

A Simple 3-Approximation of Minimum Manhattan Networks

Bernhard Fuchs and Anna Schulze
Zentrum für Angewandte Informatik Köln,
Weyertal 80, 50931 Köln, Germany
bfuchs@zpr.uni-koeln.de and schulze@zpr.uni-koeln.de

July 29, 2008

Abstract

Given a set P of n points in the plane, a *Manhattan network* of P is a network that contains a rectilinear shortest path between every pair of points of P . A *minimum Manhattan network* of P is a Manhattan network of minimum total length.

It is unknown whether it is NP-hard to construct a minimum Manhattan network. The best approximations published so far are a combinatorial 3-approximation algorithm in time $O(n \log n)$, and an LP-based 2-approximation algorithm. We present a new combinatorial 3-approximation for this problem in time $O(n \log n)$. Both our algorithm and its analysis are considerably simpler than the prior 3-approximation.

Keywords: minimum Manhattan networks, approximation algorithms, rectilinear routing, VLSI design, 1-spanner

1 Introduction

Connecting a given set of points with minimum total length is a key problem in VLSI design. In the routing process a set of components, e. g., transistors or gates, have to be connected by wires. The wires are allowed to run in two perpendicular directions. The wire length significantly affects the power consumption and the time to spread the signal across the chip. Minimum Steiner trees minimize the total wire length. Manhattan networks impose an additional constraint. In contrast to Steiner trees they must contain a *shortest* path between each pair of points. This reflects the requirement to transmit signals between pairs of components on the chip fast. Manhattan networks are also defined in the *rectilinear metric* where one is allowed to use only horizontal and vertical lines. A *minimum Manhattan network* is a Manhattan network of minimum total length. See Figure 1 (a) and (b) for an example.

Another application of Manhattan networks is described by Lam et al. [PLA03], who use a variant of the Manhattan network problem to align gene sequences. They ask only for certain node pairs to be connected by shortest paths and solve the problem by a modification of an algorithm of Gudmundsson et al. [GLN01].

Manhattan networks fit in the concept of spanners. Given a set P of points in the plane, a given metric, and a number $t \in \mathbb{R}$ with $t \geq 1$, a network is a t -spanner for P under the given metric, if for each pair of points $p, q \in P$, there exists a (p, q) -path in the network of length at most t times the distance between p and q under the appropriate norm. A Manhattan network is a 1-spanner in the rectilinear metric. Spanners are commonly known and first introduced for the Euclidean metric by Chew [Che89]. They are studied extensively, see for instance the survey

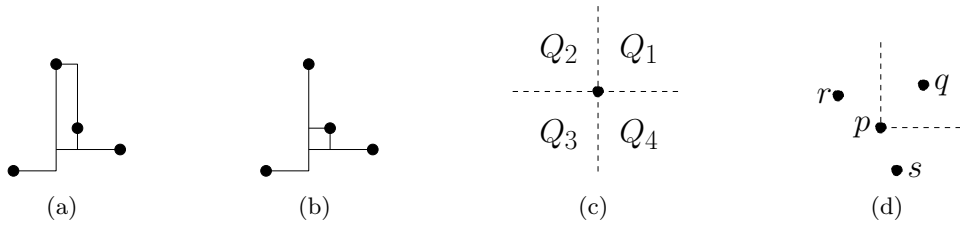


Figure 1: (a) A Manhattan network. (b) A minimum Manhattan network. (c) The quadrants of p . (d) Global and local neighborhood.

of Eppstein [Epp00] or the book of Narasimhan and Smid [NS07]. Note that the Euclidean 1-spanner is the trivial complete graph.

Up to now it is not known whether the minimum Manhattan network problem is NP-hard. Furthermore, it is not known whether a polynomial time approximation scheme exists. Gudmundsson et al. [GLN01] introduced an 8-approximation with running time $O(n \log n)$ and a 4-approximation running in time $O(n^3)$. Kato et al. [KIA02] proposed a 2-approximation with running time $O(n^3)$, however the proof of the correctness seems to be incomplete [BWWS06]. Chepoi et al. [CNV08] presented a 2-approximation based on LP-rounding. Their LP consists of $O(n^3)$ variables and constraints. In his PhD thesis Nouioua [Nou05] gave a 2-approximation that runs in $O(n \log n)$. Based on the primal-dual method, in contrast to the approach of Chepoi et al. the algorithm avoids to solve an LP. Unfortunately, the thesis is written in French. Benkert et al. [BWWS06] gave a 3-approximation with running time $O(n \log n)$. Seibert and Unger [SU05] presented an approximation algorithm and claimed that it yields a 1.5-approximation. As remarked by Chepoi et al. [CNV08] both the description of the algorithm and the performance guarantee are somewhat incomplete and not fully understandable. We show by a counterexample that an important intermediate step is incorrect, see Section 3.

We present a 3-approximation for minimum Manhattan networks with a running time of $O(n \log n)$. Our algorithm and in some parts also the analysis is similar to the one of Benkert et al. [BWWS06]. The plane is partitioned into regions belonging to two disjoint areas. In the first phase of the algorithm line segments in the first area are included and in the second phase in the second area. For each of these two areas the approximation ratio is examined separately. The main difference to the algorithm of Benkert et al. [BWWS06] lies in the first phase. Our algorithm selects line segments in a very simple way such that both the algorithm as well as the analysis are quite easy in contrast to the approach of Benkert et al. [BWWS06]. We will discuss similarities and differences below.

2 Definitions

We denote by p_x the x - and by p_y the y -coordinates of a point p . Each pair of points p and q spans a unique closed axis-parallel rectangle $R(p, q)$ with p and q as corners. Call $R(p, q)$ *critical* if it does not contain any other point of P . This term was first introduced by Seibert and Unger [SU05]. To get shortest paths it suffices to consider critical rectangles: If a rectangle $R(p, q)$ contains a further point $r \in P$ then it suffices to consider the two rectangles $R(p, r)$ and $R(r, q)$. Obviously a minimum Manhattan networks needs to contain line segments of length at least $|p_x - q_x|$ and of length $|p_y - q_y|$ in the corresponding dimensions for any rectangle $R(p, q)$. For a point $p \in \mathbb{R}^2$ we denote by $Q_1(p), \dots, Q_4(p)$ the four open quadrants of p . See also Figure 1 (c).

