Boundary Labeling with Octilinear Leaders

Article *in* Algorithmica · July 2008

DOI: 10.1007/s00453-009-9283-6 · Source: DBLP

4 authors, including:

Michael A. Bekos
University of Tuebingen
112 PUBLICATIONS **634** CITATIONS

SEE PROFILE

Martin Nöllenburg
TU Wien
139 PUBLICATIONS **1,014** CITATIONS

SEE PROFILE

Antonios Symvonis
National Technical University of Athens
134 PUBLICATIONS **1,086** CITATIONS

SEE PROFILE

Algorithmica (2010) 57: 436–461
DOI 10.1007/s00453-009-9283-6

Boundary Labeling with Octilinear Leaders

Michael A. Bekos · Michael Kaufmann · Martin Nöllenburg · Antonios Symvonis

Received: 9 September 2008 / Accepted: 19 January 2009 / Published online: 5 February 2009

Abstract An illustration with textual labels may be hard to read if the labels overlap parts of the illustration. Boundary labeling addresses this problem by attaching the labels to the boundary of a rectangle that contains all features. Then, each feature should be connected to its associated label through a polygonal line, called *leader*, such that no two leaders intersect.

In this paper we study the boundary labeling problem with octilinear leaders, i.e., leaders involving horizontal, vertical, and diagonal segments. In order to produce crossing-free boundary labelings, we combine different pairs of octilinear leaders. Thus, we are able to overcome infeasibility problems that might arise if only a single type of leader is allowed. Our main contribution is a new algorithm for solving the total leader length minimization problem (i.e., the problem of finding a crossing-free boundary labeling, such that the total leader length is minimized) assuming labels of uniform size. We also present an NP-completeness result for the case where the labels are of arbitrary size.

The work of M. Bekos and A. Symvonis is funded by the project PENED-2003. PENED-2003 is co-funded by the European Social Fund (75%) and Greek National Resources (25%). The work of M. Nöllenburg is supported by the German Research Foundation (DFG) under grant WO 758/4-3.

M.A. Bekos (✉) · A. Symvonis
School of Applied Mathematical & Physical Sciences, National Technical University of Athens, 15780 Zografou, Athens, Greece
e-mail: mikebekos@math.ntua.gr

A. Symvonis
e-mail: symvonis@math.ntua.gr

M. Kaufmann
Institute for Informatics, University of Tübingen, Sand 13, 72076 Tübingen, Germany
e-mail: mk@informatik.uni-tuebingen.de

M. Nöllenburg
Faculty of Informatics, Karlsruhe University, P.O. Box 6980, 76128 Karlsruhe, Germany
e-mail: noellenburg@iti.uka.de

Keywords Boundary labeling · Leaders · Length minimization · Map labeling

1 Motivation

Placing extra information—usually in the form of textual labels—next to features of interest within an illustration constitutes an important task in the process of information visualization. The interest in algorithms that automate this task has increased, due to the large number of applications that stem from diverse areas such as cartography, geographical information systems, etc.

Current research on map labeling has been devoted to labeling point-features of a map, so that each label is placed next to the point that it describes (an extensive bibliography about map labeling is maintained by Strijk and Wolff [17]). In this case, the basic requirement is that the labels should be pairwise disjoint. However, this is not always possible, e.g., in the case where the labels are too large or the feature set is too dense. In practice, large labels are quite usual, e.g., in medical atlases and technical drawings. In such drawings, a commonly used approach is to explain certain features of the drawing with blocks of text, arranged on its boundary. Towards this direction, Bekos et al. [6] focussed on *boundary labeling* and were the first to algorithmically study this labeling approach. In boundary labeling, the labels are attached to the boundary of a rectangle R enclosing all features and each feature is connected with its associated label by using polygonal lines, called *leaders*.

Several authors have proposed algorithms to produce boundary labelings in different settings [5, 6, 8, 12]. Recently, Benkert et al. [8] studied the boundary labeling problem with do-leaders, i.e., polygonal lines consisting of two line segments, where the first line segment is "diagonal" (d) to the side of R containing the label it leads to, whereas the second one is orthogonal (o) to that side (see Figs. 1 and 2). do-leaders maintain a uniform shape and result in simple and easy-to-read labelings that are used in real applications (see for example Fig. 1). However, in the work reported in [8], Benkert et al. studied the case where the labels can be attached only to one side of the rectangle R and they state that the production of a boundary labeling with such leaders is not always feasible. Extending their work, we examine the case of four-sided boundary labeling. We also introduce two new types of leaders and we show that by combining them, the boundary labeling problem becomes always feasible. To the best of our knowledge, this is also the first attempt, where different types of leaders are combined to produce boundary labelings.

2 Problem Definition

The *input* of a boundary labeling problem consists of a set $P = \{s_1, s_2, \ldots, s_n\} \subseteq \mathbb{R}^2$ of n points (referred to as *sites*), where $s_i = (x_i, y_i)$, $i = 1, 2, \ldots, n$. The site set P is enclosed in an axis-parallel rectangle $R = [0, W] \times [0, H]$, which is called *enclosing rectangle*. Each site s_i is associated with an axis-parallel, rectangular label l_i of dimensions $w_i \times h_i$.

The *output* of a boundary labeling problem is a placement of the labels at distinct positions on the boundary of the enclosing rectangle R and a set of leaders connecting

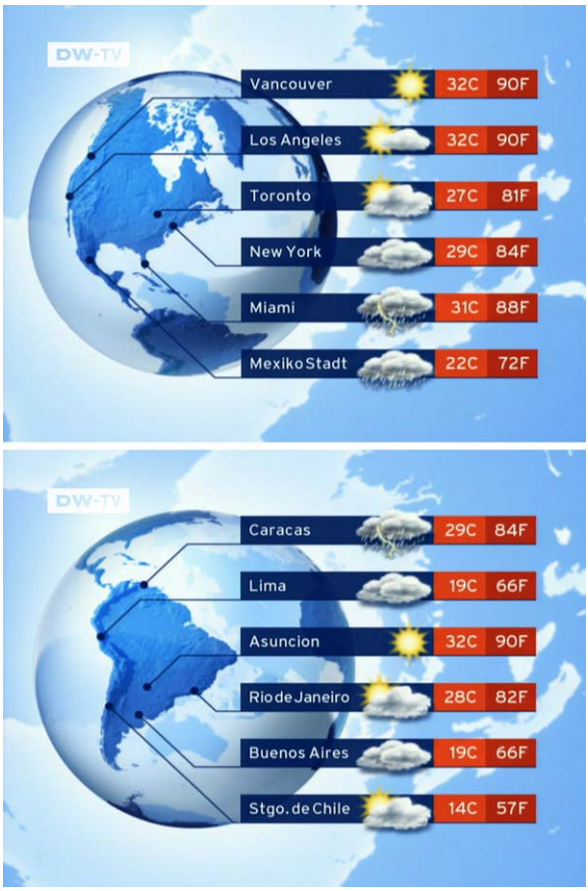


Fig. 1 Boundary labeling applied in a weather forecast map; courtesy of DW-TV

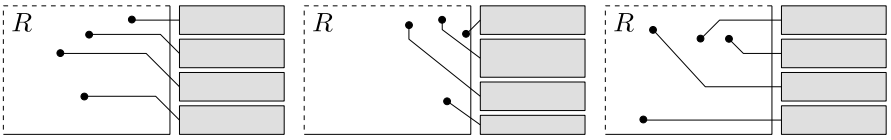
each site with its associated label, so that (i) the labels do not overlap each other and (ii) the leaders do not intersect or overlap each other. Such labelings are referred to as *valid boundary labelings* (or simply as *valid labelings*).

Following the naming scheme of Bekos et al. [6], we focus on three different types of leaders each of which consists of two line segments, where the first one is incident to the site and directed towards the label and the second one incident to the label:

od-leaders: The first line segment of an *od*-leader is orthogonal (o) to the side of R containing the label it leads to. Its second line segment is "diagonal" (d) to that side (see Fig. 2a).

pd-leaders: The first line segment of a *pd*-leader is parallel (p) to the side of R containing the label it leads to. Its second line segment is "diagonal" (d) to that side (see Fig. 2b).

do-leaders: The first line segment of a *do*-leader is "diagonal" (d) to the side of R containing the label it leads to. Its second line segment is orthogonal (o) to that side (see Fig. 2c).



(a) *od*-leaders (b) *pd*-leaders (c) *do*-leaders

Fig. 2 Different types of leaders

In general, the labels are of arbitrary size (*non-uniform labels*; see Fig. 2b). We separately consider the case where the labels are of the same width and height (*uniform labels*; see Figs. 2a and c). The point where each leader touches its associated label (referred to as *port*) can be *fixed*, e.g., the middle point of the label's side that faces the enclosing rectangle R (see Figs. 2a–c) or *sliding*, i.e., any point of the label's side. Throughout this paper, we assume fixed ports and we explicitly state it when our algorithms also work for sliding ports. The labels are usually attached to one, two or all four sides of the enclosing rectangle and are either placed at predefined locations (*fixed labels*) along the sides or can slide (*sliding labels*).

Keeping in mind that we want to obtain simple and easy-to-read labelings, we consider the *leader length minimization problem*, i.e., the problem of determining a valid labeling, such that the total leader length is minimized. This aesthetic criterion is traditionally associated with several optimization problems in the graph drawing literature [2, p. 15], [13, p. 20]. It is important from an aesthetic and cognitive sense, since it is generally agreed that having short edges in the drawing is always preferred from long edges that are difficult-to-follow for the eye especially in the case where they also have bends.

2.1 Preliminaries

We denote the number of sites (and consequently the number of labels) by n. We also denote by c_i the leader of site s_i. We use the notation (p, q) for the unique *do*- or *pd*-leader between points p and q (depending on the context *od*-leaders must be used instead of *do*-leaders). In that sense $c_i = (s_i, l_i)$, where l_i is (the port of) the label of s_i. A set of sites is considered to be in *general positions* if (i) no two sites share the same x- or y-coordinate, (ii) no two sites lie on the same diagonal line and (iii) the horizontal, vertical and diagonal lines that pass through the ports of the labels do not coincide with the sites. In order to avoid leader overlaps, we usually assume that the input site set P is in general positions. We also assume that the sites, the leader bends and label corners have integer coordinates. Consider a leader c_i which originates from site s_i and is connected to a label l_i on the right side AB of R (i.e., A and B are corners of R; see Fig. 3). The horizontal line which coincides with s_i divides the plane into two half-planes (see the dashed line l of Fig. 3). We say that leader c_i is *oriented towards* corner A of the enclosing rectangle R if both corner A and the port of label l_i are in the same half-plane, otherwise, we say that leader c_i is *oriented away* from corner A.

