Karlsruhe Institute of Technology

Labeling Maps with Free Spaces

Bachelor Thesis of

Joachim Priesner

At the Department of Informatics
Institute for Theoretical Informatics

Advisors: Dipl.-Inform. Benjamin Niedermann
Dr. Martin Nöllenburg

Reviewers: Prof. Dr. Dorothea Wagner
Prof. Dr. Peter Sanders

1st June 2013 – 30th September 2013

Abstract

Map labeling is the problem of visualizing a given set of points in the two-dimensional plane, called *sites*, each of which has an associated name, in such a way that the viewer can easily identify the name of a site. The sites themselves are fixed, but the position of the labels may vary. Its usages range from labeling actual maps to labeling diagrams, photos, and technical drawings.

Often, labels are not placed directly next to the sites but connected via a line called a *leader*. The labels are then placed somewhere else on the map, preferably grouped together in a *labeling area*. Common criteria include the absence of leader crossings and the minimization of the leader length.

In the variants of the problem studied until now, the labels are mainly placed on the outer sides of the rectangle containing the sites. We present an algorithm for labeling sites on one side of the instance, where the labeling area is bounded by a rectilinear polygon. This allows the labels to better utilize the free space of the instance. The labels are connected to the sites via leaders consisting of two horizontal and a vertical line segment.

We also study the problem of labeling sites outside of a rectangle to two adjacent sides of this rectangle. We study two variants: the number of possible label positions being greater than and this number being equal to the number of sites, each time with and without minimization of the leader length.

Lastly, we study boundary labeling on free spaces inside a map, for the case of rectangular labeling areas situated at an arbitrary place on the map. We study the problem for one such labeling area and give a short outlook on two and more labeling areas.

Deutsche Zusammenfassung

Das Problem der Kartenbeschriftung besteht darin, eine gegebene Menge von Orten – Punkten in der zweidimensionalen Ebene, denen jeweils ein Name zugeordnet ist – so zu visualisieren, dass der Betrachter den Namen eines Ortes schnell identifizieren kann. Dabei ist die Position der Orte fest, die der Beschriftungen frei wählbar. Die Anwendungen umfassen sowohl die Beschriftung von Landkarten im eigentlichen Sinne als auch die Beschriftung von Diagrammen, Bildern und technischen Zeichnungen.

Oft werden die Beschriftungen nicht direkt neben dem zugehörigen Ort platziert, sondern durch eine sogenannte Führungslinie mit diesem verbunden. Die Beschriftungen selbst werden dann in einer anderen Region der Karte untergebracht, bevorzugt gruppiert in einem Beschriftungsbereich. Gebräuchliche Qualitätskriterien für eine Lösung sind Kreuzungsfreiheit und minimale Länge der Führungslinien.

In den bisher betrachteten Problemvarianten werden die Beschriftungen meist an den Außenseiten des Rechtecks angebracht, das die Orte enthält. Wir stellen einen Algorithmus vor, der die Beschriftungen auf einer Seite der Instanz platziert, wobei der Beschriftungsbereich durch ein rechtwinkliges Polygon begrenzt ist. Auf diese Weise können die Beschriftung den zur Verfügung stehenden Platz der Instanz besser ausnutzen. Die Beschriftungen werden mit den Orten durch Führungslinien verbunden, die aus zwei horizontalen und einer vertikalen Strecke bestehen.

Wir betrachten außerdem das Problem, die Beschriftungen auf zwei aneinanderliegenden Seiten eines Rechtecks zu platzieren, außerhalb dessen die Orte liegen. Wir betrachten sowohl die Variante mit mehr möglichen Beschriftungspositionen als Orten als auch mit gleich vielen Beschriftungspositionen wie Orten, jeweils mit und ohne Längenminimierung.

Zuletzt beschäftigen wir uns mit Randbeschriftungen auf freien Bereichen innerhalb einer Karte. Wir betrachten den Fall eines rechteckigen Bereiches, der beliebig auf der Karte platziert ist, und geben einen kurzen Ausblick auf die Problemstellung mit mehreren solchen Bereichen.

To my advisors Benjamin and Martin:

Many thanks for your careful supervision, great ideas, and constructive feedback!

Andreas Gemsa also gave some helpful hints on labeling with more than one free space.

I declare that I have developed and written the enclosed thesis by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 30th September 2013

. . . . . .

(**Joachim Priesner**)

Contents

1. Introduction

In this thesis, we will work on a particular type of *map labeling problem*. These are problems where each one of a fixed set of points in the plane has to be labeled with a text naming or describing that point.

The term map labeling stems from the art of cartography, as one would expect. The problem of labeling maps is almost as old as the concept of a map itself. While the careful eye of a designer can still produce the most aesthetically pleasing results, the process is also very time-consuming and expensive. If more than a few sites are involved, it also becomes very hard for humans to reach optimal solutions, e.g. maximizing the number of sites that are labeled [Imh75].

Although often called map labeling, the problem is applicable to all sorts of two-dimensional images to be labeled, such as diagrams, photos and other images – in short, anywhere text is utilized to describe features in the two-dimensional plane.

A need for algorithmic solutions to these kinds of problems has therefore arisen with greater problem size on the one hand: The origin of algorithmic map labeling dates back to 1990, when Rudi Krämer, working for the city of Munich, wanted to label a map with groundwater level checkpoints and asked his former professor Kurt Mehlhorn for help (cited after [Kau09]).

On the other hand, algorithmic solutions have also become necessary because an increasing number of imagery is computer generated: Online mapping services like Google Maps and the OpenStreetMap project generate their map images from an ever-changing source of vector elements. In map applications on mobile devices, only vector data may be downloaded to save bandwidth which is then rendered on the device. Labeling therefore has to be done almost in real time with a quality one would expect from a traditional cartographer-generated map.

Thirdly, automatic map labeling enables users that do not have design experience to create fast and aesthetically pleasing labelings, e.g. for presentation slides, as in Figure 1.1.

With this wide range of applications, many general problems in this area have already been solved (or proven to be $\mathcal{NP}$-complete). For example, the decision variant of the internal map labeling problem, where a corner of the label is positioned at the site to be labeled and the number of labeled sites has to be maximized, is $\mathcal{NP}$-complete and no polynomial approximation with a factor greater than $\frac{1}{2}$ exists [FW91]. A detailed bibliography is available at [WS09], an overview of external map labeling problems is given in [Kau09].

