Small Manhattan Networks and Algorithmic Applications for the Earth Mover's Distance

Joachim Gudmundsson* Oliver Klein† Christian Knauer† Michiel Smid‡

Abstract

Given a set S of n points in the plane, a Manhattan network on S is a (not necessarily planar) rectilinear network G with the property that for every pair of points in S the network G contains a path between them whose length is equal to the Manhattan distance between the points. A Manhattan network on S can be thought of as a graph $G = (V, E)$ where the vertex set V corresponds to the points of S and a set of Steiner points S'. The edges in E correspond to horizontal and vertical line segments connecting points in $S \cup S'$. A Manhattan network can also be thought of as a 1-spanner (for the L_1-metric) for the points in S.

We will show that there is a Manhattan network on S with $O(n \log n)$ vertices and edges which can be constructed in $O(n \log n)$ time. This allows us to to compute the L_1-Earth Mover's Distance on weighted planar point sets in $O(n^2 \log^3 n)$ time, which improves the currently best known result of $O(n^4 \log n)$. At the expense of a slightly higher time and space complexity we are able to extend our approach to any dimension $d \geq 3$. We will further show that our construction is optimal in the sense that there are point sets in the plane where every Manhattan network needs $\Omega(n \log n)$ vertices and edges.

1 Introduction

The problem to compute a minimum length Manhattan network is a well-researched area, see Gudmundsson et al. [7], Benkert et al. [1] and Chepoi et al. [3]. Even though the problem has received considerable attention the variant of minimizing the number of vertices and edges of the graph has not been considered (to the best of the authors' knowledge).

Here we will show that for every point set in the plane there is a Manhattan network with $O(n \log n)$ vertices and edges. This graph can be constructed in $O(n \log n)$ time. We will also show that this upper bound is tight, meaning that there are point sets where every Manhattan network on these points will need at least $\Omega(n \log n)$ vertices and edges. As it turns out, the Manhattan network constructed is not planar. We will show that if we force the network to be planar, there are point sets where every Manhattan network needs at least $\Omega(n^2)$ vertices and edges. Further we will show how to generalize the construction of the network to higher dimensions. Finally, we will show that one can reduce the time to compute the L_1-Earth Mover's Distance (EMD) for weighted point sets. The EMD is a useful distance measure for, e.g., shape matching, color-based image retrieval and music score matching, see Cohen and Guibas [4], Giannopoulos and Veltkamp [5], Graumann and Darell [6], and Typke, Giannopoulos, Veltkamp, Wiering and van Oostrum [11] for more information. Work on the optimization problem for the EMD under transformations has been done by Cabelloet al. [2] and Klein and Veltkamp [9]. An upper bound for the time to compute the EMD is $O(n^4 \log n)$ using a strongly polynomial minimum cost flow algorithm by Orlin [10]. Cabello et al. [2] gave a $(1+\varepsilon)$-approximation algorithm with runtime $O(n^2 \varepsilon^{-2} \log^2(n\varepsilon^{-1}))$. Recently, Indyk [8] gave an $O(n \log^{O(1)} n)$-time randomized $O(1)$-approximation algorithm if the two point sets consist of an equal number of points in $\mathbb{R}^2$ of weight 1. Using the Manhattan network as a 1-spanner for the L_1-distance, we can compute the L_1-EMD in d dimensions in $O(n^2 \log^{2d-1} n)$ time using Orlin's algorithm on the reduced graph. This improves the previously best known runtime of $O(n^4 \log n)$ significantly. Further, it immediately leads to a $\sqrt{2}$-approximation with the same runtime for the important case when the EMD is based on the Euclidean distance. This algorithm is conceptually easier than the slightly faster $(1+\varepsilon)$-approximation given by Cabello et al. [2].

2 Manhattan Networks

We will start formulating and proving the main result of this paper.

