Multi-stack Boundary Labeling Problems⋆

Michael A. Bekos[1], Michael Kaufmann[2],
Katerina Potika[1], and Antonios Symvonis[1]

[1] National Technical University of Athens,
School of Applied Mathematical & Physical Sciences
{mikebekos, symvonis}@math.ntua.gr, epotik@cs.ntua.gr
[2] University of Tübingen, Institute for Informatics
mk@informatik.uni-tuebingen.de

Abstract. The *boundary labeling* problem was recently introduced in [5] as a response to the problem of labeling dense point sets with large labels. In boundary labeling, we are given a rectangle R which encloses a set of n sites. Each site is associated with an axis-parallel rectangular label. The main task is to place the labels in distinct positions on the boundary of R, so that they do not overlap, and to connect each site with its corresponding label by non-intersecting polygonal lines, so called *leaders*. Such a label placement is referred to as *legal label placement*.

In this paper, we study boundary labeling problems along a new line of research. We seek to obtain labelings with labels arranged on more than one stacks placed at the same side of R. We refer to problems of this type as *multi-stack boundary labeling problems*.

We present algorithms for *maximizing the uniform label size* for boundary labeling with two and three stacks of labels. The key component of our algorithms is a technique that combines the merging of lists and the bounding of the search space of the solution. We also present NP-hardness results for multi-stack boundary labeling problems with labels of variable height.

1 Introduction

A common task in the process of information visualization is the placement of extra information, usually in the form of text labels, next to the features of a drawing (diagram, map, technical or graph drawing). When the labels are small and the features are sparsely distributed in the drawing, it may be feasible to place most labels next to the features so that the labels do not overlap with each other and they do not obscure other drawing features. Obtaining optimal label placements with respect to some optimization criterion is, in general, NP-hard [8]. An extensive bibliography about map labeling can be found at [12].

In the case of very large labels (or, equivalently, dense feature sets), it is usually impossible to find a label placement, i.e. to place each label next to the feature.

In response to this problem, Bekos, Kaufmann, Symvonis and Wolff [5] (an extended journal version appears in [4]) proposed the *boundary labeling model*. In this model the labels are placed on the boundary of a rectangle enclosing all features and each label is connected to its associated feature with polygonal lines, called *leaders*. If the labels are non overlapping and the leaders non intersecting we have a *legal labeling* or a *legal label placement*. The boundary labeling model is a realistic model for medical atlases and technical drawings, where certain features of a drawing are explained by blocks of text placed outside the drawing so that no part of the drawing is obscured. SmartDraw [10] provides boundary labelings in a primitive form based on labeling templates. It does not support any form of automated boundary labeling optimization. Bler [6] supports the boundary labeling process and facilitates the annotation of drawings with text labels.

Sites model features of the drawing. If they model a *point-feature* (e.g., a city on a map) they are naturally represented as points (see points in rectangle R of Fig. 1, 2 and 5). So, in its simplest form, a boundary labeling problem specifies as part of its input a set P of n points $p_i = (x_i, y_i)$ on the plane in general position, i.e. no three points lie on a line and no two points have the same x- or y-coordinate. Another interesting variation is the one with two candidate points on the plane for each site (see Fig. 3). In practice, several times we want to associate a label with an *area-feature* (e.g., a region on a map). To keep things simple, we specify these regions by a closed polygonal line or by a line segment internal to the feature area, and assume that the site "slides" along the boundary of the polygon or on the line segment (see Fig. 4).
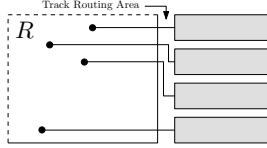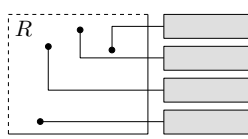

Fig. 1. Type-*opo* leaders
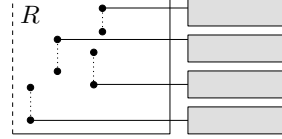
Fig. 2. Type-*po* leaders
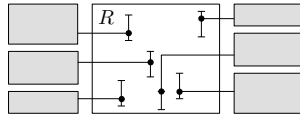
Fig. 3. Sites with 2 candidate points

Fig. 4. Sites are vertical line segments
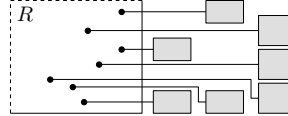
Fig. 5. Three stacks of labels

In general, each site p_i has a corresponding axis-parallel rectangular, open label l_i of width w_i and height h_i. The labels are to be placed around an axis-parallel rectangle $R = [l_R, r_R] \times [b_R, t_R]$ of height $H = t_R - b_R$ and width $W = r_R - l_R$ which contains all sites $p_i \in P$. While in the general case the labels are of *variable dimensions*, we also consider the restricted cases where the labels are of *uniform size* (height and/or width), or of *maximum uniform size*.

