Journal of Algorithms and Computation

journal homepage: http://jac.ut.ac.ir

Minimum Spanning Tree of Imprecise Points Under L_1-metric

Amir Mesrikhani*[1], Mohammad Farshi†[2] and Behnam Iranfar‡[2]

[1,2,3]Combinatorial and Geometric Algorithms Lab, Department of Computer Science, Yazd University, Yazd, Iran.

ABSTRACT

Let S be a set of imprecise points that is represented by axis-aligned pairwise disjoint squares in the plane. A precise instance of S is a set of points, one from each region of S. In this paper, we study the optimal minimum spanning tree (*OptMST*) problem on S. The *OptMST* problem looks for the precise instance of S such that the weight of the MST in this instance, maximize (Max-MST) or minimize (Min-MST) between all precise instances of S under L_1-metric. We present a $(\frac{3}{7})$-approximation algorithm for Max-MST. This is an improvement on the best-known approximation factor of 1/3. If S satisfies k-separability property (the distance between any pair of squares are at least $k.a_{max}$ where a_{max} is the maximum length of the squares), the factor parameterizes to $\frac{2k+3}{2k+7}$. We propose a new lower bound for Min-MST problem on S under L_1-metric where S contains unit squares and provide an approximation algorithm with $(1+2\sqrt{2})$ asymptotic factor.

Keyword: Minimum spanning tree, Imprecise point set, Approximation algorithm.

AMS subject Classification: 05C78.

ARTICLE INFO

Article history:
Received 12, February 2019
Received in revised form 17, November 2019
Accepted 19 December 2019
Available online 31, December 2019

*Corresponding author: A. Mesrikhani. Email: mesrikhani@gmail.com
†m.farshi@gmail.com
‡biranfar@gmail.com

1 Introduction

In the recent years, processing and managing imprecise points are challenging problems in computational geometry. The position of the points is usually imprecise because of a sensor error, the noise of transmitting data and security. For example, the Global Positioning System (GPS) is a navigation system that provides the location of a query place based on a distance from the satellites. Computing exact distance is impossible and measured approximately. So the GPS reports the position of the query point with some error. However, most of the algorithms in computational geometry work based on the assumption that the position of points (and also computations) is exact. Thus, they may fail with imprecise input [13, 10]. In the geometric context, different models are introduced for imprecision like the region-based model and linear parametric geometric uncertainty model (LPGUM) [7, 3]. In the region-based model, an imprecise point is modeled as a geometric region like square, disk and etc. These geometric regions have a constant complexity, but in LPGUM, a point is represented by a convex shape with $k \geq 3$ vertices.

Finding a network that connects set of points with minimum cost, is the main application of minimum spanning tree (MST). Computing MST under L_1 metric is so necessary for designing VLSI circuits [12]. Therefore, it is important to compute MST under imprecision. Obviously computing a lower and upper bound for the weight of MST as the best and worst case, estimate the cost of the network for the imprecise points. So it is necessary to compute which precise instances of imprecise points maximize or minimize MST.

1.1 Problem Definition

Let $S = \{s_1, \ldots, s_n\}$ be a set of n imprecise points and $S' = \{p_1, \ldots, p_n\}$ be a precise instance from S (i.e. $p_i \in s_i$ for $i = 1, \ldots, n$). For each two points $p(p_x, p_y), q(q_x, q_y) \in \mathbb{R}^2$, the L_1-distance between p and q is defined as follows:

$$d_1(p, q) = |p_x - q_x| + |p_y - q_y|.$$

Suppose $w(T)$ is the weight of the minimum spanning tree T. The weight between each two points is the L_1-distance between them. In this paper, we consider the following two problems.

Problem 1 (Max-MST problem) Given an imprecise point set S, compute a precise instance S' of S such that the weight of the MST on S' is maximized (under L_1-metric) between all precise instances of S.