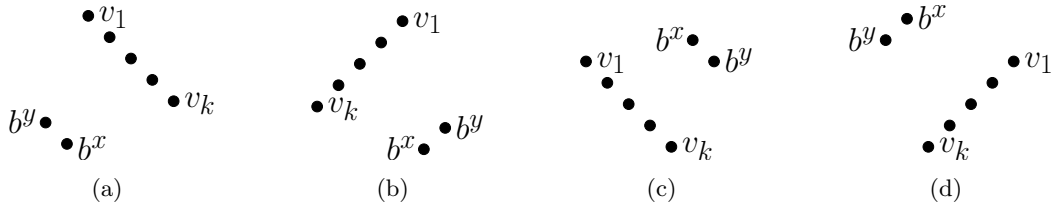


Figure 2: The four different cases of staircases.

We call two points $p, q \in P$ (w. l. o. g. $p_x \leq q_x$ and $p_y \leq q_y$) *x-neighbored* if there is no further point r with $p_x < r_x < q_x$ or $(r_x = p_x \vee r_x = q_x) \wedge (p_y < r_y < q_y)$ and *y-neighbored* if there is no further point r with $p_y < r_y < q_y$ or $(r_y = p_y \vee r_y = q_y) \wedge (p_x < r_x < q_x)$. We sometimes consider local neighborhood. More precisely, for a point $p \in P$ we consider the *x-* or *y-*neighbored point in one of its quadrants. See for example Figure 1 (d). The *x-*neighbored point of p in the first quadrant of p is the point q , whereas the globally *x-*neighbored points of p are the points r and s . Nevertheless, if not stated otherwise we consider global neighborhood. The area defined by *x-* or *y-*neighbored points plays an important role both in our algorithm and for the analysis of the approximation ratio.

Definition 1. The neighbored point area \mathcal{N} of a point set $P \subseteq \mathbb{R}^2$ is the union of all rectangles defined by neighbored points. That is,

$$\mathcal{N} = \cup_{p,q \in P} \text{x- or y-neighbored} R(p, q).$$

Almost all approximation algorithms for minimum Manhattan networks use staircases, but the definition of a staircase is not standardized. To get a clearer definition we define only one of four symmetric cases of a staircase shown in Figure 2. We define the staircase type as shown in Figure 2 (a).

Definition 2. For each point $p \in P$ the *x-base point* b^x is the *x-neighbored point* in $Q_3(p)$. The *y-base point* b^y is the *y-neighbored point* in $Q_3(p)$. Two points belong to the same staircase if they have the same base points.

We also consider sequences with only one point as a staircase sequence.

To make the definition of a staircase clear, we first name two important properties of staircases which clarify their structure.

Observation 3. Each sequence point forms a critical rectangle with each base point.

Proof. Assume to the contrary that a sequence point does not form a critical rectangle with one of its base points. That is, the rectangle contains at least one further point. Thus, the base point is neither *x-* nor *y-*neighbored to the sequence point contradicting the definition of a staircase. \square

In the following we assume the sequence points (v_1, \dots, v_k) sorted by increasing *x*-coordinate. The next remark displays the typical structure of a staircase as depicted in Figure 2 (a).

Remark 4. The set of points (v_1, \dots, v_k) with the same set of base points form a sequence monotone increasing in *x-* and monotone decreasing in *y-*direction.

Our definition of a staircase implies that the areas $\mathcal{A}_1, \mathcal{A}_2$ and \mathcal{A}_3 drawn in Figure 3 (a) do not contain a point of P . Assume to the contrary that a point p lies in the area \mathcal{A}_1 . Then for

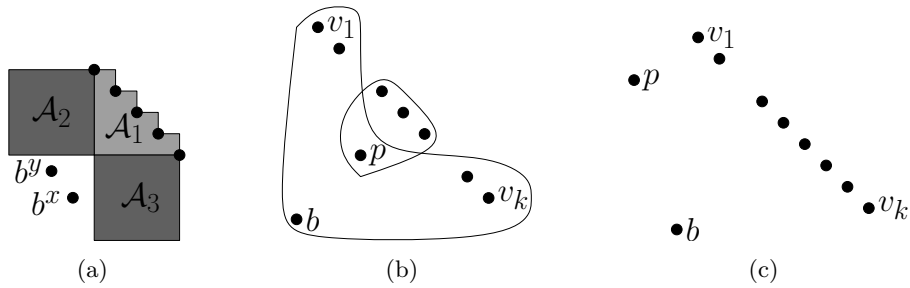


Figure 3: Splitting staircases.

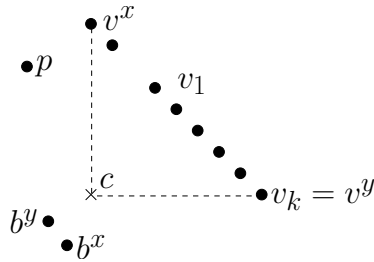


Figure 4: The definition of a cross point.

at least one of the sequence points v_i the x - or y -neighborhood point in $Q_3(v_i)$ is p and not as required b^x or b^y . The staircase would split into at least two staircases (see Figure 3 (b)). If one of the areas \mathcal{A}_2 or \mathcal{A}_3 contains a point p , then the staircase would split into at least two staircase, too (see Figure 3 (c)).

We denote the points v_1 and v_k of a staircase sequence (v_1, \dots, v_k) as the *outer points* of the sequence. Let b^x and b^y be the base points of a staircase with staircase sequence (v_1, \dots, v_k) . Let v^x be the x -neighborhood point of b^x in $Q_1(b^x)$ and let v^y be the y -neighborhood point of b^y in $Q_1(b^y)$. Note that v_1 and v^x and also v_k and v^y can be different. We define an auxiliary point c denoted as *cross point* by $c = (v^x, v^y)$. See Figure 4.

Our algorithm partitions the global Manhattan network problem into disjoint local Manhattan network problems for staircases by inserting line segments which separate the staircases of each other. We construct a boundary for each staircase which is defined as follows:

Definition 5. A staircase boundary for a sequence (v_1, \dots, v_k) of a staircase with base points b^x and b^y is defined in the following way: For any consecutive sequence points v_i and v_{i+1} , $1 \leq i \leq k-1$, the staircase boundary contains exactly one shortest path between the two points. Furthermore, the staircase boundary contains exactly one shortest path between v_1 and b^x and between v_k and b^y .