Each site is connected with its corresponding label in a simple and elegant way by using polygonal lines, called *leaders*. In our approach we have leaders that consist of a single straight line segment or a sequence of rectilinear segments. When a leader is rectilinear, it consists of a sequence of axis-parallel segments either parallel (p) or orthogonal (o) to the side of R containing the label it leads to. The *type* of a leader is defined by an alternating string over the alphabet $\{p, o\}$. We focus on leaders of types-*opo* and *po*, see Fig. 1 and 2, respectively. Furthermore, we assume that each type-*opo* leader has its parallel p-segment (or equivalently its both bends) outside R, routed in the so-called *track routing area*. We consider type-*o* leaders to be of type-*opo* and of type-*po* as well.

A further refinement of the labeling model has to do with the sides of the enclosing rectangle containing the labels. Labels can be placed on one or more sides of R (in Fig. 1, 2, 3 and 5 all labels are placed on the east side of R). In order to allow for *greater numbers* of *larger labels*, we might have the labels arranged in more than one stack at each side of the enclosing rectangle. This paper is devoted to the case of *multi-stack labelings*. Figure 5 shows a labeling where the labels occupy three stacks to the east side of R. Notice that in the case of multiple stacks of labels (say m stacks), a leader of type-*opo* can have its p segment either in between R and the first stack (called *first track routing area*) or between the i-th and the $(i+1)$-th stack, where $i < m$ (called $(i+1)$-*th track routing area*).

Each leader that connects a site to a label, touches the label on a point on its side that faces R, this point is called *port*. We can assume either *fixed ports*, i.e. the leader is only allowed to use a fixed set of ports on the label side (a typical case is where the leader uses the middle point of the label side) or *sliding ports* where the leader can touch any point of the label's side. The labelings in Fig. 1, 2 and 5 use fixed ports, while in Fig. 3 and 4 they use sliding ports.

Keeping in mind that we want to obtain simple and easy to visualize labelings, the following criteria can be adopted from the areas of VLSI and graph drawing: *minimizing the total number of bends* of the leaders, *minimizing the total leader length* and *minimizing the maximum leader length*. An additional criterion that we consider is the *maximization of the uniform label size*. This is a quite common optimization criterion in the map labeling literature. In this paper, we seek to obtain labelings with labels of maximum uniform size arranged on more than one stacks of labels at the same side of R.

This paper is structured as follows: Section 2, reviews preliminary results required for the development of our algorithms. In Section 3, we present algorithms for obtaining multi-stack labelings of maximum uniform label size for the cases of two and three stacks of labels arranged at the same side of R. In Section 4, we present several NP-hardness results for non-uniform labels placed in two stacks. We conclude in Section 5 with open problems and future work.

Previous Work
Most of the known results on boundary labeling with point sites were presented in [4]. A legal labeling, on one side with type-*opo* (type-*po*) leaders can be achieved in $O(n \log n)$ time (in $O(n^2)$ time, respectively), whereas on all four

sides with type-*opo* leaders in $O(n \log n)$ time. The problem of minimizing the total number of leader bends on one side with type-*opo* leaders can be solved in $O(n^2)$ time. The minimization of the total leader length when uniform sized labels can be placed on two opposite sides of R with either type-*opo* and type-*po* leaders needs $O(n^2)$ time. For the similar problem, where non-uniform labels can be placed on two opposite sides of R and the leaders are of type-*opo*, $O(nH^2)$ time is needed. An algorithm for minimizing the total leader length on four sides with type-*opo* leaders in $O(n^2 \log^3 n)$ ($O(n^3)$) time for fixed ports (sliding ports) is presented for points in [1] and for polygons in [3].

2 Preliminaries

Throughout the paper we use lists that contain pairs of integers describing different label placements. Given a pair (a, b) of integers, a and b are referred to as the *first* and the *second coordinate* of the pair, respectively. Inspired by an idea of Stockmeyer [11] which was subsequently used by Eades et. al. [7], we manage to keep the length of each list bounded by pruning pairs that cannot occur in an optimal solution.