Problem 2 (Min-MST problem) Given an imprecise point set S, compute a precise instance S' of S such that the weight of the MST on S' is minimized (under L_1-metric) between all precise instances of S.

In this paper, we study the Max-MST and the Min-MST problem, for a set of axis-aligned pairwise disjoint squares under L_1-metric. The disjoint squares mean that the interior of any two squares does not have any intersection, but the boundary of them may intersect.

1.2 Related Work

Computing minimum spanning tree (MST) is a classical and well-known problem and several efficient algorithms proposed for it [2]. The MST problem was studied in the various imprecise models. Under imprecision, the goal is finding a precise instance from imprecise regions to maximize (Max-MST) or minimize (Min-MST) the weight of the MST between all precise instances. In 2010, Löffler and van Kerveld shows NP-hardness of computing Min-MST for a set of geometric regions that are not pairwise disjoint [9]. Dorrigive et.al. [5] prove that finding Min-MST or Max-MST for pairwise disjoint regions, is NP-hard too in 2012. Several approximation algorithms are proposed for Min or Max-MST. Dorrigive et.al. define Max-MST problem firstly and give an algorithm with $\frac{1}{2}$-approximation factor under L_2-metric for the pairwise disjoint disks. Their algorithm is so simple and picks the center of each disk and compute the MST on these points. Let r_{max} be a maximum radius of the disks. If the minimum distance of any two disks is at least $k.r_{max}$ for $k > 0$ (k-separability property), then the approximation factor of their algorithm is $(1 - \frac{2}{k+4})$ [5]. Yang et.al. give a 3-approximation algorithm for Min-MST for set of unit disks [14]. If the disks satisfy k-separability property, picking center of each disk, yields $(1 + \frac{2}{k})$-approximation algorithm [5].
Bartal et.al. [1] studied MST under LPGUM model. They addressed the problem of testing the stability of Euclidean MST for a set of n LPGUM points with complexity k. In this problem, an arbitrary point is selected from each imprecise region and the MST is computed on these points. The goal is deciding whether we can select other points and get a smaller weight for MST or not. If the answer is no, then this MST is the Min-MST for a set of n LPGUM points. Otherwise, other points yield the Min-MST but there is not any polynomial time algorithm to compute them. They provided an $O(k \log k)$ time algorithm for comparing the weight of two edges and $O(n^3 k \log k)$ time algorithm for testing the stability of the MST. Under the stochastic model of input where the points being active with some probability, an interesting problem is computing expected weight of MST [8, 15]. Kamousi et.al. [8] considered expected MST problem and showed that it is a polynomial time problem for dimension $d \geq 2$. For $d = 2$, they proposed an $O(n^4)$ time constant factor approximation algorithm. Another well-studied problem under imprecision is traveling salesman problem (TSP). In the Euclidean setting, Mitchell [11] provided the first constant-factor approximation algorithm for a set of arbitrary disjoint regions. de Berg et.al. [4] considered TSP in the region-based model and proposed PTAS for imprecise points that modelled with disjoint convex regions.

1.3 Our Results

In this paper, we study both Max-MST and Min-MST under L_1-metric, and get the following results:

1. A $(\frac{3}{7})$-approximation algorithm for Max-MST where imprecise points are modeled as axis-aligned pairwise disjoint squares. This is an improvement on the best-known approximation factor of 1/3.

2. A $(\frac{2k+3}{2k+7})$-approximation algorithm for Max-MST where squares satisfy k-separability property.

3. We propose a new lower bound for Min-MST problem for a set of axis-aligned pairwise disjoint unit squares under L_1-metric and provide $(1+2\sqrt{2})$-approximation algorithm for Min-MST problem.

Outline of the paper. This paper organizes as follows. In Section 2, we describe two approximation algorithms for Max-MST. Then we study Min-MST in Section 3 and present an algorithm that approximates the solution with $(1 + 2\sqrt{2})$ factor. Finally, we conclude the paper in Section 4.