Theorem 1 *Let S be a set of n points in the plane. Then, there is a Manhattan network on S with*

*National ICT Australia Ltd, Sydney, Australia. NICTA is funded through the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council. joachim.gudmundsson@nicta.com.au

†Institut für Informatik, Freie Universität Berlin. This research was supported by the Deutsche Forschungsgemeinschaft within the European graduate program 'Combinatorics, Geometry and Computation' (No. GRK 588/3), {oklein,christian}@inf.fu-berlin.de

‡This research was supported by NSERC. School of Computer Science, Carleton University, Ottawa. michiel@scs.carleton.ca

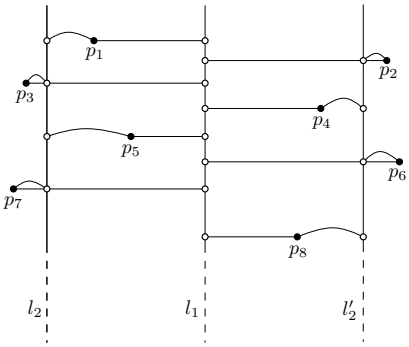

Figure 1: Construction of the Network.

$O(n \log n)$ vertices and edges. It can be computed in $O(n \log n)$ time.

Proof. Let S be a set of n points in the plane and assume that the points are sorted w.r.t. y-coordinates $S = \{p_1, \ldots, p_n\}$. Otherwise, the sorting can be done in $O(n \log n)$ time. In the following, L always denotes a list of points which is sorted by y-coordinate. The i-th point in L will be denoted by $L[i]$. Now, run the following routine on S.

Algorithm 1 (ConstructNetwork(L))
1. Find median p^* with respect to x-coordinate.
2. Set $L_1 := \emptyset, L_2 := \emptyset$.
3. For $i = 1, \ldots, |L|$ do
   (a) Construct vertex $v[i] := (p^*_x, L[i]_y)$.
   (b) Construct edge $e_h[i] := (L[i], v[i])$.
   (c) If $i \geq 2$:
   Construct edge $e_v[i] := (v[i-1], v[i])$.
   (d) If $L[i]_x \leq p^*_x$: add $L[i]$ at the end of L_1
   Else: add $L[i]$ at the end of L_2.
4. ConstructNetwork(L_1)
5. ConstructNetwork(L_2)

See Figure 1 for an illustration of the algorithm. We have to prove that the algorithm constructs a Manhattan network.

Let $p, q \in S$ be two arbitrary points. Let p^* be the first point chosen as a median in Step 1 with $p_x \leq p^*_x \leq q_x$. Clearly, p and q are both contained in L. W.l.o.g., let $p = L[i] =: p_i$ and $q = L[j] =: p_j$ with $i < j$. In Step 3, p_i is considered before p_j. Therefore, by construction, there are vertices $v[i], v[j]$, edges $(v[i], p_i)$, $(v[j], q_j)$ and a y-monotone sequence of vertices $v[i], \ldots, v[j]$. Now, the sequence $p_i, v[i], \ldots, v[j], p_j$ is an x- and y-monotone path consisting of two horizontal edges connected by a path of vertical edges and therefore a Manhattan path. This proves that the resulting graph is a Manhattan network on S.

The median in a list of $k := |L|$ numbers can be computed in $O(k)$. Steps (a) to (d) can be done in constant time. Therefore, the runtime of Algorithm 1 without the two recursive calls is $O(k)$. The insertion in the lists L_1, L_2 is done in sorted order with respect to the y-coordinate. No re-sorting is needed after the initial sorting step.

The overall runtime can be described by the recursion $T(n) = O(n) + 2 \cdot T(n/2)$, which gives $T(n) = O(n \log n)$. The number of Steiner points and edges in the construction obeys the same recursion, since in every recursive call of Algorithm 1 $O(k)$ vertices and edges are added. □

In practice, paths with a small number of links are often desirable. We will show how to construct a network such that for every pair of points there is a shortest path with a small number of links. Let $\alpha(n)$ denote the inverse of Ackermann's function, see [12].

