Polynomial Time Algorithms for Three-label Point Labeling*

Rob Duncan† Jianbo Qian ‡ Antoine Vigneron § Binhai Zhu ¶

Abstract

In this paper, we present an $O(n^2 \log n)$ time solution for the following multi-label map labeling problem: Given a set S of n distinct sites in the plane, place at each site a triple of uniform squares of maximum possible size such that all the squares are axis-parallel and a site is on the boundaries of its three labeling squares. We also study the problem under the discrete model, i.e., a site must be at the corners of its three label squares. We obtain an optimal $\Theta(n \log n)$ time algorithm for the latter problem.

1 Introduction

Map labeling is a popular problem on information visualization in our daily life. It is an old art in cartography and finds new applications in recent years in GIS, graphics and graph drawing [1, 2, 5, 6, 7, 11, 13, 15, 16, 18, 19, 22, 27, 29, 30]. Among many problems in map labeling, labeling points is of special interest to many practitioners and theoreticians. In the paper by Formann and Wagner [11], any point site can only be labeled with 4 candidate (axis-parallel) squares each of which has a vertex anchored at the site and the objective is to maximize the square size. See Figure 1, I.a, for an instance of the problem. (In general we call this kind of model *discrete*, i.e., each site has only a constant number of candidates.) Even this seemingly simplest version is shown to be NP-complete and moreover; it is NP-hard to approximate within factor 2 [11]. (For details regarding NP-completeness readers are referred to [12].) In the past several years more generalized models have been proposed. The basic idea is to allow each site to have an infinite number of possible candidate labels (see [10, 14, 17, 26, 30]). This model is more natural than the previous discrete models (like the one in [11]) and has been coined as the *sliding model* in [17]. On the other hand, designing efficient algorithms for map labeling under the sliding model is a new challenge to map labeling researchers. We briefly review some recent results on labeling points under the sliding model.

Before our review, we briefly define some necessary concepts in approximation algorithms as most of the problems in labeling points try to maximize the size of the labels. An approximation algorithm for a (maximization) optimization problem Π provides a **performance guarantee** of ρ if for every instance I of Π, the solution value returned by the approximation algorithm is at least $1/\rho$ of the optimal value for I. For the simplicity of description, we simply say that this is a factor ρ approximation algorithm for Π.

In [10], Doddi et al. designed several approximation algorithms for labeling points with arbitrarily oriented squares and circles (though the constant factors are impractical: 36.6 and 29.86 respectively).

*This research is supported by NSF of China, Hong Kong RGC CERG grants CityU1103/99E, HKUST6088/99E and DAG 00/01.EG27. Part of this research was done when the second and the fourth author were with City University of Hong Kong.

†Department of Computer Science, Montana State University, Bozeman, MT 59717-3880, USA.

‡Department of Mathematics, Shandong University, Jinan, China.

§Department of Computer Science, Hong Kong University of Science and Technology.

¶Department of Computer Science, Montana State University, Bozeman, MT 59717-3880, USA.

They also presented bicriteria PTAS for the problems. The former bound was improved to 12.95 by Zhu and Qin [31] and significantly to 5.09 by Doddi et al. most recently [9]. The latter bound on labeling points with circles was improved to 19.35 by Strijk and Wolff [26] and recently to 3.6 by Doddi et al. [9]. If any labeling square must be along a fixed direction (e.g., axis-parallel), Zhu and Qin showed that it is possible to have a factor-4 approximation [31]. In [17] van Kreveld et al. proved that it is NP-hard to decide whether a set of points can all be labeled with axis-parallel unit squares under the sliding model. See Figure 1, I.b, for an example of labeling points with sliding axis-parallel squares. (In fact, in [17], van Kreveld et al. tried to maximize the number of sites labeled instead of the size of the labels.) In [26] Strijk and Wolff used similar ideas to prove that the problem of labeling points with maximum size uniform circles is NP-hard. This explains why it is meaningful to study approximation algorithms for these problems.