2 Max-MST

Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint squares that each $s_i \in S$ represent an imprecise point. In this section, two approximation algorithms are proposed for the Max-MST problem with $\frac{1}{3}$ and $\frac{3}{7}$ factors. Dorrigive et.al. [5] proposed a simple algorithm for pairwise disjoint disks and consider the center of the disks as a precise instance. By applying this algorithm on S, a $\frac{1}{3}$-approximation algorithm can be obtained. We improve this factor to $\frac{3}{7}$ in Section 2.2.

2.1 A $(\frac{1}{3})$-Approximation Algorithm

The algorithm simply picks a center of each $s_i \in S$ as the precise instance S' and MST on S' is computed by Kruskal algorithm [5]. The Theorem 2.1 shows that this algorithm approximate the optimal Max-MST within a factor of $\frac{1}{3}$.

Theorem 2.1. *Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint squares. Let T_c be a MST on the center of the squares and T_{opt} be the optimal Max-MST. Then $w(T_c) \geq \frac{1}{3} w(T_{opt})$.*

Proof. Let O be a precise instance that, yields T_{opt}. Let T be a spanning tree with the same topology as T_c but it builds on O. Since T and T_{opt} build on the same points and T_{opt} is a MST, we have:

$$w(T_{opt}) \leq w(T). \tag{1}$$

Let $p_i, p_j \in O$ (selected from $s_i, s_j \in S$ respectively) be two arbitrary points such that the edge $e_{opt} = (p_i, p_j)$ belongs to T. e_{cen} is the edge connecting the center of s_i (i.e. $c_i(x_i, y_i)$) and s_j (i.e. $c_j(x_j, y_j)$) (see Figure 1). Suppose a_i and a_j be the lengths of s_i and s_j respectively where $a_i \leq a_j$. The ratio between the weight of e_{cen} and e_{opt} is:

$$\frac{w(e_{cen})}{w(e_{opt})} = \frac{a_i/2 + a_j/2 + c}{2a_i + a_j} \quad where \quad c = |y_i - y_j|. \tag{2}$$

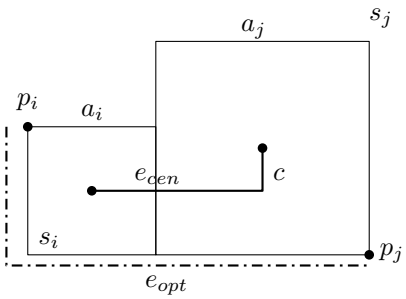

Figure 1: Illustration of the proof of Theorem 2.1.

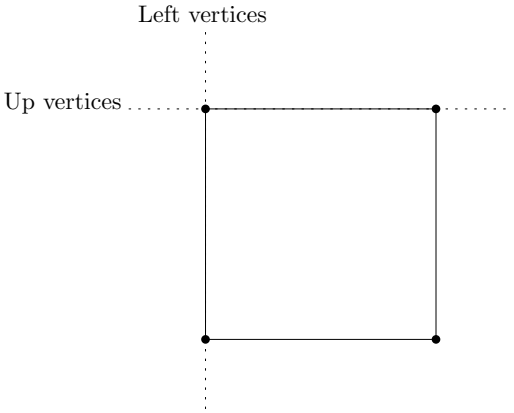

Figure 2: Labeling of the squares vertices with Up or Left.

Let a_{max} be the maximum length of squares. To get the smallest value for Equation 2, we set $c = 0$. Since $a_i \leq a_j$ we have:

$$\frac{a_i/2 + a_j/2 + c}{2a_i + a_j} \geq \frac{a_i/2 + a_j/2}{2a_i + a_j} \geq \frac{a_{max}}{3a_{max}} = \frac{1}{3}. \tag{3}$$

This case may occur for any pair of squares. Since T and T_c have the same topology, the ratio between $w(T_c)$ and $w(T)$ is:

$$\frac{w(T_c)}{w(T)} \geq \frac{1}{3}. \tag{4}$$

By Equation 1 we conclude that:

$$w(T_c) \geq \frac{1}{3}w(T) \geq \frac{1}{3}w(T_{opt}).$$

□

Running Time Analysis. Computing S' takes $O(n)$ time by simply checking all squares and for computing the MST on S', a complete graph must be considered. So the running time of $(\frac{1}{3})$-approximation algorithm is $O(n^2 \log n)$.

