Mixed Map Labeling

Maarten Löffler[1], Martin Nöllenburg[2], and Frank Staals[1](✉)

[1] Department of Information and Computing Sciences,
Utrecht University, Utrecht, The Netherlands
{m.loffler,f.staals}@uu.nl

[2] Institute of Theoretical Informatics,
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
noellenburg@kit.edu

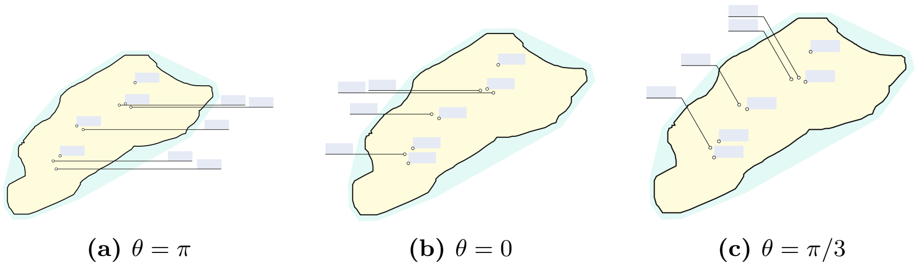
Abstract. Point feature map labeling is a geometric problem, in which a set of input points must be labeled with a set of disjoint rectangles (the bounding boxes of the label texts). Typically, labeling models either use internal labels, which must touch their feature point, or external (boundary) labels, which are placed on one of the four sides of the input points' bounding box and which are connected to their feature points by crossing-free leader lines. In this paper we study polynomial-time algorithms for maximizing the number of internal labels in a mixed labeling model that combines internal and external labels. The model requires that all leaders are parallel to a given orientation $\theta \in [0, 2\pi)$, whose value influences the geometric properties and hence the running times of our algorithms.

1 Introduction

Annotating features of interest in information graphics with textual labels or icons is an important task in information visualization. One classical application, whose principles easily generalize to the labeling of other illustrations, is map labeling, where labels are mostly placed internally in the map. Common cartographic placement guidelines demand that each label is placed in the immediate neighborhood of its feature and that the association between labels and features is unambiguous, while no two labels may overlap each other [12,21]. Point feature labeling has been studied extensively in the computational geometry literature, but also in the application areas. It is known that maximizing the number of non-overlapping labels for a given set of input points is NP-hard, even for very restricted labeling models [9,18]. In terms of labeling algorithms, several approximations, polynomial-time approximation schemes (PTAS), and exact approaches are known [1,7,15,22], as well as many practically effective heuristics, see the bibliography of Wolff and Strijk [23]. If, however, feature points lie too dense in the map or if their labels are relatively large, often only small fractions of the features obtain a label, even in an optimal solution.

An alternative labeling approach using external instead of internal labels is known as *boundary labeling* in the literature. This labeling style is frequently used

© Springer International Publishing Switzerland 2015
V.Th. Paschos and P. Widmayer (Eds.): CIAC 2015, LNCS 9079, pp. 339–351, 2015.
DOI: 10.1007/978-3-319-18173-8_25



(a) $\theta = \pi$ **(b)** $\theta = 0$ **(c)** $\theta = \pi/3$

Fig. 1. A sample point set with mixed labelings of three different slopes. In (a) five external labels are necessary, whereas (b) and (c) require only four external labels. The slope in (c) yields aesthetically pleasing results.

when annotating anatomical drawings and technical illustrations, where different and often small parts are identified using labels and descriptive texts outside the actual picture, which are connected to their features using leader lines. While the association between points and external labels is often more difficult to see, the big advantages of boundary labeling are that even dense feature sets can be labeled and that larger labels can be accommodated on the margins of the illustration. Many efficient boundary labeling algorithms are known. They can be classified by the leader shapes that are used and by the sides of the picture's bounding box that are used for placing the labels [2,5,6,10,11,14,20].

The combination of internal and external labeling models using internal labels where possible and external labels where necessary seems natural and has been proposed as an open problem by Kaufmann [13]; however, only few results are known in such *hybrid* or *mixed* settings. In a mixed labeling, the final image will be an overlay of the original map or drawing, a collection of internal text labels, and a collection of leaders leading out of the image. Depending on the application, we may wish to forbid intersections between some or all of these layers. When no additional intersection constraints are imposed, the problem reduces to classical internal map labeling. Under the very natural restriction that leaders cannot intersect internal labels, the internal labels need to be placed carefully, as not every set of disjoint internal labels creates sufficient gaps for routing leaders of the prescribed shape from all remaining feature points to the image boundary. Löffler and Nöllenburg [16] studied a restricted case of hybrid labeling, where a partition of the feature points into points with internal fixed position labels and points with external labels to be connected by one-bend orthogonal leaders is given as input. They presented efficient algorithms and hardness results, depending on three different problem parameters. Bekos et al. [3] studied a mixed labeling model with fixed-position internal labels and external labels on one or two opposite sides of the bounding box, connected by two-bend orthogonal leaders. Their goal is to maximize the number of internally labeled points, while labeling all remaining points externally. Polynomial and quasi-polynomial-time algorithms, as well as an approximation algorithm and an ILP formulation were presented.

