New Algorithms for Two-Label Point Labeling*

Zhongping Qin† Alexander Wolff‡ Yinfeng Xu§ Binhai Zhu¶

Abstract

Given a label shape L and a set of n points in the plane, the two-label point-labeling problem consists of placing $2n$ non-intersecting translated copies of L of maximum size such that each point touches two unique copies—its labels.

In this paper we give new and simple approximation algorithms for L an axis-parallel square or a circle. For squares we improve the best previously known approximation factor from $\frac{1}{3}$ to $\frac{1}{2}$. For circles the improvement from $\frac{1}{2}$ to roughly 0.5125 is less significant, but the fact that $\frac{1}{2}$ is not best possible is interesting in its own right. For the decision version of the latter problem we have an NP-hardness proof that also shows that it is NP-hard to approximate the label size beyond a factor of ≈ 0.7321. We keep the $O(n \log n)$ time and $O(n)$ space bounds of the previous algorithms.

The algorithm we propose for the two-square labeling problem actually solves a different problem. It labels points with rectangles of height-width ratio 2 of maximum size in one of four positions. It is the first point-labeling algorithm that combines the following properties. It allows more than two label positions *and* solves the size maximization problem optimally in polynomial time. This algorithm also improves the approximation factor of the only known algorithm for Knuth and Raghunathan's Metafont labeling problem from $\frac{1}{3}$ to $\frac{1}{2}$.

1 Introduction

Label placement is one of the key tasks in the process of information visualization. In diagrams, maps, technical or graph drawings, features like points, lines, and polygons must be labeled to convey information. The interest in algorithms that automate this task has increased with the advance in type-setting technology and the amount of information to be visualized. Due to the computational complexity of the label-placement problem, cartographers, graph drawers, and computational geometers have suggested numerous approaches, such as expert systems [AF84, DF89], zero-one integer programming [Zor90], approximation algorithms [FW91, DMM+97, WW97, ZP99], simulated annealing [CMS95] and force-driven algorithms [Hir82] to name only a few. The ACM Computational Geometry Impact Task Force report [C+96] denotes label placement as an important research area. Manually labeling a map, e.g., is a tedious task that is estimated to take 50 % of total map production time [Mor80].

In this paper we deal with a relatively new variant of the general label placement problem, namely the two-label point-labeling problem. It is motivated by maps used for weather forecasts, where each city must be labeled with two axis-parallel labels that contain the city's name and its predicted temperature or rainfall probability.

*This research was conducted during a visit of Z. Qin and Y. Xu to City University of Hong Kong and of A. Wolff to Hong Kong University of Science and Technology. Our work was supported by NSF of China, grant No. 19731001, and by the Hong Kong RGC CERG grants CityU-1103/99E and HKUST-6144/98E.

†Dept. of Mathematics, Huazhong University of Science and Technology, Wuhan, China, and Dept. of Computer Science, City University of Hong Kong. Email: zqin@cs.cityu.edu.hk

‡Institute of Mathematics and Computer Science, Ernst Moritz Arndt University Greifswald, Jahnstraße 15a, D-17489 Greifswald, Germany. Email: awolff@mail.uni-greifswald.de

§School of Management, Xi'an Jiaotong University, Xi'an, People's Republic of China. Email: yfxu@xjtu.edu.cn

¶Dept. of Computer Science, City University of Hong Kong, Kowloon, Hong Kong, China. Email: bhz@cs.cityu.edu.hk

The two-label point-labeling problem is a variant of the one-label problem that allows sliding. Sliding labels can be attached to the point they label anywhere on their boundary. They were first considered by Hirsch [Hir82] who gave an iterative algorithm that uses repelling forces between labels in order to eventually find a placement without or only few intersecting labels. Van Kreveld et al. gave a polynomial time approximation scheme and a fast factor-2 approximation algorithm for maximizing the number of points that are labeled by axis-parallel sliding rectangular labels of common height [vKSW99]. They also compared several sliding-label models with so-called fixed-position models where only a finite number of label positions (or *label candidates*) per point is considered, usually a small constant like four [FW91, CMS95, WW97]. Sliding rectangular labels have also been considered for labeling rectilinear line segments [KSY99]. Another generalization was investigated in [DMM$^+$97, ZQ00], namely arbitrarily oriented sliding labels.

Point labeling with circular labels, though not as relevant for real-world applications as rectangular labels, is a mathematically interesting problem. The one-label case has already been studied extensively [DMM$^+$97, SW99, DMM00]. For maximizing the label size, the best approximation factor now is $\frac{1}{3.6}$ [DMM00].

The two- or rather multi-label labeling problem was first considered by Kakoulis and Tollis who presented two heuristics for labeling the nodes and edges of a graph drawing with several rectangles [KT98]. Their aim was to maximize the number of labeled features. The algorithms are based on their earlier work; one is iterative, while the other uses a maximum-cardinality bipartite matching algorithm that matches cliques of label candidates with the elements of the graph drawing that are to be labeled. They do not give any runtime bounds or approximation factors.