2.2 A $(\frac{3}{7})$-Approximation Algorithm

To improve the approximation factor, we must choose the precise instance wisely. So we sort the squares to get some order for choosing a point from each square. Squares were sorted based on a Left or Up vertex respect to (x or y)-coordinate (See Figure 2). It is clear that a square has two Left and two Up vertices. **Definition 1** Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint squares. Suppose $x(i)$ is the x-coordinate of the Left vertices of $s_i \in S$ and $y(i)$ is y-coordinate of the Up vertices of s_i. We say $s_i \prec s_j$ iff $(x(i) < x(j))$ or $(x(i) = x(j)\ and\ y(i) > y(j))$.

Algorithm 1: $(\frac{3}{7})$-approximation algorithm

Input : S, a set of n axis-aligned pairwise disjoint squares.
Output: $T_G = (V, E)$.

```
1  Sort S based on ≺ in an ascending order. Let s_1, ..., s_n be the sorted list;
2  V, E = ∅;
3  Let p_1 and p_2 be the two points from s_1, s_2 that yield maximum distance between
   them;
4  V = V ∪ {p_1, p_2};
5  E = E ∪ {(p_1, p_2)};
6  i = 3;
7  while i ≤ n do
8      Pick a point from s_i that has a maximum distance to p_{i-1} and call it p_i;
9      V = V ∪ {p_i};
10     E = E ∪ {(p_{i-1}, p_i)};
11     i = i + 1;
12 return T_G;
```

Consider the ascending order of S based on $\prec$. The idea of the algorithm 1 is selecting a point from each square to maximize L_1-distance between any two consecutive squares, then connect selected points based on the sorted order of corresponding squares and denote it by T_G (See Figure 4). The selected points are the precise instance of S. It is clear that the points of the precise instance, are vertices of the squares and T_G is a spanning tree. We claim that T_G approximates the optimal solution of the Max-MST within a factor 3/7.

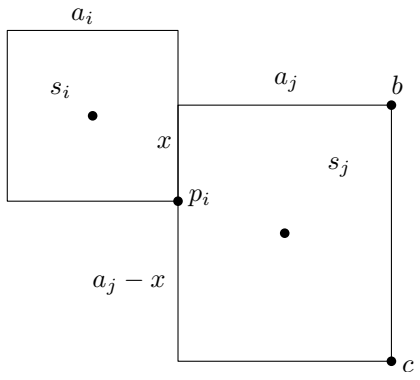

Figure 3: Illustration of the proof of Theorem 2.2.

Theorem 2.2. *Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint squares. If T_G is the output of algorithm 1 and T_{opt} be the optimal Max-MST, then $w(T_G) \geq \frac{3}{7} w(T_{opt})$.*

Proof. Let O be a precise instance, that yields T_{opt}. Let T be a spanning tree with the same topology as T_G but it builds on O and let a_i and a_j $(a_i \leq a_j)$ be the lengths of s_i

and s_j, where s_i and s_j are two consecutive squares base on $\prec$. Suppose $p_i = (x_i, y_i)$ is the selected point from s_i. By the Algorithm 1, two choices are available for s_j. These choices are denoted by $b = (x_b, y_b)$ and $c = (x_c, y_c)$ where b and c are the vertices of s_j. Let $x = |x_i - x_b|$, so we have $a_j - x = |x_i - c_x|$ (see Figure 3). To obtain the worst-case that yields the minimum approximation factor, we consider two cases:

1. $x \geq a_j - x$

   By the Algorithm 1, b is chosen from s_j. The $x \geq a_j - x$ satisfies if $a_j/2 \leq x \leq a_j$. So the ratio between the edge (p_i, b) and the optimal edge is:

   $$\frac{x + a_j}{-x + 2a_i + 2a_j}. \tag{5}$$

   The Equation (5) is a nondecreasing function base on x. Therefore, it gets the minimum value in $x = a_j/2$. Let a_{max} be the maximum length of the squares. Since $a_i \leq a_j$, the Equation (5) can be bounded as follows:

   $$\frac{x + a_j}{-x + 2a_i + 2a_j} \geq \frac{a_j/2 + a_j}{-a_j/2 + 2a_i + 2a_j} \geq \frac{(3/2)a_{max}}{(7/2)a_{max}} = \frac{3}{7}. \tag{6}$$

   This case may occur for any pair of the squares. Since T and T_G have the same topology, the ratio between $w(T_G)$ and $w(T)$ is:

   $$\frac{w(T_G)}{w(T)} \geq \frac{3}{7}. \tag{7}$$

   By the Equation (1) we conclude that:

   $$w(T_G) \geq \frac{3}{7} w(T) \geq \frac{3}{7} w(T_{opt}).$$

2. $x < a_j - x$

   The point c was chosen from s_j by Algorithm 1. The $x < a_j - x$ satisfies if $0 \leq x < a_j/2$. So the ratio between the edge (p_i, c) and the optimal edge is:

   $$\frac{-x + 2a_j}{-x + 2a_i + 2a_j}. \tag{8}$$

   The Equation (8) is a nonincreasing function base on the x. So the lower bound for the minimum value can be obtained by $x = a_j/2$. Similar to case 1 we can bound Equation (8) by 3/7. Therefore, by Equation (1), $w(T_G) \geq \frac{3}{7} w(T_{opt})$.

□

Running Time Analysis. For computing precise instance, the squares must be sorted based on $\prec$. This step takes $O(n \log n)$ and T_G is built in $O(n)$ time. So the running time of $(\frac{3}{7})$-approximation algorithm is $O(n \log n)$. So our algorithm approximates the weight of the Max-MST faster than $(\frac{1}{3})$-approximation algorithm.

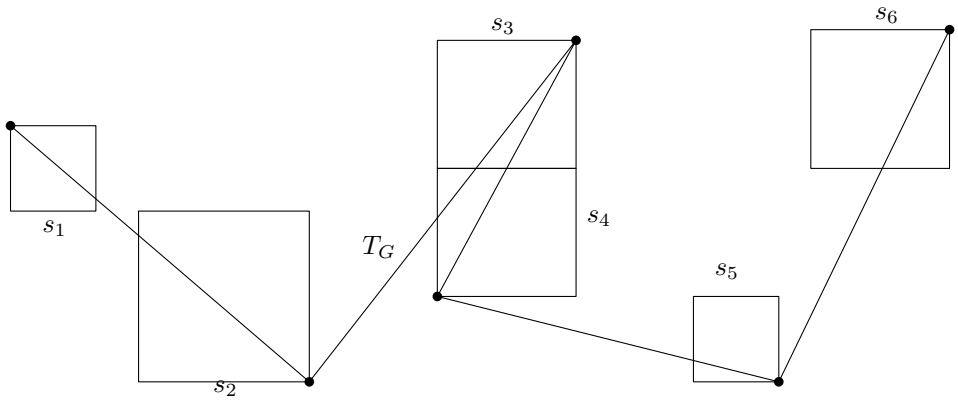

Figure 4: By running Algorithm 1, V consist of bold points and T_G connect all squares respect to sorted order $S = \{s_1, s_2, s_3, s_4, s_5, s_6\}$.