Contribution. In this paper, we extend the known results on mixed map labeling as follows. We present a mixed labeling model, in which each point is assigned either an axis-aligned fixed-position internal label (e.g., to the top right of the point) or an external label connected with a leader of slope θ, where $\theta \in [0, 2\pi)$ is an input parameter defining the unique leader direction for all external labels, measured clockwise from the negative x-axis (see Fig. 1). In this model, we present a new dynamic-programming algorithm to maximize the number of internally labeled points for any given slope θ, including the left- and right-sided case ($\theta = 0$ or $\theta = \pi$), which was studied by Bekos et al. [3]. While for the right-sided case Bekos et al. provided a faster $O(n \log^2 n)$-time algorithm, where n is the number of input points, our algorithm improves upon their pseudo-polynomial $O(n^{\log n+3})$-time algorithm for the left-sided case. We solve this problem in $O(n^3(\log n + \delta))$ time, where $\delta = \min\{n, 1/d_{\min}\}$ is the inverse of the distance $d_{\min}$ of the closest pair of points in $\mathcal{P}$ and expresses the maximum density of $\mathcal{P}$ (Section 2). In the general case it turns out that the set of slopes can be partitioned into twelve intervals, in each of which the geometric properties of the possible leader-label intersections are similar for all slopes. Depending on the particular slope interval, the amount $\iota(n, \delta, \theta)$ of "interference" between sub-problems varies. This significantly affects the algorithm's performance and leads to running times between $O(n^3 \log n)$ and $O(n^3(\log n + \iota(n, \delta, \theta))) = O(n^7)$ (Section 3). Moreover, we can use our algorithm to optimize the number of internal labels over all slopes θ at an increase in running time by a factor of $O(n^2)$, as is shown in Section 4.1. Omitted proofs can be found in the full version [17].

Problem Statement. We are given a map (or any other illustration) $\mathcal{M}$, which we model for simplicity as a convex polygon (this is easy to relax to larger classes of well-behaved domains), and a set $\mathcal{P}$ of n points in $\mathcal{M}$ that must be labeled by rectangular labels (the bounding boxes of the label texts). In addition, we are given a leader slope $\theta \in [0, 2\pi)$. For simplicity we assume that θ is none of the slopes defined by two points in $\mathcal{P}$. We discuss in Section 4.4 how to remove this restriction. There are two choices for assigning a label to a point $p \in \mathcal{P}$: either we assign an *internal label* λ_p on $\mathcal{M}$ in a one-position model, or an *external label* outside of $\mathcal{M}$ that is connected to p with a leader γ_p. An internal label λ_p is a rectangle that is anchored at p by its lower left corner. A leader γ_p is a line segment of slope θ inside $\mathcal{M}$; it may bend to the horizontal direction outside of $\mathcal{M}$ in order to connect to its horizontally aligned label, see Fig. 1c. So in this model, the labeling is fixed once the choice for an internal or external label has been made for each point $p \in \mathcal{P}$. For a *valid* label assignment we require that (i) the internal labels do not overlap each other or the leaders, and that (ii) the leaders themselves do not intersect each other. Fig. 1 shows valid mixed labelings for three different slopes.

Given a set of points $P \subseteq \mathcal{P}$, let $\Lambda(P) = \{\lambda_p \mid p \in P\}$ denote the set of (candidate) labels corresponding to the points in P and let $\Gamma(P) = \{\gamma_p \mid p \in P\}$ denote the set of (candidate) leaders corresponding to the points in P. A *labeling* of $\mathcal{P}$ is a partition of $\mathcal{P}$ into sets $\mathcal{I}$ and $\mathcal{E}$, the points in $\mathcal{I}$ labeled internally, the points in $\mathcal{E}$ labeled externally, such that no two labels in $\Lambda(\mathcal{I})$ intersect, no two leaders in $\Gamma(\mathcal{E})$ intersect, and no label from $\Lambda(\mathcal{I})$ intersects a leader from $\Gamma(\mathcal{E})$.

For ease of presentation we first assume that all labels have the same size, which, without loss of generality, we assume to be 1×1. Hence, an internal label λ_p is a unit square with its bottom left corner on p. This may be a realistic model in some settings (e.g., unit-size icons as labels), but generally not all labels have the same size. We will sketch how to relax this restriction in Section 4.3.

Each leader γ_p can be split into an *inner* part (or *inner leader*), which is a line segment of slope θ from p to the intersection point with the boundary of $\mathcal{M}$, and an *outer* part (or *outer leader*) from the boundary of $\mathcal{M}$ to the actual label. We focus our attention on the inner leaders as they determine how $\mathcal{P}$ is separated into different subinstances. Hence we can basically think of the leaders as half-lines with slope θ. For completeness, we explain a simple method of routing the outer leaders in Section 4.2.

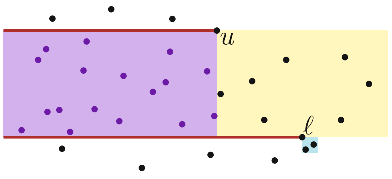
It is well known that in general not all points in $\mathcal{P}$ can be assigned an internal label. The corresponding label number maximization problem is NP-hard [9,18], even if each label has just one candidate position [16]. If, however, all labels have the same position (e.g., to the top left of the anchor points) and no input point may be covered by any other label, the one-position case can be solved efficiently by first discarding all labels containing an anchor point and then applying a simple greedy algorithm on the resulting staircase patterns [16]. On the other hand, it is also known that any instance can be labeled with external labels using efficient algorithms [4,5]. Mixed labelings combine both label types and sit between the two extremes of purely internal and purely external labeling [3,16]. Here we are interested in the *internal label number maximization problem*, which was first studied for $\theta \in \{0, \pi\}$ by Bekos et al. [3]: Given a map $\mathcal{M}$, a set of points $\mathcal{P}$ in $\mathcal{M}$ and a slope $\theta \in [0, 2\pi)$, we wish to find a valid mixed labeling that maximizes the number $|\mathcal{I}|$ of internally labeled points.