The boundary of a staircase is not unique. See Figure 5 for different examples of staircase boundaries for the same point set.

As required later, we want to define a variant of a staircase boundary which we call *extended staircase boundary*. The only difference in the definition for an extended staircase boundary is that we do not require a shortest path between the outer points and the base points. Nevertheless, there has to be a path lying in a certain area. To define this area we first expand the base rectangles calling them *extended base rectangles*.

Definition 6. Let (v_1, \dots, v_k) be a staircase sequence with base points b^x and b^y . Let r^y be the y -neighborhood point in $Q_2(v_1)$. If there exist no points in $Q_2(v_1)$ then let $r^y = v_1$. Let r^x

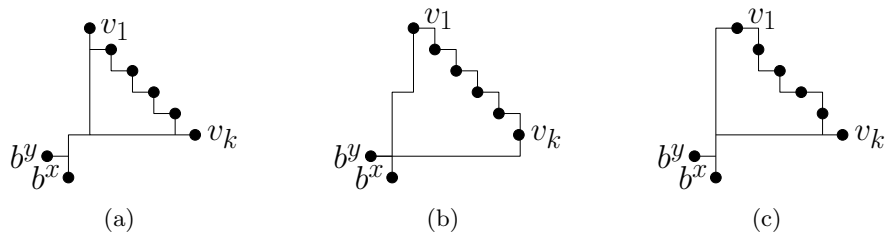


Figure 5: Different staircase boundaries for the same staircase

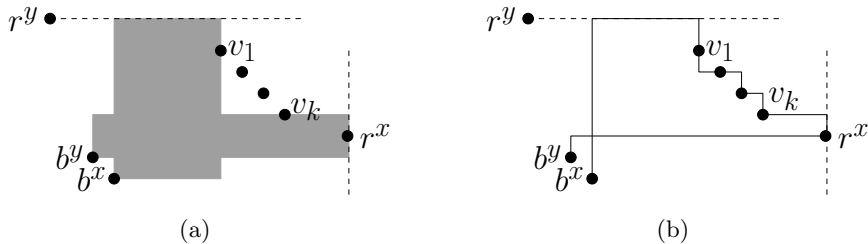


Figure 6: (a) Extended base rectangles. (b) Extended staircase boundary

be the x -neighborhood point in $Q_4(v_k)$. If there exist no points in $Q_4(v_k)$ then let $r^x = v_k$. The extended x -base rectangle is the rectangle $R((v_{1_x}, r_y^y), b^x)$ defined by the point (v_{1_x}, r_y^y) and the x -base point b^x . The extended y -base rectangle is the rectangle $R((r_x^x, v_{k_y}), b^y)$ defined by the y -base point b^y and the point (r_x^x, v_{k_y}) .

See Figure 6 (a) for an example of of the extended base rectangles. Now, we can define the extended staircase boundary.

Definition 7. An extended staircase boundary for a sequence (v_1, \dots, v_k) of a staircase with base points b^x and b^y is a set of line segments defined in the following way: For any consecutive sequence points v_i and v_{i+1} , $1 \leq i \leq k - 1$, the staircase boundary contains exactly one shortest (v_i, v_{i+1}) -path. Furthermore, the staircase boundary contains exactly one (v_1, b^x) -path inside the extended x -base rectangle and one (v_k, b^y) -path inside the extended y -base rectangle.

See Figure 6 (b) for an example of an extended staircase boundary.

Since the region surrounded by the staircase boundary plays an important role, we want to name it by the following definition:

Definition 8. Given a staircase S and a boundary B . The (extended) staircase area of S w. l. o. g. B is the interior of B .

It is essential for our algorithm that after we assigned an (extended) staircase boundary for each staircase it suffices to compute Manhattan networks for the staircases to achieve a Manhattan network for the complete instance. Note, that after we computed an (extended) staircase boundary for each staircase, each Manhattan network for a staircase lies completely inside the staircase area and that each area is the staircase area of exact one staircase. Thus, we split the problem into a set of independent subproblems of Manhattan networks for staircases.

3 The Algorithm of Seibert and Unger [SU05]

In this section we point out that the proof of the approximation ratio of the algorithm presented by Seibert and Unger [SU05] is not correct. For this we name two different points of

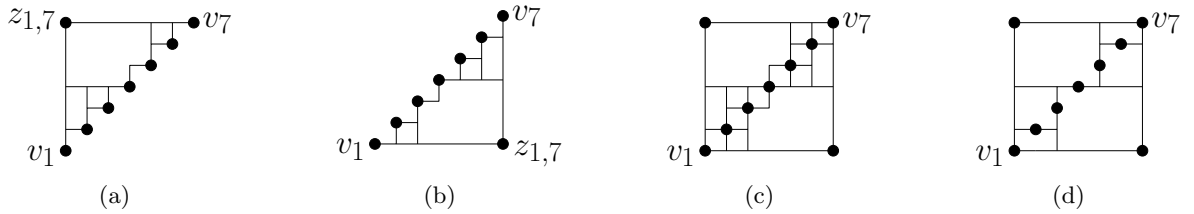


Figure 7: (a) and (b) Two staircases with the same sequence and the independently computed solutions of the algorithm of Seibert and Unger. (c) The composed solution. (d) A minimum Manhattan network for the same point set.

their analysis, at which it is incorrect. They first compute two sets of vertical and horizontal line segments by two plane sweeps. The sweeps fix line segments for neighbored points such that for two neighbored points p and q there is at most one line segment of $\partial R(p, q)$ in the respective dimension fixed. A minimum Manhattan network also contains the lengths of these line segments, thus the two sets together contain segments of total length at most the length of a minimum Manhattan network. They choose the cheaper of the two sets to use at most half the length of the optimal solution. They claim that this chosen set separates all staircases of each other and that they can independently compute Manhattan networks for the staircases. Since the chosen set of line segments of the first step do not contain a staircase boundary for each staircase, they also have to insert remaining line segments of the boundaries. They claim that this can be done independently of the other staircases (and staircase boundaries) and that this costs all in all at most the length of a minimum Manhattan network. This is the crux in their argumentation because the boundaries are not independent. They compute for each staircase a minimum Manhattan network containing also a boundary for the staircase with a dynamic program. For a staircase sequence (v_1, \dots, v_k) of points lying top-right which have to be connected to a point bottom-right of them, they select a help point $z_{1,k} = (v_{k_x}, v_{1_y})$ and compute a Manhattan network for (v_1, \dots, v_k) and $z_{1,k}$. In this step they do not consider computed boundaries of other staircases. They argue that solving all sub-problems of this type costs all in all not more than the optimal solution. But this is not necessarily true and it is not clear how to fix this issue. See Figure 7 for an example consisting of two staircases. First, the algorithm of Seibert and Unger computes for the two staircases independently minimum Manhattan networks including the boundary for each staircase (Figure 7 (a) and (b)). The resulting network consists of these two computed Manhattan networks for the staircases (Figure 7 (c)). Obviously, the optimal solution (Figure 7 (d)) is shorter than the solution composed of the two staircases computed separately by the algorithm of Seibert and Unger in contradiction to their claim. For this example the algorithm of Seibert and Unger inserts too many boundary segments. In the optimal solution the line segments of the boundary between the sequence points is shared by the two staircases (see Figure 7 (d)).