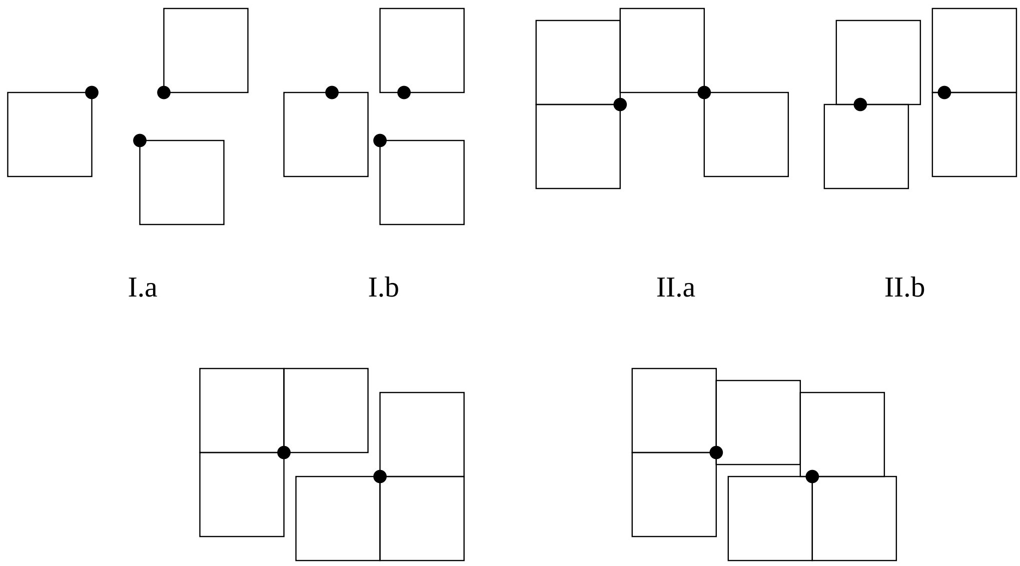


Figure 1. Examples for one-label, two-label and three-label point labeling.

As another kind of generalization for map labeling, recently Zhu and Poon studied the problem of labeling point sites with axis-parallel uniform square pairs and circle pairs. The motivation is that in some applications we need two labels for a site [20] (like labeling a map for weather reporting or labeling a bilingual city map). They obtained factor-4 and factor-2 algorithms for the two problems respectively, besides presenting a bicriteria approximation scheme [30]. For map labeling with uniform square pairs, Zhu and Qin [31] first improved the approximation factor of [30] from 4 to 3. Then Qin et al. further improved the factor to 2 [23]. More recently, Spriggs proved that the problem is NP-hard and, in fact, NP-hard to approximate within a factor of 1.33 [24]. See Figure 1, II.a and II.b for examples of labeling points with uniform square pairs under the discrete and sliding model respectively.

For map labeling with uniform circle pairs, Qin et al. first improved the 2 approximation factor to 1.96 and proved that problem is NP-hard, in fact, NP-hard to approximate within a constant factor of $\delta > 1$ [23]. The 1.96 factor was recently improved to 1.686 by Spriggs and Keil [25] and then by Wolff et al. to 1.5 [28]. There are still some gaps between the lower and upper bounds for both of the two problems.

In this paper, we study the problem of labeling point sites with uniform square triples. (See Figure 1, III.a, III.b for examples of labeling points with uniform square triples under the discrete and sliding model respectively.) The problem is interesting both in application and theory. In practice, many weather reporting programs on TV need to label a city with three labels: its name, temperature and chance of rainfall. In theory, the problem of labeling point sites with uniform squares is NP-hard under either the discrete model [11] or the sliding model [17]. Also, the problem of labeling point sites with uniform square pairs is NP-hard under both the discrete and sliding model [24]. On the other hand, labeling point sites with four squares is trivially polynomial solvable (the solution is decided by the closest pair of the point set, under the L_∞ metric). The labeling problems for points with circles and circle pairs are both NP-hard [26, 23]. Finally, we remark that it is impossible to label a point with three or more non-overlapping circles.

This paper is organized as follows. In Section 2, we formally define the new problem of labeling points with uniform square triples under the discrete and sliding models. In Section 3, we present an optimal solution for the problem under the discrete model. In Section 4, we present a polynomial time solution for the problem under the sliding model. In Section 5, we conclude the paper.

2 Preliminaries

In this section we formally define the problems to be studied. We also make some definitions related to our algorithms. The MLUST problem (Map Labeling with Uniform Square Triples) is defined as follows:
Instance: A set S of points (sites) p_1, p_2, ..., p_n in the plane.
Problem: Does there exist a set of n triples of axis-parallel squares of maximum size (i.e., length of a side) L^* each of which is placed at each input site $p_i \in S$ such that no two squares intersect in their interiors and no site is contained in any square.

Notice that we can have two versions of the problem: the problem can be under the discrete model in which every site must be at the corners of its three labeling squares and the problem can also be under the sliding model in which every site can be on the boundaries of its three labeling squares. Because of the nature of the problem even under the sliding model every site must be at the corners of at least two of its labeling squares. We present below a few definitions which will be used in later sections.

Given a set S of n sites in the plane, the closest pair of S under L_∞ metric, $D_\infty(S)$, is defined as the minimum L_∞-distance between any two points in S. Clearly $D_\infty(S)$ can be computed in $O(n \log n)$ time with standard algorithm in computational geometry [21].

3 Map Labeling With Uniform Square Triples (MLUST) Under the Discrete Model

In this section we present the details of a polynomial time algorithm for the MLUST problem under the discrete model. Because of the nature of the problem, the metric discussed in this paper is L_∞ unless otherwise specified. Let $D_\infty(S)$ be the closest pair of S under the L_∞-metric. Let l^* denote the size of each square in the optimal solution of the discrete MLUST problem. The following lemma is easy to prove.

Lemma 1 $D_\infty(S)/2 \leq l^* \leq D_\infty(S)$.

In the following we show how to decide whether a set of points S can be labeled with square triples with edge length l, $D_\infty(S)/2 \leq l \leq D_\infty(S)$. For any two points $p_i, p_j \in S$, let $d_\infty(p_i, p_j)$ denote the L_∞-distance between them. Let C_i denote the L_∞-circle centered at point $p_i \in S$ with radius l. Clearly, the circle C_i contains no other point from the input set S except its own center. Note that each circle C_i can be partitioned into four L_∞-circles with radius $l/2$, which are geometrically squares with edge length l. We loosely call them *sub-squares* of C_i. Basically, to label p_i we need to select three out of four sub-squares in each C_i so that no two sub-squares intersect each other.