For the two problems that we will consider in this paper, namely maximizing the size of axis-parallel square and circular labels, two per point, Zhu and Poon gave the first approximation algorithms [ZP99]. They achieved approximation factors of $\frac{1}{4}$ and $\frac{1}{2}$ for square and circle pairs, respectively. Both algorithms rely on the fact that there is a region around each input point or pair of input points, such that the labels can be placed within this region and without intersecting the regions of any other points. The size of the region and thus the labels is a constant fraction of an upper bound for the maximum label size. Recently Zhu and Qin improved the result for pairs of square labels to a factor of $\frac{1}{3}$ [ZQ00]. They exploit the structure of a graph that has a node for each input point and an edge for each pair of points closer than $\frac{2}{3}$ times an upper bound for the maximum label size.

In this paper we give new and simple approximation algorithms for the two-square and the two-circle point-labeling problem. For squares we improve the approximation factor of Zhu and Qin's algorithm from $\frac{1}{3}$ to $\frac{1}{2}$. For circles we present a factor-0.5125 approximation algorithm. There, the improvement over Zhu and Poon's factor-$\frac{1}{2}$ approximation algorithm is less significant, but the fact that $\frac{1}{2}$ is not best possible is interesting in its own right. For the decision version of the two-circle labeling problem we have an NP-hardness proof that also shows that it is NP-hard to approximate the label size beyond a factor of ≈ 0.7321. Other than all previous approximation algorithms for two-label placement, our new algorithms do not necessarily have to compute an upper bound for the maximum label size explicitly. We keep the $O(n \log n)$ time and $O(n)$ space bounds of the previous algorithms.

Although the two new algorithms use completely different approaches, in both cases the improved approximation factor is made possible by restricting the search to a subset of the solution space. Within this subset, optimal solutions can be computed easily and their labels are at most by the above mentioned constant factors off the maximum label size. The subproblem we solve optimally for square labels is a special case of the point-labeling problem that can be solved optimally in polynomial time although it allows *four* label positions per point. Our algorithm for this subproblem also improves the approximation factor of the only known algorithm [FW93] for Knuth and Raghunathan's Metafont labeling problem [KR92] from $\frac{1}{3}$ to $\frac{1}{2}$.

This paper is structured as follows. In Section 2 we present a new algorithm for two-square point labeling and the new algorithm for one-rectangle four-position labeling on which the former is based. We also show how the latter algorithm can be used for solving the Metafont labeling problem approximately. In Section 3 we give a new algorithm for two-circle labeling and state our non-approximability result.

Throughout this paper we consider labels being topologically *open*, and we define the *size* of a solution to be the diameter in the case of circular labels and the length of the shorter label edge in the case of rectangular labels. We refer to a label placement as *feasible* if no two labels intersect and as *optimal* if additionally labels have the largest possible size. We will only consider $n > 2$.

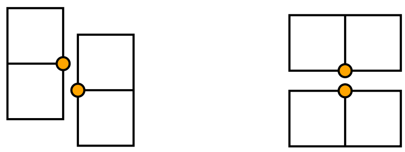
2 Two-Square Labeling

The problem we consider in this section is defined as follows.

Definition 1 (two-square point-labeling problem) *Given a set P of n points in the plane, find a set of $2n$ axis-parallel, uniform, non-intersecting, maximum-size open squares, such that each point touches two unique squares.*

The uniqueness contraint does not forbid a point to touch more than two squares, but ensures there is a map that assigns to each point exactly two squares that touch it.

The first approximation algorithm for the two-square point-labeling problem was suggested by Zhu and Poon [ZP99]. This algorithm labels pairs of points in order of increasing distance. The labels are placed to the left of the left point and to the right of the right point — except when the two points lie on a vertical, see Figure 1 (a) and (b), respectively. In other words, the algorithm does not really place two square labels at each point but one rectangle of height-width ratio 2 or 1/2. The rectangle is attached to its point in the midpoint of either of its long edges, i.e. the algorithm uses only *four* of the infinitely many possible label positions.



(a) (b)

Figure 1: Label placement of the two-square labeling algorithm of Zhu and Poon.

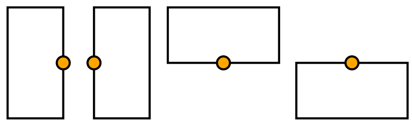


Figure 2: Label candidates in the one-rectangle four-position problem.

We use this observation as follows. First we device an $O(n \log n)$-time algorithm for the one-rectangle four-position point-labeling problem, where all points are labeled with maximum-size rectangles in one of the four discrete positions depicted in Figure 2. Then we show how an optimal solution for the two-square labeling problem can be transformed into a solution of the one-rectangle labeling problem by using rectangles of half the square size, see Figure 3. Thus the one-rectangle labeling algorithm already yields a factor-$\frac{1}{2}$ approximation algorithm for the two-square labeling problem.

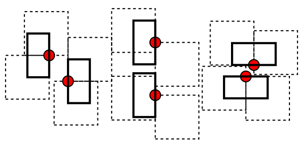


Figure 3: Mapping a two-square labeling to a one-rectangle four-position labeling of half the size.