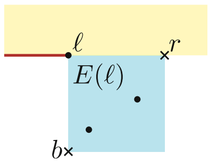
2 Leaders from the Left

We start with the case that $\theta = 0$, i.e., all leaders are horizontal half-lines leading from the points to the left of $\mathcal{M}$. Our approach for maximizing the number of internal labels is to process the points in $\mathcal{P}$ from right to left and to recursively determine the optimal rightmost unprocessed point p to be assigned an external label. Since no leader may cross any internal label, the leader γ_p decomposes the current instance left of p into two (almost) independent parts, one above γ_p and one below. As it turns out, a generic subinstance can be defined by an upper and a lower leader shielding it from the outside and additional information about at most one point outside the subinstance. The problem is then solved using dynamic programming.

2.1 Geometric Properties

Let $p = (p_x, p_y)$ be a point in the plane and let $L_p = \{q \mid q_x < p_x\}$ and $R_p = \{q \mid q_x > p_x\}$ denote the half-planes containing all points strictly to the left and to the right of p, respectively. Analogously, we define the half-planes T_p and





(a) **(b)**

Fig. 2. The slab $\overline{S(\ell,u)}$ in yellow, the region $S(\ell,u)$ and its points from $\mathcal{P}$ in purple, and the region $E(\ell)$ in blue

B_p above and below p, respectively. Let $\overline{S(\ell,u)} = T_\ell \cap B_u$ denote the horizontal slab defined by points ℓ and u (with $\ell_y < u_y$), and let $S(\ell,u) = \overline{S(\ell,u)} \cap L_\ell \cap L_u$ denote the set of points in this slab that lie to the left of both ℓ and u, see Fig. 2. We define $P_{\ell,u}$ as the subset of $\mathcal{P}$ in $S(\ell,u)$ including ℓ and u, i.e., $P_{\ell,u} = \mathcal{P} \cap (S(\ell,u) \cup \{\ell,u\})$. With some abuse of notation we will sometimes also use L_p, R_p, T_p, and B_p to mean the subset of $\mathcal{P}$ that lies in the respective half-plane rather than the entire half-plane.

Recall that $\delta = \min\{n, 1/d_{\min}\}$ is a parameter that captures the maximum density of $\mathcal{P}$ as the inverse of the smallest distance $d_{\min}$ between any two points in $\mathcal{P}$. We can use δ to bound the number of points in a unit square that may be labeled internally.

Lemma 1. *At most $O(\delta)$ points in any unit square have a label λ that does not contain another point in $\mathcal{P}$.*

Next, we characterize which leaders or labels outside of $\overline{S(\ell,u)}$ can interfere with a potential labeling of $P_{\ell,u}$ assuming ℓ and u are labeled externally.

Lemma 2. *Let $\ell, u \in \mathcal{P}$, let $(\mathcal{I}', \mathcal{E}')$, with $\ell, u \in \mathcal{E}'$ be a labeling of $P_{\ell,u}$. There is no point in $T_u \cup B_\ell$ whose leader intersects a label from $\Lambda(\mathcal{I}')$ and there is no point in T_u whose label intersects a label from $\Lambda(\mathcal{I}')$.*

It is not true, however, that labels for points in B_ℓ cannot intersect labels for $P_{\ell,u}$. Still, the influence of B_ℓ is very limited as the next lemma shows. Let $E(\ell)$ denote the open unit square with top-left corner ℓ, i.e., $E(\ell) = R_\ell \cap B_\ell \cap L_r \cap T_b$, where $r = (\ell_x + 1, \ell_y)$ and $b = (\ell_x, \ell_y - 1)$. See Fig. 2b.

Lemma 3. *Let $\ell, u \in \mathcal{P}$, let $(\mathcal{I}', \mathcal{E}')$, with $\ell, u \in \mathcal{E}'$ be a labeling of $P_{\ell,u}$, and let $(\mathcal{I}'', \mathcal{E}'')$ denote a labeling of $\mathcal{P} \cap B_\ell \cup \{\ell\}$ with $\ell \in \mathcal{E}''$. There is at most one point $p \in \mathcal{I}''$ whose label may intersect a label of $\mathcal{I}'$, and $p \in E(\ell)$.*

From Lemma 2 and Lemma 3 it follows that if ℓ and u are labeled externally, there is at most one point r below ℓ that can influence the labeling of the points in $S(\ell,u)$.

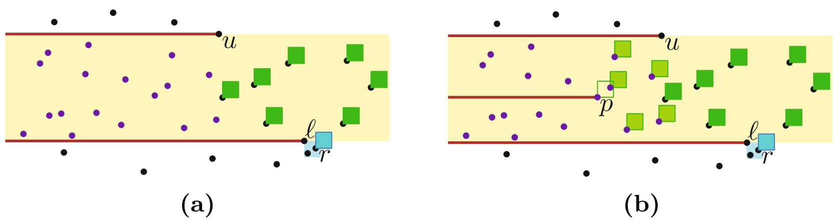


Fig. 3. (a) $\Phi(\ell, u, r)$ expresses the maximum number of points in $S(\ell, u)$ (purple), that can be labeled internally in the depicted situation. (b) The rightmost point p that is labeled with an external label decomposes the problem into two subproblems (the orange and blue points).