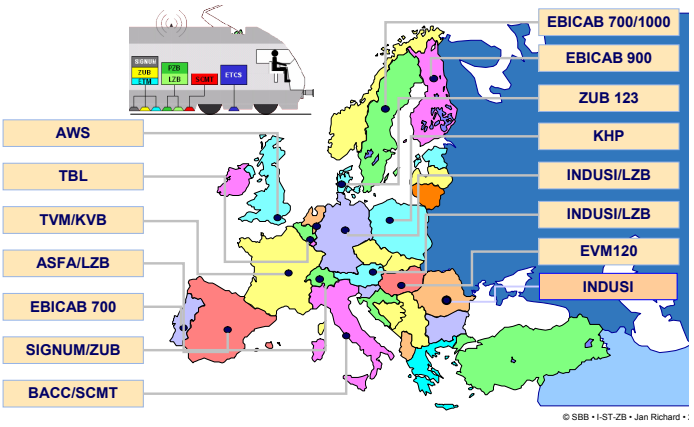


Figure 1.1: An example of external map labeling with *opo* and *popo* leaders (probably created manually): Overview of train protection systems used in Europe, from a presentation by Swiss Federal Railways.[1]

Figure 1.2: Example of internal map labeling: Map of German capital cities.[2]

We can generally divide the problems into *internal labeling* problems, where the labels touch the sites directly, for example with a corner or a side of the rectangular label, and *external labeling* or *boundary labeling* problems, where the labels are connected to the site with a curve called a *leader*. Usually the shape of the leaders is restricted in the problem definition. Common shapes include *po* and *opo* leaders, consisting of two and three axis-parallel line segments respectively.

As a third category, problems that allow both types of labelings, so-called *mixed labeling* problems, have also been researched. An overview is given in [BKPS11].

To improve legibility in external labelings, it is often required that the leaders do not cross at all or that the number of leader crossings is minimized. In mixed labeling problems, the additional problem of avoiding crossings between a leader and an internal label has to be considered. Other quality criteria for external labelings include the total length of the leaders, the maximum length of a leader, and the total or maximum number of bends of the leaders.

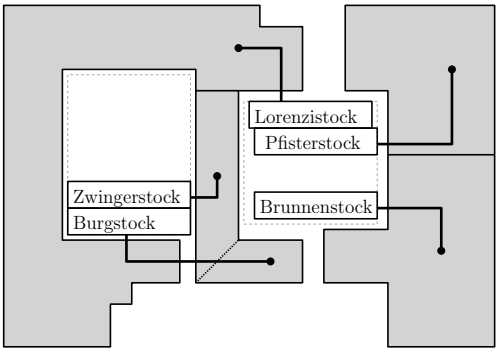
For some applications of map labeling, not only static maps are considered. Computing labelings that work well with operations like zooming in and out of a map [NPS10] or panning [Nie12] have practical applications e.g. in car navigation systems or online mapping services.

In this thesis, we will study external labeling problems with *opo* and *po* leaders for static maps. In the previous work on this problem area, the sites and the corresponding label positions have been strictly separated, for example by the sites being contained in a rectangle and labels being allowed only outside this rectangle [Kau09]. This allows easy solutions, but limits the quality of the drawings in some cases, as leaders might grow unnecessarily long.

[1] Image taken from: J. Richard, "ETCS – Eine Standortbestimmung," 2009. Available online: `http://www.gdi-adi.ch/uploads/media/091117_GdI_Vortrag_ETCS.pdf`, retrieved on 28 September 2013.

[2] Map source: "Germany location map," 2013, by Wikimedia users SokoWiki et al., from Wikimedia Commons, licensed under GFDL. Available online: `https://commons.wikimedia.org/wiki/File:Germany_location_map.svg`, retrieved on 28 September 2013.

Figure 1.3: Example of external labeling with free spaces inside the map: Schematic overview of the buildings surrounding the "Alter Hof" in Munich. The labeling areas are surrounded with dotted lines.



In contrast, we will focus on problem definitions that do not have this strict separation. This problem is mentioned in the open problems section of [Kau09] as "[allowing] the placement of leader-connected labels not only at the boundary but anywhere in the map where there is empty space". A free space is a region with arbitrary shape and at an arbitrary position on the map which does not contain any sites.

In Chapter 2 we solve a labeling problem with *opo* leaders where the labels may be placed anywhere inside a simple rectilinear polygon at the left or right side of the instance. We present a polynomial time algorithm for this problem. Related work includes [Jam12], where the one-sided labeling problem with convex boundaries is solved using *po* leaders, but labels are still restricted to be stacked on top of each other.

In Chapter 4, we solve this problem where the free space consists of one rectangle, and give an outlook on the variant with two or more rectangles. The groundwork for this is laid in Chapter 3, where we study a variant of the two-sided labeling problem and give an efficient algorithm that runs in $\mathcal{O}(n \log n)$ for leader length minimization in the simple case where there are as many sites as there are possible label positions.

1.1 Definitions

The general labeling problem consists of a set of **sites**, points in the two-dimensional plane, each of which has an associated text or description. The process of labeling consists of placing a label for some or all of the sites at a place where it can be identified as belonging to that site. A **label** is an axis-parallel rectangle containing the text that describes the site.

To facilitate the solution, we will often assume that all labels have the same size. The **position of a label** is defined as the position of its top-left corner. All relative positions such as to the right, to the left, above and below are defined in respect to that position.

There are multiple methods of associating a label to its site. The main difference is between internal and external labelings. An **internal label** is a label touching the site directly, for example with a corner or a side of the rectangle. An **external label** can be placed anywhere and is connected to the site with a **leader**, a curve with the site as one endpoint. We denote the leader between site s and label ℓ by $\lambda(s, \ell)$.

Accordingly, an **internal labeling problem** will allow only internal labels, while an **external labeling problem** will only allow external labels. A **mixed labeling problem** allows both types. In this thesis, we only consider external labeling problems.

Leaders are categorized according to their shape. Although in theory arbitrary curves are possible, leaders are mostly constrained to polygonal chains. These are often characterized by their relative position of their segments to the side of the label the leader is connected to: p means a parallel segment, o means an orthogonal segment. An *opo* leader is therefore a leader consisting of an orthogonal segment (starting at the site), then a parallel segment, and another orthogonal segment connected to the label. Other types include straight-line leaders (s) and diagonal leader segments (d).

In external labeling problems, labels are grouped at a convenient place, e.g. at the border of the map, in an area called the labeling area: A **labeling area** L of an instance of an external labeling problem is a subset of $\mathbb{R}^2$ so that in each solution all label must be completely contained in L.

Often, we will limit the set of points where a leader may connect to the label by assigning **ports** to the label. For example, we could demand that the leader connects to the label at the midpoint of a rectangle side. If a label has only one port, we often identify the port with the label and only consider ports which we then also denote by the letter ℓ.

To improve the legibility of the drawings, we do not allow leaders to be positioned arbitrarily close to other leaders or labels, instead we require a spacing around the leaders. Given a labeling problem $\mathcal{P}$ with fixed label height h, the **leader spacing** parameter of $\mathcal{P}$ is an amount of space ε ($0 < \varepsilon \leq \frac{1}{2}h$) at both sides of each leader which must be kept free of other leaders or labels.

Leader spacing also has an effect on what we consider as a leader touching a label or another leader. Two labels **touch each other** if their intersection is a point or a line segment. In a labeling instance with leader spacing ε, a leader **touches** another leader or a label if the intersection of the strip of width 2ε around the leader and the other leader or the label is a line segment. A leader **touches** the outline of the labeling area if it intersects the outline (in this case, there is no ε spacing between leader and outline).