Definition 1. *A list L of pairs of integers is **sorted** if the pairs it contains are lexicographically sorted in decreasing order with respect to their first coordinate and in increasing order with respect to their second coordinate.*

Definition 2. *Let (a, b) and (c, d) be pairs of integers.*

$$(a, b) \text{ \textbf{dominates} } (c, d) \iff a \geq c \text{ and } b \geq d.$$

Suppose that we have to solve a problem where the search space of the solution consists of pairs of integers, and let f be a monotone function computing a minimization objective on pairs from the solution search space. If (a, b) and (c, d) represent possible solutions and (a, b) dominates (c, d), then the pair (a, b) can never be involved in an optimal solution and may be safely removed from the solution set. Given a list L of pairs of integers, a pair $(a, b) \in L$ that does not dominate any other pair in L is called an *atom* (with respect to L).

In our algorithms we maintain lists (of pairs) that contain only atoms. A frequently performed operation is the merging of two lists of atoms, resulting in a new list of atoms. The merging algorithm resembles the merging step of merge sort algorithm. It supports the following lemmas:

Lemma 1. *k sorted lists $L_1, L_2, \ldots, L_k$, $k \geq 2$, of atoms can be merged in $O((k-1) \sum_{i=1}^{k} |L_i|)$ time into a new sorted list L of at most $\sum_{i=1}^{k} |L_i|$ atoms.*

Lemma 2. *Let A and B be two finite sets of integers and let $L = \{(a, b) \mid a \in A \text{ and } b \in B\}$ be a list of atoms. Then, $|L| \leq \min(|A|, |B|)$.*

Finally, we present some notation and terminology that we use in the description of our algorithms. We say that a pair (a, b) obeys the *boundary conditions*, if

$a \leq H$ and $b \leq H$, where H is the height of the enclosing rectangle. We also define operator $\oplus_H : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, where:

$$a \oplus_H b = \begin{cases} a + b, \text{ if } a + b \leq H \\ \infty, \quad \text{otherwise} \end{cases} .$$

3   Label Size Maximization

3.1   Two Stacks of Labels on the Same Side

We consider boundary labeling with type-*opo* leaders, where the labels are placed at two stacks on the same side (say, the east side) of the rectangle R. We assume that all labels have the same size (width and height) and we seek to maximize the uniform height h of all labels, so that a legal labeling exists. To determine the maximum value of h, we apply a binary search on all possible discrete values for height h. We assume the more general case of sliding ports. Additionally, the type-*opo* leaders connecting sites to labels that are at the second stack are allowed to bend either in the first or in the second track routing area.
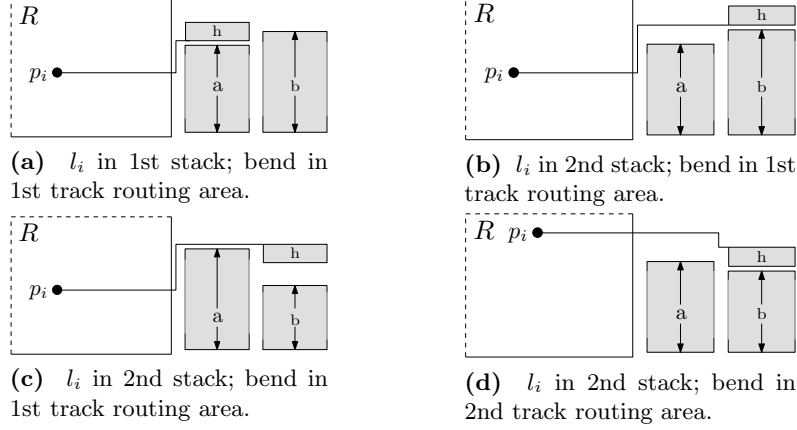
Observe that, in any legal one-side labeling with type-*opo* leaders, the vertical order of the sites is identical to the vertical order of their corresponding labels on both stacks. This, together with the assumption that no two sites share the same y-coordinate, guarantees that leaders do not intersect. So, we assume that the sites are sorted according to increasing y-coordinate.

For a fixed h, we propose a dynamic programming algorithm that outputs a boolean value, which indicates whether there exists a legal label placement, when all sites have labels of height h. Imagine that a label placement L is given, then we say that a pair (a, b) *describes* L, if a (b) is the highest occupied y-coordinate of the first (respectively second) stack. Our algorithm maintains a table T of size $(n+1) \times (n+1)$, where each entry $T[i, k]$, $i \geq k$, of table T contains a list of atoms (a, b) decribing the label placement of the first i sites when k out of them have leaders bending in the second track routing area. List $T[i, k]$ is empty, when it is impossible to place the first i labels, with k leaders bending in the second track routing area.