After computing Manhattan networks for staircases, they have to insert further line segments to establish remaining connections. They claim that this can be done without consuming (together with the previously computed Manhattan networks for staircases) more than the length of a minimum Manhattan network. See Figure 8. Assume the set of vertical line segments is cheaper than the horizontal ones. They first compute Manhattan networks for (v_1, \dots, v_4) and $z_{1,4}$ and for (v'_1, \dots, v'_4) and $z'_{1,4}$ independently. For the latter point set the Manhattan network includes also the line segment l'_h . Afterwards they have to connect $z_{1,4}$ to the point p by inserting the horizontal line segment l_h . A minimum Manhattan network for this instance contains only the line segment l_h and not l'_h in contradiction to their claim.

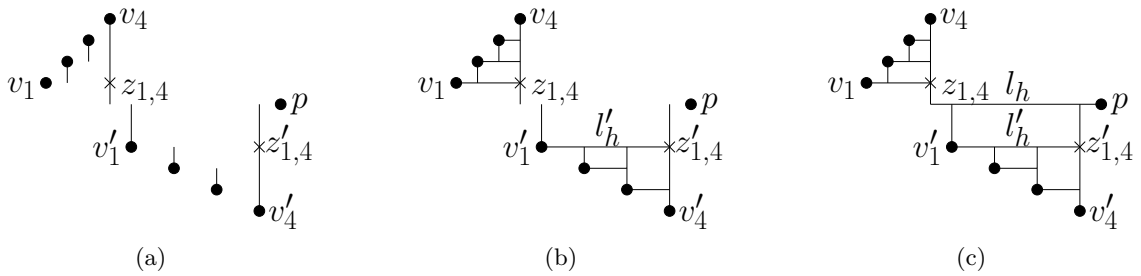


Figure 8: (a) The vertical line segment computed by the algorithm of Seibert and Unger. (b) The minimum Manhattan networks for the staircases. (c) The remaining connections.

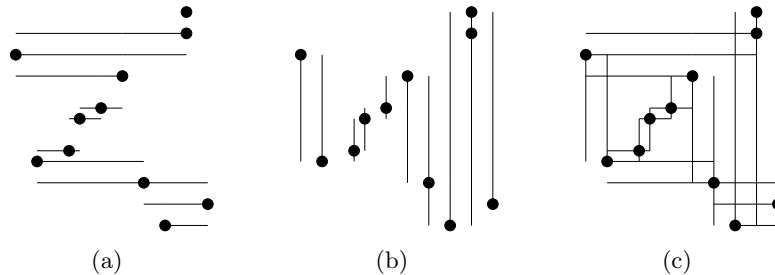


Figure 9: Line segments inserted by the sweep steps.

4 A 3-Approximation of Minimum Manhattan Networks

Our 3-approximation algorithm for minimum Manhattan networks proceeds in two phases. In the first phase we compute a basic set of line segments in each of the two dimensions. These segments ensure that only sequence points for staircases remain unconnected to the appropriate base points and that the segments constitute an extended staircase boundary with smallest area relative to the neighbored point area \mathcal{N} . More precisely, we do not need to insert line segments inside \mathcal{N} in Phase II. Furthermore, we do need line segments of a minimum Manhattan network inside \mathcal{N} to justify segments inserted by Phase II because each staircase area \mathcal{A}_{app} computed in Phase I is contained in a staircase area \mathcal{A}_{opt} inside \mathcal{N} of a minimum Manhattan network ($\mathcal{A}_{app} \subseteq \mathcal{A}_{opt}$). In the second phase we compute Manhattan networks for the staircases. The general approach to partition the problem in a set of Manhattan network problems for staircases is used by all combinatorial approaches (see for example [BWWS06], [GLN01] or [KIA02]). We now describe our approach in more detail. We first examine the points from bottom to top. For two y -neighbored points p and q considered by this sweep we insert the horizontal boundary segments of the rectangle $R(p, q)$ into our network. Afterwards we perform an analogous sweep from left to right. Similarly, for two x -neighbored points p and q considered by the sweep we insert the vertical boundary segments of the rectangle $R(p, q)$. See Figure 9 (a) to (c) for an example of such a sweep in the two directions. After sorting the points which can be done with running time $O(n \log n)$, the sweeps can be performed in time $O(n)$. After these two sweeps the only remaining part to get shortest paths between all point pairs are shortest paths between sequence points and their appropriate cross point. Furthermore, the up to now identified line segments contain a boundary for each staircase. Afterwards, we identify all staircases. For this purpose we assign to each point its two base points and then consider the points with the same base points as a staircase. This can be done in time $O(n)$. In the last step we compute for each staircase (given a staircase boundary) a Manhattan network. As stated in Section 5 we can compute a 2-approximation for a staircase in time $O(n \log n)$ for n input points. Altogether, we

achieve a Manhattan network for the input points. See Algorithm 1 for a detailed description.

Algorithm 1 MANHATTAN NETWORK APPROXIMATION

Require: A set $P \subseteq \mathbb{R}^2$ of points.

Phase I:

- 1: Set $CR = \emptyset$.
- 2: Sweep over the points of P bottom-up. Let p be the currently considered point and q be the previously processed point. Add to CR the horizontal line segments of $\partial R(p, q)$.
- 3: Sweep over the points of P from left to right. Let p be the currently considered point and q be the previously processed point. Add to CR the vertical line segments of $\partial R(p, q)$.