Theorem 2 *Let S be a set of n points in the plane. Then, there is a Manhattan network on S with $O(n \log n)$ vertices and edges, where the number of edges on a shortest Manhattan path between any pair of points is bounded by $O(\alpha(n))$. The network can be computed in $O(n \log n)$ time.*

Proof. Consider one call of Algorithm 1. The Manhattan path between two input points p_i, p_j with $i < j$ always has the form $p_i, v[i], \ldots, v[j], p_j$, where $v[i], \ldots, v[j]$ is a sequence of Steiner points lying on a vertical line. Now, using a result of Yao [12], we can compute $O(k)$ edges in $O(k)$ time, each connecting two Steiner points, such that for any pair of Steiner points the number of links on the shortest path is $O(\alpha(k))$. That is, the total length of any path constructed is $O(\alpha(n)) + 2 = O(\alpha(n))$. Since we can compute these $O(k)$ edges in every recursive call in $O(k)$ time, the asymptotic runtime and number of Steiner points does not change. □

At the expense of a slightly higher runtime we can reduce the length of a shortest Manhattan path to a constant number of edges.

Theorem 3 *Let S be a set of n points in the plane. Then, there is a Manhattan network on S with $O(n \log^2 n)$, $O(n \log n \log\log n)$ and $O(n \log n \log^* n)$ vertices and edges where the number of edges on a shortest Manhattan path between any pair of points is bounded by $6, 7$ and 8, respectively. The runtimes are linear in the number of vertices and edges.*

Proof. The proof is analogous to that of Theorem 2, again using results of Yao [12]. □

The upper bound given in Theorem 1 is tight.

Theorem 4 *There are n-point sets in $\mathbb{R}^2$ where every Manhattan network needs $\Omega(n \log n)$ vertices and edges.*

Proof. We construct a point set P in general position, such that any Manhattan network for P consists of $\Omega(n \log n)$ vertices and edges. We assume that n is a power of two. Let ℓ be a vertical line separating P into two point sets $U := \{u_1, \ldots, u_{n/2}\}$ and $V := \{v_1, \ldots, v_{n/2}\}$, such that the points $u_1, v_1, u_2, v_2, \ldots, u_{n/2}, v_{n/2}$ are sorted by y-coordinates, from top to bottom, see Figure 2. For

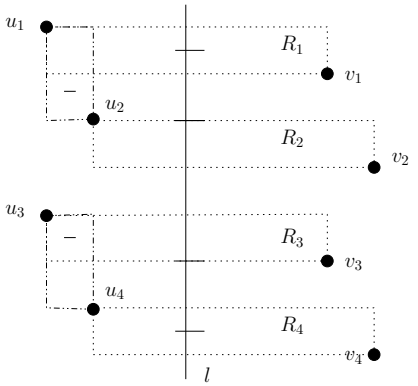

Figure 2: Lower Bound

$1 \leq i \leq n/2$, let R_i be the axes-parallel rectangle with top-left corner u_i and bottom-right corner v_i. Any Manhattan network on P must contain a path between u_i and v_i that crosses ℓ and is completely contained in R_i. Since the rectangles R_i are pairwise disjoint, it follows that any Manhattan network on P contains at least $n/2$ edges that cross ℓ. Observe that this remains true if we move the points of U and V horizontally, as long as U stays to the left of ℓ and V stays to the right of ℓ. Thus, we can move the points of U, such that they can be split into two subsets U_1 and U_2 that are separated by a vertical line ℓ' such that the sorted y-order alternates between a point in U_1 and a point in U_2. Any Manhattan network on P must contain at least $n/4$ edges that cross ℓ' and that are distinct from the above $n/2$ edges. Similarly, we can move the points of V, and split them into two subsets V_1 and V_2 that are separated by a vertical line ℓ'' in such a way that any Manhattan network on P must contain at least $n/4$ edges that cross ℓ'' and are distinct from the above $n/2 + n/4$ edges. We continue this moving in a recursive way and it can be shown that all these edges are distinct. We omit a rigorous proof due to space limitations. Then, it follows that the number $T(n)$ of vertices and edges in any Manhattan network on the final set P satisfies $T(n) \geq n/2 + 2 \cdot T(n/2)$, which proves that $T(n) = \Omega(n \log n)$. □

If the network is required to be planar the lower bound can be improved.