Consider a site s_i that has to be connected to a label l_i on the right side AB of the enclosing rectangle R. The lines that pass through the port of label l_i and form 45°,

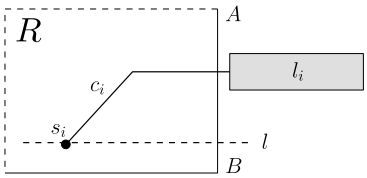


Fig. 3 c_i is oriented towards corner A

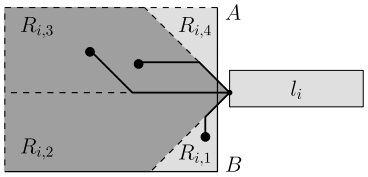


Fig. 4 Connecting site s_i to label l_i

90° and 135° angles with the left side of label l_i, partition the enclosing rectangle into four regions $R_{i,1}$, $R_{i,2}$, $R_{i,3}$, and $R_{i,4}$, as in Fig. 4. If the site s_i lies within a region incident to A or B (i.e., $R_{i,1}$ or $R_{i,4}$; refer to the light-gray colored regions of Fig. 4), then it can only be connected to label l_i using a pd-leader. Otherwise (i.e., site s_i lies within $R_{i,2}$ or $R_{i,3}$; refer to the dark-gray colored regions of Fig. 4), it can be connected to l_i using either a do- or an od-leader. Also, observe that connecting a site to its label with a do-leader, requires the same leader length as with an od-leader. So, depending on the location of site s_i, one has to use an appropriate leader to connect it to its label l_i.

We can define a metric $d_{\text{oct}} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$, such that given two points p and q in $\mathbb{R}^2$, $d_{\text{oct}}(p, q)$ equals the length of the shortest pd-, od- or do-leader between them. Let $|pq|_x$ and $|pq|_y$ be the absolute distances in the x- and y-coordinates, respectively between p and q. Then, the length of the d-segment of the leader connecting p and q is $\sqrt{2}\min(|pq|_x, |pq|_y)$. Similarly, the length of the p- or o-segment of the leader is $\max(|pq|_x, |pq|_y) - \min(|pq|_x, |pq|_y)$. Let $d_1 : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ and $d_\infty : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ be the Manhattan and maximum metrics, respectively.[1] Then, we can express d_{oct} as a linear combination of those metrics (which guarantees that d_{oct} satisfies the metric axioms) as follows:

$$\begin{aligned} d_{\text{oct}}(p, q) &= (\sqrt{2} - 1)\min(|pq|_x, |pq|_y) + \max(|pq|_x, |pq|_y) \\ &= (\sqrt{2} - 1)(d_1(p, q) - d_\infty(p, q)) + d_\infty(p, q) \\ &= (\sqrt{2} - 1)d_1(p, q) + (2 - \sqrt{2})d_\infty(p, q) \end{aligned}$$

3 Previous Work and Our Results

Several authors have proposed algorithms to produce valid boundary labelings in different settings. The first results that come with asymptotic bounds on the running time

[1] Recall that $d_1(x, y) = |x_1 - y_1| + |x_2 - y_2|$ and $d_\infty(x, y) = \max|x_i - y_i|$, where $x = (x_1, x_2)$ and $y = (y_1, y_2)$.

were presented by Bekos et al. [6] (see also [7], who studied a variety of models based on the type of the leaders (i.e., s-leaders[2], po-leaders[3] or opo-leaders[4]), the location (i.e., one, two-opposites or all four sides of R) and the size (i.e., uniform of non-uniform) of the labels. In their work, they presented several algorithms for obtaining valid labelings of either minimum total leader length or of minimum total number of leader bends. In subsequent work, Bekos et al. studied variations of boundary labeling where the labels are arranged in multiple stacks on one side of the enclosing rectangle [4], or where the features to be labeled occupy polygonal regions [5].

Kao, Lin and Yen [12] presented another variation of boundary labeling (referred to as *many-to-one boundary labeling*), where several sites are associated with the same label. Unlike the conventional boundary labeling, the many-to-one boundary labeling inevitably leads to crossings among leaders. Therefore, in their work, they presented several algorithms, approximations and heuristics for minimizing the total number of leader crossings.

Recently, Benkert et al. [8] studied the boundary labeling problem with po- and do-leaders, in the case where uniform labels are allowed to be placed on one side of the enclosing rectangle. They presented algorithms for minimizing the total leader length and they further formulated the boundary labeling problem as an optimization problem, where the objective function is a general quality function which evaluates the niceness of the resulting labeling and then, using dynamic programming, presented several results for obtaining optimal solutions.

Table 1 reviews the most related previous results on boundary labeling and it also presents our contribution (shaded in gray problems).

This paper is structured as follows: In Sect. 4 we prove that the problem of determining a valid boundary labeling of minimum total leader length with do- and pd-leaders and non-uniform labels is NP-complete. In Sects. 5 and 6, we present polynomial time algorithms for obtaining either optimal (in terms of total leader length) or simply valid boundary labelings with labels of uniform size. We conclude in Sect. 7 with open problems and future work.

4 Boundary Labeling with Non-uniform Labels

In this section, we consider the boundary labeling problem with labels of non-uniform size. We are given a set P of n sites $s_i, i = 1, 2, \ldots, n$, each associated with an axis-parallel, rectangular label l_i of height h_i. The labels are allowed to be placed on the right side of the enclosing rectangle R. We further assume fixed label ports, i.e., each leader is connected to its corresponding label using the middle point of the label side that faces the enclosing rectangle. We prove the following theorem:

[2] An s-leader consists of a single line segment from the site to the label.

[3] A po-leader consists of two line segments: The first one is parallel (p) to the side of R containing the label it leads to, whereas the second one is orthogonal (o) to that side.

[4] An opo-leader consists of three line segments: The first one and the third ones are orthogonal (o) to the side of R containing the label it leads to, whereas the second one is parallel (p) to that side.

Table 1 Running times of known algorithms and new algorithms presented in this paper (in big-Oh-Notation) for various versions of boundary labeling, where ϵ is an arbitrarily small positive constant. The time bounds in square parentheses refers to the case of non-uniform labels. The algorithms for the problems marked by * do not always result in valid labelings. Entries in column "Valid solution" are filled only if we can compute a valid solution asymptotically faster than a length-optimal solution

Leader type	# Rect. sides	Valid solution		Length-optimal solution		
				Fixed ports	Sliding ports	Ref.
s	1	$n \log n$	Ref. [7]	$n^{2+\epsilon}$	n^3	[7]
s	4	$n \log n$	Ref. [7]	$n^{2+\epsilon}$	n^3	[7]
po	1			$n \log n$	$n \log n$	[8]
po	2			n^2	n^2	[7]
opo	1	$[n \log n]$	Ref. [7]	$n \log n$	$[n^2]$	[7]
opo	2			n^2 $[nH^2]$	n^2	[7]
opo	4	$n \log n$	Ref. [7]	$n^2 \log^3 n$	n^3	[7]
do	1			$n^{2\star}$	$n^{2\star}$	[8]
do-pd	1			n^3 [NP-complete]	n^3	Thm. 1, 2
od-pd	1	$\log n$	Thm. 5	n^3	n^3	Thm. 2
do-pd	2			n^3	n^3	Thm. 3
od-pd	2	$\log n$	Thm. 6	n^3	n^3	Thm. 3
od-pd	4	$\log n$	Thm. 6	n^3	n^3	Thm. 4

Theorem 1 *Given a set P of n sites, a label l_i of height h_i for each site s_i and an integer $k \in \mathbb{Z}^+$, it is NP-complete to decide whether there exists a valid boundary labeling with do- and pd-leaders of total leader length no more than k.*

Proof For simplicity, we relax the general position restriction i.e., we assume that the sites can be placed in arbitrary positions. At the end of the proof, we briefly describe how it can be adopted to also hold for sites in general position.

Membership of *NP* follows from the fact that a non-deterministic algorithm only needs to guess a positioning of the labels on the boundary of R, a set of leaders connecting each site with its associated label and check in polynomial time that (i) the labels do not overlap each other, (ii) the leaders do not intersect each other and (iii) the sum of the lengths of all leaders is not more than k.

We will reduce the following single-machine scheduling problem (known as *total discrepancy problem* [11]) to our problem: We are given a set J of $2n + 2$ jobs $J_0, J_1, J_2, \ldots, J_{2n}, J_{2n+1}$, which are to be executed on one machine non-preemptively. Each job J_i is associated with a known deterministic processing time $p_i \in \mathbb{N}$, such that $0 < p_0 < p_1 < \cdots < p_{2n}$ and $p_{2n+1} = 2$. We are also given a single *preferred midtime* $M \in \mathbb{Z}^+$, which corresponds to the time at which we would like the first half of each of the first $2n + 1$ jobs (i.e., $J_0, J_1, \ldots, J_{2n}$) to be completed. Without loss of generality, we assume that M is large (e.g. $M > \sum_{i=0}^{2n} p_i$). Given a schedule σ, we denote the *starting* (*completion*) *time* of job J_i in σ by $S_i(\sigma)$ ($C_i(\sigma)$)

and we use $M_i(\sigma)$ to denote its *midtime*, i.e., $M_i(\sigma) = S_i(\sigma) + p_i/2$, or equivalently, $M_i(\sigma) = C_i(\sigma) - p_i/2$. Under a schedule σ:

- A job J_i, $i = 0, 1, \ldots, 2n$ is considered to be *on-time* if its midtime $M_i(\sigma)$ is equal to the preferred midtime M and in this case, it incurs no penalty. On the other hand, if the midtime $M_i(\sigma)$ of J_i commences prior to M (exceeds M), an earliness (tardiness) penalty $E_i(\sigma) = M - M_i(\sigma)$ ($T_i(\sigma) = M_i(\sigma) - M$) incurs.
- The last job J_{2n+1} is considered to be on-time if its midtime $M_{2n+1}(\sigma)$ is equal to $M' = 2M + p_{2n+1}/2 = 2(M+1)$, and as in the previous case it incurs no penalty. If its midtime $M_{2n+1}(\sigma)$ commences prior to M' (exceeds M'), an earliness (tardiness) penalty $E_{2n+1}(\sigma) = M' - M_{2n+1}(\sigma)$ ($T_{2n+1}(\sigma) = M_{2n+1}(\sigma) - M'$) incurs.