Apart from the improved approximation factor, our approach has the advantage that we do not have to compute an upper bound for the maximum label size explicitly. Both previous algorithms [ZP99, ZQ00] first have to determine such an upper bound before they can actually place labels whose size depends on this upper bound. Zhu and Poon use $D_{3,\infty}$, the minimum over the diameters of all 3-subsets of the input points, as an upper bound for the maximum label size. $D_{3,\infty}$ can be computed in $O(n \log n)$ time [DLSS95]. Instead of computing $D_{3,\infty}$ we preprocess the input by computing relevant adjacency information, i.e. for each input point we find a constant number of rectilinear nearest neighbors (in the L_∞-metric). For this task a simple $O(n \log n)$ algorithm is known [For92]. Of course $D_{3,\infty}$ can be computed in linear time from nearest neighborhood data, but we make better use of this information by solving the following problem optimally.

Definition 2 (one-rectangle four-position point-labeling problem) *Given n points in the plane, find a set of n congruent, axis-parallel, non-intersecting, maximum-size open rectangles of height-width ratio 2 or 1/2, such that each point touches a rectangle in the midpoint of one of its longer edges, and each rectangle touches a point.*

The decision version of this problem is the question whether a set of points can be labeled with congruent rectangles of a *given* size. If we encode the four label candidates of each point p by *two* Boolean variables p_1 and p_2 as in Figure 4, we can construct a 2-SAT formula that is equivalent to the decision version of our problem. Thus it can be solved in time and space linear in the number of clauses [EIS76]. Here the number of clauses is at most three times the number of pairs of intersecting label candidates, which must be computed beforehand. A similar strategy has been applied to encode label positions for labeling rectilinear line segments with rectangles of maximum height [PZC98].

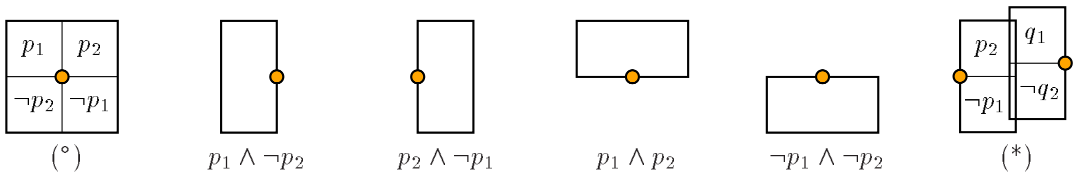


Figure 4: We encode the four label positions of a point p by the values of two Boolean variables p_1 and p_2 ($^\circ$). A label intersection (*) can then be read as $\neg((p_2 \wedge q_1) \vee (p_2 \wedge \neg q_2) \vee (\neg p_1 \wedge \neg q_2))$. This is equivalent to the 2-SAT formula $(\neg p_2 \vee \neg q_1) \wedge (\neg p_2 \vee q_2) \wedge (p_1 \vee q_2)$.

In order to device an algorithm that maximizes the rectangle size and runs in $O(n \log n)$ time, we use the same strategy as two existing algorithms. The algorithms *AS4a* [FW91] and *B* [WW97] also try to maximize the label size. They solve the one-square four-position point-labeling problem, where labels are restricted to uniform *squares*, and each label must be placed such that one of its *corners* coincides with the point it labels. Like in our problem each point has four label candidates. However, in their problem the candidates only intersect at their borders, while in ours each candidate is completely contained in (the closure of) two others. This is why we can solve our problem optimally, while *AS4a* and *B* are factor-$\frac{1}{2}$ approximation algorithms.

Algorithm B proceeds as follows. First it uses the above mentioned simple sweep-line algorithm to detect a constant number of rectilinear nearest neighbors for each point [For92]. Then B computes a list of *conflict sizes* from the set N of pairs of neighboring points. A conflict size is a label size for which two label candidates touch but do not contain any input points. Algorithm B does a binary search on the list of conflict sizes. This is sufficient since the intersection graph of the label candidates does not change between two conflict sizes. For each conflict size, B solves the decision problem in three steps. First the algorithm extracts the intersection graph for the current label size from N. Then it uses certain (partly heuristical) rules to simplify the intersection graph until each point has at most *two* label candidates left. Finally B makes a 2-SAT clause for each edge of the intersection graph and tries to find a satisfying truth assignment for the resulting 2-SAT formula. If one exists, it corresponds to a label placement and the search is continued with a greater label size, otherwise with a smaller one.

Asymptotically B does not need more space than the size of N, which is linear in n. Clearly the binary search consists of $O(\log n)$ tests. Again due to the linear size of N, each test takes $O(n)$ time. This adds up to $O(n \log n)$ time in total.

Our algorithm for one-rectangle four-position point labeling differs from B only in that it is not necessary to reduce the number of candidates of each point from four to two, since we can immediately encode our label candidates by a 2-SAT formula. All that remains to show is that as for one-square four-position point labeling it is sufficient to consider the conflict sizes induced by a constant number of rectilinear nearest neighbors per point.