Phase II:

- 4: Set $MN = \emptyset$.
 - 5: **for** each Staircase S **do**
 - 6: Compute with Algorithm 2 a Manhattan network MS of the staircase S with the staircase boundary computed in Phase I.
 - 7: Set $MN = MN \cup MS$.
 - 8: **return** $MN \cup CR$.
-

To analyze the algorithm we can interpret our strategy in such a way that we partition the plane into regions belonging to two disjoint areas. The first one is the neighbored point area \mathcal{N} , the second one the complement $\mathbb{R}^2 \setminus \mathcal{N}$. In the first phase we include line segments in the first area and in the second phase in the second area. For each of these two areas we examine the approximation ratio separately. This strategy for the analysis is similar to the one of Benkert et al. [BWWS06]. They also consider x - or y -neighbored points. Whereas we include for pairs of x -neighbored points p and q both horizontal line segments of $\partial R(p, q)$ into our network, they first include only one of the two segments. A symmetric statement holds for y -neighbored points. They also get a staircase boundary for each staircase. Unfortunately, since they include for two, w.l.o.g. x -neighbored, points p and q only one of the two vertical boundary segments of $R(p, q)$, these segment do not guarantee a partition into two disjoint areas for which they can bound the length of the inserted segments separately. Notice, that we get two disjoint areas since we insert both vertical boundary segments to maximize the area isolated in the first phase. They have to insert further line segments. This stepwise proceeding complicates the analysis and the algorithm in comparison to our approach.

To prove the correctness of Algorithm 1 we first show that neighboring pairs are connected by a shortest path after Phase I. We then show that for pairs not connected after Phase I it suffices to add shortest paths from sequence to cross points.

Lemma 9. *After Phase I of the MANHATTAN NETWORK APPROXIMATION algorithm two x - or y -neighbored points are connected by line segments of CR forming a shortest path.*

Proof. Let p and q be neighbored in x -direction. W.l.o.g. $p_x \leq q_x$ and $p_y \leq q_y$. If the points are neighbored also in y -direction then CR contains all line segments of $\partial R(p, q)$. Thus, assume p and q are not neighbored in y -direction. There exists a point r with $p_y < r_y < q_y$. W.l.o.g. let $r_x < p_x$ and r be the topmost one. See Figure 10. Thus, r is y -neighbored either to q or to another point s in $Q_4(q)$. Since r and s are y -neighbored, the horizontal segments of $\partial R(r, s)$ are inserted into CR . Together with the vertical segments of $\partial R(p, q)$ they constitute a shortest path between p and q . \square

After showing the general statement that neighbored points are connected, we now prove that the line segments added during the sweeps of steps 2 and 3 yield us an extended staircase

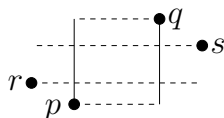


Figure 10: Proof of Lemma 9.

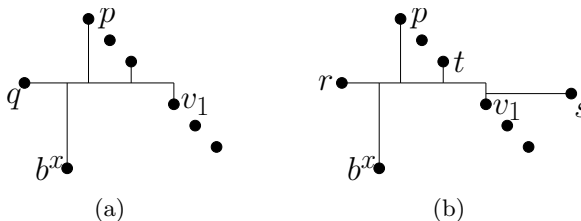


Figure 11: Proof of Lemma 10.

boundary for each staircase. That is, we get paths from the outer points to the base points inside the extended base rectangles (as we prove in Lemma 10) and shortest paths between consecutive sequence points (see Lemma 11).

Lemma 10. *After Phase I of the MANHATTAN NETWORK APPROXIMATION algorithm the outer points of a staircase sequence are connected to both base points by paths in the appropriate extended base rectangles. There exist shortest paths between the base points and the cross point.*

Proof. Let (v_1, \dots, v_k) be a staircase sequence with base points b^x and b^y . If b^x and v_1 are x -neighbored then they are connected by a shortest path by Lemma 9. Thus assume b^x and v_1 are not x -neighbored. Let v^x be the point x -neighbored to b^x on the right of b^x . See Figure 11. Since v^x is x -neighbored to b^x we know that there exists a shortest (v^x, b^x) -path. Consider the point q y -neighbored to v_1 above v_1 . We distinguish two cases. First assume q lies in $q_2(v_1)$. (Note q is the point r^y defined in Definition 6.) Since v_1 is the outer point of the staircase and v^x is x -neighbored to b^x the point q lies on the left of b^x . Again there exists a shortest (v^x, v_1) -path due to the neighborhood of q and v_1 . Together with the shortest (p, b^x) -path we get a (v_1, b^x) -path in the extended x -base rectangle. See Figure 11 (a).

Now assume the point q lies to the right of v_1 . Since v^x does not belong to the staircase sequence (v_1, \dots, v_k) there is a point above v_1 and to the left of b^x . Let r be the one with smallest y -coordinate. (Note r is the point r^y defined in Definition 6.) The point r is y -neighbored below itself to a point s on the right of v_1 . There exists a shortest (r, s) -path. Since b^x is not globally x -neighbored and v_1 is an outer point the point t x -neighbored to v_1 on the left of v_1 lies above r . Again there exists a shortest (v_1, t) -path. Together with the shortest (v^x, b^x) - and (r, s) -path we get a (v_1, b^x) -path inside the extended x -base rectangle. See Figure 11 (b).

In almost the same manner we can prove that we get a (v_k, b^y) -path inside the extended y -base rectangle. These two paths together establish also a (v_1, b^y) - and a (v_k, b^x) -path. Since the algorithm inserts the vertical line segments of $\partial R(b^x, v^x)$ and the horizontal of $\partial R(b^y, v^y)$ we get shortest paths between the base points and the cross point. \square

Lemma 11. *After Phase I of the MANHATTAN NETWORK APPROXIMATION algorithm two consecutive points v_i and v_{i+1} of a staircase sequence are connected by a shortest path.*

Proof. W.l.o.g. let v_i and v_{i+1} be part of a staircase as depicted in Figure 12 (a), let v_i being on the left of v_{i+1} . Assume v_i and v_{i+1} are not connected after Phase I. Thus, by Lemma 9 v_i