The objective is to determine a schedule σ, so that the total earliness-tardiness penalty $\sum_{i=0}^{2n+1}(E_i(\sigma) + T_i(\sigma)) = \sum_{i=0}^{2n}|M - M_i(\sigma)| + |M' - M_{2n+1}(\sigma)|$ is minimized. Garey, Tarjan and Wilfong [11] proved that an optimal schedule σ_{opt} of the first $2n+1$ jobs $J_0, J_1, \ldots, J_{2n}$ has the following properties:

(1) σ_{opt} does not have any gaps between the jobs.
(2) $M_0(\sigma_{\text{opt}}) = M$, i.e., the job with scheduled midpoint M is the task with the shortest processing time.
(3) If $A(\sigma_{\text{opt}}) = \{J_i : M_i(\sigma_{\text{opt}}) < M\}$ & $B(\sigma_{\text{opt}}) = \{J_i : M_i(\sigma_{\text{opt}}) > M\}$, then $|A(\sigma_{\text{opt}})| = |B(\sigma_{\text{opt}})| = n$.
(4) $\sigma_{\text{opt}} = [A_n, A_{n-1}, \ldots, A_1, J_0, B_1, B_2, \ldots, B_n]$, where $\{A_i, B_i\} = \{J_{2i}, J_{2i-1}\}$, i.e., if $A_i = J_{2i}$ then $B_i = J_{2i-1}$ otherwise $A_i = J_{2i-1}$ and $B_i = J_{2i}$.
(5) The minimum total earliness-tardiness penalty is equal to

$$ETP = \sum_{i=1}^{n}(p_{2i} + p_{2i-1})(n - i + 1/2) + np_0$$

Garey, Tarjan and Wilfong [11] further proved that a schedule of all jobs (i.e., including the $(2n+2)$-th job) with total earliness-tardiness penalty equal to ETP should also have the above properties for the first $2n+1$ jobs and therefore that the total cost due to the J_{2n+1} should be zero. However, the decision problem whether such a schedule exists is *NP*-complete [11].

The reduction we propose can be achieved in linear time. Let I_S be an instance of the total discrepancy problem mentioned above. We proceed to construct an instance I_L of our problem as follows: For each job J_i, $i = 0, 1, \ldots, 2n$, we introduce a site s_i placed at point $(2n + 1 - i, M)$, i.e., the sites that correspond to the first $2n+1$ jobs are collinear, lie on the horizontal line $y = M$ and the horizontal distance between two consecutive sites is one unit. We also introduce a site s_{2n+1} associated with job J_{2n+1} and placed at $(2n+1, M')$. For $i = 0, 1, \ldots, 2n+1$, the label l_i associated with site s_i has height h_i equal to the processing time p_i of job J_i. The bottom left corner of the enclosing rectangle R is $(0, M - \sum_{i=0}^{2n+1} p_i)$. The height H of R is equal to $\sum_{i=0}^{2n+1} p_i$, which ensures that all labels can be placed at the right side of R without gaps between them. We seek to exclude the case where a site can be connected to its label through a pd-leader. So, the enclosing rectangle should be of appropriate width. We set its width W to be equal to $H/2 + 2n + 1$ (see Fig. 5). This ensures that the

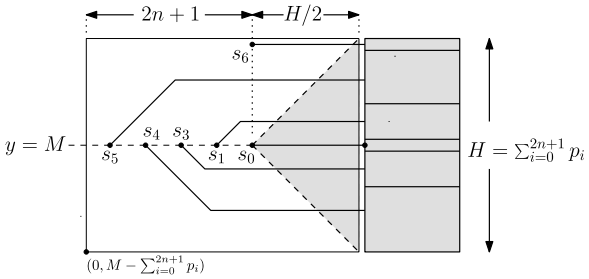


Fig. 5 For each job J_j, $i = 0, 1, \ldots, 2n$ we introduce a site s_i placed at $(2n + 1 - i, M)$

gray-colored triangular area contains no sites and therefore, all sites can be connected to their associated labels through do-leaders only.

In the following, we show that a schedule σ of I_S with total earliness-tardiness penalty ETP implies a valid labeling L of I_L with total leader length $(\sqrt{2}-1)ETP+(2n+1)(W-n-1)+H/2$ and vice versa.

Suppose that we can derive a schedule σ of I_S with total earliness-tardiness penalty ETP. We place each label l_i, so that the y-coordinate of its bottom boundary edge is equal to $S_i(\sigma)$, i.e., equal to the starting time of job J_i under schedule σ. We connect each of these labels to their associated sites by using do-leaders, except labels l_0 and l_{2n+1} which can be connected using o-leaders, since the jobs J_0 and J_{2n+1} do not incur penalties in σ. Obviously, the length of the leader from the site s_{2n+1} is equal to $H/2$ (see Fig. 5). From the location of the labels, it follows that the length of the d-segment of any other leader c_i (i.e., $i \neq 2n+1$) which connects site s_i to its associated label, is equal to $\sqrt{2}\,|M - M_i(\sigma)|$. The length of its o-segment is equal to $W-(2n+1-i)-|M-M_i(\sigma)|$. This implies that the total length of leader c_i is equal to $(\sqrt{2}-1)\,|M-M_i(\sigma)|+W-(2n+1-i)$. However the total earliness-tardiness penalty of schedule σ is equal to ETP, which implies that $\sum_{i=0}^{2n}|M-M_i(\sigma)| = ETP$. Summing up the length of all leaders, the total leader length of labeling I_L is equal to $(\sqrt{2}-1)ETP+(2n+1)(W-n-1)+H/2$. Also, from the fourth property of schedule σ, it follows that I_L is valid.

Suppose now that there exists a valid labeling L of I_L with total leader length $(\sqrt{2}-1)ETP+(2n+1)(W-n-1)+H/2$. Due to the position of the sites, the construction ensures that labeling L contains only do-leaders. We further observe that in any valid labeling of I_L, the leader c_{2n+1} from the site s_{2n+1} should be of an o-leader (and thus its label should be placed topmost as in Fig. 5), since otherwise at least one leader would cross c_{2n+1}. We proceed to derive a schedule σ for I_S as follows: We set the starting time $S_i(\sigma)$ of each job J_i to be equal to the y-coordinate of the bottom boundary edge of label l_i. Since the leader c_{2n+1} is an o-leader, the total cost due to the J_{2n+1} should be zero in σ. Let d_{s_i} and o_{s_i} be the d- and o-segment of leader c_i, respectively. Then, simple geometric properties show that:

- $length(d_{s_i}) = \sqrt{2}\,penalty(J_i)$, $i = 0, 1, \ldots, 2n$.
- $length(o_{s_i}) = W-(2n+1)+i-penalty(J_i)$, $i = 0, 1, \ldots, 2n$.

From the above relations, it follows easily that the total earliness-tardiness penalty of I_S is ETP.

Finally, we note that for the case where the general position restriction holds, the sites have to have distinct y-coordinates. To achieve this, we can modify the

construction by distributing the sites in a strip of height $2n + 1$ (each site now has a distinct y-coordinate) and by appropriately scaling up the size of all labels. □

Note The *NP*-completeness result of Theorem 1 also holds in the case of boundary labelings with *po* leaders. The proof is almost identical. Instead of measuring the length of each leader using the Euclidean metric, we have to use the Manhattan metric.

5 Boundary Labeling with Uniform Labels

Theorem 1 implies that, unless $P = NP$, we cannot efficiently determine an optimal solution of the boundary labeling problem with non-uniform labels. Therefore, we proceed to consider the case of uniform labels, which is a reasonable assumption, since in real applications the labels usually contain single line texts (for example a place name or an integer used as a legend).

Let $P = \{s_1, s_2, \ldots, s_n\}$ and $L = \{l_1, l_2, \ldots, l_n\}$ be the sets of sites and labels, respectively. We assume that the sites are in general positions and the labels are placed in fixed positions on the boundary of R. For simplicity, we assume fixed label ports. At Sect. 5.4, we explain how to extend the presented algorithm to also support sliding ports. Since the labels are of uniform size, each site s_i can be connected to any label l_j. We seek to connect each site s_i to a label l_j, so that the total leader length is minimized. Our approach is outlined in Algorithm 1.

Initially, we construct a complete weighted bipartite graph $G = (P \cup L, E, w)$ between all sites $s_i \in P$ and all labels $l_j \in L$, where $E = \{(s_i, l_j); s_i \in P, l_j \in L\}$ and $w : E \to \mathbb{R}$ is a cost function (see step A of Algorithm 1). Each edge $e_{ij} = (s_i, l_j) \in E$ of G is assigned a weight $w(e_{ij}) = d_{ij}$, where d_{ij} is equal to the length of the leader which connects site s_i with label l_j. Recall that the type of the leader that will be used to connect site s_i to label l_j depends on their relative positions, as stated in Sect. 2.1. Also, recall that if a site can be connected to its associated label with a *do*-leader, it can also be connected using an *od*-leader. However, in both cases the total length required is the same and consequently, the edge e_{ij} is assigned the same weight, regardless the type of the leader that will be eventually used (i.e., *do* or *od*).