Assuming that we have placed the labels for the first $i-1$ sites, we try to place the label l_i of the i-th site. Label l_i can be placed at the first or second stack. Additionally, if l_i is to be placed at the second stack, then we have to check whether this can be done with a leader bending in the first or second track routing area. Obviously, such placements can be obtained from label placements of the first $i-1$ sites with either k or $k-1$ leaders bending in the second track routing area.

Label l_i is placed at the first stack: Let $T_1[i, k]$ be a list of pairs (a, b) describing the label placement of the first i sites when the i-th site has its label at the first stack and k leaders have their bends in the second track routing area. $T_1[i, k]$ can be computed based on entry $T[i-1, k]$ (see Fig. 6a), as follows:

$$T_1[i, k] = \{(a \oplus_H h, b) : \; \forall (a, b) \in T[i-1, k]\}$$

(a) l_i in 1st stack; bend in 1st track routing area.

(b) l_i in 2nd stack; bend in 1st track routing area.

(c) l_i in 2nd stack; bend in 1st track routing area.

(d) l_i in 2nd stack; bend in 2nd track routing area.

Fig. 6. Different placements obtained for the label of site i. In Figures 6a, 6b and 6c: $(a,b) \in T[i-1,k]$, whereas in Fig. 6d: $(a,b) \in T[i-1,k-1]$.

Label l_i is placed at the second stack - bend at the first track routing area: Let $T_{21}[i,k]$ be a list of pairs (a,b) describing the label placement of the first i sites when the i-th site has its label at the second stack using a leader bending at the first track routing area and k leaders have their bends in the second track routing area. Again, $T_{21}[i,k]$ can be computed based on entry $T[i-1,k]$. If for some pair $(a,b) \in T[i-1,k]$ it holds that $a \leq b$ (i.e. the first stack is lower or equal than the second stack), then a pair $(b, b \oplus_H h)$ is added in $T_{21}[i,k]$ (see Fig. 6b). Else pair $(a, \max\{b \oplus_H h, a\})$ is added in $T_{21}[i,k]$ (see Fig. 6c). Therefore, $T_{21}[i,k]$ is computed as follows:

$$T_{21}[i,k] = A_{21}[i,k] \cup B_{21}[i,k],$$

where:

$$A_{21}[i,k] = \{(b, b \oplus_H h) : \ \forall (a,b) \in T[i-1,k] \text{ s.t. } a \leq b\}$$
$$B_{21}[i,k] = \{(a, \max\{b \oplus_H h, a\}) : \ \forall (a,b) \in T[i-1,k] \text{ s.t. } a > b\}$$

Label l_i is placed at the second stack - bend at the second track routing area: Let $T_{22}[i,k]$ be a list of pairs (a,b) describing the label placement of the first i, when the i-th site has its label placed at the second stack using a leader bending at the second track routing area and k leaders have their bends in the second track routing area. $T_{22}[i,k]$ is computed based on entry $T[i-1,k-1]$ (see Fig. 6d), as follows:

$$T_{22}[i,k] = \{(y_i, b \oplus_H h) : \ \forall (a,b) \in T[i-1,k-1] \text{ s.t. } a < y_i\}$$

All pairs (∞, a), (a, ∞) can be removed from lists $T_1[i,k]$, $T_{21}[i,k]$ and $T_{22}[i,k]$, in linear time, since they do not capture possible placements. The implied lists are merged into list $T[i,k]$ of atoms, based on Lemma 1. We can easily show that $|T[i,k]| \leq 2|T[i-1,k]|+3$. This implies that $|T[n,k]| = O(2^n)$, $n \geq k$. Also, by Lemma 2, we have that $|T[n,k]| \leq H$. However, by employing the following

Lemma 3, we can improve on both of these bounds. Its correctness can easily be shown inductively, by proving that the distinct values that both coordinates of the pairs in $T[i,k]$ can receive are drawn from the sets $\{0, h, 2h, \ldots, ih\}$, $\{y_1, y_2, \ldots, y_i\}$, and $\bigcup_{j=1}^{i} \{y_j + h, y_j + 2h, \ldots, y_j + (i-1)h\}$.

Lemma 3. *List $T[n,k]$, $n \geq k$ contains $O(n^2)$ pairs.*

To prove the correctness of our algorithm, consider a pair $(a,b) \in T[i,k]$ that dominates pair $(c,d) \in T[i,k]$. Assume, for the sake of contradiction, that pair (a,b) yields a solution and pair (c,d) does not. That means that, for at least one pair out of $\{(y_i, b+h), (b, b+h), (a, \max\{b+h, a\}), (a+h, b)\}$ the boundary condition holds while the boundary condition does not hold for any of the pairs $\{(y_i, d+h), (d, d+h), (c, \max\{d+h, c\}), (c+h, d)\}$. This is impossible since $a \geq c$ and $b \geq d$. Therefore (a,b) can never be involved in an optimal solution and can be discarded. This implies that each list $T[i,k]$ should only contain atoms.