We now present our algorithm. First, we compute the following multiple intersection graph $G_M(S, l)$. C_i $(1 \leq i \leq n)$ are the vertices for $G_M(S, l)$. There is an edge between C_i, C_j if they intersect and only one pair of sub-squares of them overlap. If at least two (and at most three) pairs of sub-squares of C_i and C_j overlap, then we draw two edges between C_i and C_j. (Recall that two squares overlap if they have a common interior point.)

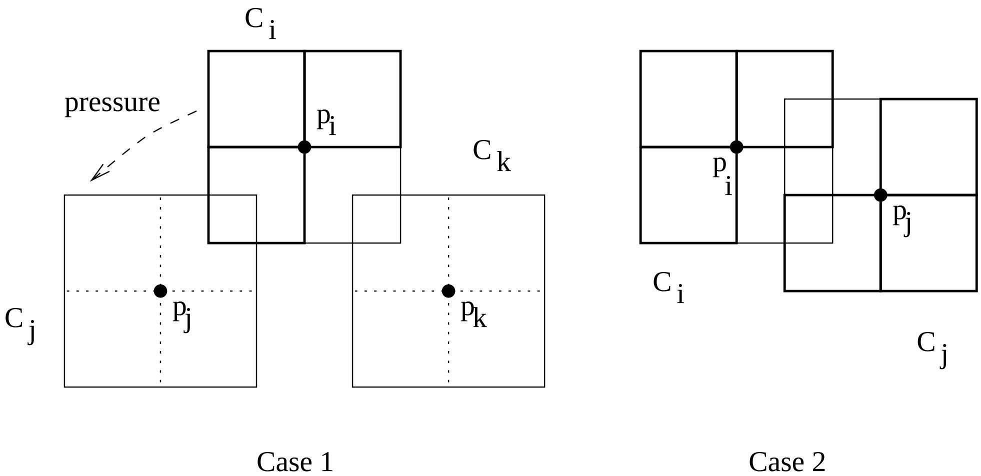


Figure 2. Illustration for the proof of Lemma 3.2.

Lemma 2 *There is a valid labeling for S with square triples of size l if and only if every connected component of $G_M(S, l)$ has at most one cycle.*

Proof: We refer to Figure 2. By the definition of the problem, if we have a cycle in $G_M(S, l)$ which contains no multiple edge then when we label p_i the labeling will generate 'pressure' to either C_j or C_k. In other words, one of the sub-squares of either C_j or C_k will be 'destroyed' and cannot be used as legal label for either p_j or p_k anymore. This holds for all the nodes involved in that cycle. (In Figure 2, Case 1, the labeling of p_i generates 'pressure' on C_j.) If C_i and C_j form a cycle with two edges, similar claim holds, except that we need to label p_i (p_j) with a sub-square which only generates one 'pressure' to C_j (C_i). (See Figure 2, Case 2.) Now we continue with our proof.

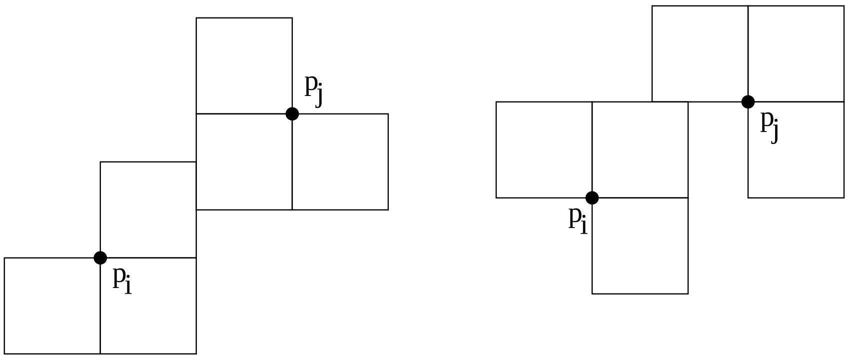
(*Necessity*.) Assume that there is a valid labeling for S with square triples of size l. Then for each C_i, it receives at most one 'pressure', i.e., at most one of its four sub-squares cannot be used as a legal label for p_i. Clearly, that implies that every connected component of $G_M(S, l)$ contains at most one cycle.

(*Sufficiency*.) It is easy to see that a leaf dangling at a cycle has no influence over the labeling of the centers of those nodes involved in a cycle. So we assume that there is no leaf node in any connected component of $G_M(S, l)$. Now if each connected component of $G_M(S, l)$ is a cycle then obviously we can label the centers of all those nodes C_i involved in that cycle. □

Clearly the graph $G_M(S, l)$ has a vertex degree of at most 12. So the graph is of linear size and checking whether the graph contains more than one cycle can be done in linear time with standard graph algorithms. To obtain a polynomial time solution for MLUST under the discrete model, what we are going to do is to prove that there are only a polynomial number of candidates for l^*. Among them, the largest will give us the size of the optimal solution. Let the coordinates of p_i be $(x(p_i), y(p_i))$ and let

$d_{min}(p_i, p_j)$=min$\{|x(p_i) - x(p_j)|, |y(p_i) - y(p_j)|\}$. (Note that $d_\infty(p_i, p_j)$=max$\{|x(p_i) - x(p_j)|, |y(p_i) - y(p_j)|\}$.) We have the following lemma.