2. External *opo* labelings with rectilinear labeling area

In this chapter we will use a certain type of labeling area to accommodate our labels:

Definition 2.0.1. *A **rectilinear labeling area** (L, R) is a rectilinear simple polygon consisting of two y-monotone polygonal chains L and R at the left and right.*

We also restrict ourselves to leaders that go to the right. A left-sided problem can be solved in the same way by mirroring it horizontally.

Definition 2.0.2. *A **right-sided** labeling problem is a problem where the label for a site may only placed to the right of that site. A **left-sided** labeling problem is defined accordingly.*

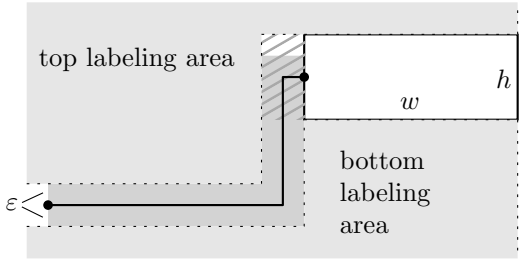
2.1 Problem definition

We consider a right-sided external labeling problem with rectilinear labeling area A, uniform label size $w \times h$ and leader spacing ε.

Each of the n sites is connected to its corresponding label by a leader of type *opo*. The leaders are crossing-free. The last segment of the *opo* leader has fixed length ε so that the leader connects directly to the label, as shown in Figure 2.1. Furthermore, the two last segments of a leader must lie completely inside A.

The strip of height h and width 2ε to the left of the label (shaded in Fig. 2.1) must be kept free of other labels. This improves legibility as a label now cannot horizontally touch

Figure 2.1: A label together with an *opo* leader and the top and bottom labeling areas defined by this label



another label directly. It also slightly simplifies our algorithm by eliminating some corner cases.

Problem 2.1.1 (RECTILINEAR AREA *opo* LABELING decision problem)**.** *Given a rectilinear labeling area A, a label size $w \times h$, leader spacing ε and a set of n sites to the left of the labeling area with minimum vertical distance ε, is it possible to label all sites in the fashion described above?*

The minimum vertical distance of ε between the sites ensures leader spacing.

2.2 A dynamic programming algorithm

Our algorithm for RECTILINEAR AREA *opo* LABELING will be based on partitioning an instance via so-called top and bottom labels. To simplify definitions and proofs, we first introduce the concept of dummy labels.

Definition 2.2.1 (Dummy sites and labels)**.** *Given a* RECTILINEAR AREA *opo* LABELING *instance with labeling area A, let (x_t, y_t) be the top-left site of A and (x_b, y_b) the bottom-left site of A.*

*Then the **dummy labels** $\ell_{\mathcal{T}}$ and $\ell_{\mathcal{B}}$ are two fixed labels with straight-line leaders that are connected to the **dummy sites** $(x_t, y_t + \varepsilon)$ and $(x_b, y_b - \varepsilon)$, respectively. The labels' horizontal position is 2ε to the right of the rightmost site of A.*

Note that these dummy labels are not legally placed with regard to the constraints for RECTILINEAR AREA *opo* LABELING. They are not included in the solution and just serve as placeholders.

Our algorithm will place labels in the areas to the left of other labels and either above or below them and their leaders. These areas are defined as follows:

Definition 2.2.2 (Top and bottom labeling areas)**.** *Given a leader $\lambda := \lambda(s, \ell)$, the **top (bottom) labeling area** of λ is the area above (below) ℓ and the ε space around the leader λ, bounded at the right by the right-hand side of ℓ, and unbounded to the left and to the top (bottom).*

Now we can define the area in which labels in a sub-instance defined by two other labels can be legally placed:

Definition 2.2.3. *The **labeling area** $\mathcal{L}(A, s_T, \ell_T, s_B, \ell_B)$ defined by a rectilinear labeling area A, a* top site s_T, top label ℓ_T, bottom site s_B *and* bottom label ℓ_B *is defined as the intersection of A, the bottom labeling area of $\lambda(s_T, \ell_T)$ and the top labeling area of $\lambda(s_B, \ell_B)$.*

This is the area in which a label for a site s that lies vertically between s_T and s_B can be placed (without taking into account the label size and the leader) so that it lies to the left of both ℓ_T and ℓ_B and vertically between ℓ_T and ℓ_B and their leaders. One example is given in Figure 2.2.

Note that the dummy labels do not influence a labeling area. In particular, the labeling area defined by A, $\ell_{\mathcal{T}}$ and $\ell_{\mathcal{B}}$ is A. This is exactly the purpose of the dummy labels: to define a labeling area for a label when no other label lies to the right and above/below it.

We now show that the number of positions a label can take can be restricted without affecting the solvability of the problem. We first introduce the concept of reference heights:

Figure 2.2: Visualization of $\mathcal{L}(A, \ell_T, p_T, \ell_B, p_B)$. The dummy labels $\ell_\mathcal{T}$ and $\ell_\mathcal{B}$ are also shown.

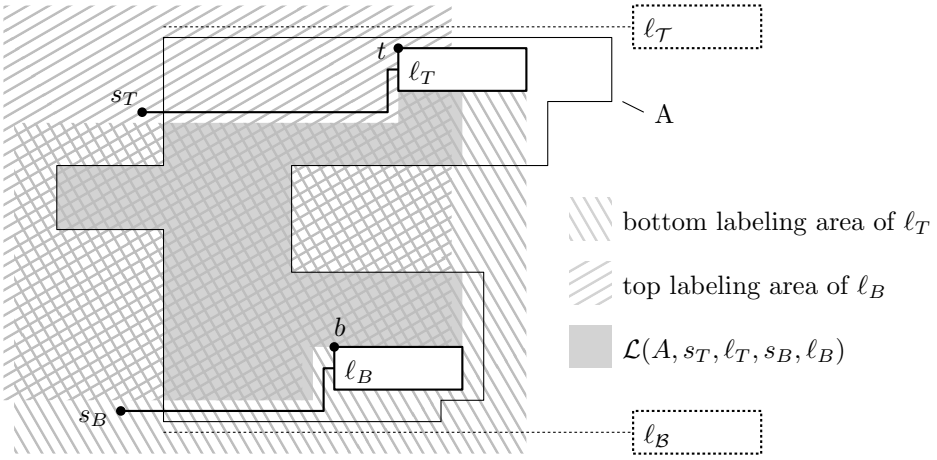
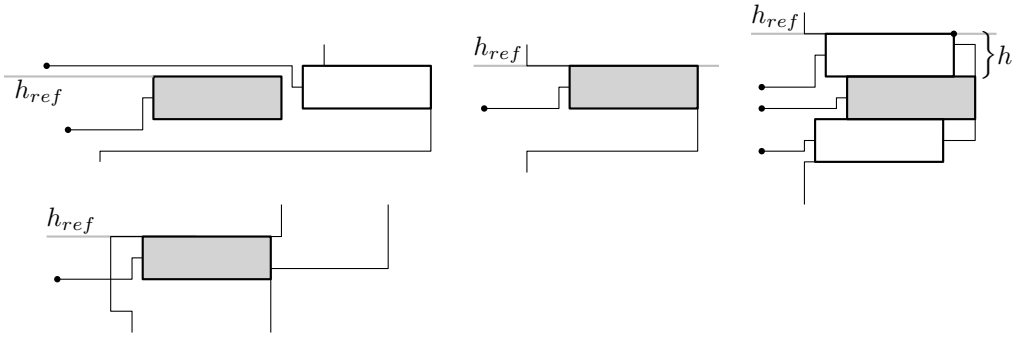


Figure 2.3: Some possible positions of a label (marked in gray) in a position-restricted solution to RECTILINEAR AREA *opo* LABELING.