2.2 Computing an Optimal Labeling

We define $\Phi(\ell, u, r)$, with $\ell, u \in \mathcal{P}$, and $r \in E(\ell) \cup \{\perp\}$ as the maximum number of points in $S(\ell, u)$ that can be labeled internally, given that

(i) the points ℓ and u are labeled externally,
(ii) all remaining points in $\overline{S(\ell, u)} \setminus S(\ell, u)$ have been labeled internally, and
(iii) point r is labeled internally. If $r = \perp$ then no point in $E(\ell)$ is labeled internally.

See Fig. 3(a) for an illustration. Furthermore, given ℓ, r, and a point $p \in T_\ell$ we define $\varrho(p, \ell, r)$ to be the topmost point in $E(p) \cap (T_\ell \cup \{r\})$ if such a point exists. Otherwise we define $\varrho(p, \ell, r) = \perp$.

Lemma 4. *For any $\ell, u \in \mathcal{P}$, and $r \in E(\ell) \cup \{\perp\}$, we have that $\Phi(\ell, u, r) = |S(\ell, u)|$, or $\Phi(\ell, u, r) = |R_p \cap S(\ell, u)| + \Phi(\ell, p, r) + \Phi(p, u, r')$, where p is the rightmost point in $S(\ell, u)$ with an external label and $r' = \varrho(p, \ell, r)$.*

Proof. et $(\mathcal{I}^*, \mathcal{E}^*)$ be an optimal labeling of $S(\ell, u)$ that satisfies the constraints *(i)–(iii)* on $\Phi(\ell, u, r)$, i.e., $\Phi(\ell, u, r) = |\mathcal{I}^*|$. In case $\mathcal{E}^* = \emptyset$, we have $\Phi(\ell, u, r) = |S(\ell, u)|$ and the lemma trivially holds. Otherwise, there must be a rightmost point $p \in \mathcal{E}^*$ with an external label. Consider the partition of $\mathcal{I}^*$ at point p into the lower left part $B^* = B_p \cap L_p \cap \mathcal{I}^*$, the upper left part $T^* = T_p \cap L_p \cap \mathcal{I}^*$, and the right part $R^* = R_p \cap \mathcal{I}^*$, see Fig. 3b. We show that $|R^*| = |R_p \cap S(\ell, u)|$, $|B^*| = \Phi(\ell, p, r)$, and $|T^*| = \Phi(p, u, r')$, which proves the lemma.

Since p is the rightmost point with an external label it follows that all points in $S(\ell, u)$ right of p are labeled internally. Hence, $R^* = R_p \cap S(\ell, u)$.

Next, we observe that $\mathcal{L}_B = (B^*, S(\ell, p) \setminus B^*)$ as a sub-labeling of $(\mathcal{I}^*, \mathcal{E}^*)$ forms a valid labeling of $S(\ell, p)$, so by Lemma 3 there is at most one point $\hat{r}$ below ℓ that can influence the labeling of $S(\ell, p)$. This point $\hat{r}$, if it exists, lies in $E(\ell)$. By constraint *(iii)* point r lies in $E(\ell)$ or $r = \perp$ and no point in $E(\ell)$ is labeled internally, and thus r can be the only point in $E(\ell)$ labeled internally, i.e., $\hat{r} = r$. So, we have that *(i)* ℓ and p are labeled externally, *(ii)* all points in $\overline{S(\ell, p)} \setminus S(\ell, p)$ are labeled internally, and *(iii)* point r is the only internally labeled point in $E(\ell)$. Thus the definition of Φ applies and we obtain $|B^*| \leq \Phi(\ell, p, r)$.

Lemmas 2 and 3 together imply that any labeling of $S(\ell, p)$ is independent from any labeling of $S(p, u)$. Thus, it follows that $\mathcal{L}_B$ is an optimal labeling of $S(\ell, p)$ (given the constraints), since otherwise $(\mathcal{I}^*, \mathcal{E}^*)$ could also be improved. Thus $|B^*| \geq \Phi(\ell, p, r)$ and we obtain $|B^*| = \Phi(\ell, p, r)$.

Finally, we consider the upper left part $\mathcal{T}^*$. By Lemma 3 there is at most one point r' in B_p with an internal label that can influence the labeling of $S(p, u)$ and we have $r' \in E(p)$. We need to show that $r' = \varrho(p, \ell, r)$. Then the rest of the argument is analogous to the argument for B^*.