Lemma 3 *The size of the optimal solution for MLUST under the discrete model is equal to either $D_\infty(S)$, or $d_\infty(p_i, p_j)/2$ or $d_{min}(p_i, p_j)$ for some i, j, provided that its value is bounded by $D_\infty(S)/2$ and $D_\infty(S)$.*



(a) (b)

Figure 3. Illustration for the proof of Lemma 3.3.

Proof: Notice that if the size of the optimal solution for MLUST under the discrete model is not $D_\infty(S)$ then the reason why we cannot increase the size of the optimal solution must be the following: one of the labels of p_i already touches the label of p_j. In Figure 3 (a), the optimal size is $d_\infty(p_i, p_j)/2$ and in Figure 3 (b) it is $d_{min}(p_i, p_j)$. (For clarity of the figure, we do not show other sites and their labeling in Figure 3.) Because all the labels must have the same size, the lemma simply follows. □

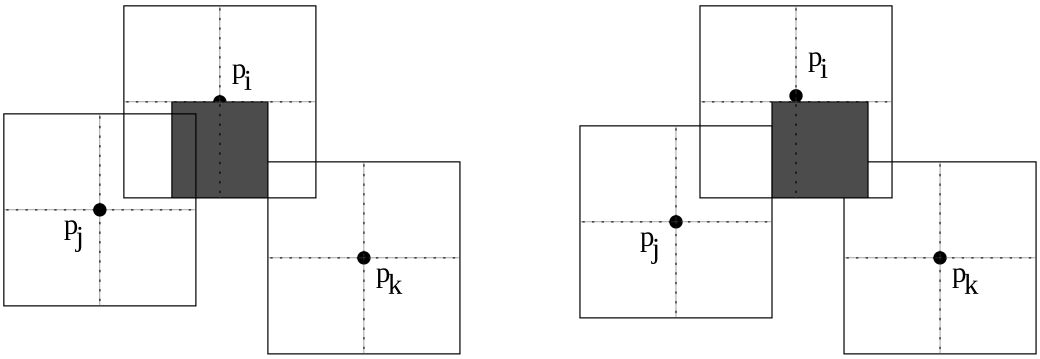
Lemmas 3.1, 3.2 and 3.3 naturally give us the following algorithm. For each site p_i, we look at the axis-parallel square C centered at p_i with edge length $4D_\infty(S)$. Clearly any point p_k out of this square would be at a distance longer than $2D_\infty(S)$, which implies $d_\infty(p_i, p_k)/2$ cannot be the optimal solution value for the problem. As any two sites in C are at least $D_\infty(S)$ distance away, we only need to consider a constant number (24) of points in S which are the closest to p_i. (Overall, for all p_i this can be computed in $O(n \log n)$ time using standard techniques [8].) For each such point p_j, we simply measure $d_\infty(p_i, p_j)$ and $d_{min}(p_i, p_j)$. If $d_\infty(p_i, p_j)/2$ or $d_{min}(p_i, p_j)$ is out of the range $[D_\infty(S)/2, D_\infty(S)]$ then throw it away as a valid candidate. Eventually we have at most $2 \times 24n + 1 = O(n)$ number of candidates. We sort them into a list $l_1, ..., l_{O(n)}$ and then we run a binary search over this list to decide the maximum value l^* such that a valid labeling for S with this size exists. As the decision step, following Lemma 3.2, takes $O(n)$ time, the whole algorithm takes $O(n \log n)$ time. It is easy to show that $\Omega(n \log n)$ is a lower bound under the algebraic decision tree model for the MLUST problem, by a reduction from the element uniqueness problem: given a set of real numbers $\{x_1, ..., x_n\}$, there are two elements which are equal if and only if the MLUST problem for point set $\{(x_1, 0), ..., (x_n, 0)\}$ has a zero solution, under either the discrete or sliding models. Summarizing the above results, we have the following theorem.

Theorem 1 *For any given set of n points in the plane, the above algorithm, which runs in $\Theta(n \log n)$ time, produces an optimal solution for the MLUST problem under the discrete model.*

4 Map Labeling With Uniform Square Triples (MLUST) Under the Sliding Model

In this section, we shall proceed with the more interesting problem of labeling a set of points with sliding square triples. Because of the nature of the problem, not all of the three labels for a site p_i can slide; in fact, only one of them can. We hence call the two discrete sub-squares in the label of p_i *base sub-squares* or *base labels*. Let L^* be the optimal solution of the problem MLUST under the sliding model. We have a lemma similar to Lemma 3.1.

Lemma 4 $D_\infty(S)/2 \leq L^* \leq D_\infty(S)$.



(a) (b)

Figure 4. A pressure-releasing operation.