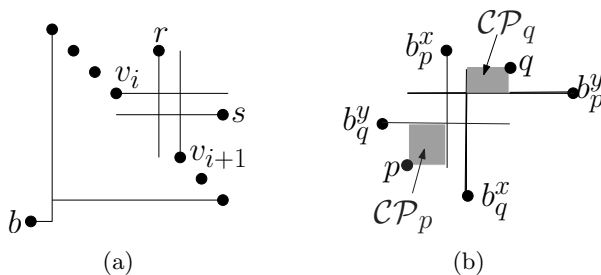


Figure 12: (a) Proof of Lemma 11. (b) Proof of Lemma 12.

and v_{i+1} are neither x - nor y -neighborhood. Since v_i and v_{i+1} belong to the same staircase and are consecutive in the sequence, the point r x -neighbor to v_{i+1} on the left is above v_i . See Figure 12 (a). The vertical segments of $\partial R(r, v_{i+1})$ are contained in CR . Again, since v_i and v_{i+1} belong to the same staircase and v_i and v_{i+1} are not y -neighborhood there exists a y -neighborhood point s of v_i below and to the right of v_{i+1} . The horizontal segments of $\partial R(s, v_i)$ are contained in CR . Thus, by these four line segments v_i and v_{i+1} are connected by a shortest path. \square

By Lemma 10 and 11 we get an extended staircase boundary for each staircase after Phase I. In the next lemma we point out that it suffices to compute Manhattan networks for the staircases to get a Manhattan network for the complete instance.

Lemma 12. *If after Phase I of the MANHATTAN NETWORK APPROXIMATION algorithm two points p and q forming a critical rectangle are not connected by a shortest path the only missing parts to obtain such a shortest path are connections between p and a cross point of a staircase p belongs to and between q and a cross point of a staircase q belongs to.*

Proof. If p and q x - or y -neighborhood then there exists a shortest path by Lemma 9. Thus, assume that p and q neither x - nor y -neighborhood.

We consider the two possible cases whether p and q belong to the same staircase or not. First, assume p and q belong to the same staircase. By Lemma 11 if p and q are consecutive points of the staircase sequence then they are connected by a shortest path. Thus, assume one point, w.l.o.g. p , is a base point of a staircase sequence q belongs to. To get a shortest path between p and q the only remaining part is a shortest path between the cross point and the point q . (Note that between p and the cross point is already a shortest path).

Next, assume p and q do not belong to the same staircase. W.l.o.g. let p and q be arranged as in Figure 12 (b). Let b_p^x and b_p^y be the two base points of p in the first quadrant of p and let b_q^x and b_q^y be the two base points of q in the third quadrant of q . The sweep inserts a vertical segment incident to b_p^x with end point on the same height or below p and a vertical segment incident to b_q^x with end point on the same height or above q . Similarly, the sweep inserts a horizontal segment incident to b_p^y with end point on the same width or on the left of p and a horizontal segment incident to b_q^y with end point on the same width or on the right of q . The cross point of the staircase p belongs to, lies in the area \mathcal{CP}_p marked in Figure 12 (b). Similarly, the cross point of the staircase q belongs to, lies in the area \mathcal{CP}_q .

By Lemma 10 there is a shortest path between the cross point and the appropriate base points and thus also between the two cross points. To achieve a shortest (p, q) -path it suffices to insert shortest paths between p and q to their cross points. This proves the Lemma. \square

Now we can prove that our algorithm computes a Manhattan network.

Theorem 13. *For a point set $P \subseteq \mathbb{R}^2$ the MANHATTAN NETWORK APPROXIMATION algorithm computes a Manhattan network.*

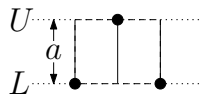


Figure 13: Proof of Theorem 14.

Proof. To prove that we get a Manhattan network we have to consider critical rectangles only. By Lemma 12 the only points constituting critical rectangles which are not connected after Phase I are sequence points missing a shortest path to an appropriate cross point. By Lemma 10 the outer points of a sequence are connected to the base points by paths through the cross point after Phase I of the algorithm. In step 6 we compute a Manhattan network for each staircase to add missing shortest paths between the sequence points and the cross points. Altogether, we obtain a Manhattan network for P . \square

Next, we show the approximation factor of the algorithm.

Theorem 14. *The MANHATTAN NETWORK APPROXIMATION algorithm computes a Manhattan network for P with total length at most 3 times the length of a minimum Manhattan network for P .*

Proof. To achieve the approximation ratio we have to account each length of a line segment of a minimum Manhattan network at most three times. Our strategy of the proof is to partition the plane into two parts and to compare the length of a minimum Manhattan network in each part separately. As mentioned earlier this strategy was also used by Benkert et al. [BWWS06]. First, we want to remind that we only need to consider shortest paths for point pairs constituting critical rectangles. The first area we consider is the neighbored point area \mathcal{N} formed by the union of all critical rectangles of x - or y -neighbored points. By Lemma 12 except for staircases all points forming critical rectangles are connected by line segments of CR . Consider the horizontal sweep of the algorithm in step 3. For each pair of neighbored points p and q we add the vertical edges of the rectangle $R(p, q)$. In the minimum Manhattan network the length of $|p_y - q_y|$ is required to connect the points. Thus, for this rectangle we insert at most twice the length of the required length. However, we must consider also neighbored critical rectangles having common boundaries as depicted in Figure 13. We add three line segments to CR while it suffices to include the middle of the three vertical line segments. Generally for k points alternating on the lines L and U our algorithm adds k line segments of length a while $\lfloor k/2 \rfloor$ suffice. The ratio $k/\lfloor k/2 \rfloor$ is largest for $k = 3$ and achieves the value 3. Thus, we add in Phase I line segments of length at most 3 times the length of segments constituting a minimum Manhattan network inside the neighbored point area \mathcal{N} .

For any two w.l.o.g. x -neighbored points p and q we know that there exists a minimum Manhattan network that adds vertical line segments on the boundary of $R(p, q)$ of length $|p_y - q_y|$. We add both boundary segments. Thus, it would not be reasonable to add further vertical segments inside $R(p, q)$. Therefore, we maximize the area separated by the sweep (this is the neighbored point area \mathcal{N}) and minimize the remaining area ($\mathbb{R}^2 \setminus \mathcal{N}$). The remaining area is the union of the staircase areas defined by the staircase boundaries given by line segments of Phase I. At this, each staircase area is accounted as an open area.