Definition 2.2.4. *The **reference heights** of a* RECTILINEAR AREA *opo* LABELING *instance with labeling area A and sites $(x_1, y_1), \ldots, (x_n, y_n)$ are the y coordinates of A's horizontal segments and the numbers $y_1 - \varepsilon, \ldots, y_n - \varepsilon$.*

Those reference heights will later define the top position of "label stacks", sets of labels that touch each other vertically. The outline y coordinates correspond to labels touching the outline, the numbers $y_i - \varepsilon$ correspond to labels touching another label's leader.

Now we can define a solution in which each label has a finite number of possible positions:

Definition 2.2.5. *A solution to a* RECTILINEAR AREA *opo* LABELING *instance I is called **position-restricted** if for each label ℓ_i connected to site s_i*

1. *there exists a reference height h_{ref} of I so that the vertical distance between the line $y = h_{ref}$ and ℓ_i's position is a multiple of h.*
2. *there exists a dynamic labeling area $A_i = (L_i, R_i)$ defined by A, ℓ_i, top label ℓ_T and bottom label ℓ_B, both to the right of ℓ_i, and the sites s_T, s_B and s_i connected to them such that*
   a) *the right-hand side of ℓ_i or the leader $\lambda(s_i, \ell_i)$ touches R_i.*
   b) *for each label ℓ_j to the left of ℓ_i connected to a site s_j above (below) s_i, the top and bottom labels $\ell_{T'}$ and $\ell_{B'}$ of the dynamic labeling area A_j are connected to sites $s_{T'}$, $s_{B'}$ that lie vertically between s_i and s_T (s_B), inclusive.*

Some valid positions of a label in a position-restricted solution are shown in Figure 2.3.

Each label ℓ_i satisfying condition 2*b*) splits the instance in two independent sub-instances: one "top instance" containing all labels of sites between s_T and s_i, and one "bottom instance" containing all labels of sites between s_i and s_B. Both instances contain only labels to the left of ℓ_i.

Theorem 2.2.6. *For any solvable instance of* Rectilinear area *opo* labeling *there exists a position-restricted solution.*

Proof. Given a valid solution, we transform it into a position-restricted one as follows:

Algorithm 1: Transforming a valid solution into a position-restricted one

```
1 Order the labels from top to bottom by the order of their respective sites
2 repeat
3 |   Move each of the labels (starting at the top) upwards as far as legally possible
  |   (i. e. until it touches another label, its leader or the outline of the labeling area).
4 |   Move each of the labels (starting at the top) to the right as far as legally possible.
5 until none of the labels has changed its position in the current iteration
```

The algorithm terminates because the labeling area is bounded to the top and to the right, therefore each of the finitely many labels has a finite number of positions after moving (restricted only by the outline and other labels, of which one must touch the outline). Because moving is done only to the top and right, no previous position of a label is repeated after moving it.

As only legal positions of labels are considered when moving them, the result is again a legal labeling.

After the algorithm has terminated, each label is in one of the desired positions:

- It cannot be moved upwards. This means that its top edge touches *a*) a horizontal edge of the labeling area (in this case h_{ref} is the vertical position of the edge), or *b*) another label's leader (h_{ref} is the y coordinate of the other label's site minus ε), or *c*) another stack of labels of which the uppermost is in position *a*) or *b*). The first condition is therefore satisfied.
- It cannot be moved to the right. This means that the label itself or its leader touches either another leader/label or the outline of the labeling area.

We can now define the label numbering and the labeling area A_i for each label. We number the labels from ℓ_1 to ℓ_n from right to left (labels with the same horizontal position can be numbered arbitrarily) and number the labels' sites accordingly (s_1 to s_n). The top label ℓ_t for A_i is the label for the site above s_i that is closest to s_i and whose label lies to the right of ℓ_i. Similarly, the bottom label for A_i is the label for the site below s_i that is closest to s_i and whose label lies to the right of ℓ_i. The labels ℓ_t and ℓ_b are always well-defined since at least the dummy labels always satisfy the condition.

To prove that label ℓ_i or its leader touches $\mathcal{R}_i$, note that if the label or leader touches the right outline of A, the condition is fulfilled, since both the top and the bottom label of A_i lie to the right of ℓ_i and therefore the segment of R which ℓ_i touches is also part of A_i.

Otherwise, ℓ_i or its leader touches another label ℓ_j whose site s_j is above or below s_i. All labels of sites between s_i and s_j cannot be placed to the right of ℓ_i because there is no space between ℓ_i or its leader and the leader of ℓ_j, and leader crossings are not allowed. Therefore ℓ_j must be the top or bottom label of A_i, and any label or leader touching ℓ_j also touches $\mathcal{R}_i$.

We are left with condition 2*b*), of which we prove the first case (the second case is similar). Let label ℓ_j be as described. The site s'_t of A_i's top label ℓ'_t must lie above s_i because it already lies above s_j. The bottom label ℓ'_b of A_i cannot belong to a site s'_k below s_i because ℓ_i lies to the right of ℓ_j and s_i is closer to s_j than s_k. □

Theorem 2.2.6 leads to a dynamic programming algorithm, shown in Algorithm 2 and using the function RectilinearLabelingSolution as the main part of the algorithm.

In the proof of the theorem we showed that each valid solution can be divided into two smaller instances by one label which is one of the rightmost labels in the instance. We therefore need to enumerate all possible candidates for this rightmost label and all possible vertical positions for each candidate. For each possible vertical position, we place the label as far to the right as possible. The horizontal position of the top and bottom label also plays a role in determining whether there is enough space to accommodate all labels.

Function RectilinearLabelingSolution(I, $\mathcal{T}$, s, (s_{top}, ℓ_{top}), (s_{bottom}, ℓ_{bottom}))

Data: I is a RECTILINEAR AREA *opo* LABELING instance with a labeling area A, site set $\{s_1, \ldots, s_n\}$, dummy labels $\ell_\mathcal{T}$ and $\ell_\mathcal{B}$ and size parameters w, h and ε. $\mathcal{T}$ is the dynamic programming table. s is a site. s_{top} and s_{bottom} are two sites connected to the labels ℓ_{top} and ℓ_{bottom} positioned at p_{top} and p_{bottom}.