Each of the $(n+1) \times (n+1)$ entries of T is computed in $O(n^2)$ time. Thus, our algorithm terminates after $O(n^4)$ time. For a fixed label height h, the algorithm outputs a boolean value, which indicates whether there exists a legal label placement. This is done by identifying whether there exists a non-empty list $T[n,j]$, with $0 \leq j \leq n$. By using an extra table of the same size as T, our algorithm can easily be modified, such that it also computes the label and leader positions.

Theorem 1. *Given a rectangle R of integer height H and a set $P \subset R$ of n points in general positions, there exists an $O(n^4 \log H)$ time algorithm that produces a legal multi-stack labeling with two stacks of labels on the same side of R and with type-opo leaders such that the uniform integer height of the labels is maximum.*

Proof. In order to solve the *label size maximization problem*, we can simply apply a binary search on all possible discrete values for height h. To complete the proof, observe that $\frac{H}{n} \leq h \leq \frac{2H}{n}$. $\qquad\square$
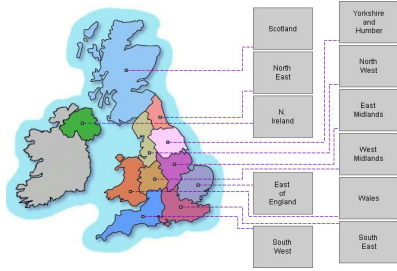
Fig. 7. A regional map of UK
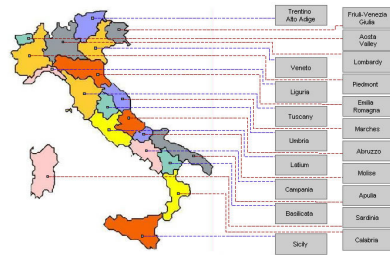
Fig. 8. A regional map of Italy

Sample Labelings
Figures 7 and 8 are produced from the algorithm of Section 3.1 and depict two
regional maps of UK and Italy, respectively. The labels occupy two stacks on the
east side of the enclosing rectangle. In both labelings the label size is maximum.

3.2   Three Stacks of Labels on the Same Side

In this section, we extend the algorithm of Section 3.1 to support an additional
stack of labels. We consider the case, where leaders connected to labels at the
i-th stack are restricted to bend in the i-th track routing area. The objective,
again, is to maximize the uniform height h of all labels.

Theorem 2. *Given a rectangle R of integer height H and a set $P \subset R$ of
n points in general positions, there exists an $O(n^4 \log H)$ time algorithm that
produces a legal multi-stack labeling with three stacks of labels on the same side
of R and with type-opo leaders such that the uniform integer height of the labels
is maximum and the leaders connected to labels at the i-th stack are restricted
to bend in the i-th track routing area.*

Proof. We use dynamic programming algorithm employing a table T of size
$(n + 1) \times (n + 1) \times (n + 1)$. For each $i \geq k + m$, entry $T[i, k, m]$ contains a
list of pairs (a, b), where a (b) is the y-coordinate of the first (second) stack,
that is needed to place the first i labels, when m labels are placed in the third
stack, k labels are placed in the second stack and $i - k - m$ labels are placed in
the first stack. Note that the height at the third stack is mh, since all leaders
connected to labels of the third stack are restricted to bend in the third track
routing area. List $T[i, k, m]$ is empty, when it is impossible to route the first i
labels using k labels in the second stack and m in the third stack. This implies
that table entries $T[i, k, m]$, where $i < k + m$, contain empty lists. Following
similar arguments as in Section 3.1, entry $T[i, k, m]$ can be computed based on
the following recurrence relation:

$$T[i, k, m] = \text{MERGE}\{T_1[i, k, m], T_2[i, k, m], T_3[i, k, m]\} \tag{1}$$

where:

$T_1[i, k, m] = \{(a \oplus_H h, b) : \ \forall (a, b) \in T[i - 1, k, m]\}$

$T_2[i, k, m] = \{(y_i, b \oplus_H h) : \ \forall (a, b) \in T[i - 1, k - 1, m] \text{ s.t. } a < y_i\}$

$T_3[i, k, m] = \{(y_i, y_i) : \ \forall (a, b) \in T[i - 1, k, m - 1], \ \text{s.t. } mh \leq H \text{ and } (a, b) < (y_i, y_i)\}$