We claim that r' is the topmost point in $E(p) \cap (T_\ell \cup \{r\})$. Assume that $r' \notin T_\ell$, which means $r' \in B_\ell$. We know that γ_ℓ does not intersect $\lambda_{r'}$ and hence $r' \in R_\ell$. This means that $r' \in E(p) \cap B_\ell \cap R_\ell =: X$ and since p lies to the top-left of ℓ we have $X \subseteq E(\ell)$. By definition r is the only point with an internal label in $E(\ell)$ and hence $r' = r$. So if $r' \neq r$ we have $r' \in E(p) \cap T_\ell$. Now assume that $r' \neq \varrho(p, \ell, r)$. Then there is another point $q \in E(p) \cap T_\ell$ above r'. This point q must be labeled externally since no two points in $E(p)$ can be labeled internally. This is a contradiction since by definition p is the rightmost externally labeled point in $S(\ell, u)$ and by constraint *(ii)* all points in $\overline{S(\ell, u)} \setminus S(\ell, u)$ are labeled internally. So indeed $r' = \varrho(p, \ell, r)$ and the same arguments as for B^* can be used to obtain $|T^*| = \Phi(p, u, r')$. □

Let ℓ, $u \in \mathcal{P}$, and $p \in S(\ell, u)$. We observe that $|S(\ell, p)|$ and $|S(p, u)|$ are strictly smaller than $|S(\ell, u)|$. Thus, Lemma 4 gives us a proper recursive definition for Φ:

$$\Phi(\ell, u, r) = \max \Big\{ \Psi(S(\ell, u)), \\ \max_{p \in S(\ell, u)} \{\Psi(R_p \cap S(\ell, u)) + \Phi(\ell, p, r) + \Phi(p, u, \varrho(p, \ell, r))\} \Big\},$$

where

$$\Psi(P) = \begin{cases} |P| & \text{if all labels in } \Lambda(P \cup \{r\} \cup (\overline{S(\ell, u)} \setminus S(\ell, u))) \text{ are pairwise} \\ & \text{disjoint, and their intersection with } \gamma_\ell \text{ and } \gamma_u \text{ is empty,} \\ -\infty & \text{otherwise.} \end{cases}$$

We can now express the maximum number of points in $\mathcal{P}$ that can be labeled internally using Φ. We add two dummy points to $\mathcal{P}$ that we assume are labeled externally: a point p_∞ that lies sufficiently far above and to the right of all points in $\mathcal{P}$, and a point $p_{-\infty}$ below and to the right of all points in $\mathcal{P}$. The maximum number of points labeled internally is then $\Phi(p_{-\infty}, p_\infty, \perp)$.

We use dynamic programming to compute $\Phi(\ell, u, r)$ for all $\ell, u \in \mathcal{P} \cup \{p_\infty, p_{-\infty}\}$ with $\ell_y < u_y$ and $r \in E(\ell) \cup \{\perp\}$. By finding the maximum in a set of linear size, each value $\Phi(\ell, u, r)$ can be computed in $O(n)$ time, given that the values $\Phi(\ell', u', r')$ for all subproblems have already been computed and stored in a table and the relevant values for the functions ϱ and f have been precomputed. There are $O(n)$ choices for each of ℓ and u; further there are $O(\delta)$ choices for the point r given ℓ

since r is labeled internally and we know from Lemma 1 that there are at most $O(\delta)$ points in $E(\ell)$ as candidates for an internal label. This results in an $O(n^3\delta)$ time and $O(n^2\delta)$ space dynamic-programming algorithm. We show next that the preprocessing of ϱ and f can be done in $O(n^3 \log n)$ time.

To compute $\Phi(\ell, u, r)$ we actually have to compute $\varrho(p, \ell, r)$ and $\Psi'(p, r) := \Psi(R_p \cap S(\ell, u))$ for all points $p \in S(\ell, u)$. We can preprocess all points in $\mathcal{P}$ in $O(n \log n)$ time, such that we can compute each $\varrho(p, \ell, r)$ in $O(1)$ time as follows. First, we compute and store for each point $p \in \mathcal{P}$ the topmost point $q_p \in \mathcal{P}$ in $E(p)$. This requires n standard priority range queries that take $O(n \log n)$ time in total using priority range trees with fractional cascading [8, Chapter5]. To compute $\varrho(p, \ell, r)$ we then check if q_p lies above ℓ. If it does, we have $\varrho(p, \ell, r) = q_p$. Otherwise, the only candidate point for $\varrho(p, \ell, r)$ is r and we can check in $O(1)$ time if r lies in $E(p)$. This takes $O(1)$ time for each triple (p, ℓ, r) and $O(n^2\delta)$ time in total.

Next, we fix ℓ and u, and compute a representation of Ψ' in $O(n \log n)$ time, such that for each $p \in S(\ell, u)$ and $r \in E(\ell) \cup \{\perp\}$ we can obtain $\Psi'(p, r)$ in constant time.

We start by computing the values $\Psi'(p, \perp)$, for all p. We sweep a vertical line from right to left. That is, we sort all points in $\overline{S(\ell, u)}$ by decreasing x-coordinate, and process the points in that order. The status structure of the sweep line contains the number of points N in $S(\ell, u)$ right of the sweep line, and a (semi-)dynamic data structure $\mathcal{T}$, which stores the labels from the points right of the sweep line, and can report all labels intersected by an (axis-parallel) rectangular query window. All labels are unit squares, so λ_r intersects a label λ_q if and only if λ_r contains a corner point of λ_q. Furthermore, we only ever insert new labels (points) into $\mathcal{T}$, thus it suffices if $\mathcal{T}$ supports only insert and query operations. It follows that we can implement $\mathcal{T}$ using a semi-dynamic range tree using dynamic fractional cascading [19]. In this data structure insertions and queries take $O(\log n)$ time.