2.3 Parametrized Approximation Factor

Let a_{max} be the maximum length of the squares. S satisfies k-separability property if the minimum distance between any pair of the squares is at least $k.a_{max}$. If S satisfies k-separability property and by using Algorithm 1, then the following result can be obtained:

Theorem 2.3. *Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint squares that satisfies k-separability property ($k > 0$). If T_G denotes the output of the Algorithm 1 and T_{opt} is the optimal Max-MST, then $w(T_G) \geq \frac{2k+3}{2k+7} w(T_{opt})$.*

Proof. Consider two consecutive squares s_i and s_j (base on $\prec$) where the length of s_i (i.e. a_i) is no more than the length of s_j (i.e. a_j). If a be the selected point of s_i then two choices are available for s_j. We denote these choices by $b = (x_b, y_b)$ and $c = (x_c, y_c)$ (see Figure 5). Let $x = |x_i - x_b|$, so we have $a_j - x = |x_i - c_x|$. Similar to the proof of Theorem 2.2, the worst-case for two squares can be obtained by setting $x = a_j/2$. So the ratio between the edge (a, b or c) to the optimal edge can be bounded as follows:

$$\frac{k.a_{max} + x + a_j}{k.a_{max} - x + 2a_i + 2a_j} \geq \frac{k.a_{max} + \frac{3}{2}a_j}{k.a_{max} + 2a_i + \frac{3}{2}a_j} \geq \frac{2k+3}{2k+7}. \tag{9}$$

This case may occur for any pair of squares, so by the Equation (1), we have $w(T_G) \geq \frac{2k+3}{2k+7} w(T_{opt})$.

□

3 Min-MST

Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint unit squares that represent n imprecise points. In the following, we give an approximation algorithm for Min-MST problem with $1 + 2\sqrt{2}$ asymptotic approximation factor.

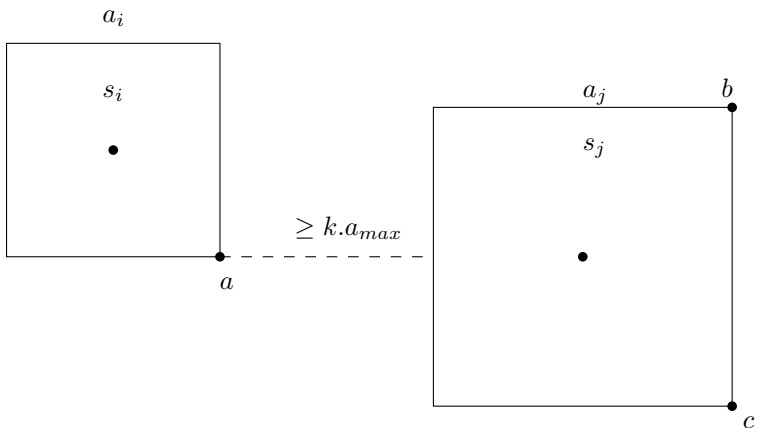

Figure 5: Illustration of the proof of Theorem 2.3.

Definition 2 Let $S = \{s_1, \ldots, s_n\}$ be a set of n axis-aligned pairwise disjoint unit squares and d_{ij} be the minimum L_1-distance between $s_i, s_j \in S$. Define the complete graph $G = (V, E)$ where $v_i \in V$ corresponds to $s_i \in S$ and the weight of the edge $e_{ij} = (v_i, v_j)$ is d_{ij}. The minimum connecting tree (MCT) is the minimum spanning tree of G. Let L_{MCT} denote the weight of MCT.
In computing MCT, more than one point is selected from the squares, So the lower bound of the weight of Min-MST is L_{MCT}. Otherwise, we can get a smaller spanning tree on S that contradicts the definition of MCT. Let L_{opt} be the weight of Min-MST. We extend the conjecture of Fraser [6] for the lower bound of the Min-MST for squares under L_1-metric. Figure 6 shows the arrangement of the squares that yields the lower bound of the weight of Min-MST. For the even number of squares, every fourth square from left to right based on $\prec$ order are joined with an edge weight 2. The first and the last two squares do not increase the weight of Min-MST. So the total weight of the Min-MST is $[\frac{n}{2}] - 2$. For the odd number of squares, the last one base on $\prec$ order does not increase the weight of the Min-MST, so $[\frac{n-1}{2}] - 1 \leq [\frac{n}{2}] - 1$ holds for the lower bound of the Min-MST.
Conjecture 1. Let $S = \{s_1, \ldots, s_n\}$ be a set of $n \geq 5$ axis-aligned pairwise disjoint unit squares. The following holds for L_{opt}.