Result: Whether a valid solution exists for the given sub-instance.

```
1  if T[s, s_top, p_top, s_bottom, p_bottom] = ⊥        // if table entry not yet computed
2  then
3      V ← valid label positions for s in L(A, s, s_top, ℓ_top, s_bottom, ℓ_bottom)
4      if V = ∅ then
5          T[s, s_top, p_top, s_bottom, p_bottom] ← false
6      else
7          U ← sites between s_top and s
8          L ← sites between s and s_bottom
9          if U = ∅ ∧ L = ∅ then
10             T[s, s_top, p_top, s_bottom, p_bottom] ← true
11         else
12             for p ← V do
13                 ℓ ← a label at position p connected to s
14                 T[s, s_top, p_top, s_bottom, p_bottom] ←
                   ∨_{s_j ∈ U} RectilinearLabelingSolution(I, T, s_j, (s_top, ℓ_top), (s, ℓ)) ∧
                   ∨_{s_j ∈ L} RectilinearLabelingSolution(I, T, s_j, (s, ℓ), (s_bottom, ℓ_bottom))
15 return T[s, s_top, p_top, s_bottom, p_bottom]
```

The sites s_{top} and s_{bottom} passed to RectilinearLabelingSolution are the sites whose labels ℓ_{top} and ℓ_{bottom} (with positions p_{top} and p_{bottom}) restrict the current instance. The site s is the site whose label shall be placed as the rightmost label in this situation.

Algorithm 2: Dynamic programming solution for Rectilinear Labeling

Data: A RECTILINEAR AREA *opo* LABELING instance I with labeling area A, site set S, dummy labels $\ell_\mathcal{T}$ and $\ell_\mathcal{B}$ connected to the sites $s_\mathcal{T}$ and $s_\mathcal{B}$ and size parameters w, h and ε.

Result: Whether a valid solution exists for the instance I.

```
1  initialize T with ⊥
2  return ∨_{s ∈ S} RectilinearLabelingSolution(I, T, s, (s_T, ℓ_T), (s_B, ℓ_B))
```

The following theorem is the main result of this chapter, stating that the problem we considered can be solved in polynomial time. This is however mostly a theoretical result, as the exponents of the polynomial running time are still quite large.

Theorem 2.2.7. *Algorithm 2 solves the* Rectilinear area *opo* labeling *decision problem in* $\mathcal{O}(n^8 \cdot m^3 \cdot (n+m)^3)$ *time.*

Proof. The correctness of the algorithm follows immediately from the proof of theorem 2.2.6. For each solvable instance of Rectilinear area *opo* labeling we proved that there exists a position-restricted solution S. As the algorithm tries all possible positions for each label, it will eventually find the label's position in S. Each rightmost label also divides the problem into two independent sub-instances. This allows us to calculate the solution recursively in line 14 of the function RectilinearLabelingSolution.

For the running time, let us first remember that n is the number of sites to be labeled. Let m denote the complexity of the labeling area, that is, the number of line segments defining the polygon.

The defining factors for the running time are *a*) the size of the dynamic programming table, *b*) finding the valid label positions in line 3 of function RectilinearLabelingSolution and *c*) iterating over them in line 12. All three depend on n_v, the number of valid vertical positions for a label.

In the function RectilinearLabelingSolution, the number $|V|$ of possible vertical positions for the current label depends on the current parameters, but the worst-case number n_v is equal for each label. There are $n+m$ reference heights and n possible positions of a label relative to one such height, in total $n_v = (n+m) \cdot n$ vertical positions.

For *a*) the size of the table we also have to take into account the number n_h of horizontal positions a label can take. As a label cannot be moved to the right in a position-restricted solution, it either directly touches the right outline or it touches a sequence of labels touching each other, of which the rightmost touches the right outline.

As we can see in table 2.1, a label being touched by another label causes the second label to be placed $w + 2\varepsilon$ units to the left of the first label, while the label being touched by a leader causes a ε shift (or no shift at all). Therefore, for each of the at most m segments of the right outline, the possible positions of any label are $2 \cdot (a \cdot \varepsilon + b \cdot (w + 2\varepsilon))$ to the left of the segment, where $a, b < n$, a is the number of leaders touching another leader in the sequence and b is the number of labels touching another leader. The factor 2 accounts for the fact that the label touching the outline may do so either with the label or with the leader.

Thus we have $n_h = m \cdot 2n^2$ and the maximum size of the dynamic programming table is $n \cdot (n \cdot n_v \cdot n_h + 1) \cdot (n \cdot n_v \cdot n_h + 1)$. Note that we need to add 1 because the top and bottom label can also be a dummy label.

For *b*) we have to calculate how long it takes to find the horizontal position for a label corresponding to a given vertical position. Figure 2.4 illustrates this problem. As we can see, only the outline between the top and the bottom label plays a role in finding the right position. The top and bottom label each add three segments to the left or right outline (depending on whether the leader bends upwards or downwards).

To find the possible horizontal positions, it suffices to find the x coordinate x_{l1} of the rightmost vertical segment of the left outline and x_{r1} of the leftmost vertical segment of the right outline that lies within the interval defined by h_{top} and h_{bottom}, the top and bottom vertical positions of the label. These segments restrict the horizontal position of

Table 2.1: The different ways a label can touch another label or the outline.

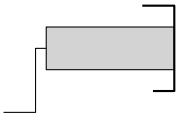
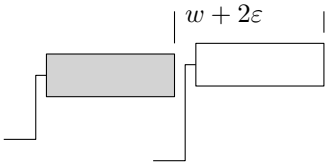
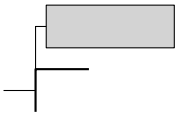
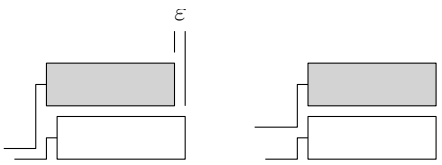
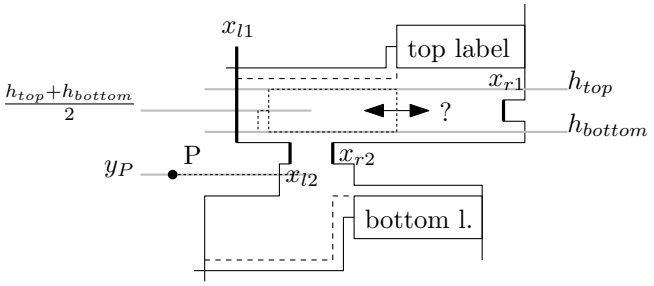
	... the right outline	... another label or leader
label touching ...		
leader touching ...		

Figure 2.4: Finding the horizontal position for a label with given vertical position. The two upper thick lines define x_{l1} and x_{r1}, while the two lower thick lines define x_{l2} and x_{r2}.



the label. The leader's position is restricted by the segments which lie in the interval defined by $\frac{h_{top}+h_{bottom}}{2}$ (the y coordinate of the label's midpoint) and y_s (the y coordinate of s). The x coordinates of these segments are called x_{l2} and x_{r2}. To find these coordinates, it suffices to iterate over the left and right outline, a $\mathcal{O}(m)$ operation.