We proceed by computing a minimum-cost bipartite matching on G, i.e., we compute a matching between the sites and the labels that minimizes the total distance of the matched pairs (see step B of Algorithm 1). Since G is a complete bipartite graph, a perfect matching always exists. Then, we can obtain a labeling M of minimum total leader length as follows: If an edge $e_{ij} = (s_i, l_j) \in E$ is selected in the matching, then we connect site s_i with label l_j using a leader of length $w(e_{ij})$ (see step C of Algorithm 1). Labeling M is of optimal total leader length, but it might contain crossing leaders. In the remainder of this section, we will prove that all crossings can be eliminated while keeping the total leader length unchanged, i.e., equal to that of M (see step D of Algorithm 1), in the case where the leaders are (i) either *do*- and *pd*-leaders or (ii) *od*- and *pd*-leaders and the labels are allowed to be attached to one or two opposite sides of R. In the more general case, where the labels are allowed to be placed to all four sides of the enclosing rectangle R, we will prove this result only for combinations of *od*- and *pd*-leaders.

Algorithm 1: Generic Min-Length Algorithm

input : A set $P = \{s_1, \ldots, s_n\}$ of n sites and a set $L = \{l_1, \ldots, l_n\}$ of n uniform labels placed on the boundary of R.
output : A crossing free boundary labeling of minimum total leader length.
require: If the labels are placed on one or two opposite sides of R, then the leaders should be i) either of do- and pd-leaders or ii) od- and pd-leaders. Otherwise, the leaders should be od- and pd-leaders only.

Step A: *Construct a complete weighted bipartite graph.*
Construct a complete weighted bipartite graph $G = (P \cup L, E, w)$ between all sites $s_i \in P$ and all labels $l_j \in L$. The weight $w(e_{ij})$ of an edge $e_{ij} = (s_i, l_j) \in E$ is the length of the leader, say d_{ij}, which connects s_i with l_j.
Step B: *Compute a Minimum Cost Bipartite Matching.*
Compute a minimum-cost perfect bipartite matching $\mathcal{M}$ of G, i.e., compute a matching between sites and labels that minimizes the total distance of the matched pairs.
Step C: *Obtain an optimal boundary labeling M.*
foreach (edge $e_{ij} = (s_i, l_j) \in \mathcal{M}$) **do**
Connect site s_i to label l_j s.t. $length(c_i) = w(e_{ij})$
Step D: *Eliminate crossings.*
Eliminate all crossings among pairs of leaders and obtain a valid boundary labeling M', keeping the total leader length unchanged, i.e., equal to that of M.

Note Algorithm 1 can be also applied for the case where more uniform labels are available at the boundary of the enclosing rectangle R than the number of sites. In this case, the perfect bipartite matching of Step B will assign one distinct label to each site, while the remaining labels will not be used.

5.1 One-Sided Boundary Labeling

We first describe how to eliminate all crossings of labeling M (obtained from Step C of Algorithm 1), assuming that the labels are allowed to be attached to one side of the enclosing rectangle R, say the right side AB. Note that labeling M is of minimum total leader length and the leaders used to produce it in Step C of Algorithm 1, are (i) either do- and pd-leaders or (ii) od- and pd-leaders. Our aim is to eliminate all crossings and obtain a valid labeling M' that keeps the total leader length unchanged.

Lemma 1 *Let M be an optimal one-sided boundary labeling either with do- and pd-leaders or with od- and pd-leaders (which may contain crossings) obtained from Step C of Algorithm 1. Let c_i and c_j be a pair of intersecting leaders originating from sites s_i and s_j, respectively. Then the following hold*:

(i) *Leaders c_i and c_j are oriented towards the same corner of R.*
(ii) *Leaders c_i and c_j are of the same type.*

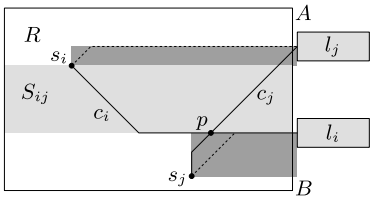


Fig. 6 The total length of two oppositely directed leaders can be decreased. The bounding boxes of the new (*dotted*) leaders are shaded in dark gray

(iii) *Leaders c_i and c_j can be rerouted so that they do not cross each other, the sum of their leader length remains unchanged, their type remains unchanged and they are oriented towards the same corner of R.*

Proof We will only show the proof for the combination of do- and pd-leaders. Analogous reasoning yields the result for od- and pd-leaders.

(i) For the sake of contradiction let's assume that c_i and c_j are directed into opposite directions. Figure 6 shows an example of a pd- and a do-leader that intersect in a point p. We can assume that c_i is directed towards B and c_j is directed towards A; thus we know $y(l_i) < y(l_j)$ and $y(s_j) < y(s_i)$. The length of leader c_i is $d_{\text{oct}}(s_i, l_i) = d_{\text{oct}}(s_i, p) + d_{\text{oct}}(p, l_i)$ and the length of c_j is $d_{\text{oct}}(s_j, l_j) = d_{\text{oct}}(s_j, p) + d_{\text{oct}}(p, l_j)$. Now the triangle inequality yields $d_{\text{oct}}(s_i, l_j) \leq d_{\text{oct}}(s_i, p) + d_{\text{oct}}(p, l_j)$ and $d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_j, p) + d_{\text{oct}}(p, l_i)$ for the dotted leaders in the figure, that is, $d_{\text{oct}}(s_i, l_j) + d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_i, l_i) + d_{\text{oct}}(s_j, l_j)$. In fact, the dotted leaders are shorter than c_i and c_j: the point p is not inside the bounding box (shaded in dark gray in Fig. 6) of at least one of the dotted new leaders since $p \in S_{ij} = (\mathbb{R} \times [\max(y(s_j), y(l_i)), \min(y(s_i), y(l_j))]) \cap R$, the horizontal strip that is occupied by both c_i and c_j, see Fig. 6. Therefore, the pd- or do-path via p that leaves the bounding box of site and label must be longer than the direct pd- or do-path inside the bounding box. This is a contradiction to the optimality of M.

(ii) To show property (ii) we can now assume that both leaders are directed towards corner A. Further assume that c_i is a do-leader and c_j is a pd-leader as depicted in Fig. 7. Doing an exhaustive case analysis we show that in all cases the leader length can be reduced, which contradicts the optimality of the initial solution. In Fig. 7a the crossing is between the d-segment of c_i and the p-segment of c_j. By swapping the label assignment (dotted leaders), the total leader length decreases: the new leader from s_i to l_j has the same length as the combination of the segments (s_i, p) and (p, l_j), but the new leader from s_j (or $\hat{s}_j$) to l_i is shorter than (s_j, p) (or $(\hat{s}_j, p)$) and (p, l_i) as it shortcuts the detour via p. Note that the difference between s_j and $\hat{s}_j$ is that (s_j, l_i) is a do-leader while $(\hat{s}_j, l_i)$ remains a pd-leader. This is because $s_j \in R_{i,2}$ while $\hat{s}_j \in R_{i,1}$.

Figures 7b and 7c show a crossing between the o-segment of c_i and the p-segment of c_j. The difference is that in the first case we have $s_i \in R_{j,1}$ and in the second case $s_i \in R_{j,2}$; this yields either a pd- or a do-leader for s_i after the swap (dotted leaders). In both cases the length of the new leader from s_i to l_j is smaller than the length of (s_i, p) and (p, l_j), and the length of the new leader from s_j (or $\hat{s}_j$) to l_i is smaller than the length of (s_j, p) (or $(\hat{s}_j, p)$) and (p, l_i).

In the last case (Fig. 7d) the crossing is between the o-segment of c_i and the d-segment of c_j. Similar to Fig. 7a the length of the new leader from s_i to l_j equals the

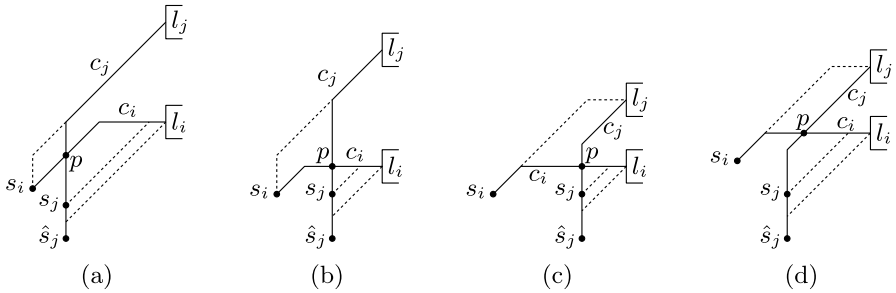


Fig. 7 The total length of two leaders of different type that cross can be decreased

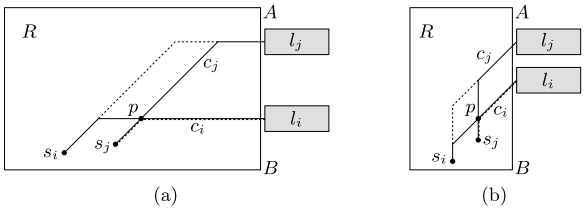


Fig. 8 Swapping two crossing leaders removes the crossing and keeps the total length

length of (s_i, p) and (p, l_j), while the new leader from s_j (or $\hat{s}_j$) to l_i is shorter than (s_j, p) (or $(\hat{s}_j, p)$) and (p, l_i). So for any pair of different type crossing leaders we can strictly decrease their total length. This contradicts the optimality of M and thus shows property (ii).