List $T_1[i, k, m]$ of Eq. 1 captures placements of the i-th label at the first stack.
Similarly, list $T_2[i, k, m]$ of Eq. 1 captures placements of the i-th label at the
second stack. Since we assumed that leaders connected to labels at the second
stack are restricted to bend in the second track routing area, this is possible
only for pairs $(a, b) \in T[i - 1, k - 1, m]$ with $a \leq y_i$. Finally, list $T_3[i, k, m]$ of
Eq. 1 captures placements of the i-th label at the third stack. This is possible
only for pairs $(a, b) \in T[i - 1, k, m - 1]$ with $(a, b) \leq (y_i, y_i)$. To compute entry
$T[i, k, m]$, we first remove all pairs (∞, a), (a, ∞) from lists $T_1[i, k, m]$, $T_2[i, k, m]$

and $T_3[i, k, m]$ and then we merge the implied lists to $T[i, k, m]$ of atoms, based on Lemma 1.

Lemma 4. *For $n \geq k + m$, $|T[n, k, m]| \leq n + 1$.*

Proof. Lists $T_2[i, k, m]$ and $T_3[i, k, m]$ contain pairs of numbers with the same first coordinate. This implies that they contribute at most one atom, while list $T_1[i, k, m]$ contains at most i elements, since $|T[i-1, k, m]| \leq i$. Thus, $T[i, k, m] \leq i + 1$. $\qquad \square$

Each of the $(n + 1) \times (n + 1) \times (n + 1)$ entries of T is computed in $O(n)$ time. Thus, our algorithm terminates after $O(n^4)$ time. For a fixed label height h, the algorithm outputs a boolean value, which indicates whether there exists a legal label placement. This is done by identifying whether there exists a non-empty list $T[n, i, j]$, with $0 \leq i + j \leq n$. By using an extra table of the same size as T, our algorithm can easily be modified, such that it also computes the label and leader positions. In order to solve the *label size maximization problem*, we can simply apply a binary search on all possible discrete values for height h. To complete the proof, observe that $\frac{H}{n} \leq h \leq \frac{3H}{n}$. $\qquad \square$

4   Computational Complexity of the Multi-stack Labeling Problem

In this section, we investigate the computational complexity of several multi-stack boundary labeling problems with either type-*opo* or *po* leaders and labels of arbitrary size, which can be placed at two stacks on the same side of the enclosing rectangle. Without loss of generality, we assume that the labels are located on the east side of the enclosing rectangle. We consider several different type of sites. In the most applicable case, site s_i is associated with a point $p_i = (x_i, y_i)$ on the plane. However, we also consider the cases, where site s_i is associated with either two candidate points $p_i^1 = (x_i^1, y_i^1)$ and $p_i^2 = (x_i^2, y_i^2)$ on the plane (see Fig. 3) or with a vertical line segment, so that the site "slides" along the boundary of the proposed line segment (see Fig. 4). The assumed models are quite general, since we allow sliding labels with sliding ports.

4.1   Line Sites with Type-*opo* Leaders at Two Stacks on One Side

We focus on type-*opo* leaders, where each site s_i can slide along a line segment parallel to the y-axis and is associated with a label l_i of height h_i. We seek to find a legal labeling.

Theorem 3. *Given a rectangle R of height H, a set $P \subset R$ of n line segments (sites) that are parallel to the y-axis and a label of height h_i for each site $s_i \in P$, it is NP-hard to place all labels at two stacks on one side of R with non-intersecting type-opo leaders.*

Proof. We reduce the PARTITION problem [9] to our problem. The PARTITION problem is defined as follows: Given positive integers $a_1, a_2, \ldots, a_m$, is there a subset I of $J = \{1, 2, \ldots, m\}$ such that $\sum_{i \in I} a_i = \sum_{i \in J - I} a_i$?

Our site set $P = \{s_1, s_2, \ldots, s_m\}$ consists of m (parallel to y-axis) line segments of identical length $H = \frac{1}{2} \sum_{i \in J} a_i$ (H: height of R). Each site s_i is also associated with a label l_i of height a_i. Both stacks contribute $2H$ height, which is equal to the sum of all label heights.