Lemma 1 *Let r_{opt} be the size of an optimal solution for the one-rectangle four-position point-labeling problem. Then for any label size $\rho \le r_{\text{opt}}$ the label candidates of a point $p \in P$ can intersect only candidates of the 17 points $q \in P \setminus \{p\}$ that are closest to p in the L_∞-metric.*

Proof. Refer to Figure 5 (a). There all points that intersect one of the four label candidates of p must lie within region A. The labels of these points must in turn lie inside region B. The area of B is $36\rho^2$, thus if B contains more than 18 input points there cannot be a solution for them with labels of size $2\rho^2$. ◻

This observation is analogous to Lemma 1 in [FW91]. The algorithm $AR2$ in that paper, however, needs $O(n \log n)$ time for the *decision* version of a problem similar to our one-rectangle four-position labeling problem. (There only two square labels per point are allowed.) By a more careful analysis as in Theorem 1 in [WW97] one can show that it is sufficient to consider much fewer rectilinear nearest neighbors per point. From Lemma 1 the time and space complexity of our one-rectangle four-position labeling algorithm follow immediately. Thus we have

Lemma 2 *An optimal solution of the one-rectangle four-position point-labeling problem can be computed in $O(n \log n)$ time using linear space.*

Following the ideas outlined above we get

Theorem 1 *A feasible solution of the two-square point-labeling problem with squares of at least half the optimal size can be computed in $O(n \log n)$ time using linear space.*

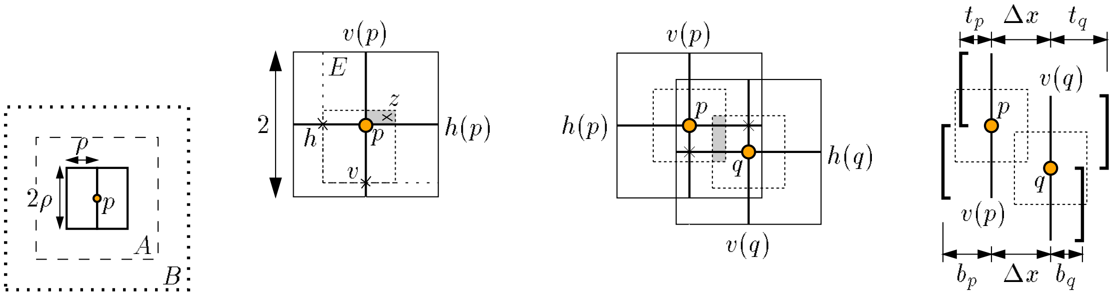
Proof. Given Lemma 2, we only have to show that an optimal solution of the one-rectangle four-position labeling problem always represents a feasible solution of the two-square labeling problem of at least half the optimal size. In other words, if r_{opt} and s_{opt} are the sizes of optimal solutions of the one-rectangle and the two-square labeling problem for the same point set P, then we have to prove $r_{\text{opt}} \ge s_{\text{opt}}/2$. To show this we map a hypothetical optimal solution of the latter problem, i.e. a set $\mathcal{S}_{\text{opt}}$ of $2n$ squares of size s_{opt} into a feasible solution of the former problem, namely a set $\mathcal{R}$ of n rectangles of size $s_{\text{opt}}/2$. Since the labels in $\mathcal{R}$ cannot be larger than r_{opt}, the size of labels in an optimal solution of the one-rectangle labeling problem, we have $r_{\text{opt}} \ge s_{\text{opt}}/2$.

The mapping is simple: for each point p we choose from its four label candidates the rectangle R_p such that a rectangle of twice the size of R_p (with p as stretching center) has the largest area of intersection with the two square labels of p in $\mathcal{S}_{\text{opt}}$. For our proof it does not matter that R_p is not uniquely defined.

It remains to show why R_p does not intersect the label R_q of some point $q \in P \setminus \{p\}$. Let $s_{\text{opt}} = 1$; the instance can always be scaled such that this is true. Recall that all labels in $\mathcal{S}_{\text{opt}}$ and $\mathcal{R}$ are topologically open. For $A \subseteq \mathbb{R}^2$ let $\overline{A}$ be the topological closure of A. For $p = (x_p, y_p)$ define the line segments $v(p)$ and $h(p)$ as the intersection of an open unit disk with the vertical and the horizontal through p, respectively. Let T_p and B_p be the top- and bottommost square labels of p in $\mathcal{S}_{\text{opt}}$; if their y-coordinates are the same, let T_p be the leftmost. Let $S_p = T_p \cup B_p$. We will use the same notations for q.

Observe that $v(p) \subseteq \overline{S_p}$ or $h(p) \subseteq \overline{S_p}$, otherwise there are points $h \in h(p) \setminus \overline{S_p}$ and $v \in v(p) \setminus \overline{S_p}$ that delimit region E in Figure 5 (b). Since E must contain S_p completely, we would have $T_p \cap B_p \ne \emptyset$.

Now suppose $R_p \cap R_q \ne \emptyset$. Then the distance $d_\infty(p, q)$ of p and q in the L_∞-metric is less than 1 since all points in R_p (R_q) have distance less than $\frac{1}{2}$ from p (q). For this reason $v(p)$ and $h(q)$ as well as $v(q)$ and $h(p)$ intersect, see Figure 5 (c). Since S_p and S_q do not intersect by definition,



(a) (b) (c) (d)

Figure 5: (a) Only the labels of the 31 nearest neighbors of a point p can intersect those of p. (b) If there are points $v \in v(p) \setminus \overline{S_p}$ and $h \in h(p) \setminus \overline{S_p}$ then there is a point $z \in T_p \cap B_p$. (c) If the rectangular labels R_p and R_q intersect, then $v(p) \cap h(q) \neq \emptyset$ and $v(q) \cap h(p) \neq \emptyset$. (d) t_p (b_p) is the horizontal distance between the left edge of T_p (B_p) and $v(p)$.

the observation above yields that $v(p)$ is contained in $\overline{S_p}$ and $v(q)$ in $\overline{S_q}$ or $h(p)$ is contained in $\overline{S_p}$ and $h(q)$ in $\overline{S_q}$. W.l.o.g. we may assume the former, otherwise we rotate the whole instance by 90 degrees around p. Thus T_p lies above p, and B_p below p; the same holds for q. We can also assume that $x_p \leq x_q$, otherwise we mirror our instance.