Our general idea is the same as that for the discrete case. We first try to design a decision procedure which can decide for any $l \in [D_\infty(S)/2, D_\infty(S)]$ whether a valid labeling of S with sliding square triples of size l exists. We follow the same procedure as in the previous section to build a multi-graph $G_M(S,l)$. (The nodes of $G_M(S,l)$ are the squares C_i whose centers are p_i, for all $p_i \in S$.) However, because of the difference between the two models we cannot immediately have a lemma similar to Lemma 3.2. The reason is that the sliding of some label for p_i might simply terminate any 'pressure' its neighbor carries over to it. In Figure 4 (a), if we label p_i using the shaded sliding label then the 'pressure' from p_k vanishes and in Figure 4 (b), if we label p_i using the shaded sliding label then the 'pressure' from p_j vanishes. We call (p_j, p_i, p_k) a critical triple if C_j intersects C_i and C_i intersects C_k. A *pressure-releasing* operation on a critical triple p_j, p_i, p_k is that we label p_i with sliding labels such that the labels generate minimum 'pressure' on either p_j or p_k, i.e., the number of either p_j or p_k's sub-squares destroyed by the labels of p_i is minimized and furthermore, the area of both p_j and p_k's sub-squares destroyed by the labels of p_i is also minimized. (The first condition implies that a pressure-releasing operation destroys either 0 or 1 sub-square of p_j or p_k. The second condition implies that if a pressure-releasing operation has to destroy a sub-square of p_j or p_k then it will destroy the minimum area of it. Finally, it is clear that we can perform at most $O(1)$ pressure-releasing operations on any given critical triple.) In this case we call C_i a *cycle-breaker* in $G_M(S,l)$ and clearly a cycle-breaker will terminate some pressure along some cycle in $G_M(S,l)$ if we perform a pressure-releasing operation on its center. Therefore, we have the following revised version of Lemma 3.2, whose proof is straightforward.

Lemma 5 *There is a valid labeling for S with sliding square triples of size l if and only if every connected component of $G_M(S,l)$ has at most one cycle after a set of pressure-releasing operations are enumerated.*

Notice that different from the discrete MLUST problem, this lemma does not give us a static algorithm as does Lemma 3.2. In fact, at the first glance, it seems that the above lemma gives us an exponential solution. However, we will make use of a specific property of $G_M(S,l)$ to obtain a polynomial time solution.

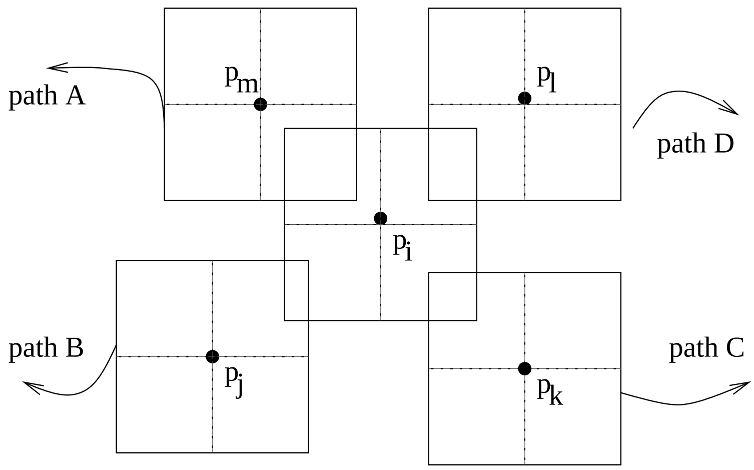


Figure 5. An example for the states of C_i.

What we do is as follows. As the vertex degree of any node in $G_M(S,l)$ is at most a constant (12), we can fix any node C_i and identify all possible states of it. That state is determined by the position of the two base sub-squares as well as along which path adjacent to C_i the 'pressure' will be generated. For example, in Figure 5, if we use $C_i(1), C_i(2), C_i(3)$ and $C_i(4)$ to indicate the corresponding four sub-squares of C_i which are located at the northeast, northwest, southwest and southeast corners of p_i, then the corresponding states for C_i are: $(\{C_i(1), C_i(2)\}, \{A, D\})$, $(\{C_i(2), C_i(3)\}, \{A, B, C\})$, $(\{C_i(2), C_i(3)\}, \{A, B, D\})$, $(\{C_i(3), C_i(4)\}, \{A, B, C\})$, $(\{C_i(3), C_i(4)\}, \{B, C, D\})$, $(\{C_i(4), C_i(1)\}, \{A, C, D\})$ and $(\{C_i(4), C_i(1)\}, \{B, C, D\})$. In this case, $(\{C_i(1), C_i(2)\}, \{A, D\})$ means that the base sub-squares for p_i will be $C_i(1)$ and $C_i(2)$, and this state will generate pressure on path A and D (because in this case there is a gap of at least l between C_j and C_k). Clearly we have $O(1)$ number of states for C_i.

What we do next is to fix a state of C_i and traverse the graph by following those paths carrying 'pressures' generated so far. Suppose that after visiting C_j and successfully labeling p_j (i.e., the current state of p_j is *safe*), we reach at a vertex C_k. If we can find a valid labeling of p_k taking into consideration all the pressures generated on p_k so far, then we set the state of p_k as safe and we continue our traversal. If we reach a *dead* state at p_k, i.e., no valid labeling of p_k exists (in other words, at least two of p_k's sub-squares are destroyed), then we backtrack to C_j and traverse the edge (C_j, C_k) by starting at a different safe state of p_j, if there exists one. If we backtrack to p_i and try out all its safe states and still cannot find a valid labeling for all the sites in S, then a valid labeling of S with square triples of size l does not exist.