Let (v_1, \dots, v_n) be a staircase sequence with base points b^x and b^y and let \mathcal{A}_{app} be an the staircase area of the boundary computed by Algorithm 1. A minimum Manhattan network contains shortest (b^x, v_1) - and (b^y, v_n) -paths and shortest (v_i, v_{i+1}) -paths, $1 \leq i < n$, inside the neighbored point area \mathcal{N} constituting a staircase boundary B_{opt} inside \mathcal{N} . Let \mathcal{A}_{opt} the staircase area of B_{opt} . By the construction of the sweeps we get $\mathcal{A}_{app} \subseteq \mathcal{A}_{opt}$. Furthermore,

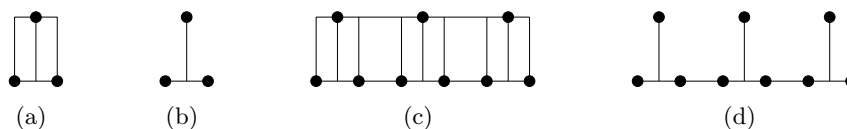


Figure 14: (a) and (c) A Manhattan network computed by our approximation. (b) and (d) A minimum Manhattan network.

all line segments of a minimum Manhattan network in the interior of the area \mathcal{A}_{opt} belong only to paths between sequence points and the base points. No such line segment contributes to a shortest path between two points already connected by line segments in CR . Thus, we can use the total length of a minimum Manhattan network inside \mathcal{A} to prove the approximation factor for staircases given the staircase boundaries. That is, we partitioned the problem in two parts which can be considered separately. As stated above for the first part we add at most three times the length of the line segments constituting a minimum Manhattan network in this area. By Theorem 16 we can compute a Manhattan network for staircases with approximation ratio two.

Altogether, our algorithm approximates minimum Manhattan networks with a ratio of three. \square

The core of the proof is that our Phase I yields a 3-approximation outside of staircases and minimizes the staircase areas. That is, we get a partitioning into two disjoint parts (defined by critical rectangles of neighbored points and the staircase areas) which can be examined separately. Thus, together with the standard 2-approximation for staircases we get a 3-approximation altogether.

Theorem 15. *The running time of the MANHATTAN NETWORK APPROXIMATION algorithm is $O(n \log n)$.*

Proof. First of all we must sort the points of P horizontally and vertically. This can be done in time $O(n \log n)$. The sweeps in steps 2 and 3 then can be performed in time $O(n)$. To find all staircases, we assign for each point its two base points. This can be done by a sweep in time $O(n)$. Afterwards, we look from the perspective of the base points and consider all points with the same base points. These points form the staircase sequence of the staircase. This can be done in running time $O(n)$. The running time to compute a 2-approximation for a Manhattan network of a staircase given the staircase boundary is $O(k \log k)$ for k sequence points by Theorem 17. Each sequence point can only belong to at most four different staircases, thus we get a total running time of $O(n \log n)$ to compute Manhattan networks for all staircases.

This yields the total running time for the algorithm of $O(n \log n)$. \square

See Figure 14 (a) and (b) for an example that the approximation ratio of our algorithm is tight. We use three vertical segments instead of one. If we consider to fix this by the observation that we do not need the outer line segments we see by the example in Figure 14 (c) and (d) that redundant line segments can also lie in the interior of the instance.

5 A 2-Approximation of Staircases

In this section we deal with the problem to compute Manhattan networks for staircases given the staircase boundary. The algorithm partitions recursively a staircase into two new staircases. This proceeding is also known as *thickest-first* partitioning and was analyzed to yield a

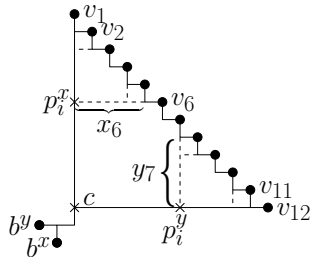


Figure 15: The definition of x_i and y_i .

2-approximation by Gudmundsson et al. [GLN01] or Benkert et al. [BWWS06]. Gudmundsson et al. [GLN01] prove that given a staircase boundary a rectangulation of the polygon defined by the boundary with minimum total edge length is a minimum Manhattan network for the points defining the staircase boundary. Lingas et al. show that a minimum rectangulation of a rectilinear polygon can be computed in time $O(n^4)$. They state, that for a special case of so-called *histograms* this can be computed in $O(n^3)$. They introduced a 2-approximation for rectangulations with running time $O(n \log n)$. We make other demands on the staircase boundary and thus we cannot use the algorithm of Gudmundsson et al. [GLN01] directly. Furthermore, we achieve the result by computing a Manhattan network directly without the detour to the rectangulation. The algorithm of Benkert et al. [BWWS06] to compute Manhattan networks for staircase boundaries has more analogies to our than the the one of Gudmondsson et al. . For the purpose of self containment and since both the definitions of staircases and the boundary of staircases are not standardized we specify the algorithm adapted to our conditions.

Given the staircase boundary, for a point v_i of the staircase sequence we denote by x_i the missing line segment of a shortest path to the vertical left boundary edge, by y_i we denote the missing line segment of a shortest path to the horizontal bottom boundary edge. See Figure 15 for an illustration. Note, that we do not count the length of the possibly used boundary edges between v_i and the left and right boundary, respectively. Let p_i^x be the intersection of x_i with the vertical left boundary edge and p_i^y the intersection of y_i with the horizontal bottom boundary edge.

Our algorithm partitions the staircase into two staircases for which networks are computed recursively. In the partitioning step two new edges are inserted, each of them being a new boundary edge of one of the two new staircases. We get as input an (extended) staircase boundary of a staircase with sequence (v_1, \dots, v_n) and base points b^x and b^y . We consider the segments x_i and y_i , $1 \leq i \leq n$ to connect a point v_i to the left and bottom boundary segment, respectively. We select the point v_i , $1 \leq i < n$, for which $|x_i| \leq |y_i|$ and $|x_{i+1}| \geq |y_{i+1}|$ holds and add the segments x_i and y_{i+1} to our network and compute recursively Manhattan networks for the staircases with sequence (v_1, \dots, v_i) and the one with sequence (v_{i+1}, \dots, v_n) . As illustrated in Figure 15 we insert edges x_6 and x_7 and get two new staircases with sequences (v_1, \dots, v_6) and (v_7, \dots, v_{12}) . See Algorithm 2 RECURSION FOR STAIRCASES for a detailed description. For this, assume the cross point lies on the origin.