Let $\Delta x = x_q - x_p$. Clearly $0 \leq \Delta x \leq d_\infty(p, q) < 1$. Let t_p (b_p) be the horizontal distance between the *left* edge of T_p (B_p) and $v(p)$, see Figure 5 (d). Similarly, let t_q (b_q) be the horizontal distance between the *right* edge of T_q (B_q) and $v(q)$.

Now a simple packing argument will yield the contradiction. Since T_p and T_q do not intersect, we have $t_p + \Delta x + t_q \geq 2$. Since B_p and B_q do not intersect, we have $b_p + \Delta x + b_q \geq 2$. These inequalities sum up to $\Sigma := t_p + t_q + b_p + b_q \geq 4 - 2\Delta x$. Since $0 \leq \Delta x < 1$ we only have to consider the following two cases:

A) $\frac{1}{2} \leq \Delta x < 1$. Then $\Sigma > 2$ and $t_p + b_p \leq 1$, otherwise — due to our mapping — R_p lies completely left of $v(p)$ and cannot intersect R_q due to $\Delta x \geq \frac{1}{2}$. Similarly $t_q + b_q \leq 1$, otherwise R_q lies completely to the right of $v(q)$. Thus $t_p + t_q + b_p + b_q \leq 2$, contradicting $\Sigma > 2$.

B) $0 \leq \Delta x < \frac{1}{2}$. Then $\Sigma > 3$. Since t_p, t_q, b_p, and b_q are all at most 1, we get $t_p + b_p > 1$ and $t_q + b_q > 1$. But then R_p lies completely to the left of $v(p)$ and R_q to the right of $v(q)$ — which means they do not intersect since we assumed $x_p \leq x_q$.

This completes the proof of Theorem 1. ◻

Obviously the one-rectangle four-position algorithm solves an instance P of the two-square labeling problem optimally if P has an optimal solution that uses only the four label positions depicted in Figure 1, as in a rectangular grid $G_n = \{(x, 2y) \mid 1 \leq x, y \leq n\}$ for $n \geq 2$.

However, our new one-rectangle four-position algorithm can also be used to find approximate solutions for another problem, namely the optimization version of the Metafont labeling problem [KR92]. Knuth developed the program Metafont as a tool for font design. In their paper, Knuth and Raghunathan introduced the name *problem of compatible representatives* for a large class of combinatorial problems that are NP-hard in general. This class of problems has been studied independently in the artificial intelligence community under the name *constraint satisfaction problem* (CSP). Knuth and Raghunathan investigated several special cases of the problem of compatible representatives, among them the Metafont labeling problem, where a set of n grid points is to be labeled with n 2×2 squares such that each square touches exactly one point in the center of one of its edges, see Figure 6. They showed that this one-square four-position point labeling problem is NP-complete. Note that the one-rectangle four-position problem considered in this section is *not* a more general case of the Metafont labeling problem; each of our rectangular label candidates is the union of two other candidates of the same point, while this is not the case with the square Metafont labels.

Formann and Wagner saw the connection between the Metafont labeling problem and the cartographic map-labeling problem they had attacked earlier [FW91]. They defined a maximization version of the Metafont labeling problem by dropping the grid constraint that comes from the Metafont application [FW93]. The problem they considered is as follows.

Definition 3 (Metafont optimization problem) *Given n points in the plane, find a set of n uniform, axis-parallel, non-intersecting, maximum-size open squares, such that each point touches* exactly *one square in the midpoint of one of its edges.*

They gave a factor-$\frac{1}{3}$ approximation algorithm that uses the same techniques and has the same time and space complexity as the algorithm *AS4a* cited above [FW91].

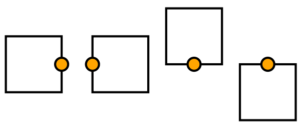


Figure 6: The four label positions allowed in Metafont labeling.

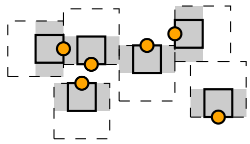


Figure 7: Maping an optimal Metafont solution via a rectangle solution to a Metafont solution of half the optimal size.

Obviously we can approximate the Metafont optimization problem exactly in the same way as the two-square point-labeling problem, namely by running our one-rectangle four-position labeling algorithm. The mapping that transforms an optimal Metafont labeling to a feasible rectangle labeling is the same as for the two-square problem. Here, however, it's trivial to show the feasibility of the resulting rectangle labeling, since each rectangle (shaded grey) is completely contained in a Metafont label (dashed), and no two Metafont labels intersect, see Figure 7. Finally we map each rectangle back to an inscribed square of half the optimal size (printed bold) in one of the four allowed positions of Figure 6.