Suppose that there exists a subset I of $J = \{1, 2, \ldots, m\}$ such that $\sum_{i \in I} a_i = \sum_{i \in J-I} a_i$. Without loss of generality, we further suppose that $|I| \geq |J - I|$. For each site s_i with $i \in I$, we choose to place its label at the first stack. The remaining labels are placed at the second stack. The leaders of the sites with labels at the first stack are of type-o. In this case, the ports of both sites and labels can be chosen arbitrarily. Since the labeling is *tight* (i.e. the sum of the label heights on each stack is equal to H), we use the fact that the labels are open and use the gaps between them as *corridors* to route the leaders, that connect sites with labels at the second stack. Since we assumed that $|I| \geq |J - I|$, there exist enough corridors to route all leaders: The leader which corresponds to the lowest label that has not been routed yet, can use the lowest available corridor. In this case the site ports are defined based on the corridors, whereas the label ports can be chosen arbitrarily again. $\square$

4.2 Two Candidate Points with Type-*opo* Leaders at Two Stacks on One Side

We will show that the problem remains NP-hard even if we restrict ourselves in sites, which may have two candidate points, i.e. leader of site s_i connects either point $p_i^1 = (x_i^1, y_i^1)$ or point $p_i^2 = (x_i^2, y_i^2)$ with label l_i. To show NP-hardness, we reduce the following variant of Partition to our problem. The NP-hardness of this problem follows easily from the EVEN ODD PARTITION problem (see [9] pp. 223).

Lemma 5 (RPartition). *Given $2m$ non-negative integers $a_1, a_2, \ldots a_{2m}$, the problem of finding a subset I of $J = \{1, 2, \ldots 2m\}$ such that the following three conditions are satisfied is NP-hard. 1) I contains exactly one of $\{2i-1, 2i\}$ for $i = 1, 2, \ldots m$. 2) $\sum_{i \in I} a_i = \sum_{i \in J-I} a_i$ and 3) $\sum_{i \in I \& i \leq k} a_i < \sum_{i \in J-I \& i \leq k} a_i$ for $k = 2, 4, \ldots 2m - 2$.*

Theorem 4. *Given a rectangle R of height H, a set $P \subset R$ of n sites, each associated with two candidate points, and a label of height h_i for each site $s_i \in P$, it is NP-hard to place all labels at two stacks on one side of R with non-intersecting type-opo leaders.*

Proof. Let $A = \{a_1, a_2, \ldots a_{2m}\}$ be an instance of RPARTITION. We will construct an instance B of our problem as follows: Let C be a very large number, e.g. $C = (2m + 1)^2 \sum_{i \in J} a_i$. Set $P = \{s_1, s_2, \ldots, s_{2m}\}$ consists of $2m$ sites. Site s_i is associated with $p_i^1 = (x_i, y_i^1)$ and $p_i^2 = (x_i, y_i^2)$. Consecutive sites s_{2i-1} and s_{2i}, $i = 1, 2, \ldots, m$, form m parallelograms r_i, $i = 1, 2, \ldots, m$, such that $y_{2i-1}^1 < y_{2i}^1 < y_{2i-1}^2 < y_{2i}^2$ and $|y_{2i-1}^2 - y_{2i}^1| = \frac{a_{2i-1}+a_{2i}}{2} + 1$. We assume that parallelogram r_{i-1} is placed lower than r_i. The vertical distance between two consecutive parallelograms is C, whereas the vertical distance between the

bottommost (topmost) parallelogram r_1 (r_m) and the bottommost (topmost respectively) side of the enclosing rectangle R is $C/2$. The height of the enclosing rectangle is $H = m(C + 1) + \frac{1}{2} \sum_{i \in J} a_i$. The label l_i of site s_i has height $h_i = C + a_i + 1$, thus $\sum_{i \in J} h_i = 2m(C+1) + \sum_{i \in J} a_i$. Observe, that both stacks contribute $2H$ height, which is equal to the sum of all label heights.

One can see that the construction ensures that the same number of labels are placed at the two stacks and that all leaders should bend in the first track routing area. Two consecutive sites s_{2i-1} and s_{2i}, $i = 1, 2, \dots m$ can not have their labels both at the same stack, because at least one corridor is lost and therefore at least one label at the second stack can not be routed. To avoid leader crossings, the order of indices should be preserved at both stacks, i.e. if $i < j$ then label l_i will be stacked lower than l_j. To connect all sites with their labels, it must either hold $\sum_{i \in I \& i \leq k} h_i < \sum_{i \in J-I \& i \leq k} h_i$ or $\sum_{i \in I \& i \leq k} h_i > \sum_{i \in J-I \& i \leq k} h_i$, for all $k = 2, 4, \dots 2m - 2$, which is equivalent to condition (3) of RPARTITION. The indices of the sites with labels at the first stack imply the partition I of J.