Let (v_1, \dots, v_n) be a staircase sequence with base points b^x and b^y . Let \mathcal{A}_{app} be the staircase area of the staircase boundary given to Algorithm 2 RECURSION FOR STAIRCASES for the instance. A minimum Manhattan network contains shortest (b^x, v_1) - and (b^y, v_n) -paths and shortest (v_i, v_{i+1}) -paths, $1 \leq i < n$, inside the neighbored point area \mathcal{N} constituting a staircase boundary B_{opt} inside \mathcal{N} . Let \mathcal{A}_{opt} the staircase area of B_{opt} .

Theorem 16. *Algorithm 2 RECURSION FOR STAIRCASES computes a Manhattan network for a staircase with total length at most 2 times the length of a minimum Manhattan*

Algorithm 2 RECURSION FOR STAIRCASES

Require: A staircase sequence (v_1, \dots, v_n) , $n \geq 3$, with boundary.

- 1: Set $SC = \emptyset$.
 - 2: Let v_i, v_{i+1} be the unique pair of neighbors with $|x_i| \leq |y_i|$ and $|x_{i+1}| \geq |y_{i+1}|$.
 - 3: Set $SC = SC \cup \{x_i\} \cup \{y_{i+1}\}$.
 - 4: Let SC' be the staircase recursively computed for the staircase sequence (v_1, \dots, v_i) .
 - 5: Let SC'' be the staircase recursively computed for the staircase sequence (v_{i+1}, \dots, v_n) .
 - 6: **return** $SC \cup SC' \cup SC''$
-

network for it if $\mathcal{A}_{app} \subseteq \mathcal{A}_{opt}$ holds.

Proof. Due to the fact that we recursively call the algorithm and in each call, we connect two points of the sequence with to the base, we get a Manhattan network for the staircase.

We prove the ratio between the length of the solution of Algorithm 2 and the length of a minimum Manhattan network to be two by the use of an inductive argument over the number of sequence points. Assume we insert in the i -th step a segment x_k and a segment y_{k+1} . Let x_i^{opt} and y_i^{opt} , $1 \leq i \leq n$, the segments x_i and y_i for B_{opt} . We get $|x_i| \leq |x_i^{opt}|$ and $|y_i| \leq |y_i^{opt}|$ because $\mathcal{A}_{app} \subseteq \mathcal{A}_{opt}$ holds. If we insert x_i then it holds $|x_i| \leq |y_i|$ and therefore $|x_i| \leq \min\{|x_i^{opt}|, |y_i^{opt}|\}$. In an analogous manner it holds $|y_{i+1}| \leq \min\{|x_{i+1}^{opt}|, |y_{i+1}^{opt}|\}$ if we insert y_{i+1} . To connect v_i and v_{i+1} to the cross point a minimum Manhattan network needs at least the length of $\min\{|x_i^{opt}| + |y_{i+1}^{opt}|, |y_i^{opt}| + |x_{i+1}^{opt}|\}$. If the minimum is adopted for $|x_i^{opt}| + |y_{i+1}^{opt}|$ then our algorithm takes the right choice. Otherwise we get the following estimation:

$$\begin{aligned} |x_i| + |y_{i+1}| &\leq \min\{|x_i^{opt}|, |y_i^{opt}|\} + \min\{|x_{i+1}^{opt}|, |y_{i+1}^{opt}|\} \\ &\leq \min\{|x_{i+1}^{opt}|, |y_i^{opt}|\} + \min\{|x_{i+1}^{opt}|, |y_i^{opt}|\} \\ &\leq 2 \min\{|y_i^{opt}|, |x_{i+1}^{opt}|\} \end{aligned}$$

. So in step k we insert at most two times the required length in the minimum Manhattan network to connect v_i to the base. Furthermore, we split the staircase in two disjoint staircase with smaller number of sequence points. According to the induction hypotheses for these two staircases the approximation ratio holds. \square

Theorem 17. *The running time of the RECURSION FOR STAIRCASES algorithm is $O(n \log n)$.*

Proof. The only time consuming work to be done is step 2 which can be executed by binary search in time $O(\log n)$. The recursion have to be proceeded $O(n)$ times. Together we get the running time of the algorithm to be $O(n \log n)$. \square

6 Acknowledgement

The authors thank two anonymous referees for helpful remarks and pointing out reference [Nou05].

References

- [BWWS06] M. Benkert, A. Wolff, F. Widmann, and T. Shirabe, *The minimum Manhattan network problem: Approximations and exact solutions*, Computational Geometry: Theory and Applications **35** (2006), no. 3, 188–208.

- [Che89] L. P. Chew, *There are planar graphs almost as good as the complete graph*, Journal of Computer and System Sciences **39** (1989), no. 2, 205–219.
- [CNV08] V. Chepoi, K. Nouioua, and Y. Vaxès, *A rounding algorithm for approximating minimum Manhattan networks*, Theoretical Computer Science **390** (2008), no. 1, 56–96.
- [Epp00] D. Eppstein, *Spanning trees and spanners*, Handbook of Computational Geometry (Jörg-Rüdiger Sack and Jorge Urrutia, eds.), Elsevier, 2000, pp. 425–461.
- [GLN01] J. Gudmundsson, C. Levcopoulos, and G. Narasimhan, *Approximating a minimum Manhattan network*, Nordic Journal of Computing **8** (2001), no. 2, 219–232.
- [KIA02] R. Kato, K. Imai, and T. Asano, *An improved algorithm for the minimum Manhattan network problem*, Proceedings of the 13th International Symposium on Algorithms and Computations (ISAAC 2002), Lecture Notes in Computer Science, vol. 2518, Springer-Verlag, 2002, pp. 344–356.
- [Nou05] K. Nouioua, *Enveloppes de Pareto et Réseaux de Manhattan: Caractérisations et algorithmes*, Ph.D. thesis, Université de la Méditerranée, 2005.
- [NS07] G. Narasimhan and M. Smid, *Geometric spanner networks*, Cambridge University Press, New York, NY, USA, 2007.
- [PLA03] L. Pachter, F. Lam, and M. Alexandersson, *Picking alignments from (Steiner) trees*, Journal of Computational Biology **10(3-4)** (2003), 509–520.
- [SU05] S. Seibert and W. Unger, *A 1.5-approximation of the minimal Manhattan network problem*, Proceedings 16th International Symposium on Algorithms and Computation (ISAAC 2005), Lecture Notes in Computer Science, vol. 3827, Springer-Verlag, 2005, pp. 246–255.