This mapping automatically ensures the new uniqueness requirement of Definition 3, namely that each input point touches *exactly* one square label in the midpoint of one of its edges. In a solution produced via the rectangle-labeling algorithm that is not necessarily the case, but the algorithm can be adjusted to disallow label placements where two labels completely share a long edge. Thus we can summarize as follows.

Theorem 2 *A feasible solution of the Metafont optimization problem with squares of at least half the optimal size can be computed in $O(n \log n)$ time using linear space.*

3 Two-Circle Labeling

The problem we consider in this section is defined as follows.

Definition 4 (two-circle point-labeling problem) *Given a set P of n points in the plane, find a set of $2n$ uniform, non-intersecting, maximum-size open circles such that each point touches exactly two circles.*

Zhu and Poon [ZP99] have suggested the first approximation algorithm for this problem. Their algorithm always finds a solution of at least half the optimal size. The algorithm is very simple; it relies on the fact that D_2, the minimum Euclidean distance between any two points in P is an upper bound for the optimal label size (i.e. diameter), see Figure 9. On the other hand, given two points p and q in P, open circles $C_{p,D_2/2}$ and $C_{q,D_2/2}$ with radius $\frac{1}{2}D_2$ centered at p and q do not intersect. Thus if each point is labeled within its circle, no two labels will intersect. Clearly this allows labels of maximum diameter $\frac{1}{2}D_2$, i.e. half the upper bound for the optimal label size.

The difficulty of the problem immediately comes into play when increasing the label diameter d beyond $\frac{1}{2}D_2$, since then the intersection graph of the circles $C_{p,d}$ of all points p in P changes abruptly; the maximum degree jumps from 0 to 6.

Our approach also assigns each point a certain region such that no two regions intersect and each point can be labeled within its region. The regions we use are not circles but the cells of the *Voronoi diagram* of P, a well-known multi-purpose geometrical data structure [Aur91]. It is the dual of the *Delaunay triangulation* $\mathrm{DT}(P)$ that our algorithm needs to compute in a preprocessing step. $\mathrm{DT}(P)$ is a planar geometric graph with vertex set P and edges for each pair of points that can be placed on the boundary of an open disc that does not contain any other points of P [Aur91]. For the description of our algorithm, see Figure 8, we need the following notation.

Definition 5 *Let* $d = \frac{D_2}{1+\cos 18°} \approx 0.5125\, D_2$. *An edge of the Delaunay triangulation* $\mathrm{DT}(P)$ *of* P *is* short *if its length is less than* $2d \approx 1.025\, D_2$, long *otherwise.*

Given two lines that intersect at an angle of less than 90 degrees, we call the union of the two smaller regions into which the plane is divided a wedge. *Our wedges are topologically open.*

Given two non-parallel short edges incident to a point p in P, we say that the wedge defined by the two lines containing the edges is free *if it does not contain any short edges incident to p.*

Let $\mathrm{Vor}(p) = \{x \in \mathbb{R}^2 \mid d(x,p) < d(x,q)\ \forall q \in P \setminus \{p\}\}$ *be the* Voronoi cell *of p, where $d(x,y)$ denotes the Euclidean distance of two points x and y in the plane.*

```
Two_Circle_Point_Labeling(P)

Compute DT(P) and D_2 (the length of the shortest edge in DT(P)).
Let d = D_2/(1+cos 18°) ≈ 0.5125 D_2 be the label diameter.
Delete all long Delaunay edges (i.e. of length ≥ 2d).
for all p ∈ P do
    Compute the largest free wedge W of p.
    Place the centers of the labels of p on the bisector of W at distance d/2 from p.
end
```

Figure 8: Our two-circle point-labeling algorithm.

In order to show the correctness of our algorithm we first determine the maximum degree in the reduced Delaunay triangulation and then give a lower bound for the size of the largest free wedge of a point p in P.

Fact 1 *The angle between two short edges incident to a point p in P is at least* $2 \arcsin \frac{D_2}{4d} \approx 58.4°$.

Lemma 3 *Each point p in P is incident to at most six short edges.*

Proof. Consider an annular ring R with diameters D_2 and $2d$ around p (including the inner and excluding the outer circle). R contains all points in P that share a short edge with p. However, due to Fact 1, R cannot contain more than six points whose pairwise distance is at least D_2. ♂

Lemma 4 *Each point p in P has a free wedge with an angle of at least 36 degrees.*

Proof. Due to Lemma 3 we know that p has $k \le 6$ short edges. There are $k' \le k$ lines that go through these edges, and there are k' wedges defined by pairs of neighboring lines (if none of them forms an angle ≥ 90 degrees, but then we would be done). Due to the pigeon-hole principle there must be a wedge with an angle of at least 36 degrees if $k' \le 5$. (Each wedge contributes two angles!) So we only have to consider the case $k = k' = 6$ (i.e. each line contains exactly one short edge) and all of the six wedges have an angle of less than 36 degrees. Then, however, one of the wedges must be delimited by two short edges on the same side of p, which is a contradiction to Fact 1. ♂

It remains to show that, by taking advantage of the largest free wedge, the labels of each point are placed neatly within its Voronoi cell.