(iii) We now proceed to show property (iii), i.e., how crossings between two leaders of the same type and with the same orientation can be removed without increasing the sum of their lengths and without changing their type and orientation. Figure 8a shows the case of two do-leaders and Fig. 8b shows two pd-leaders that intersect. In both cases the total length of the swapped dotted leaders equals the sum of the lengths of c_i and c_j: we know that $d_{\text{oct}}(s_i, l_i) = d_{\text{oct}}(s_i, p) + d_{\text{oct}}(p, l_i)$ and $d_{\text{oct}}(s_j, l_j) = d_{\text{oct}}(s_j, p) + d_{\text{oct}}(p, l_j)$ since p lies on c_i and c_j. By the triangle inequality we have $d_{\text{oct}}(s_i, l_j) \leq d_{\text{oct}}(s_i, p) + d_{\text{oct}}(p, l_j)$ and $d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_j, p) + d_{\text{oct}}(p, l_i)$. Now $d_{\text{oct}}(s_i, l_j) + d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_i, l_i) + d_{\text{oct}}(s_j, l_j)$ and since M is an optimal solution the two sums must indeed be equal. Furthermore, neither orientation nor leader type has been changed. This concludes the proof of the lemma. □

Lemma 2 *Let M be an optimal one-sided boundary labeling either with do- and pd-leaders or with od- and pd-leaders (which may contain crossings) obtained from Step C of Algorithm 1. We can always determine a crossing-free labeling M' with total leader length equal to that of M (step D of Algorithm 1). Moreover, labeling M' can be obtained in $O(n^2)$ time.*

Proof By Lemma 1, it follows that leaders involved in a crossing are of the same type and oriented towards the same corner of R. We show how to eliminate all crossings of labeling M by rerouting the crossing leaders. Our method performs four passes over the sites. In the first and second pass, we eliminate all crossings among the

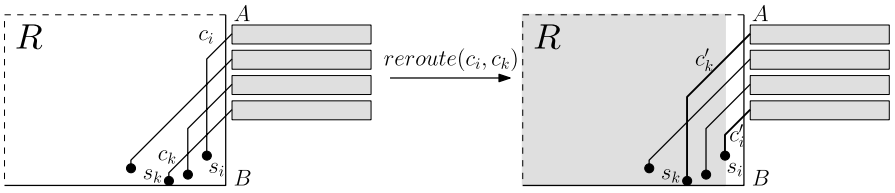


Fig. 9 Rerouting the crossing pd-leaders c_i and c_k

pd-leaders, which are oriented towards the top right and bottom right corner of R, respectively. In the third and fourth pass, we eliminate all crossings among the remaining leaders (i.e. either do-leaders or od-leaders), which are oriented towards the top right and bottom right corner of R, respectively. For simplicity, we will restrict our proof for combinations of do- and pd-leaders.

As already mentioned, in the first pass we consider the sites whose leaders are pd-leaders oriented towards the top right corner, say A, of the enclosing rectangle R. We examine these sites from right to left. We are interested only in those sites, that have crossing leaders. Let s_i be the first such site and let c_i be the leader that connects it to its corresponding label on the right side AB of R (see the left part of Fig. 9). By Lemma 1(i) and (ii), all leaders that intersect c_i are also pd-leaders and oriented towards corner A. Let s_k be the site whose leader c_k intersects c_i and its label is placed bottommost. From Lemma 1(iii), it follows that we can reroute leaders c_i and c_k so that the total leader length remains unchanged (see the right part of Fig. 9). Note that the rerouting possibly eliminates more than one crossing but, in general, it may also introduce new crossings with other pd-leaders, oriented towards corner A.[5] However, the crossings are now located to the left of the vertical line that coincides with s_i (within the gray-colored region of Fig. 9). Continuing in the same manner, the line which forms the region containing the crossings in the right-to-left pass is pushed to the left (i.e., the area of this region is reduced at each iteration, in the right-to-left pass), which guarantees that all crossings—among pd-leaders that are oriented towards corner A—are eventually eliminated. The second pass is handled in a similar fashion.

In the third pass, we seek to eliminate the crossings among the sites whose leaders are do-leaders oriented towards the top right corner A of the enclosing rectangle R. To achieve this, we follow a similar approach as in the first and second pass, i.e., we examine the sites in turn and we perform appropriate reroutings. More precisely, in this case, we examine the sites according to the order in which they are intersected by the sweep line $\ell : y = x$, which sweeps the plane in the north-west direction. Then, assuming that s_i is the first site in the order, whose leader c_i is involved in a crossing, and c_k is the leader whose label is placed bottommost and intersects c_i, we proceed to reroute c_i and c_k (see Fig. 10). By Lemma 1(iii), the total leader length remains unchanged. Also, the crossings are located to the left of the line that coincides with the

[5] From Lemma 1(iii), it follows that the new leaders c'_i and c'_k are also pd-leaders, oriented towards corner A. This implies that crossings with pd-leaders oriented away from corner A or od/do-leaders cannot occur, since that would violate the optimality of the initial labeling M.

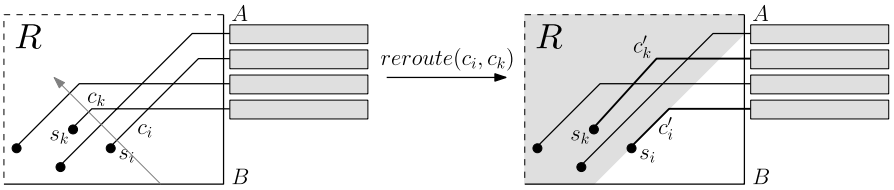


Fig. 10 Rerouting the crossing do-leaders c_i and c_k

d-segment of the new leader c'_i (within the gray-colored region of Fig. 10). Continuing in the same manner, the line which forms the region containing the crossings is pushed to the left, towards the top left corner of R, which guarantees that all crossings will eventually be eliminated. The fourth pass is handled in a similar fashion.

When the four independent passes over the site set are completed, we have eliminated all crossings, resulting in a labeling M' without any crossings and of total leader length equal to that of M, i.e., of minimum total leader length. To complete the proof of the lemma, it remains to explain how to obtain the new labeling M' in $O(n^2)$ time, given that labeling M is of minimum total leader length. At each pass, we sort the site set appropriately. This can be done in $O(n \log n)$ time. At each iteration over the sorted sets of sites, we are interested in finding a specific leader, which crosses the leader of the site that we currently consider. In a straight-forward manner, this can be computed in $O(n)$ time.[6] This results in a total of $O(n^2)$ time for each pass and, consequently, for the elimination of all crossings. □

Theorem 2 *Given a site set P of n sites and a set L of n labels of uniform size placed at fixed positions on one side of the enclosing rectangle R, we can compute a valid boundary labeling of minimum total leader length with either do- and pd-leaders or with od- and pd-leaders in $O(n^3)$ time.*

Proof In Step A of Algorithm 1, we construct a complete weighted bipartite graph $G = (P \cup L, E, w)$ between all sites $s_i \in P$ and all labels $l_j \in L$, where the weight of an edge $e_{ij} = (s_i, l_j) \in E$ is the length of the leader connecting site s_i to label l_j. The computation of each edge weight requires constant time. Hence, the construction of G can be done in $O(n^2)$ time. In Step B of Algorithm 1, we have to compute a minimum cost bipartite matching on the graph G, which can be done efficiently by means of the Hungarian method in $O(n^3)$ time [14]. Note that we cannot use Vaidya's algorithm [15] to reduce the time complexity of Step B, since the leaders are neither straight lines (Euclidean metric) nor rectilinear (Manhattan metric). The solution obtained from Step C of Algorithm 1 is optimal. However, it may contain crossings. In Step D of Algorithm 1, the crossings are eliminated in $O(n^2)$ time. Thus, the total time complexity of Algorithm 1 for the case of one-sided boundary labeling with either do- and pd-leaders or with od- and pd-leaders is $O(n^3)$. □

[6] We note that it is also not too difficult to achieve $O(1)$ amortized time. We do not present the details since the established time complexity will not play the dominant role in the asymptotic analysis of Algorithm 1.

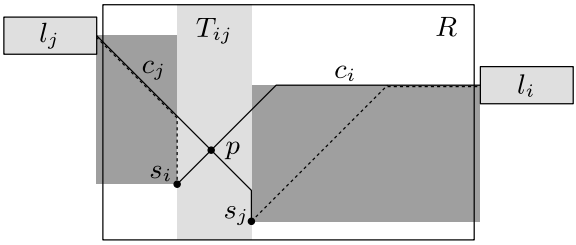


Fig. 11 Crossing leaders to opposite sides can be shortened

5.2 Two-Sided Boundary Labeling

In this subsection, we consider the case where the labels are allowed to be attached to two opposite sides of the enclosing rectangle R. To cope with this case, we use Algorithm 1 to obtain a boundary labeling M (not necessarily crossing-free) of minimum total leader length. This can be done in $O(n^3)$ time. We observe that a possible crossing, between two leaders c_i and c_j that lead to labels located at opposite sides of the enclosing rectangle, cannot occur (as an example see Fig. 11), since the rerouting of the leaders c_i and c_j results in a solution with smaller total leader length. This result is summarized in the following lemma.

Lemma 3 *In an optimal two-sided boundary labeling, crossings between leaders that connect labels located at opposite sides of the enclosing rectangle cannot occur.*

Proof Let's assume that in an optimal solution two leaders c_i and c_j intersect, where c_i is connecting s_i to its label l_i on the right side of R and c_j is connecting s_j to l_j on the left side of R. An example is given in Fig. 11, where c_i is a do-leader and c_j is a pd-leader. Since c_i and c_j intersect it follows directly that $x(s_i) < x(s_j)$. The intersection point p must be located in the vertical strip $T_{ij} = ([x(s_i), x(s_j)] \times \mathbb{R}) \cap R$. Now rerouting the two leaders as (s_i, l_j) and (s_j, l_i) (dotted leaders in Fig. 11) yields $d_{\text{oct}}(s_i, l_j) \leq d_{\text{oct}}(s_i, p) + d_{\text{oct}}(p, l_j)$ and $d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_j, p) + d_{\text{oct}}(p, l_i)$ by the triangle inequality. Therefore, $d_{\text{oct}}(s_i, l_j) + d_{\text{oct}}(s_j, l_i) \leq d_{\text{oct}}(s_i, l_i) + d_{\text{oct}}(s_j, l_j)$. More precisely, this inequality is strict, since p is not inside the bounding box (shaded in gray in Fig. 11) of at least one of the dotted new leaders, i.e., the path from the site to its label via p is longer than the corresponding pd- or do-leader. This contradicts the optimality of the solution and thus there cannot be a crossing between two leaders connecting to labels at opposite sides of R. □

From Lemma 3, it follows that we can independently eliminate the crossings along the two opposite sides of R. The following theorem summarizes our result.