The final horizontal position of the label is $\min(x_{r2} + \varepsilon, x_{r1} - w)$. If it is less than $x_{l1} + \varepsilon$ or $x_{l2} + \varepsilon$, there is no legal horizontal position for the label at this vertical position.

Thus the running time for *b*) and *c*): For each entry that has to be computed, it takes $\mathcal{O}(n_v \cdot m)$ steps to find the valid positions of the label, and $\mathcal{O}(n_v)$ steps to iterate over them, which gives a $\mathcal{O}(n_v \cdot m)$ cost of calculating each table entry.

This gives us a total running time of $\mathcal{O}(n^3 \cdot n_v^3 \cdot n_h^2 \cdot m) = \mathcal{O}(n^8 \cdot m^3 \cdot (n+m)^3)$ for the algorithm. □

Note that the solution itself is not returned by the algorithm. However, this can easily be accomplished by storing the position of the placed label in the dynamic programming table instead of just *true*. We can then recreate the placement of the labels via an additional pass through the table that takes $\mathcal{O}(n)$ steps.

3. External *po* labelings on adjacent sides of a rectangle

3.1 Problem definition

In this chapter we work on the problem of external boundary labeling, where the boundary consists of two adjacent sides of a rectangle R. This has already been solved for the case that the sites lie inside R [KNR+13]. In our variant of the problem, the sites lie outside R.

On the one hand, this section lays the groundwork for the next chapter in which we will make extensive use of this problem. On the other hand, the problem might arise in some cases where space is limited, as in Figure 3.1

Problem 3.1.1 (Two-sided convex rectilinear boundary labeling (2CRB))**.** *Let R be a rectangle, $\ell_1, \ldots, \ell_m$ be m ports that lie on two adjacent sides r_1 and r_2 of R (but not on the corner) and H_i, $i \in \{1, 2\}$, be the open half-plane defined by r_i that does not intersect R. Furthermore, let $s_1, \ldots, s_n$ be n ($n \leq m$) sites that lie in $H_1 \cup H_2$.*

The problem is to connect the sites $s_1, \ldots, s_n$ to the ports using crossing-free po leaders.

In the variant of the problem we will be studying, no two sites and no site and a port may share an x or y coordinate. To achieve a leader spacing of ε, sites and ports must have a minimum vertical and horizontal distance of ε.

If we use coordinates in the following text, we will always assume a Cartesian coordinate system where the origin is the intersection point of r_1 and r_2, the positive x axis is r_1 and the positive y axis is r_2. We will use the words "horizontal", "vertical", "up" and "down" accordingly.

We partition the set of sites of an instance into the three disjoint sets P_1, P_2 and Q of sites that lie in $H_1 \setminus H_2$, $H_2 \setminus H_1$ and $H_1 \cap H_2$, respectively. Any site in P_i, $i \in \{1, 2\}$, can be connected only to ports on r_i with a *po* leader, while the sites in Q can be connected to all ports. Figure 3.2 illustrates this partition.

We also denote the ports that lie on r_i by ports(r_i) for $i \in \{1, 2\}$.

In the following section, we will show how to solve the Two-sided convex rectilinear boundary labeling problem by splitting it into two one-sided labeling problems. One-sided boundary labeling is a well-known problem for which efficient algorithms exist. It

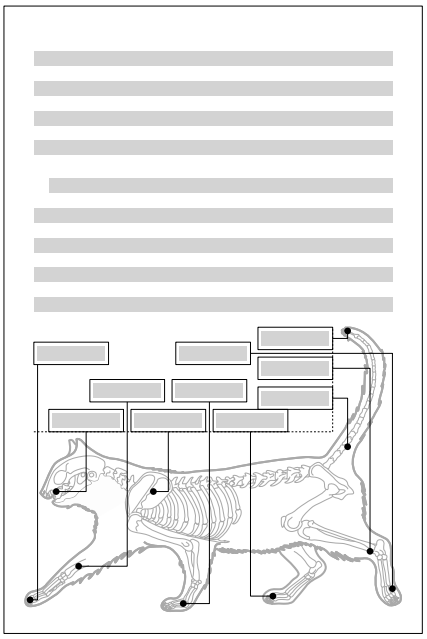


Figure 3.1: Example application of 2CRB labeling in a book layout.[1]

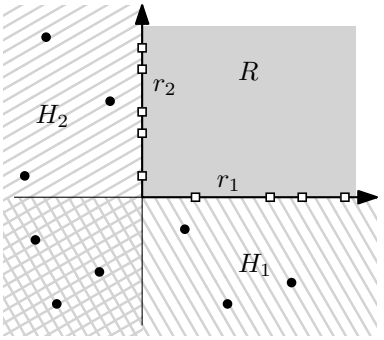


Figure 3.2: Illustration of a TWO-SIDED CONVEX RECTILINEAR BOUNDARY LABELING instance and how it is partitioned.

can be solved in $O(n^3)$ time for general badness functions, and in $\mathcal{O}(n \log n)$ time for leader length minimization [BHKN08].

When solving an instance in this way, we have to decide for each point in Q whether to connect it to a port on r_1 or r_2. This leads to a partition of Q into two disjoint sets Q_1 and Q_2 of sites that are to be connected to ports on r_1 and r_2, respectively.

We first prove that Q_1 and Q_2 can be separated geometrically and furthermore that the partition can be chosen to avoid crossings between leaders of sites in Q. This allows us to solve the basic problem. For the additional goal of leader length minimization, we show that the total leader length does not depend on the partition itself, but rather on the partition size.

3.2 Solution for the basic problem

We want to partition Q into two sets P_1 and P_2 of sites to be connected to ports on r_1 and r_2, respectively. We have some constraints for a partition that leads to a valid solution:

Definition 3.2.1. *Let I be an instance of* TWO-SIDED CONVEX RECTILINEAR BOUNDARY LABELING*, with the sites partitioned into P_1, P_2 and Q as described above. For $i \in \{1, 2\}$, let m_i be the number of ports that lie on r_i.*

*A **valid partition** of I is a partition (Q_1, Q_2) of Q so that $|P_i| + |Q_i| \leq m_i$ for $i \in \{1, 2\}$.*

*A **crossing-free partition** of I is a partition (Q_1, Q_2) of Q so that for $i \in \{1, 2\}$ the following condition holds: If all sites in Q_i are connected to ports on r_i, then for $j \in \{1, 2\}$, for any two sites $s_j \in Q_j$ and any two ports ℓ_j on r_j, the leaders $\lambda(s_1, \ell_1)$ and $\lambda(s_2, \ell_2)$ do not intersect.*

A valid partition ensures that there are enough ports to accomodate all sites. It is immediately obvious that a solution can only exist if a valid partition of Q exists.