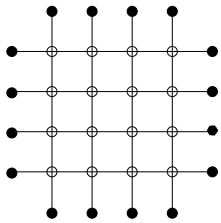
Figure 3: Planar Construction.

Theorem 5 *There are n-point sets in $\mathbb{R}^2$ where every planar Manhattan network needs $\Omega(n^2)$ vertices and edges.*

Proof. Let the set P of points in $\mathbb{R}^2$ be defined as follows, see Figure 3:

$$P := \bigcup_{i=1}^{n-1} (\{(\frac{i}{n}, 0)\} \cup \{(\frac{i}{n}, 1)\} \cup \{(0, \frac{i}{n})\} \cup \{(1, \frac{i}{n})\})$$

Let G be a Manhattan network for this point set. There must be a Manhattan path between every pair of points $(\frac{i}{n}, 0), (\frac{i}{n}, 1)$ and $(0, \frac{i}{n}), (1, \frac{i}{n})$. These paths have to be straight lines, since in the first case the x-coordinate and in the second case the y-coordinate is the same. This forces the $O(n^2)$ cross points of the straight lines to be Steiner points. □

A point set in general position giving the same lower bound can be constructed easily by perturbing the points slightly.

3 Higher Dimensions

The extension of the definition of a Manhattan path and therefore of a Manhattan network to dimensions $d \geq 3$ is straightforward. In dimension d we can use a similar divide-and-conquer approach as in the plane.

Theorem 6 *Let S be a set of n points in $\mathbb{R}^d$. Then, there is a Manhattan network on S with $O(n \log^{d-1} n)$ vertices and edges. It can be computed in $O(n \log^{d-1} n)$ time.*

Proof. Consider Algorithm 2 for point sets in $\mathbb{R}^d$:

Algorithm 2 (ConstructNetwork(L, d))
1. Find median p^* with respect to the d-th coordinate.
2. Project any point on the hyperplane containing p^* and orthogonal to the d-th coordinate. Let $\mathcal{P}$ be the set of projected points.
3. Add an edge between the original points and their projection.
4. ConstructNetwork($\mathcal{P}, d-1$) (Compute the Manhattan network on this hyperplane).
5. $L_1 := \{p \in L : p_d \leq p^*_d\}$. $L_2 := L \setminus L_1$.
6. ConstructNetwork(L_1, d)
7. ConstructNetwork(L_2, d)

Except for the recursive calls in the algorithm, any call can be done in $O(|L|)$ time. There are three recursive calls, one call of the routine for the same number of points in one dimension less and two calls for the number of points halved in the same dimension. Analogous to the earlier proof, the runtime of this can be expressed like

$$\begin{aligned} T(n,d) &= O(n) + T(n,d-1) + 2 \cdot T(n/2,d) \\ &= O(n \log^{d-1} n). \end{aligned}$$

The bound on the number of points and edges follows analogously. □

4 Earth Mover's Distance

We will now show that we can reduce the time to compute the L_1-Earth Mover's Distance on weighted point sets to $O(n^2 \log^{2d-1} n)$, which improves the previously best known result of $O(n^4 \log n)$.

A set $A = \{a_1, \ldots, a_n\}$ is called a weighted point set in $\mathbb{R}^d$ if $a_i = (p_i, \alpha_i)$ for $i = 1, \ldots, n$, where p_i is a point in $\mathbb{R}^d$ and $\alpha_i \in \mathbb{R}_0^+$ its corresponding weight; $W^A = \sum_{i=1}^n \alpha_i$ denotes the total weight of A. Now, let $A = \{(p_i, \alpha_i)_{i=1,\ldots,n}\}$ and $B = \{(q_j, \beta_j)_{j=1,\ldots,m}\}$ be two weighted point sets with total weights W^A, $W^B \in \mathbb{R}^+$ and $m \leq n$. Let $D : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_0^+$ be a distance measure on $\mathbb{R}^d$. The D-EMD between A and B is defined as