Theorem 3 *Given a site set P of n sites and a set L of n labels of uniform size, placed at fixed positions, on two opposite sides of the enclosing rectangle R, we can compute a valid boundary labeling of minimum total leader length with either do- and pd-leaders or with od- and pd-leaders in $O(n^3)$ total time.*

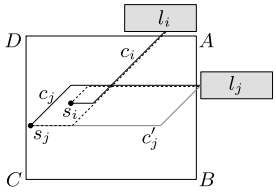


Fig. 12 An instance that does not have a valid labeling with *do*- and *pd*-leaders

5.3 Four-Sided Boundary Labeling with *od*- and *pd*-Leaders

In this subsection, we consider the general case of determining a valid boundary labeling of minimum total leader length with *od*- and *pd*-leaders, where the labels are allowed to be attached to all four sides of the enclosing rectangle R. Note that it is essential to use *od*-leaders and not *do*-leaders in combination with the *pd*-leaders. The reason is that there are instances that do not have a valid labeling consisting of *do*- and *pd*-leaders only, see Fig. 12. In this example the black leaders c_i and c_j intersect. But swapping the label assignment the two dotted leaders still intersect. The only way to avoid the crossing is to use c_i and the gray *od*-leader c'_j instead of the *do*-leader c_j.

Again, we use Algorithm 1 to obtain a labeling M of minimum total leader length. By Lemma 1, it follows that crossing leaders that connect labels placed at the same side of R are of the same type and oriented towards the same corner of R. Crossings between leaders that connect labels placed at opposite sides of R cannot occur, because of Lemma 3. For the case where two crossing leaders connect to labels placed at two adjacent sides of R, we can show the following lemma.

Lemma 4 *Let M be an optimal four-sided boundary labeling with od- and pd-leaders (which may contain crossings) obtained from Step C of Algorithm 1. Let c_i and c_j be a pair of intersecting leaders originating from sites s_i and s_j, respectively. Let l_i and l_j be their associated labels, which lie on two adjacent sides of the enclosing rectangle R. Then the following hold*:

(i) *Leaders c_i and c_j are oriented towards the same corner of R.*
(ii) *Leaders c_i and c_j are of different type (i.e., the diagonal segments of the leaders of c_i and c_j are parallel to each other, whereas the non-diagonal ones are either horizontal or vertical).*
(iii) *Leaders c_i and c_j can be rerouted so that they do not cross each other, the sum of their leader length remains unchanged, their type remains unchanged and they remain oriented towards the same corner of R.*

Proof We can assume that the two adjacent sides of R are the top and the right side, so that their incident corner is A. Let c_i be the leader to the top and c_j the leader to the right.

(i) To prove property (i), we lead any combination of different orientations to a contradiction. If c_i is oriented towards A and c_j towards B we can reuse Lemma 1 by a simple reduction to the one-sided case. The idea is to extend the diagonal segment of the leader c_i towards A and the side AB of R until they intersect. This is illustrated

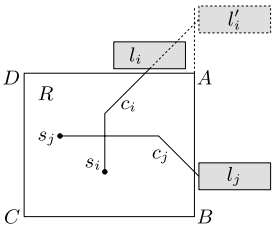


Fig. 13 An *od*-leader to the top side of R is extended into a *pd*-leader to the right side of R

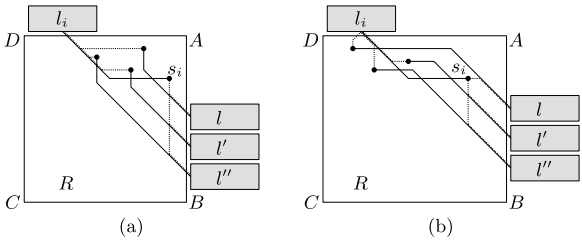


Fig. 14 Two intersecting *pd*-leaders (**a**) and a pair of intersecting *pd*- and *od*-leaders (**b**) directed towards D and B can be shortened by swapping the label assignment

in Fig. 13. The intersection point of the extension lines can be seen as the port of a virtual label l'_i, which is now on the right side of R. Also note that, due to the fact that we only use *od*- and *pd*-leaders, any leader connecting virtual label l'_i will use the diagonal and thus will also pass through the original port for l_i. For this situation Lemma 1(i) gives a contradiction to the optimality of M. The same reduction holds for the symmetric case where c_i is oriented towards D and c_j towards A.

The final case we need to consider is that c_i is oriented towards D and c_j towards B. Figure 14 shows the case where c_i is a *pd*-leader. It can either be intersected by a *pd*-leader directed towards corner B (Fig. 14a shows all three possibilities) or by an *od*-leader oriented towards B (Fig. 14b shows all three possibilities). In both examples the fact that their intersection point lies outside the bounding boxes of the swapped (dotted) leaders yields a contradiction to the optimality of M using the triangle inequality. The case where c_i is an *od*-leader is analogous.

(ii) To prove property (ii), we can assume that both c_i and c_j are oriented towards corner A. First consider the case that both leaders are *pd*-leaders. In that case, however, it is impossible that there is an intersection between c_i and c_j since c_i is inside $R_{i,4}$, c_j is inside $R_{j,1}$, but $R_{i,4} \cap R_{j,1} = \emptyset$, see Fig. 15. The other case, in which both c_i and c_j are *od*-leaders, is again reduced to the one-sided situation by extending c_i, the *od*-leader to the top, such that it becomes a *pd*-leader to the right (recall Fig. 13). Now Lemma 1(ii) yields that if there is a crossing between two leaders going to the same side of R then both leaders must be of the same type—but this is not the case for the *od*-leader c_j and the *pd*-extension of c_i to the right. This completes the proof of property (ii).

(iii) Property (iii) can be shown in a similar way as in Lemma 1(iii). □

Lemma 5 *Let M be an optimal four-sided boundary labeling with od- and pd-leaders (which may contain crossings) obtained from Step C of Algorithm 1. We can determine a valid labeling M' with total leader length equal to that of M (step D of Algorithm 1). Moreover, labeling M' can be obtained in $O(n^2)$ time.*

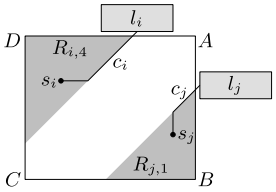


Fig. 15 Two pd-leaders towards corner A cannot intersect

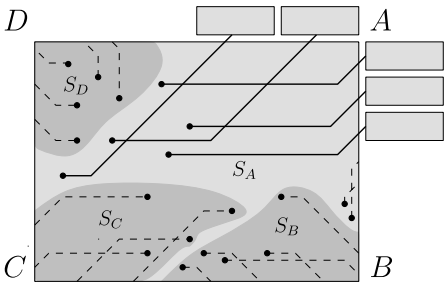


Fig. 16 Sets S_A, S_B, S_C and S_D

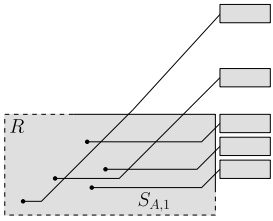
Proof We partition the set of sites into four disjoint sets S_A, S_B, S_C and S_D, each of those contains the sites whose leaders are oriented towards the top-right, bottom-right, bottom-left and top-left corner of R, respectively (see Fig. 16). From Lemma 1 (one-sided case), Lemma 3 (case of two opposite sides) and Lemma 4 (case of two adjacent sides), it follows that in an optimal labeling, possible crossings can only occur between leaders that are oriented towards the same corner of R. Therefore, we can independently eliminate the crossings at each of the sets S_A, S_B, S_C and S_D.

We describe in detail how to eliminate the crossings of set S_A. The remaining cases are treated similarly. Recall that the set S_A contains sites whose leaders are oriented towards the top-right corner of R. We proceed to partition the set S_A into two disjoint subsets $S_{A,1}$ and $S_{A,2}$, as follows: Set $S_{A,1}$ ($S_{A,2}$) contains the sites of S_A whose leaders are either (i) od-leaders leading to a label placed at the right (top) side of R or (ii) pd-leaders leading to a label placed at the top (right) side of R. In Fig. 16, the sites that constitute set $S_{A,1}$ are the ones whose leaders are drawn as solid lines.

From Lemma 4(ii), it follows that the leaders, which are involved in a crossing and lead to labels placed at two adjacent sides of R, should be of different type. Furthermore, crossing leaders that connect labels placed at the same side of R, should be of the same type. This directly follows from Lemma 1(ii). Hence, we can independently eliminate the crossings at each of the sets $S_{A,1}$ and $S_{A,2}$.

Consider the set $S_{A,1}$ and let $s \in S_{A,1}$ be a site whose leader c is a pd-leader. Leader c leads to a label placed at the top side of R. By extending appropriately its d-segment, leader c can be viewed as an od-leader leading to a label at the right side of R (see Fig. 17). This implies that we can make use of the algorithm described in the proof of Lemma 2 to eliminate all crossings of $S_{A,1}$. Since all leaders of the sites of $S_{A,1}$ have the same orientation (this holds because $S_{A,1} \subseteq S_A$), their d-segments are parallel to each other, which guarantees that all crossings will occur within the enclosing rectangle. This ensures that our approach will find a valid labeling. Sim-

Fig. 17 Extending a *pd*-leader



ilarly, we eliminate the crossings of the set $S_{A,2}$. The total time needed in order to eliminate the crossings at each of the sets S_A, S_B, S_C and S_D, is $O(n^2)$. Therefore, labeling M' can be obtained in $O(n^2)$ time. □

Theorem 4 *Given a site set P of n sites and a set L of n labels of uniform size placed at fixed positions on all four sides of the enclosing rectangle R, we can compute a valid boundary labeling of minimum total leader length with od- and pd-leaders in $O(n^3)$ time.*

5.4 Sliding Ports

In order to simplify the description of Algorithm 1 and make the study of the accompanying case analysis simpler, we have so far restricted our presentation to the case where the label ports are fixed, i.e., the label port is the middle point of the label's side that faces R. In the remainder of this section we briefly discuss how Algorithm 1 can be easily modified so that Theorems 2, 3 and 4 also hold for the case of sliding label ports.

In Step A of Algorithm 1, we compute a complete weighted bipartite graph G between all sites $s_i \in P$ and all labels $l_j \in L$, where the weight $w(e_{ij})$ of an edge $e_{ij} = (s_i, l_j)$ of G is equal to the length of the leader from s_i to l_j. Obviously, in the case of fixed label ports the leader from s_i to l_j is unique and thus the computation of the edge weight $w(e_{ij})$ is easy.