When we encounter a new point p, $p \notin \{\ell, u\}$ we test if the label of p intersects any of the labels encountered so far. We can test this using a range query in the tree $\mathcal{T}$. If $p \in S(\ell, u)$ we also explicitly test if λ_p intersects γ_ℓ or γ_u. If there are no points in the query range λ_p, and λ_p does not intersect γ_ℓ or γ_u we report $\Psi'(p, \perp) = N$, increment N (if applicable), and insert the corner points of λ_p into $\mathcal{T}$. If the query range λ_p is not empty, it follows that $\Psi'(p', \perp) = -\infty$, for $p' = p$ as well as for any point to the left of p. Hence, we report that and stop the sweep. Our algorithm runs in $O(n \log n)$ time: sorting all points takes $O(n \log n)$ time, and handling each of the $O(n)$ events takes $O(\log n)$ time.

Now consider a point $r \in E(\ell)$. We observe that for all points p right of r, we have that $\Psi'(p, r) = \Psi'(p, \perp) = 0$ since r is right of all points in $S(\ell, u)$. Consider the points left of r ordered by decreasing x-coordinate. There are two options, depending on whether or not λ_r intersects the label λ_p of the current point p. If λ_r intersects λ_p, we have $\Psi'(p, r) = -\infty$ as well as $\Psi'(p', r) = -\infty$ for all points p' left of p. If λ_r does not intersect λ_p we still have $\Psi'(p, r) = \Psi'(p, \perp)$. We can test if λ_r intersects any other label using a range priority query with λ_r

in (the final version of) the range tree $\mathcal{T}$. We need $O(\delta)$ such queries, which take $O(\log n)$ time each. This gives a total running time of $O(n \log n)$.

The above algorithm can also be used when the leaders have a slope $\theta \neq 0$. However, the data structure $\mathcal{T}$ that we use is fairly complicated. In this specific case where $\theta = 0$, we can also use a much easier data structure, and still get a total running time of $O(n \log n)$. Instead of using the semi-dynamic range tree as status structure, we use a simple balanced binary search tree that stores (the end-points of) a set of vertical *forbidden intervals*. When we encounter a new point p, we check if p_y lies in a forbidden interval. If this is the case then λ_p intersects another label. Otherwise we can label p internally. This set of forbidden intervals is easily maintained in $O(\log n)$ time.

We use this algorithm for every pair (ℓ, u). Hence, after a total of $O(n^3 \log n)$ preprocessing time, we can answer $\Psi'(p, r)$ queries for any p and r in constant time. This yields the following result, which improves the previously best known pseudo-polynomial $O(n^{\log n+3})$-time algorithm of Bekos et al. [3] for the left-sided case $\theta = 0$.

Theorem 1. *Given a set $\mathcal{P}$ of n points, we can compute a labeling of $\mathcal{P}$ that maximizes the number of internal labels for $\theta = 0$ in $O(n^3 \log n + n^3\delta)$ time and $O(n^2\delta)$ space, where $\delta = \min\{n, 1/d_{\min}\}$ for the minimum distance $d_{\min}$ in $\mathcal{P}$.*

3 Other Leader Directions

For other leader slopes $\theta \neq 0$ we use a similar approach as before. We consider a sub-problem $S(\ell, u)$ defined by two externally labeled points ℓ and u. We again find the "rightmost" point in the slab labeled externally. This gives us two sub-problems, which we solve recursively using dynamic programming. However, there are four complications:

- The region $E(\ell)$ containing the points "below" the slab $\overline{S(\ell, u)}$ that can influence the labeling of $S(\ell, u)$ is no longer a unit square. Depending on the orientation, it can contain more than one point with an internal label.
- In addition to the region $E(\ell)$, which contains points that can interfere with $S(\ell, u)$ from below, we now also need to consider a second region, which we call $F(u)$, containing points whose labels can interfere with a subproblem from above.
- The labels of points in $S(\ell, u)$ are no longer fully contained in the slab $\overline{S(\ell, u)}$. Hence, we have to check that they do not intersect leaders of points outside $\overline{S(\ell, u)}$.
- The regions $E(p)$ and $F(p)$ are no longer strictly to "the right" of p. Hence, for some sub-problems we may have already decided (by definition of the sub-problem) that a point $q \in E(p)$ that lies "left" of p is labeled internally. Hence, we can no longer choose q to be the rightmost point in $S(\ell, p)$ to be labeled externally.

Details on how to compute a labeling taking these complications into account can be found in the full version [17]. We obtain the following main result, where the running time is at most $O(n^7)$ for the worst case of $\delta = O(n)$:

Theorem 2. *Given a set $\mathcal{P}$ of n points and an angle θ, we can compute a labeling of $\mathcal{P}$ that maximizes the number of internal labels in $O(n^3(\log n + \iota(n,\delta,\theta)))$ time and $O(n^2\sqrt{\iota(n,\delta,\theta)})$ space, where $\delta = \min\{n, 1/d_{\min}\}$ for the minimum distance $d_{\min}$ in $\mathcal{P}$, and $\iota(n,\delta,\theta)$ models how much subproblems can influence each other. More formally,*

$$\begin{aligned}\iota(n,\delta,\theta) = \; & n^{2e'(\theta)+2f'(\theta)} + n^{2e'(\theta)+f'(\theta)}\delta^{f^*(\theta)} + n^{e'(\theta)+2f'(\theta)}\delta^{e^*(\theta)} \\ & + n^{e'(\theta)+f'(\theta)}\delta^{e^*(\theta)+f^*(\theta)} + n^{2e'(\theta)}\delta^{2f^*(\theta)} + n^{2f'(\theta)}\delta^{2e^*(\theta)} \\ & + n^{e'(\theta)}\delta^{e^*(\theta)+2f^*(\theta)} + n^{f'(\theta)}\delta^{2e^*(\theta)+f^*(\theta)} + \delta^{2e^*(\theta)+2f^*(\theta)},\end{aligned}$$

where $e^*(\theta), f^*(\theta), e'(\theta)$, and $f'(\theta)$ are all at most one.

