it fits into an $\alpha \times \alpha$ square. Let $\varepsilon > 0$ be a sufficiently small number.

The arrangement $A(0)$ is empty. The arrangement $A(n)$ consists of a unit square S_n whose upper right vertex is the origin. We add the arrangement $A_{\text{right}} := \varepsilon \cdot A((n-1)/2)$ and place it in the first quadrant at distance ε to the origin. Finally, we add the arrangement $A_{\text{left}} := (1-\varepsilon) \cdot A((n-1)/2)$ inside the square S_n so that it does not touch the boundary of S_n. See Fig. 5 for an illustration.

Let $\rho(n)$ denote the cost produced by our algorithm when applied to $A(n)$. Observe that our algorithm partitions $A(n)$ into subinstances $R_{\text{left}} = A_{\text{left}}$, $R_{\text{mid}} = \{S_n\}$, and

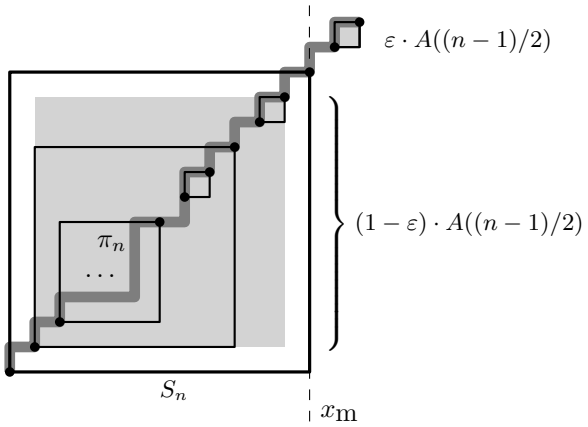


Fig. 5: Recursive construction of the arrangement $A(n)$. The gray M-path π_n shows an optimum solution. The dashed vertical line marks where the algorithm separates.

$R_{\text{right}} = A_{\text{right}}$. Solving the x-separated instance R_{mid} by our stabbing subroutine costs 1. Let $\rho(n)$ be the cost of the solution to $A(n)$ that our algorithm computes. Recursively solving R_{left} costs $(1-\varepsilon) \cdot \rho((n-1)/2)$. Recursively solving R_{right} costs $\varepsilon \cdot \rho((n-1)/2)$. Hence, the cost of the solution of our algorithm is $\rho(n) \geq 1 + \rho((n-1)/2)$. This resolves to $\rho(n) = \Omega(\log n)$.

Finally, observe that the optimum solution is a single M-path π_n of length $1 + 2\varepsilon$ going from the third to the first quadrant through the origin, see Fig. 5.

We now show that also the analysis of the of the $O(\log^2 n)$-approximation algorithm of Section 2 is tight. An important feature of the example is that it generalizes to higher dimensions, where it yields a deviation of the algorithm of Section 4 from the optimum by a factor $\Omega(\log^d n)$. This shows that the algorithm from Section 4 actually requires d in the exponent and that also this analysis is tight up to one log-factor.

Observation 2 *There are infinitely many instances where the $O(\log^2 n)$-approximation algorithm for 2D-GMMN described in Section 2 has approximation performance $\Omega(\log^2 n)$.*

Proof. Our example consists of blue and green structures, each defined recursively. The green structure $G(1)$ is empty. To create $G(m)$, we start with an outer unit square S, and place the structure $\varepsilon G(m/2)$ inside S into the top right corner and the structure $(1-\varepsilon)G(m/2)$ inside S into the bottom right corner such that a vertical line passes through S, $\varepsilon G(m/2)$, and $(1-\varepsilon)G(m/2)$.

Now we describe the blue structure $B(m,n)$. $B(m,1)$ is empty. To create $B(m,n)$ we start with a unit square S'. We place the structure $(1-\varepsilon)G(m)$ into the bottom left corner inside S' and the structure $\varepsilon^n B(m,n/2)$ into the top right corner inside S'. Finally, we place the structure $(1-\varepsilon)^n B(m,n/2)$ into the bottom left corner of $G(2)$, that is, the innermost large rectangle of the structure $(1-\varepsilon)G(m)$. No vertical line

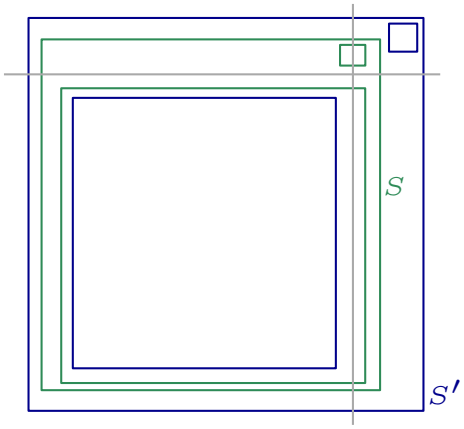


Fig. 6: Tight example for the algorithm of Section 2.

passes through S', $(1-\varepsilon)^n B(m, n/2)$, or $\varepsilon^n B(m, n/2)$. The structure $B(n, n)$ (see Fig. 6) represents our tight example.