In the case of sliding label ports, we have to compute the lengths of the shortest leaders from each site s_i to each label l_j, in order to obtain an optimal (in terms of total leader length) labeling. Towards this direction, consider a pair (s_i, l_j) consisting of a site s_i and a label l_j and assume, without loss of generality, that label l_j is on the right side AB of R. As illustrated in Fig. 18, label l_j defines a horizontal strip H_j, two diagonal strips $D_{j,1}$ and $D_{j,2}$ and four regions $R_{j,1}, \ldots, R_{j,4}$. Then, it is easy to characterize the shortest leader from site s_i to label l_j, depending on the relative positions of s_i and l_j. More precisely:

- If $s_i \in H_j$, then the shortest leader from s_i to l_j should be an o-leader.
- If $s_i \in R_{j,2} \cup D_{j,1}$ ($s_i \in R_{j,3} \cup D_{j,2}$), then the shortest leader from s_i to l_j leads to the bottom (top) corner of l_j and is an od-leader or, alternatively, a do-leader of identical length.
- If $s_i \in R_{j,1}$ ($s_i \in R_{j,4}$), then the shortest leader from s_i to l_j leads to the bottom (top) corner of l_j and a pd-leader.

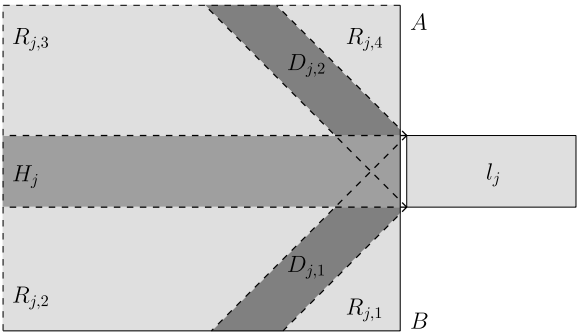


Fig. 18 Computing the shortest leader from site s_i to label l_j

From the above, we observe that, except for the case of o leaders, the label corners are used as ports. Thus, based on condition (iii) of the general position condition, we assume that *no site lies on the horizontal, vertical and diagonal lines that pass through the corners of the labels* and, consequently, there do not exist any d-leaders.

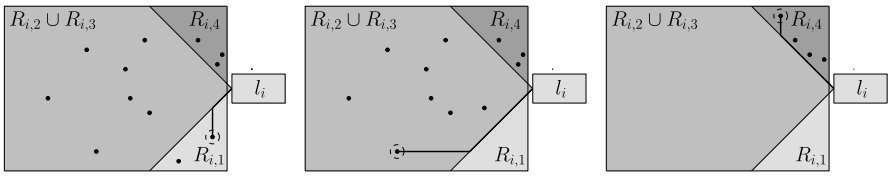
Having computed the shortest leader from each site s_i to each label l_j, we can proceed to Steps B and C of Algorithm 1. We now observe that in a solution of minimum total leader length, the o-leaders are not involved in any crossing. This can be easily established by showing that a crossing that involves at least one o-leader can be resolved so that the two leaders (i) they do not cross and (ii) their total leader length is reduced. This contradicts the optimality of the labeling.

Now, since in the crossing elimination procedure (Step D of Algorithm 1) we do not have any d-leaders or any o-leaders involved in a crossing, no new combination of crossing leader types exists. Thus, the proofs for Lemma 1, 3 and 4 which cover all existing combinations still hold.

6 An Algorithm for Obtaining Valid Boundary Labelings

In this section, we consider the problem of determining a valid boundary labeling with od- and pd-leaders, i.e., we relax the optimality constraint on the resulting labeling. Our aim is to obtain a more efficient algorithm in terms of time complexity. Again, we assume that the sites are in general positions, the labels are of uniform size, placed at fixed positions on all four sides of the enclosing rectangle R. We further assume fixed label ports, i.e., each leader is connected to its corresponding label using the middle point of the label side that faces the enclosing rectangle.

Our basic idea is simple: We first develop an algorithm which determines a valid labeling in the case where the labels are allowed to be attached to one side of the enclosing rectangle R (see Sect. 6.1). Then, using standard plane sweep algorithms [9], we partition R into four disjoint regions such that the algorithm for the one-sided case can be applied to each region separately (see Sect. 6.2).



(a) Case A. **(b)** Case B. **(c)** Case C.

Fig. 19 Illustration of the different cases of Algorithm 2

6.1 An Algorithm for Valid One-Sided Boundary Labelings

Let $P = \{s_1, s_2, \ldots, s_n\}$ and $L = \{l_1, l_2, \ldots, l_n\}$ be the sets of sites and labels, respectively. W.l.o.g. we assume that the labels are placed at the right side AB of R. Our approach is outlined in Algorithm 2.

We process the labels from bottom to top. Let l_i be the i-th label in the order. Let also P' be the set of sites that have not been routed yet. The lines that pass through the port of l_i and form 45°, 90° and 135° angles with the left side of l_i partition R into four regions $R_{i,1}$, $R_{i,2}$, $R_{i,3}$ and $R_{i,4}$ (see Fig. 19). We distinguish the following cases:

Case (a): $P' \cap R_{i,1} \neq \emptyset$.
Refer to Fig. 19a. In the case where there exist sites within the region $R_{i,1}$ that haven't been routed yet (case A of Algorithm 2), we determine the rightmost one and we proceed to connect it to label l_i, through a pd-leader.

Case (b): $P' \cap R_{i,1} = \emptyset$ *and* $P' \cap (R_{i,2} \cup R_{i,3}) \neq \emptyset$.
Refer to Fig. 19b. In the case where region $R_{i,1}$ contains no sites, whereas region $R_{i,2} \cup R_{i,3}$ does (case B of Algorithm 2), we choose to connect the bottommost site of $R_{i,2} \cup R_{i,3}$ to label l_i, through an od-leader.

Case (c): $P' \cap (R_{i,1} \cup R_{i,2} \cup R_{i,3}) = \emptyset$ *and* $P' \cap R_{i,4} \neq \emptyset$.
Refer to Fig. 19c. In the case where neither $R_{i,1}$ nor $R_{i,2} \cup R_{i,3}$ contain sites (i.e. all sites that haven't been routed yet lie within the region $R_{i,4}$; case C of Algorithm 2), we choose to connect the leftmost site of $R_{i,4}$ to label l_i, using a pd-leader.

Observe that the connection of the i-th label to its site does not affect connections with labels later in the order. Therefore, we can inductively show that this approach yields a crossing free boundary labeling. Moreover, in a straight-forward approach, Algorithm 2 needs $O(n^2)$ time. However, the time-complexity can be further improved to $O(n \log n)$ by means of more sophisticated data structures. We state the following theorem.

Theorem 5 *Given a site set P of n sites and a set L of n labels of uniform size placed at fixed positions on one side of the enclosing rectangle R, we can compute a valid boundary labeling with od- and pd-leaders in $O(n \log n)$ time.*

Proof To achieve the desired $O(n \log n)$ time complexity, we use the following data structures:

Algorithm 2: ONE-SIDED VALID ALGORITHM

input : A set $P = \{s_1, \ldots, s_n\}$ of n sites and a set $L = \{l_1, \ldots, l_n\}$ of n uniform labels placed on the right side of R, s.t. the y-coordinate of the top left corner of l_i is smaller than the y-coordinate of the top left corner of l_j, $\forall i < j$.

output : A crossing free one-sided boundary labeling with od- and pd-leaders.

$P' \leftarrow P$;

for $i = 1$ **to** n **do**

  if $(P' \cap R_{i,1} \neq \emptyset)$ **then** $s \leftarrow$ rightmost site of $P' \cap R_{i,1}$; /* Case A */

  else

    if $(P' \cap (R_{i,2} \cup R_{i,3}) \neq \emptyset)$ **then** /* Case B */

      $s \leftarrow$ bottommost site of $P' \cap (R_{i,2} \cup R_{i,3})$

    else $s \leftarrow$ leftmost site of $P' \cap R_{i,4}$; /* Case C */

  connect label l_i to site s;

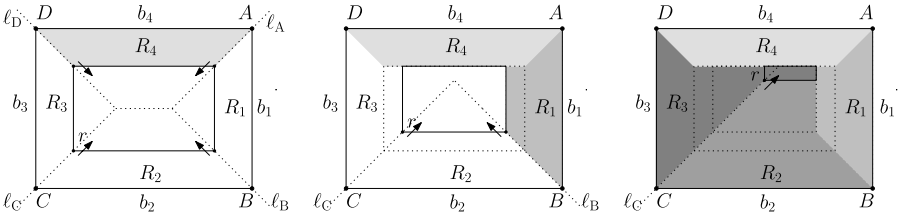
  $P' = P' - \{s\}$

1. Three sets of sites R_1, $R_{2,3}$ and R_4 to store the sites in regions $R_{i,1}$, $R_{i,2} \cup R_{i,3}$ and $R_{i,4}$ during the ith iteration of Algorithm 2, respectively. The sites in R_1 (resp. R_4) are kept sorted in decreasing (resp. increasing) order of their x-coordinates. The sites in $R_{2,3}$ are kept sorted in increasing order of their y-coordinates. These three site sets can be easily implemented by using a balanced binary search tree scheme such as AVL trees [1] or Red-Black trees [3]. Insertions, deletions, as well as search, successor and predecessor queries are supported in $O(n \log n)$ time where n is the number of elements in the data structure.
2. Recall that P is the set of sites and let L_p be the set of label ports. We use two sorted lists of points, denoted by D_1 and D_2, to store the points in $P \cup L_p$ in increasing order according to the direction of the diagonal thought the lower-right and lower-left corners of the enclosing rectangle, respectively, as we move away of them in the interior of the rectangle. Both lists can be created in $O(n \log n)$ time and support $O(1)$ successor and predecessor queries. Moreover, we also keep cross pointers between the two copies of each point in the two lists are kept we can move between lists.