At the first sight, this procedure seems to take exponential time. However, the following lemma guarantees a polynomial time solution for fixed l. (Following Lemma 4.2, if we can label all sites corresponding to nodes in $G_M(S,l)$ with square triples of size l then there must exist at least one node in $G_M(S,l)$ which admits a pressure-releasing operation.)

Lemma 6 *In the above procedure, if at each cycle-breaker C_j we minimize the out-going pressure along (C_j, C_k) while withholding the incoming 'pressure' then for each state of C_j there is a unique state for the next cycle-breaker C_k in the same cycle.*

Proof. The correctness of this lemma is due to that if a valid labeling with size l for set S exists, then there must exist one valid labeling starting at some C_i such that at each step the pressure which the current labeling generates is minimized. □

The above lemma basically shows that the procedure in the previous paragraph runs in linear time if we start at a C_i which admits a pressure-releasing operation. (For each cycle-breaker C_j we have $O(1)$ states, when traversing the graph from C_j to the next cycle-breaker C_k we can only reach exactly one state of C_k. This procedure finishes either after we try all states of C_i without finding a valid labeling for S or we terminate with a valid labeling of size l for set S.) However, as in the optimal labeling not all C_i's admit a pressure-releasing operation we need to try the above procedure $O(n)$ times — starting at every possible node in the graph. Therefore, for a fixed l deciding whether we can label S with sliding square triples of size l can be done in $O(n \times n) = O(n^2)$ time. To solve MLUST under the sliding model, we must also make sure that there are only a polynomial number of candidates for L^*. This is guaranteed with the following lemma.

Lemma 7 *The size of the optimal solution for MLUST under the sliding model L^* is equal to either the optimal solution for MLUST under the discrete model or $d_\infty(p_i, p_j)/K$ for some $p_i, p_j \in S$ and some K such that $1 \le K \le n-1$, provided that its value is bounded by $D_\infty(S)/2$ and $D_\infty(S)$.*

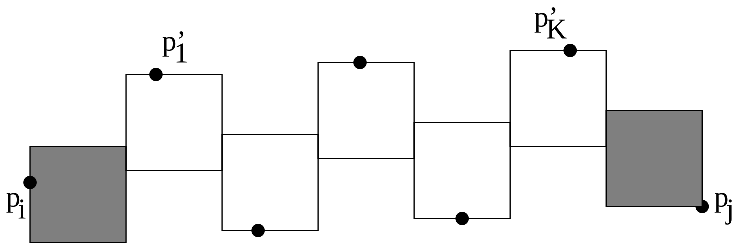


Figure 6. Illustration for the proof of Lemma 4.4.

Proof. It is only necessary to discuss the situation when L^* is not equal to the optimal solution for MLUST under the discrete model. In this situation, the reason that the optimal solution value L^* for MLUST under the sliding model cannot be increased is that there exists a series of K labels $p'_1, ..., p'_K$ touching each other and the sum of their sizes is exactly the distance between some sites p_i and p_j, i.e., each p'_k contributes some distance to fill the L_∞-gap between p_i and p_j (Figure 6). We call (p_i, p_j) an extreme pair. (Note that in Figure 6 we do not show the base labels for all the sites, for the clarity of the figure.) What remains to show is that each p'_k contributes exactly a distance of L^* to fill the L_∞-gap between p_i and p_j, which in turn implies that we do not need to consider any K which is larger than $n-1$. Assume to the contrary that this is not the case, i.e., at least one of the sites, say p'_k, would contribute $2L^*$ to fill the L_∞-gap between p_i and p_j. However, this implies that either (p_i, p'_k) or (p'_k, p_j) would be an extreme pair. □

Similar to the previous section, Lemmas 4.1, 4.2, 4.3 and 4.4 naturally give us the following algorithm. For each pair of sites p_i, p_j, we simply measure $d_\infty(p_i, p_j)$. If any of $d_\infty(p_i, p_j)/K(1 \le K \le n-1)$ is out of the range $[D_\infty(S)/2, D_\infty(S)]$ then we throw it away as a valid candidate. With i, j fixed, we have at most $O(n)$ candidates. In total we have $O(n^3)$ candidates for L^*. (We also need to test all the $O(n)$ candidates for the discrete problem.) We sort them into a list $L_1, ..., L_{O(n^3)}$ in $O(n^3 \log n)$ time and then we run a binary search over this list to decide the maximum value L^* such that a valid labeling for S with such a value exists. As the decision step, following Lemma 4.3, takes $O(n^2)$ time, the whole algorithm takes $O(n^3 \log n + n^2 \times \log n^3) = O(n^3 \log n)$ time. Summarizing the above results, we have the following theorem.