4 Extensions

So far, we have considered a stylized version of the question we set out to solve. In this section we discuss how our solution may be adapted and extended, depending on the exact requirements of the application.

4.1 Optimizing the Direction

Rather than fixing the direction for the leaders in advance, we may be willing to let the algorithm specify the optimal orientation that maximizes the number of points that can be labeled internally. Or, perhaps we wish to compute a chart that plots the maximum number of internally labeled points as a function of the leader orientation θ, leaving the final decision to the judgement of the user.

In both scenarios, we need to efficiently iterate over all possible orientations. We adapt our method straightforwardly. Let Q be the set of all $4n$ corner points of all potential labels. For every pair $p, q \in Q$ consider the slope $\theta_{p,q}$ of the line through p and q. All values $\theta_{p,q}$ partition all possible angles into $O(n^2)$ intervals. For all values θ in the same interval J, any leader γ_p intersects the same set of potential labels, so the optimal set of internal labels is constant throughout J. We compute it separately for each interval.

By applying Theorem 2, we achieve a total of $O(n^2 \cdot n^3(\log n + \iota(n,\delta,\theta))) = O(n^5(\log n + \iota(n,\delta,\theta)))$ time to compute the optimal labelings for all orientations, or to optimize the orientation by performing a simple linear scan.

4.2 Routing the Outer Leaders

Once the core combinatorial problem of deciding which points have to be labeled internally is solved, it remains to route the outer leaders and place the external labels. Since our goal in this paper is to maximize the number of internally

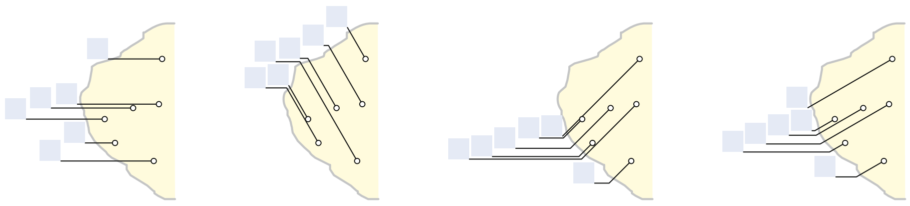


Fig. 4. Examples for routing the outer leaders and placing the external labels for different slopes

labeled points, we are only interested in finding a valid labeling, in which neither labels nor leaders intersect each other. Let us assume that $\theta \in [0, \pi/2] \cup [3\pi/2, 2\pi]$, i.e., all external labels are oriented to the left. The case of labels oriented to the right is symmetric. We consider the leaders in counterclockwise order around the boundary of $\mathcal{M}$ and place them one by one starting with the topmost leader. The first label is placed with its lower right corner anchored at the endpoint of its inner leader. For all subsequent labels we test if the label anchored at the endpoint of the inner leader intersects the previously placed label. If there is no intersection, we use that label position. Otherwise, we draw an outer leader extending horizontally to the left starting from the endpoint of the inner leader until the label can be placed without overlap. Obviously this algorithm takes only linear time. Fig. 4 shows the resulting labelings for four different slopes. We note that depending on the slope θ of the inner leaders other methods for routing the outer leaders might yield more pleasing external labelings. This is, however, beyond the scope of this paper.

4.3 Non-square Labels

Square labels are not very realistic in most map-labeling applications. Their use is justified by the observation that if all labels are homothetic rectangles, we can scale the plane in one dimension to obtain square labels without otherwise changing the problem. Nonetheless, reality is not quite that simple, for two reasons: firstly, the scaling does alter inter-point distances, so if we wish to parametrize our solution by $d_{\min}$ we need to take this into account. Secondly, in real-world applications, labels may arguably have the same height, but not usually the same width.

If all labels are homothetic rectangles with a height of 1 and a width of w, scaling the plane by a factor $1/w$ in the horizontal direction potentially decreases the closest interpoint distance by the same factor. Now, the number of points in a unit-area region that do not contain each other's potential labels is bounded by δw, immediately yielding a result of $O(n^3(\log n + \iota(n, \delta w, \theta)))$ using exactly the same approach.

When all labels have equal heights but may have arbitrary widths, we conjecture that a variation of our approach will still work, but a careful analysis of the intricacies involved is required. If the labels may also have arbitrary heights the problem is open. It is unclear if there is a polynomial time solution in this case.

4.4 Obstacles

In this paper, we have considered only abstract point sets to be labeled, using leaders that are allowed to go anywhere, as long as they do not intersect any internal labels. While this is justified in some applications (e.g., in anatomical drawings, it is common practice to ignore the drawing when placing the leaders, as they are very thin and do not occlude any part of the drawing), in others this may be undesirable (in certain map styles, leaders may be confused for region boundaries or linear features). As a solution, we may identify a set of polygonal *obstacles* in the map, that cannot be intersected by leaders or internal labels.

In this setting, obviously not every input has a valid labeling: a point that lies inside an obstacle can never be labeled, or obstacles may surround points or force points into impossible configurations in more complex ways. Nonetheless, we can test whether an input has a valid labeling and if so, compute the labeling that maximizes the number of internal labels in polynomial time with our approach.