1. $L_{opt} \geq [\frac{n}{2}] - 2 \quad for \quad even \quad n \geq 6$
2. $L_{opt} \geq [\frac{n}{2}] - 1 \quad for \quad odd \quad n \geq 5$

3.1 A $(1+2\sqrt{2})$-Approximation Algorithm

In this section, we propose an approximation algorithm for the Min-MST problem. The approximation algorithm is similar to Fraser et.al. algorithm [6] that proposed to compute Min-MST for a set of n pairwise disjoint unit disks under the Euclidean metric. The algorithm computes MCT. Consider the squares that have more than one selected point and move these points to the center of the squares. By these movements, we ensure that

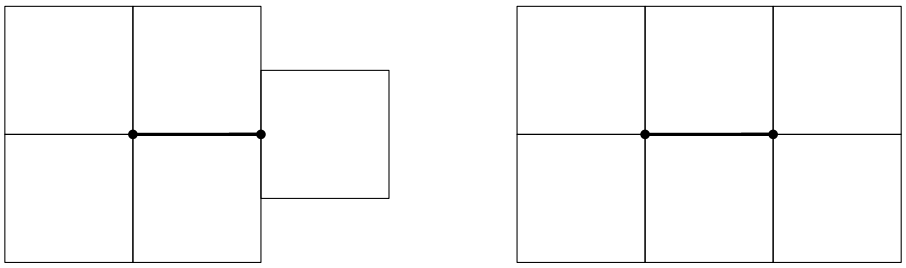

Figure 6: Arrangement of the squares correspond to Conjecture 1.

exactly one point is selected from each square. The resulting points are a precise instance of S. Finally the MST on the precise instance is returned as a solution.

Algorithm 2: $(1+2\sqrt{2})$-approximation algorithm

Input : $S = \{s_1, \ldots, s_n\}$, a set of n axis-aligned pairwise disjoint unit squares.
Output: $(1+2\sqrt{2})$-appoximate solution for Min-MST.

```
Compute MCT of S;
i = 1 and P = ∅;
while i ≤ n do
    if s_i has more than one selected point then
        Set p_i to the center of s_i;
    else
        Set p_i to the selected point of s_i;
    P = P ∪ {p_i};
    i = i + 1;
Compute MST of P and call it T_G;
return T_G;
```

3.1.1 Analyze of the Approximation Factor

It is clear that MCT has $n-1$ edges and at least two leaves, since it is a tree. By the Algorithm 2, we must move $2(n-3)$ points to the center of the squares in the worst-case. Each moving adds $\frac{\sqrt{2}}{2}$ to the weight of each $2(n-3)$ edges. Let L_c be the weight of the tree (T_c) that are created by moving points and L_G be the weight of T_G, so we have:

$$L_c \leq L_{MCT} + \sqrt{2}(n-3) + \sqrt{2} \leq L_{MCT} + \sqrt{2}(n-2). \tag{10}$$

By assuming Conjecture 1, it is clear that $\frac{n}{2} < L_{opt}$. Since $L_{MCT} \leq L_{opt}$, so we rewrite (10) as follows:

$$L_c \leq L_{opt} + \sqrt{2}(2L_{opt} + 2) < (1+2\sqrt{2})L_{opt} + 2\sqrt{2}. \tag{11}$$

By Algorithm 2, T_G computed on the points of T_c, we conclude the following result:

$$L_G \leq L_c < (1 + 2\sqrt{2})L_{opt} + 2.82. \tag{12}$$

Assuming the lower bound of Conjecture 1 for Min-MST, T_G approximates the optimal Min-MST with an asymptotic approximation factor of $(1 + 2\sqrt{2})$.