Our algorithm partitions $B(m, n)$ by placing the vertical line as shown in Fig. 6. This follows as the smaller top-right blue structure ($\varepsilon^n B(m, n/2)$) inside $B(m, n)$ has the same number of terminals as the larger blue structure ($(1-\varepsilon)^n B(m, n/2)$) inside $B(m, n)$. As for the green structure inside $B(m, n)$, that is, $((1-\varepsilon)G(m))$, each green rectangle has one terminal on the right and one terminal on the left of the vertical line. The same holds for S'. Thus we get the sub-instances $R_{\text{left}} = (1-\varepsilon)^n B(m, n/2)$, $R_{\text{mid}} = (1-\varepsilon)G(m) \cup S'$, and $R_{\text{right}} = \varepsilon^n B(m, n/2)$.

Let $b(m, n)$ be the cost of the solution to $B(m, n)$, and let $g(m)$ be the cost of the solution of $G(m)$ produced by our algorithm. Thus, we have $b(m, n) = 2 + (1-\varepsilon)g(m) + (1-\varepsilon)^n b(m, n/2) + (\varepsilon)^n b(m, n/2) \approx g(m) + b(m, n/2)$.

Observe that our algorithm solves the x-separated instance R_{mid} by placing the horizontal line as shown in Fig. 6. This follows as the smaller green structure inside R_{mid} contains the same number of terminals as the larger green structure, one terminal of the square S' of $G(m)$ is above the horizontal line, and one is below the horizontal line. Thus the sub-instances will be the xy-separated instance R'_{mid} consisting of S', $R_{\text{top}} = \varepsilon(1-\varepsilon)G(m/2)$ and $R_{\text{bottom}} = (1-\varepsilon)^2 G(m/2)$. As S' is a square of size $1-\varepsilon$, its cost is $1-\varepsilon$. Hence, the cost of the solution of our algorithm on R_{mid} is $g(m) \geq (1-\varepsilon) + (1-\varepsilon)^2 g(m/2) + \varepsilon(1-\varepsilon)g(m/2) = (1-\varepsilon)(1+g(m/2))$, which resolves to $g(m) = O(\log m)$. Plugging the value of $g(m)$ into the equation for $b(m, n)$ above, we get $b(m, n) = (\log m)(\log n)$. Setting $m = n$ shows that $b(n, n) = \log^2 n$.

An optimal 2D-GMMN solution to our example instance has total length approximately 2 since all the big squares are nested one inside the other. (We can ignore the cost of the Manhattan network inside all the small squares.)

References

1. B. Aronov, E. Ezra, and M. Sharir. Small-size ε-nets for axis-parallel rectangles and boxes. In *Proc. 41st Annu. ACM Symp. Theory Comput. (STOC'09)*, pages 639–648, 2009.
2. S. Arora. Nearly linear time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 38th Annu. IEEE Symp. Foundat. Comput. Sci. (FOCS'97)*, pages 554–563, 1997.
3. S. Arora. Approximation schemes for NP-hard geometric optimization problems: A survey. *Math. Program.*, 97(1–2):43–69, 2003.
4. M. Bateni and M. Hajiaghayi. Euclidean prize-collecting steiner forest. In *LATIN*, pages 503–514, 2010.
5. G. Borradaile, P. N. Klein, and C. Mathieu. A polynomial-time approximation scheme for Euclidean steiner forest. In *Proc. 49th Annu. IEEE Symp. Found. Comput. Sci. (FOCS'08)*, pages 115–124, 2008. Corrected Version available at http://arxiv.org/abs/1302.7270.
6. H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete Comput. Geom.*, 14(1):463–479, 1995.
7. V. Chepoi, K. Nouioua, and Y. Vaxès. A rounding algorithm for approximating minimum Manhattan networks. *Theor. Comput. Sci.*, 390(1):56–69, 2008.
8. K. Nouioua. *Enveloppes de Pareto et Réseaux de Manhattan: Caractérisations et Algorithmes*. PhD thesis, Université de la Méditerranée, 2005. Available at http://www.lif-sud.univ-mrs.fr/~karim/download/THESE_NOUIOUA.pdf.
9. S. Rao, P. Sadayappan, F. Hwang, and P. Shor. The rectilinear Steiner arborescence problem. *Algorithmica*, 7:277–288, 1992.
10. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.
11. A. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM J. Comput.*, 11(4):721–736, 1982.