Lemma 5 *Let L be an open circle of diameter d that touches a point p in P. If the center of L lies on the bisector of the largest free wedge of p, then $L \subseteq \mathrm{Vor}(p)$.*

Proof. Lemma 4 guarantees that p has a free wedge that spans an angle α of at least 36 degrees. The construction in Figure 10 shows for the extremal case $\alpha = 36°$ how d is actually chosen, namely such that the center z of the label L of p lies at the intersection of the bisector b of W and the set F of all points with equal distance to p and a line ℓ_1. This line has distance $\frac{1}{2}D_2$ from p and is perpendicular to s_1, one of the supporting lines of W. Let x be the point that lies between p and ℓ_1 on s_1 at a distance of $\frac{D_2-d}{2}$ from p. In the right-angled triangle Δxzp we then have $\cos\frac{\alpha}{2} = \frac{D_2-d}{d}$, and thus $d = \frac{D_2}{1+\cos 18°}$. For a line h let h^+ be the open halfplane that is supported by h and contains p.

Due to our construction L is contained in the union of a disc D and a kite K, see Figure 10. The disc D is centered at p and has radius $\frac{1}{2}D_2$, thus it lies completely within $\mathrm{Vor}(p)$. The kite K is the intersection of W and the two halfplanes ℓ_1^+ and ℓ_2^+, where the latter is symmetric to the former with respect to b. Note that ℓ_1 and ℓ_2 intersect the two supporting lines s_1 and s_2 of W at right angles and touch the disc D there.

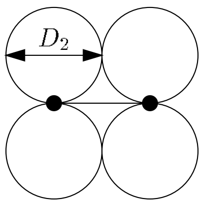


Figure 9: D_2 is an upper bound for the optimal label size.

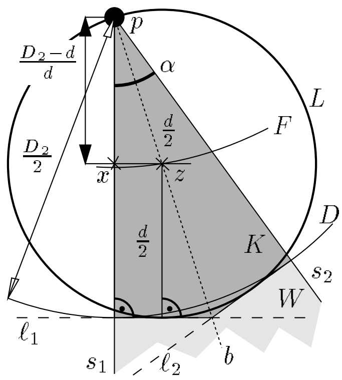


Figure 10: Label placement in a wedge with $\alpha = 36°$.

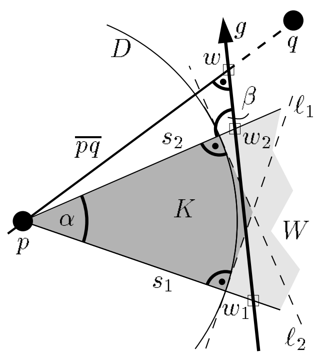


Figure 11: No Voronoi edge can intersect kite K or disc D.

Now suppose L is intersected by a Voronoi edge e of $\mathrm{Vor}(p)$. Let q be the point in P whose Voronoi cell touches that of p in e. If $d(p,q) \geq 2d$ than $d(p,e) = \frac{1}{2}d(p,q) \geq d$ and e does not intersect L. If $d(p,q) < 2d$ then q cannot lie in the wedge W, since W does not contain any short edges. So we can assume $D_2 \leq d(p,q) < 2d$ and $q \notin W$. (In this case (p,q) is a Delaunay edge; if the disc with diameter $\overline{pq}$ contained another point r of P, r would be closer than D_2 to p or q.)

Let g be the line that contains e. The halfplane g^+ contains $\mathrm{Vor}(p)$ and thus D. Let w_1 and w_2 be the points where g intersects the two supporting lines s_1 and s_2 of W, see Figure 11. Let w be the intersection of g and $\overline{pq}$. w cannot lie between w_1 and w_2, otherwise q would lie in W. So we can assume that w lies outside W and closer to w_2, say. Direct g from w_1 to w_2. By definition g intersects $\overline{pq}$, the Delaunay edge between p and q, in a right angle. Then the angle β that g and s_2 form in the triangle $\Delta p w_2 w$ must be less than 90 degrees. Since ℓ_2 is perpendicular to s_2, this means that g intersects ℓ_2 beyond w_2 — and not within W. Thus g^+ contains K, which in turn contains $L \setminus D$. This contradicts our assumption, namely that e and L intersect. ♂

Since the Voronoi cells of a point set are mutually exclusive, Lemma 5 yields the correctness of our two-circle point-labeling algorithm whose time and space complexity is obvious. Using the same upper bound for an optimal solution as in [ZP99, ZQ00], we can summarize as follows.

Theorem 3 *Our algorithm labels a set of n points with 2n circles, two per point, of diameter at least* $\frac{1}{1+\cos 18^\circ} \approx 0.5125$ *times the maximum diameter in* $O(n \log n)$ *time using linear space.*

This algorithm uses the Delaunay triangulation to compute D_2 and to label all points with labels whose size depends on D_2. A different approach would give larger labels in general, although we were not yet able to prove a better approximation factor and keep the runtime of $O(n \log n)$. Instead of computing the Delaunay triangulation and D_2 as an upper bound for the label size, we could directly compute the Voronoi diagram, label each point optimally within its Voronoi cell, and then shrink all labels to the smallest label size we have used. If the Voronoi cell is a regular pentagon both algorithms actually place labels of the same size.