Theorem 2 *For any given set of n points in the plane, there is an $O(n^3 \log n)$ time solution for the MLUST problem under the sliding model.*

In the following, we show that the problem can be solved in $O(n^2 \log n)$ time. Notice that when the sizes of the solutions under the discrete and the sliding model differ, then the size L^* of the optimal solution under the sliding model is of the form $L^* = d_\infty(p, p')/K$ where $(p, p') \in S^2$ and $K \in \{1, 2, \dots n-1\}$. In the following we show that the search for L^* can be performed efficiently, without enumerating all the possible values of L^*, by a decimation argument. The main idea is reminiscent of Blum et al. [3] algorithm for finding a median in linear time.

Let E be a finite set of pairs of real numbers. We denote $W = \sum_{(x,w)\in E} w$ and $W_{<y} = \sum_{x<y,(x,w)\in E} w$. Similarly $W_{>y} = \sum_{x>y,(x,w)\in E} w$. A weighted median m_E of E is such that $(m_E, w_E) \in E$, $W_{<m_E} \leq W/2$ and $W_{>m_E} \leq W/2$. A weighted median can be computed in $O(|E|)$ time (see the book by Cormen et al. [4] page 193).

The set of the distances $d_\infty(p, p')$ for all $(p, p') \in S^2$ is denoted by D. Initially, the search interval (l_1, l_2) is $(0, \infty)$. For all $d \in D$, we denote by $\mathcal{L}_d$ the set of the lengths d/k that belong to (l_1, l_2) and such that $k \in \{1, 2, \dots n-1\}$. We denote by x_d the median of $\mathcal{L}_d$ and w_d is the cardinality of $\mathcal{L}_d$. We can find both these values in $O(1)$ time without computing $\mathcal{L}_d$ explicitly, for instance in case $\mathcal{L}_d$ is non–empty the index k associated with its smallest element is given by $\min(n-1, \lceil \frac{d}{l_1} - 1 \rceil)$. The recursive function described below reduces the search interval (l_1, l_2) and returns the optimal size when it differs from the solution to the discrete problem.

Algorithm *decimate*(l_1, l_2)

1. compute $E = \{(x_d, w_d) \mid d \in D\}$
2. **if** $W = 0$
3.   **then** return l_1
4.   **else** compute the weighted median m_E
5.   **if** the sliding decision problem (P, m_E) has a solution
6.   **then** return $decimate(m_E, l_2)$
7.   **else** return $decimate(l_1, m_E)$

Correctness of this algorithm follows from previous discussions and the following invariant: the decision problem with parameters (S, l_1) has a solution and the one with parameters (S, l_2) has no solution. Now we prove that it runs in $O(n^2 \log n)$ time. First note that each call to *decimate*, ignoring the recursive calls, takes $O(n^2)$ time. Indeed, step 1 can be performed in constant time per $d \in D$ as was explained in the previous paragraph, and step 5 can be performed in $O(n^2)$ time. So we only need to prove that a constant fraction of $\mathcal{L} = \bigcup_{d\in D} \mathcal{L}_d$ is discarded at each recursive call.

Lemma 8 *At each recursive call to $decimate$, the cardinality of $\mathcal{L}$ shrinks by a factor at least $4/3$.*

Proof: Note that $|\mathcal{L}| = W$. We show that $|\mathcal{L} \cap (l_1, m_E)| \leq 3W/4$, the proof that $|\mathcal{L} \cap (m_E, l_2)| \leq 3W/4$ is similar. For all $d \in D$ such that $x_d \geq m_E$, at least $w_d/2$ lengths in $\mathcal{L}_d$ that are greater or equal to x_d are also greater or equal to m_e, therefore they do not appear in $\mathcal{L} \cap (l_1, m_E)$. Therefore $|\mathcal{L} \cap (l_1, m_E)| \leq W - \frac{1}{2}\sum_{x_d \geq m_E} w_d$. By the definition of the weighted median $|\mathcal{L} \cap (l_1, m_E)| \leq W - \frac{1}{4}W$. □

Putting everything together, we have the following theorem.

Theorem 3 *The MLUST problem under the sliding model can be solved in $O(n^2 \log n)$ time.*

Proof: First we solve the problem under the discrete model in $O(n \log n)$ time and obtain an optimal size l^*. Then we compute $l_1^* = decimate(0, \infty)$, it runs in $O(n^2)$ time per level of recursion, and it recurses $O(\log n)$ times by Lemma 4.6, so the whole process takes $O(n^2 \log n)$ time. Clearly, the maximum of l^* and l_1^* is L^*. □

5 Concluding Remarks

In this paper, we investigate the new problem of labeling point sites with uniform square triples, under either the discrete or sliding models. We present an optimal $\Theta(n \log n)$ time algorithm for the discrete problem and an $O(n^2 \log n)$ time solution for the problem under the sliding model. This is significantly different from the problem of labeling point sites with uniform square pairs, which is NP-hard under both the discrete and sliding models. An immediate question is whether we can reduce the gap between the $\Omega(n \log n)$ lower bound and the $O(n^2 \log n)$ upper bound for the general problem. It is interesting to know whether a near linear time algorithm can be designed. In order to obtain a subquadratic time bound, a very different method would have to be discovered that does not make use of the current $O(n^2)$ time decision algorithm. Another interesting practical extension for all the research in multi-label point labeling would be allowing the labels to have different shapes.