Suppose that we have a subset I of J of A such that all three conditions of RPARTITION are satisfied. If $i \in I$, then the label of site s_i is placed at the first stack preserving the order of indices. The remaining labels ($i \in J - I$) are placed at the second stack in the same manner. A legal labeling is obtained by taking the lowest site which has not been routed. If its label is to be placed at the second stack, use the lowest available corridor for its leader, else route it at the first stack with a type-o leader. For two consecutive sites s_{2i-1} and s_{2i} we can determine in constant time which points will be used, such that their leaders do not intersect. $\qquad\square$

4.3 Type-*po* Leaders at Two Stacks on One Side

Following similar arguments as in proof of Theorem 4, one can show that the problem remains NP-hard if we use type-po leaders, even if we restrict ourselves to a point $p_i = (x_i, y_i)$ or two candidate points $p_i^1 = (x_i^1, y_i^1)$ and $p_i^2 = (x_i^2, y_i^2)$ for each site s_i. Recall that for the case of two candidate points the leader of each site s_i connects either point p_i^1 or point p_i^2 with label l_i. The corresponding theorems follow. Detailed proofs of these theorems are given in the full version of the paper (see [2]).

Theorem 5. *Given a rectangle R of height H, a set $P \subset R$ of n points and a label of height h_i for each site $s_i \in P$, it is NP-hard to place all labels at two stacks on one side of R with non-intersecting type-po leaders.*

Theorem 6. *Given a rectangle R of height H, a set $P \subset R$ of n sites, each associated with two candidate points, and a label of height h_i for each site, it is NP-hard to place all labels at two stacks on one side of R with non-intersecting type-po leaders.*

Since, each point site can be thought as a line site of zero length, Corollary 1 follows immediately from Theorem 5.

Corollary 1. *Given a rectangle R of height H, a set $P \subset R$ of n lines and a label of height h_i for each site $s_i \in P$, it is NP-hard to place all labels at two stacks on one side of R with non-intersecting type-po leaders.*

5   Open Problems and Future Work

We presented results for the label size maximization problem and for the legal label placement for the case of two and three stacks of labels on the same side of R. No results are known regarding the total leader length minimization and the minimization of the total number of bends. Another line of research is to design good approximation algorithms that solve the problems, that are proved to be NP-hard.

References

1. M. A. Bekos, M. Kaufmann, K. Potika, and A. Symvonis. Boundary labelling of optimal total leader length. In *Proc. 10th Panhellenic Conference On Informatics (PCI2005), LNCS 3746*, pages 80–89, 2005.
2. M. A. Bekos, M. Kaufmann, K. Potika, and A. Symvonis. Multi-stack boundary labeling problems, 2006. Technical report, Available online http://www.math.ntua.gr/aarg/.
3. M. A. Bekos, M. Kaufmann, K. Potika, and A. Symvonis. Polygon labelling of minimim leader length. In *Asia Pacific Symposium on Information Visualisation (APVIS2006), CRPIT 60*, pages 15–21, 2006.
4. M. A. Bekos, M. Kaufmann, A. Symvonis, and A. Wolff. Boundary labeling: Models and efficient algorithms for rectangular maps. *Computational Geometry: Theory and Applications.* Available online http://www.sciencedirect.com/.
5. M. A. Bekos, M. Kaufmann, A. Symvonis, and A. Wolff. Boundary labeling: Models and efficient algorithms for rectangular maps. In *Proc. 12th Int. Symposium on Graph Drawing (GD'04), LNCS 3383*, pages 49–59, 2004.
6. M. A. Bekos and A. Symvonis. Bler: A boundary labeller for technical drawings. In *Proc. 13th Int. Symposium on Graph Drawing (GD'05), LNCS 3843*, pages 503–504, 2005.
7. P. Eades, T. Lin, and X. Lin. Minimum size h-v drawings. In *Advanced Visual Interfaces (Proceedings of AVI '92), volume 36 of World Scientific Series in Computer Science*, pages 386–394, 1992.
8. M. Formann and F. Wagner. A packing problem with applications to lettering of maps. In *Proc. 7th Annual ACM Symposium on Computational Geometry (SoCG'91)*, pages 281–288, 1991.
9. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979, 1979.
10. SmartDraw-7. Product web site: http://www.smartdraw.com.
11. L. Stockmeyer. Optimal orientations of cells in slicing floorplan designs. *Information and Control*, 57(2–3):91–101, 1983.
12. A. Wolff and T. Strijk. The Map-Labeling Bibliography. http://i11www.ira.uka.de/map-labeling/bibliography, 1996.