[1] Drawing based on "Skeleton diagram of a cat.svg" by Przemek Maksim, licensed under CC-BY-SA, available online at `http://commons.wikimedia.org/wiki/File:Skeleton_diagram_of_a_cat.svg`

A crossing-free partition ensures that no crossings occur between leaders of sites in Q, however it does not state anything about the leaders of sites in P_1 and P_2. A crossing-free partition can also be described geometrically:

Lemma 3.2.2. *A partition of I is crossing-free if and only if the inequality $x_2 \leq x_1 \vee y_2 \geq y_1$ is satisfied for any two sites $(x_1, y_1) = s_1 \in Q_1$ and $(x_2, y_2) = s_2 \in Q_2$.*

Proof. Let $\lambda_1 = \lambda(s_1 = (x_1, y_1), \ell_1)$ and $\lambda_2 = \lambda(s_2 = (x_2, y_2), \ell_2)$ be two leaders, where $s_i \in Q_i$ and ℓ_i lies on r_i, for $i \in \{1, 2\}$.

As the orthogonal segment of λ_1 lies in $H_1 \setminus H_2$ and the orthogonal segment of λ_2 lies in $H_2 \setminus H_1$, a crossing can only occur between the two parallel segments. The parallel segment of λ_1 is the horizontal line between (x_1, y_1) and $(0, y_1)$. The parallel segment of λ_2 is the vertical line between (x_2, y_2) and $(x_2, 0)$.

If the lines intersect, the intersection point has the coordinates (x_2, y_1) and lies on both lines. The coordinates x_1, x_2, y_1, y_2 are all negative. Therefore the lines intersect if and only if $x_1 \leq x_2$ and $y_1 \geq y_2$ which is the negation of the above inequality. □

Fortunately, valid and crossing-free partitions can be found easily:

Lemma 3.2.3. *Let I be an instance of* Two-sided convex rectilinear boundary labeling. *Then a valid and crossing-free partition of I exists if and only if $|P_i| \leq m_i$ for $i \in \{1, 2\}$. This partition can be computed in $\mathcal{O}(n \log n)$ time.*

Proof. "⇒": If $|P_i| > m_i$ for $i = 1$ or $i = 2$, then obviously $|P_i| + |Q_i| \leq m_i$ cannot be satisfied for any partition of Q, thus no valid partition exists.

"⇐": Let $\Delta_i := m_i - |P_i|$, $i \in \{1, 2\}$, be the number of ports on r_i that are not "reserved" for sites in P_i. Q_i may not contain more than Δ_i sites in a valid solution. From our premise we have $\Delta_i \geq 0$. The minimum number of sites in Q_1 is then $|Q| - \Delta_2$ and the minimum number of sites in Q_2 is $|Q| - \Delta_1$.

A simple (but not the only) way to find a valid crossing-free partition is to sort the sites ascendingly by their x coordinate and place the first few sites into Q_2 and the rest into Q_1 such that the number of sites in Q_2 satisfies the inequalities $0 \leq |Q_2| \leq |Q|$ and $|Q| - \Delta_1 \leq |Q_2| \leq \Delta_2$. The second inequality can be satisfied because $|Q| - \Delta_1 \leq \Delta_2 \Leftrightarrow |Q| - m_1 + |P_1| \leq m_2 - |P_2| \Leftrightarrow |Q| + |P_1| + |P_2| \leq m_1 + m_2 \Leftrightarrow n \leq m$, which is satisfied by the problem definition.

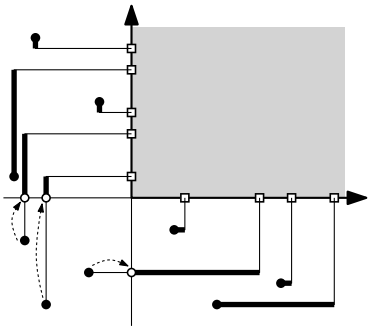
This takes $\mathcal{O}(n \log n)$ time because of the sorting. The partition is crossing-free because for any two sites $(x_1, y_1) = s_1 \in Q_1$ and $(x_2, y_2) = s_2 \in Q_2$ we have $x_2 \leq x_1$. It is also valid because $|Q_2| \leq m_2 - |P_2| = \Delta_2$ per construction, and $|P_1| + |Q_1| = |P_1| + (|Q| - |Q_2|) \leq |P_1| + \Delta_1 = m_1$. □

In the proof we have calculated the minimum and maximum cardinality of Q_1 in a valid partition. The cardinality of Q_2 is uniquely defined by the cardinality of Q_1, so it makes sense to call $|Q_1|$ the **size of the partition** (Q_1, Q_2) of Q.

Corollary 3.2.4. *Let I be an instance of the 2CRB problem. Then there are $m - n + 1$ possible differently-sized partitions of Q.*

Proof. As we saw in the proof of the previous lemma, Q_1 may contain between $|Q| - \Delta_2$ and Δ_1 sites. Thus there are $|\Delta_1 - (|Q| - \Delta_2) + 1| = |m_1 - |P_1| - |Q| + m_2 - |P_2| + 1| = |m - n + 1|$ possible partition sizes. □

Figure 3.3: A solution for the TWO-SIDED CONVEX RECTILINEAR BOUNDARY LABELING instance from Figure 3.2. The thick parts of the leaders are the only parts of the solution that influence the total leader length. The white circles are the points in the axis projected sub-instances, see Definition 3.3.6.



It follows from Lemma 3.2.3 that the 2CRB problem is not harder than a one-sided labeling problem:

Theorem 3.2.5. *Any instance of the 2CRB problem has a solution if and only if* $|P_i| \leq m_i$ *for* $i \in \{1, 2\}$*. The solution can be computed in* $\mathcal{O}(n \log n)$ *time.*

Proof. If $|P_i| > m_i$ for $i = 1$ or $i = 2$, we have no valid partitions and thus no solution. We now assume $|P_i| \leq m_i$ for $i \in \{1, 2\}$. By Lemma 3.2.3, a valid and crossing-free partition (Q_1, Q_2) of I exists.

Because the partition is valid, we have $|P_i| + |Q_i| \leq m_i$ for $i \in \{1, 2\}$. We remove arbitrary ports from r_1 and r_2 until $|P_i| + |Q_i| = m_i$ for $i \in \{1, 2\}$.

We now have two ONE-SIDED BOUNDARY LABELING instances $(P_1 \cup Q_1, \text{ports}(r_1))$ and $(P_2 \cup Q_2, \text{ports}(r_2))$ with the same number of ports and sites. As described in [BHKN08], a crossing-free solution for each of the sub-instances can be computed in $\mathcal{O}(n \log n)$ time.