$$D\text{-}EMD(A,B) = \frac{\min_{F \in \mathcal{F}} \sum_{i=1}^n \sum_{j=1}^m f_{ij} D(p_i, q_j)}{\min\{W^A, W^B\}},$$

where $F = \{f_{ij}\}$ is a feasible flow, i.e., for every $i = 1, \ldots, n$ and $j = 1, \ldots, m$ we have $f_{ij} \geq 0$, $\sum_{j=1}^m f_{ij} \leq \alpha_i$, $\sum_{i=1}^n f_{ij} \leq \beta_j$ and $\sum_{i=1}^n \sum_{j=1}^m f_{ij} = \min\{W^A, W^B\}$.

Theorem 7 *The L_1-EMD can be computed in $O(n^2 \log^{2d-1} n)$ time.*

Proof. Let A, B be weighted point sets. Using Theorem 6 we can construct a 1-spanner of the complete bipartite graph between the points of A and B for the L_1-metric in $O(n \log^{d-1} n)$ time. The number of points and edges in the resulting network is bounded by $O(n \log^{d-1} n)$. Now we proceed as in Cabello et al. [2]. By the standard method of doubling each edge and orienting the two copies in different directions we get a flow network where between any pair of points there is a directed path of minimum L_1-length. Now we can use the minimum cost flow algorithm by Orlin [10] on the 1-spanner. Given a network $G = (V, E)$, Orlin's algorithm solves the minimum cost flow problem in $O((|E| \log |V|)(|E| + \log |V|))$. Since the number of points and edges in our spanner is bounded by $|E| = |V| = O(n \log^{d-1} n)$, the overall runtime is bounded by $O(n^2 \log^{2d-1} n)$. □

Theorem 7 immediately leads to a $\sqrt{2}$-approximation with the same runtime for the important case when the EMD is based on the Euclidean distance. This algorithm is conceptually easier than the slightly faster $(1+\varepsilon)$-approximation given by Cabello et al. [2].

Acknowledgments

We thank Günter Rote for the idea in Theorem 5.

References

[1] M. Benkert, A. Wolff, F. Widmann, and T. Shirabe. The minimum Manhattan network problem: Approximations and exact solution. *Computational Geometry - Theory and Applications*, 2006.

[2] S. Cabello, P. Giannopoulos, C. Knauer, and G. Rote. Matching point sets with respect to the earth mover's distance. In *Proc. 13th ESA*, 2005.

[3] V. Chepoi, K. Nouioua, and Y. Vaxes. A rounding algorithm for approximating minimum manhattan networks. In *Proc. 8th APPROX*, 2005.

[4] S. D. Cohen and L. J. Guibas. The earth mover's distance under transformation sets. In *Proc. 7th IEEE Int. Conf. Comp. Vision*, 1999.

[5] P. Giannopoulos and R. C. Veltkamp. A pseudometric for weighted point sets. In *Proc. 7th Europ. Conf. on Comp. Vision*, 2002.

[6] K. Graumann and T. Darell. Fast contour matching using approximate earth mover's distance. In *Proc. 1991 IEEE Comp. Society Conf. on Comp. Vision and Pattern Recognition*, 2004.

[7] J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Approximating a minimum Manhattan network. *Nordic J. Comput. 8*, 2001.

[8] P. Indyk. A near linear time constant factor approximation for Euclidean bichromatic matching (cost), to appear. In *Proc. 18th Symp. on Disc. Alg.*, 2007.

[9] O. Klein and R. C. Veltkamp. Approximation algorithms for the earth mover's distance under transformations using reference points. In *Proc. 16th ISAAC*, 2005.

[10] J. B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research 41*, 1993.

[11] R. Typke, P. Giannopoulos, R. C. Veltkamp, F. Wiering, and R. Oostrum. Using transportation distances for measuring melodic similarity. In *Proc. 4th Int. Conf. on Music Information Retrieval*, 2003.

[12] A. C. Yao. Space-time trade-off for answering range queries. In *Proc. 14th STOC*, 1982.