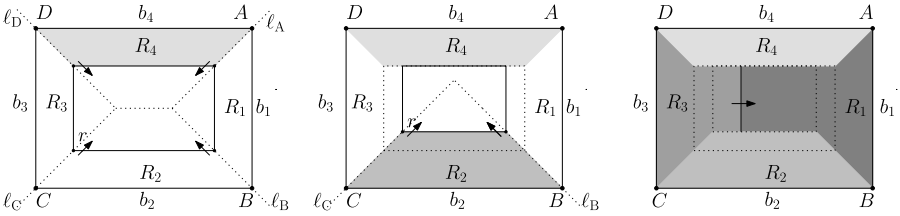
Set R_1 is initialized by inserting into it all sites of D_1 that are before the port of label l_1. Similarly, R_4 is initialized by inserting into it all sites of D_2 that are after the port of label l_1 and are not in R_1. The remaining sites are inserted in $R_{2,3}$. The initialization of all three site sets is completed in $O(n \log n)$ time.

During the ith iteration of Algorithm 2, the site s to be labeled is easily identified (and removed) from either R_1, $R_{2,3}$ or R_4, as well as from both lists D_1 and D_2. Following the deletion of site s, we have to update set R_1, $R_{2,3}$ and R_4 to prepare them for the labeling of the next site. The update of these sets can done in as following:

– Insert into R_1 all sites of list D_1 located between the ports of labels l_i and l_{i+1}. Before the selection of the ith site to be labeled, each of these sites was located in



(a) The first two formed regions are adjacent.



(b) The first two formed regions are not adjacent.

Fig. 20 Partitioning R into four disjoint regions R_1, R_2, R_3, and R_4

either $R_{2,3}$ or R_4. Thus, it has to be first deleted from the corresponding set before it is inserted to R_1.

– Insert into R_1 or $R_{2,3}$ (whichever appropriate for each site) the sites of list D_2 located between the ports of labels l_i and l_{i+1}. Before the selection of the ith site to be labeled, each of these sites was located in R_4. Thus, it has to be first deleted from it before it is inserted to the appropriate set.

During the course of the algorithm, each site moves sets at most two times, that is from R_4 to $R_{2,3}$ to R_1. Thus, the total cost for the maintenance of these three ordered set is $O(n \log n)$. □

6.2 Partitioning R into Four Disjoint Regions

We partition R into four disjoint regions R_1, R_2, R_3, and R_4 so that Algorithm 2 can be applied to each region separately. We have two requirements for a region R_i $(i = 1, 2, 3, 4)$ in the partition of R: (a) R_i must be adjacent to a specific side b_i of R and (b) each site in R_i can be connected to any label attached on b_i either through a do- or a pd-leader.

Let A, B, C, and D be the corners of R from the top right corner in clockwise order, and let $b_1 = \overline{AB}$, $b_2 = \overline{BC}$, $b_3 = \overline{CD}$ and $b_4 = \overline{DA}$ be the sides of R (see Fig. 20). We also assume that we know how many labels have to be attached to each side of R. Let n_1, n_2, n_3, and n_4 be the number of labels attached to the sides b_1, b_2, b_3, and b_4, respectively and ℓ_A, ℓ_B, ℓ_C, and ℓ_D the diagonal lines that pass through the corners A, B, C, and D, respectively and intersect R.

Let r be a rectangle which initially coincides with R. We start simultaneously sliding its vertices along the lines ℓ_A, ℓ_B, ℓ_C, and ℓ_D towards the center of R (see the left drawing of Fig. 20a). During this procedure, the sides of R, the lines ℓ_A, ℓ_B, ℓ_C,

and ℓ_D and the sides of r form four disjoint regions R_1, R_2, R_3, and R_4 adjacent to the sides b_1, b_2, b_3, and b_4, respectively. We stop sliding the vertices of r when R_i contains n_i sites of P, for some $i \in \{1, 2, 3, 4\}$ (R_4 in Fig. 20a). This is the first region of the partition. We remove the diagonal lines that are adjacent to this region and we continue by simultaneously sliding the vertices of r along the remaining diagonal lines (see the middle drawings of Fig. 20). Again we stop when one of the formed regions contains the same number of sites as the number of labels attached to the side of R to which it is adjacent (R_1 in Fig. 20a or R_2 in Fig. 20b). This constitutes the second region of the partition. Similarly, we proceed by removing the diagonal lines that are adjacent to the newly formed region and we distinguish two cases. If the two regions that are already formed are adjacent (refer to Fig. 20a), we similarly proceed by sliding the vertex of r along the remaining diagonal line. In the case where these regions are not adjacent, a horizontal or vertical sliding is required (see Fig. 20b). Using standard plane sweep algorithms [9], the partition of R can be constructed in $O(n \log n)$ time.

Theorem 6 *Given a site set P of n sites, a set L of n labels of uniform size placed at fixed positions on all four sides of the enclosing rectangle R and numbers n_1, n_2, n_3, and n_4 that express how many labels are to be attached to each side of R, we can compute a valid boundary labeling with od- and pd-leaders in $O(n \log n)$ time.*

Proof We first partition the enclosing rectangle R into 4 regions so that each region is adjacent to exactly one side of R and it contains as many sites as the number of labels at its adjacent side. Then, we apply Algorithm 2 separately to each region. Both of these steps can be completed in $O(n \log n)$ time.

To complete the proof, it remains to prove that the above procedure produces a valid labeling. To establish this, it is enough to show that each of the 4 one-sided boundary labeling problems can be solved entirely within its respective region. Without loss of generality, consider region R_1 and its respective side AB (see Fig. 20; the proof is identical for all regions) and observe that no *od*- or *pd*-leader that exits R_1 can connect a site in R_1 with a label next to side AB. Since Algorithm 2 produces a valid labeling and uses only *od*- and *pd*-leaders, it follows that these leaders must be routed entirely within R_1, as required. □

7 Conclusions

In this paper, we studied boundary labelings with octilinear leaders. Several new problems arise from our research and remain to be addressed. Among them, we distinguish the following:

– The complexity of our algorithm for the leader length minimization problem is determined in Step B of Algorithm 1, where we have to compute a minimum-cost bipartite matching. Unfortunately, we cannot use Vaidya's algorithm [15] to reduce it, since the leaders are neither straight lines (Euclidean metric) nor rectilinear (Manhattan metric). It is worth trying to derive a more efficient matching algorithm for this metric.

– In the case where the labels are of arbitrary size, we proved that the total leader length minimization problem is *NP*-complete. So, it is worth trying to derive an approximation algorithm for this problem.
– Clearly, the focus of our work was on the leader length minimization problem. The evaluation of different optimization criteria (e.g., the one of minimizing the total number of leader bends) would also be of particular interest.
– By combining different types of octilinear leaders, we proved that the boundary labeling problem becomes always feasible. However, it is intuitive that the quality of the labelings can be improved by allowing combinations of octilinear and rectilinear (*po* or *opo*) leaders. The evaluation of such mixed model would be of interest.
– Another line of research would be to try to derive labelings that combine the traditional labeling models (fixed [10] or sliding [16]) with our model, i.e., labelings that use leaders to connect labels with their corresponding sites only in the case where it is not possible to place the labels using the traditional models.

References

1. Adel'son-Vel'skii, G., Landis, E.: An algorithm for the organization of information. Sov. Math. Dokl. **3**, 1259–1263 (1962)
2. Battista, G.D., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall, New York (1999)
3. Bayer, R.: Symmetric binary b-trees: data structures and maintenance algorithms. Acta Inf. **1**, 290–306 (1972)
4. Bekos, M.A., Kaufmann, M., Potika, K., Symvonis, A.: Multi-stack boundary labeling problems. In: Proc. 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS2006). Lecture Notes in Computer Science, vol. 4337, pp. 81–92. Springer, Berlin (2006)
5. Bekos, M.A., Kaufmann, M., Potika, K., Symvonis, A.: Polygons labelling of minimum leader length. In: Kazuo, M., Kozo, S., Jiro, T. (eds.) Proc. Asia Pacific Symposium on Information Visualisation (APVIS2006), CRPIT 60, pp. 15–21 (2006)
6. Bekos, M.A., Kaufmann, M., Symvonis, A., Wolff, A.: Boundary labeling: Models and efficient algorithms for rectangular maps. In: Pach, J. (ed.) Proc. 12th Int. Symposium on Graph Drawing (GD'04). Lecture Notes in Computer Science, vol. 3383, pp. 49–59. Springer, New York (2005)
7. Bekos, M.A., Kaufmann, M., Symvonis, A., Wolff, A.: Boundary labeling: models and efficient algorithms for rectangular maps. Comput. Geom., Theory Appl. **36**, 215–236 (2007)
8. Benkert, M., Haverkort, H., Kroll, M., Nöllenburg, M.: Algorithms for multi-criteria one-sided boundary labeling. In: Proc. 15th Int. Symposium on Graph Drawing (GD'07). Lecture Notes in Computer Science, vol. 4875, pp. 243–254. Springer, Berlin (2007)
9. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry: Algorithms and Applications. Springer, Berlin (2000)
10. Formann, M., Wagner, F.: A packing problem with applications to lettering of maps. In: Proc. 7th Annu. ACM Sympos. Comput. Geom., pp. 281–288 (1991)
11. Garey, M., Tarjan, R., Wilfong, G.: One-processor scheduling with symmetric earliness and tardiness penalties. Math. Oper. Res. **13**, 330–348 (1988)
12. Kao, H.-J., Lin, C.-C., Yen, H.-C.: Many-to-one boundary labeling. In: Proc. Asia Pacific Symposium on Information Visualisation (APVIS2007), pp. 65–72. IEEE, New York (2007)
13. Kaufmann, M., Wagner, D. (eds.): Drawing Graphs: Methods and Models. Lecture Notes in Computer Science, vol. 2025. Springer, Berlin (2001)
14. Kuhn, H.W.: The Hungarian method for the assignment problem. Nav. Res. Logist. Q. **2**, 83–97 (1955)
15. Vaidya, P.M.: Geometry helps in matching. SIAM J. Comput. **18**, 1201–1225 (1989)
16. van Kreveld, M., Strijk, T., Wolff, A.: Point labeling with sliding labels. Comput. Geom., Theory Appl. **13**, 21–47 (1999)
17. Wolff, A., Strijk, T.: The map-labeling bibliography, maintained since 1996. Online: http://i11www.iti.uni-karlsruhe.de/~awolff/map-labeling/bibliography/