Running Time Analysis. Computing MCT of S takes $O(n^2 \log n)$ time and moving points to the center of the squares, takes $O(n)$ time. In the last step, MST is computed over n points, so it takes $O(n^2 \log n)$ time. Therefore, the algorithm complexity is $O(n^2 \log n)$.

4 Conclusion

In this paper, we studied the problem of computing optimal minimum spanning tree for a set of imprecise points under L_1-metric. The Max-MST and Min-MST can be defined under imprecision. We provided two approximation algorithm for Max-MST and if imprecise points satisfy k-separability property, the approximation factor can be parametrized with respect to k. An approximation algorithm was presented for Min-MST where imprecise points modelled as unit disjoint squares. There are various problems to be pursued. One interesting problem is giving an algorithm that yields a better approximation factor for Min or Max-MST. Another challenging problem is giving tight examples for proposed algorithms.

References

[1] Bartal, O. and Joskowicz, L. (2014). Euclidean minimum spanning tree with dependent uncertainties. In *Proceedings of the 30th European Workshop on Computational Geometry*.

[2] Boruvka, O. (1926). About a certain minimal problem. *Praca Moravske Prirodovedecke Spolecnosti*, 3:37–58.

[3] Davoodi, M., Mohades, A., Sheikhi, F., and Khanteimouri, P. (2015). Data imprecision under λ-geometry model. *Information Sciences*, 295:126–144.

[4] de Berg, M., Gudmundsson, J., Katz, M. J., Levcopoulos, C., Overmars, M. H., and van der Stappen, A. F. (2005). Tsp with neighborhoods of varying size. *Journal of Algorithms*, 57(1):22–36.

[5] Dorrigiv, R., Fraser, R., He, M., Kamali, S., Kawamura, A., López-Ortiz, A., and Seco, D. (2015). On minimum-and maximum-weight minimum spanning trees with neighborhoods. *Theory of Computing Systems*, 56(1):220–250.

[6] Fraser, R. (2012). *Algorithms for geometric covering and piercing problems.* PhD thesis, University of Waterloo.

[7] Joskowicz, L., Ostrovsky-Berman, Y., and Myers, Y. (2010). Efficient representation and computation of geometric uncertainty: The linear parametric model. *Precision Engineering*, 34(1):2–6.

[8] Kamousi, P., Chan, T. M., and Suri, S. (2011). Stochastic minimum spanning trees in euclidean spaces. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 65–74. ACM.

[9] Löffler, M. and van Kreveld, M. (2010). Largest and smallest convex hulls for imprecise points. *Algorithmica*, 56(2):235–269.

[10] Mesrikhani, A. and Davoodi, M. (2017). Non-zero probability of nearest neighbor searching. *Journal of AI and Data Mining*, 5(1):101–109.

[11] Mitchell, J. S. (2010). A constant-factor approximation algorithm for tsp with pairwise-disjoint connected neighborhoods in the plane. In *Proceedings of the twenty-sixth annual symposium on Computational geometry*, pages 183–191. ACM.

[12] Mittal, S. (2016). A survey of architectural techniques for managing process variation. *ACM Computing Surveys (CSUR)*, 48(4):54.

[13] Wang, Y., Li, X., Li, X., and Wang, Y. (2013). A survey of queries over uncertain data. *Knowledge and information systems*, 37(3):485–530.

[14] Yang, Y., Lin, M., Xu, J., and Xie, Y. (2007). Minimum spanning tree with neighborhoods. *Algorithmic Aspects in Information and Management*, pages 306–316.

[15] Zhou, J., He, X., and Wang, K. (2014). Uncertain quadratic minimum spanning tree problem. *Journal of Communications*, 9(5):385–390.