Due to space limitations we must refer to the full paper for the proof of the following theorem. It does not only provide strong evidence for the necessity to search for approximate solutions of the two-circle labeling problem, but it also considerably reduces the gap between our approximability result and the polynomial-time non-approximability of two-circle labeling.

Theorem 4 *It is NP-hard to decide whether a set of points can be labeled with pairs of unit circles, and it is NP-hard to approximate the two-circle point-labeling problem beyond a factor of* $\frac{2}{1+\sqrt{3}} \approx 0.7321$.

Acknowledgments

We wish to thank Otfried Cheong, Hong Kong University of Science and Technology, without whose generous support this research would not have been possible.

References

[AF84] John Ahn and Herbert Freeman. AUTONAP - an expert system for automatic map name placement. In *Proceedings International Symposium on Spatial Data Handling*, pages 544–569, 1984.

[Aur91] F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, September 1991.

[C+96] Bernard Chazelle et al. Application challenges to computational geometry: CG impact task force report. Technical Report TR-521-96, Princeton University, April 1996. `http://www.cs.princeton.edu/~chazelle/taskforce/CGreport.ps`.

[CMS95] Jon Christensen, Joe Marks, and Stuart Shieber. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics*, 14(3):203–232, 1995.

[DF89] Jeffrey S. Doerschler and Herbert Freeman. An expert system for dense-map name placement. In *Proc. Auto-Carto 9*, pages 215–224, 1989.

[DLSS95] A. Datta, H.-P. Lenhof, C. Schwarz, and M. Smid. Static and dynamic algorithms for k-point clustering problems. *J. Algorithms*, 19:474–503, 1995.

[DMM+97] Srinivas Doddi, Madhav V. Marathe, Andy Mirzaian, Bernard M.E. Moret, and Binhai Zhu. Map labeling and its generalizations. In *Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97)*, pages 148–157, 1997.

[DMM00] Srinivas Doddi, Madhav V. Marathe, and Bernard M.E. Moret. Point labeling with specified positions. In *Proc. 16th Annu. ACM Sympos. Comput. Geom. (SoCG'00)*, Hongkong, 12–14 June 2000. to appear.

[EIS76] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5:691–703, 1976.

[For92] Michael Formann. *Algorithms for Geometric Packing and Scaling Problems.* PhD thesis, Fachbereich Mathematik und Informatik, Freie Universität Berlin, 1992.

[FW91] Michael Formann and Frank Wagner. A packing problem with applications to lettering of maps. In *Proc. 7th Annu. ACM Sympos. Comput. Geom. (SoCG'91)*, pages 281–288, 1991.

[FW93] Michael Formann and Frank Wagner. An efficient solution to Knuth's METAFONT labeling problem. Manuscript available at `http://www.math-inf.uni-greifswald.de/map-labeling/papers/fw-eskml-93.ps.gz`, 1993. Fachbereich Informatik, Freie Universität Berlin.

[Hir82] Stephen A. Hirsch. An algorithm for automatic name placement around point data. *The American Cartographer*, 9(1):5–17, 1982.

[KR92] Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM J. Discr. Math.*, 5(3):422–427, 1992.

[KSY99] Sung Kwon Kim, Chan-Su Shin, and Tae-Cheon Yang. Labeling a rectilinear map with sliding labels. Technical Report HKUST-TCSC-1999-06, Hongkong University of Science and Technology, July 1999.

[KT98] Konstantinos G. Kakoulis and Ioannis G. Tollis. On the multiple label placement problem. In *Proc. 10th Canadian Conf. Computational Geometry (CCCG'98)*, pages 66–67, 1998.

[Mor80] Joel L. Morrison. Computer technology and cartographic change. In D.R.F. Taylor, editor, *The Computer in Contemporary Cartography.* Johns Hopkins Univ. Press, New York, 1980.

[PZC98] Chung Keung Poon, Binhai Zhu, and Francis Chin. A polynomial time solution for labeling a rectilinear map. *Information Processing Letters*, 65(4):201–207, 1998.

[SW99] Tycho Strijk and Alexander Wolff. Labeling points with circles. Technical Report B 99-08, Institut für Informatik, Freie Universität Berlin, April 1999.

[vKSW99] Marc van Kreveld, Tycho Strijk, and Alexander Wolff. Point labeling with sliding labels. *Computational Geometry: Theory and Applications*, 13:21–47, 1999.

[WW97] Frank Wagner and Alexander Wolff. A practical map labeling algorithm. *Computational Geometry: Theory and Applications*, 7:387–404, 1997.

[Zor90] Steven Zoraster. The solution of large 0-1 integer programming problems encountered in automated cartography. *Operations Research*, 38(5):752–759, 1990.

[ZP99] Binhai Zhu and Chung Keung Poon. Efficient approximation algorithms for multi-label map labeling. In *Proc. Tenth Annual International Symposium on Algorithms and Computation (ISAAC'99)*, Lecture Notes in Computer Science, pages 143–152, Chennai, India, 16–18 December 1999. Springer-Verlag.

[ZQ00] Binhai Zhu and Zhongping Qin. New approximation algorithms for map labeling with sliding labels. Dept. of Computer Science, City University of Hong Kong, 2000.