References

[1] P. Agarwal, M. van Kreveld and S. Suri. Label placement by maximum independent set in rectangles. *Comp. Geom. Theory and Appl.*, 11:209-218, 1998.

[2] D. Beus and D. Crockett. Automated production of 1:24,000 scale quadrangle maps. In *Proc. 1994 ASPRS/ACSM Annual Convention and Exposition*, volume 1, pages 94–99, 1994.

[3] M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest and R. E. Tarjan. Time Bounds for Selection. *J. of Computer and System Sciences*, 7(4):448-461, 1973.

[4] T. H. Cormen, C. E. Leiserson and R. L. Rivest. *Introduction to algorithms.* MIT Press, 1990.

[5] J. Christensen, J. Marks, and S. Shieber. Algorithms for cartographic label placement. In *Proc. 1993 ASPRS/ACSM Annual Convention and Exposition*, volume 1, pages 75–89, 1993.

[6] J. Christensen, J. Marks, and S. Shieber. An Empirical Study of Algorithms for Point-Feature Label Placement, *ACM Transactions on Graphics*,14:203-222, 1995.

[7] J. Doerschler and H. Freeman. A rule-based system for cartographic name placement. *CACM*, 35:68–79, 1992.

[8] A. Datta, II.-P. Lenhof, C. Schwarz, and M. Smid. Static and dynamic algorithms for k-point clustering problems, In *Proc. 3rd Worksh. Algorithms and Data Structures,* springer-verlag, LNCS 709, pages 265-276, 1993.

[9] S. Doddi, M. Marathe and B. Moret, Point set labeling with specified positions. In *Proc. 16th Annu. ACM Sympos. Comput. Geom.*, pages 182-190, June, 2000.

[10] S. Doddi, M. Marathe, A. Mirzaian, B. Moret and B. Zhu, Map labeling and its generalizations. In *Proc. 8th ACM-SIAM Symp on Discrete Algorithms (SODA'97)*, New Orleans, LA, pages 148-157, Jan, 1997.

[11] M. Formann and F. Wagner. A packing problem with applications to lettering of maps. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 281–288, 1991.

[12] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, CA, 1979.

[13] E. Imhof. Positioning names on maps. *The American Cartographer*, 2:128–144, 1975.

[14] C. Iturriaga and A. Lubiw. Elastic labels: the two-axis case. In *Proc. Graph Drawing'97*, pages 181–192, 1997.

[15] C. Jones. Cartographic name placement with Prolog. *Proc. IEEE Computer Graphics and Applications*, 5:36–47, 1989.

[16] D. Knuth and A. Raghunathan. The problem of compatible representatives. *SIAM J. Disc. Math.*, 5:422–427, 1992.

[17] M. van Kreveld, T. Strijk and A. Wolff. Point set labeling with sliding labels. *Comp. Geom. Theory and Appl.*, 13:21-47, 1999.

[18] K. Kakoulis and I. Tollis. An algorithm for labeling edges of hierarchical drawings. In *Proc. Graph Drawing'97*, pages 169–180, 1997.

[19] K. Kakoulis and I. Tollis. A unified approach to labeling graphical features. *Proc. 14th Annu. ACM Sympos. Comput. Geom.*, pages 347–356, 1998.

[20] K. Kakoulis and I. Tollis. On the multiple label placement problem. In *Proc. 10th Canadian Conf. on Comput. Geom.*, pages 66–67, 1998.

[21] F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[22] C.K. Poon, B. Zhu and F. Chin, A polynomial time solution for labeling a rectilinear map. *Inform. Process. Lett.*, 65(4):201-207, Feb, 1998.

[23] Z.P. Qin, A. Wolff, Y. Xu and B. Zhu, New algorithms for two-label point labeling. *Proc. 8th European Symp. on Algorithms (ESA'00)*, pages 368–379, Springer-Verlag, LNCS series 1879, 2000.

[24] M. Spriggs. On the complexity of labeling maps with square pairs. *manuscript*, 2000.

[25] M. Spriggs and M. Keil. A new bound for map labeling with uniform circle pairs. *Inform. Process. Lett.*, submitted, 2000.

[26] T. Strijk and A. Wolff. Labeling points with circles. Tech Report B 99-08, Institut fur Informatik, Freie Universitat, Berlin, April, 1999.

[27] F. Wagner. Approximate map labeling is in $\Omega(n \log n)$. *Inform. Process. Lett.*, 52:161–165, 1994.

[28] A. Wolff, M. Thon and Y. Xu. "A better lower bound for two-circle point labeling," In *Proc. 11th Intl Symp. on Algorithms and Computation (ISAAC'00)*, pages 422–431, Springer-Verlag, LNCS series 1969, 2000.

[29] F. Wagner and A. Wolff. Map labeling heuristics: Provably good and practically useful. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 109–118, 1995.

[30] B. Zhu and C.K. Poon. Efficient approximation algorithms for two-label point labeling, *Intl. J. Computational Geometry and Applications*, 11(4):455–464, 2001.

[31] B. Zhu and Z.P. Qin. New approximation algorithms for map labeling with sliding labels, *J. Combinatorial Optimization*, 6(1):99–110, 2002.