The main idea is to preprocess the input points in a similar way as for the arbitrary leader orientations case (Section 3). Whenever a point has a potential leader that intersects an obstacle, it must be labeled internally; similarly, whenever a point has a potential internal label that intersects an obstacle, it must be labeled externally. If we include such "forced" leaders or labels into our set of obstacles and apply this approach recursively, we will either find a contradiction or be left with a set of points whose potential leaders and potential internal labels do not intersect any obstacle, and we can apply our existing algorithm on this point set.

The same approach may be used for point sets that are not in general position: if we disallow leaders that pass through other points, they are forced to be labeled internally. Note that this again may result in situations where no valid labeling exists.

Acknowledgments. M.L. and F.S. are supported by the Netherlands Organisation for Scientific Research (NWO) under grant 639.021.123 and 612.001.022, respectively.

References

1. Agarwal, P.K., van Kreveld, M., Suri, S.: Label placement by maximum independent set in rectangles. Comput. Geom. Theory Appl. **11**(3–4), 209–218 (1998)
2. Bekos, M., Kaufmann, M., Nöllenburg, M., Symvonis, A.: Boundary labeling with octilinear leaders. Algorithmica **57**, 436–461 (2010)
3. Bekos, M.A., Kaufmann, M., Papadopoulos, D., Symvonis, A.: Combining traditional map labeling with boundary labeling. In: Černá, I., Gyimóthy, T., Hromkovič, J., Jefferey, K., Královič, R., Vukolić, M., Wolf, S. (eds.) SOFSEM 2011. LNCS, vol. 6543, pp. 111–122. Springer, Heidelberg (2011)

4. Bekos, M.A., Kaufmann, M., Symvonis, A.: Efficient labeling of collinear sites. J. Graph Algorithms Appl. **12**(3), 357–380 (2008)
5. Bekos, M.A., Kaufmann, M., Symvonis, A., Wolff, A.: Boundary labeling: Models and efficient algorithms for rectangular maps. Comput. Geom. Theory Appl. **36**(3), 215–236 (2007)
6. Benkert, M., Haverkort, H., Kroll, M., Nöllenburg, M.: Algorithms for multi-criteria boundary labeling. J. Graph Algorithms and Appl. **13**(3), 289–317 (2009)
7. Chalermsook, P., Chuzhoy, J.: Maximum independent set of rectangles. In: Discrete Algorithms (SODA 2009), pp. 892–901 (2009)
8. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry: Algorithms and Applications, 2nd edn. Springer-Verlag, Berlin, Germany (2000)
9. Formann, M., Wagner, F.: A packing problem with applications to lettering of maps. In: Computational Geometry (SoCG 1991), pp. 281–288. ACM (1991)
10. Gemsa, A., Haunert, J.-H., Nöllenburg, M.: Boundary-labeling algorithms for panorama images. In: Advances in Geographic Information Systems (SIGSPATIAL GIS 2011), pp. 289–298. ACM (2011)
11. Huang, Z.-D., Poon, S.-H., Lin, C.-C.: Boundary labeling with flexible label positions. In: Pal, S.P., Sadakane, K. (eds.) WALCOM 2014. LNCS, vol. 8344, pp. 44–55. Springer, Heidelberg (2014)
12. Imhof, E.: Positioning names on maps. The American Cartographer **2**(2), 128–144 (1975)
13. Kaufmann, M.: On map labeling with leaders. In: Albers, S., Alt, H., Näher, S. (eds.) Efficient Algorithms. LNCS, vol. 5760, pp. 290–304. Springer, Heidelberg (2009)
14. Kindermann, P., Niedermann, B., Rutter, I., Schaefer, M., Schulz, A., Wolff, A.: Two-sided boundary labeling with adjacent sides. In: Dehne, F., Solis-Oba, R., Sack, J.-R. (eds.) WADS 2013. LNCS, vol. 8037, pp. 463–474. Springer, Heidelberg (2013)
15. Klau, G.W., Mutzel, P.: Optimal labeling of point features in rectangular labeling models. Mathematical Programming **94**(2), 435–458 (2003)
16. Löffler, M., Nöllenburg, M.: Shooting bricks with orthogonal laser beams: a first step towards internal/external map labeling. In: Canadian Conf. Computational Geometry (CCCG 2010), pp. 203–206. University of Manitoba (2010)
17. Löffler, M., Nöllenburg, M., Staals, F.: Mixed map labeling. CoRR, abs/1501.06813 (2015)
18. Marks, J., Shieber, S.: The computational complexity of cartographic label placement. Technical report, Harvard University (1991)
19. Mehlhorn, K., Näher, S.: Dynamic fractional cascading. Algorithmica **5**(1–4), 215–241 (1990)
20. Nöllenburg, M., Polishchuk, V., Sysikaski, M.: Dynamic one-sided boundary labeling. In: Advances in Geographic Information Systems (SIGSPATIAL GIS 2010), pp. 310–319, November 2010
21. Reimer, A., Rylov, M.: Point-feature lettering of high cartographic quality: a multi-criteria model with practical implementation. In: EuroCG 2014, Ein-Gedi, Israel (2014)
22. van Kreveld, M., Strijk, T., Wolff, A.: Point labeling with sliding labels. Comput. Geom. Theory Appl. **13**(1), 21–47 (1999)
23. Wolff, A., Strijk, T.: The map labeling bibliography. http://i11www.iti.kit.edu/map-labeling/bibliography/