Because the individual one-sided labelings are crossing-free, leader crossings can only occur between the leaders of a site in Q_1 and a site in Q_2. But our partition of I was crossing-free which specifically rules out such crossings, thus the whole solution is crossing-free. □

3.3 Leader length minimization

We are usually interested in not just any crossing-free labeling, but in one where the total leader length is minimal. We modify our problem as follows:

Problem 3.3.1 (LENGTH-MINIMAL 2CRB LABELING)**.** *The prerequisites are the same as in the* TWO-SIDED CONVEX RECTILINEAR BOUNDARY LABELING *problem. The problem is to connect the sites* $s_1, \ldots, s_n$ *to the ports using crossing-free po leaders such that the total leader length* $\text{length}(S)$ *of the solution* S *is minimized.*

Figure 3.3 gives an idea of what possibilities we have to minimize the total leader length. Intuitively, the thin parts of the leaders are fixed in every solution; their length only depends on the sites' positions. The only way to minimize the leader length is to minimize the length of the thick leader parts.

We begin by examining the one-sided boundary labeling problem a bit closer. In the following, we assume without loss of generality that the ports of a ONE-SIDED BOUNDARY LABELING instance lie on the vertical line $x = 0$. Thus the parallel leader segments in a

solution are vertical lines, and the orthogonal leader segments are horizontal lines with length $|x_s|$, where x_s is the x coordinate of the site in question.

We first split up the leader length of a solution by examining the parallel and orthogonal leader segments:

Definition 3.3.2. *Let I be an instance of the* One-sided boundary labeling *problem with po leaders. The* ***(total) parallel segment length*** $\text{length}_p(S)$ *of a solution S is the total length of the parallel segments of the leaders. The* ***(total) orthogonal segment length*** $\text{length}_o(S)$ *is the total length of the orthogonal segments of the leaders.*

Of course we have $\text{length}(S) = \text{length}_\text{p}(S) + \text{length}_\text{o}(S)$. The orthogonal segment length is special as it does not depend on the solution:

Lemma 3.3.3. *Let S and S' be two solutions of an* One-sided boundary labeling *instance with n sites $s_1, \ldots, s_n$. Then $\text{length}_o(S) = \text{length}_o(S')$, i. e. the total orthogonal segment length only depends on the instance.*

Proof. The orthogonal segment length of a leader $\lambda(s_i, \ell)$ is $|x_i|$, where x_i is the x coordinate of the site s_i. Thus we have $\text{length}_\text{o}(S) = \sum_{i=1}^{n} |x_i| = \text{length}_\text{o}(S')$. □

It therefore makes sense to define the total orthogonal segment length not only for a solution, but also for an instance of One-sided boundary labeling:

Definition 3.3.4. *Let I be an* One-sided boundary labeling *instance with n sites $s_1, \ldots, s_n$. The* ***(total) orthogonal segment length*** *of I is defined as $\text{length}_o(I) = \sum_{i=1}^{n} |x_i|$, where x_i is the x coordinate of the site s_i.*

By the proof of Lemma 3.3.3 we have $\text{length}_\text{o}(S) = \text{length}_\text{o}(I)$ for any solution S of I.

Having examined the orthogonal leader segments, we turn to the parallel ones. Their length does depend on the solution, but their minimum length depends only on the vertical position of the sites:

Lemma 3.3.5. *Let I and I' be two instances of* One-sided boundary labeling *that differ only in the set of sites. Let $\{s_1, \ldots, s_n\}$ be the set of sites in I, and let $\{s'_1, \ldots, s'_n\}$ be the set of sites in I'. Let S and S' be a solution to I and I', respectively, with minimum total leader length.*

If for $i \in \{1, \ldots, n\}$ the sites s_i and s'_i share the same y coordinate, then $\text{length}_p(S) = \text{length}_p(S')$.

Proof. We take the solution S and create from it a solution T for I' by adding the leader $\lambda(s'_i, \ell)$ for every leader $\lambda(s_i, \ell)$ in S. Since s_i and s'_i share their y coordinate, the parallel segment length stays the same, so $\text{length}_\text{p}(T) = \text{length}_\text{p}(S)$.

T may contain crossings. As described in [BHKN08], these crossings can be resolved without increasing the total leader length, resulting in a solution T' for I' with $\text{length}(T') \leq \text{length}(T)$. Because of Lemma 3.3.3, we have $\text{length}_\text{o}(T') = \text{length}_\text{o}(T)$ and therefore $\text{length}_\text{p}(T') \leq \text{length}_\text{p}(T) = \text{length}_\text{p}(S)$.

Because S' has minimum total leader length, we have $\text{length}(T') \geq \text{length}(S')$. Subtracting $\text{length}_\text{o}(I')$ from both sides yields $\text{length}_\text{p}(T') \geq \text{length}_\text{p}(S')$, therefore we have $\text{length}_\text{p}(S) \geq \text{length}_\text{p}(S')$. Reversing the roles of S and S' in the previous transformation yields $\text{length}_\text{p}(S') \geq \text{length}_\text{p}(S)$ and thus the desired equality. □

Our next goal is to abstract from the partitions of Q and instead only consider *partition sizes*. For this it is useful to split the part of the leaders that lies in $H_1 \cup H_2$ from the rest:

Definition 3.3.6. *Let I be a* Length-minimal 2CRB labeling *instance partitioned by a valid and crossing-free partition (Q_1, Q_2) of I. Then the* ***axis-projected sub-instances*** *I_1 and I_2 of the partitioned instance I are instances of the* One-sided boundary labeling *problem defined as follows:*

For $i \in \{1, 2\}$, the ports in I_i are the ports on r_i. The sites in I_i are the sites in $P_i \cup Q'_i$, where Q'_1 is the set $(0, y_i)$ for all sites $(x_i, y_i) \in Q_1$ and Q'_2 is the set $(x_i, 0)$ for all sites $(x_i, y_i) \in Q_2$.

This means that the points in Q_1 are projected onto the y axis, while the points in Q_2 are projected onto the x axis. Figure 3.3 visualizes this projection.

It is obvious that a solution to the axis-projected sub-instances yields a solution for I by connecting the points in Q to their respective projections with a straight line.

Our next step is to express the leader length of a solution of a 2CRB instance in terms of the leader lengths of the axis-projected sub-instances:

Lemma 3.3.7. *Let I be a* Length-minimal 2CRB labeling *instance partitioned by a valid and crossing-free partition (Q_1, Q_2) of I. Let I_1, I_2 the axis-projected sub-instances of the partitioned instance.*

Let S' be a solution with minimum total leader length for the partitioned instance I, and let S_1, S_2 be solutions with minimum total leader length for I_1 and I_2, respectively.

Then $\text{length}(S') = \text{length}_p(S_1) + \text{length}_p(S_2) + X$ *where X is independent from the partition (Q_1, Q_2) or the solution S'.*

Proof. We claim that

$$\begin{aligned} X &= \sum_{(x,y)\in P_1\cup Q_1} |y| + \sum_{(x,y)\in P_2\cup Q_2} |x| + \sum_{(x,y)\in Q_1} |x| + \sum_{(x,y)\in Q_2} |y| \\ &= \sum_{(x,y)\in P_1} |y| + \sum_{(x,y)\in P_2} |x| + \sum_{(x,y)\in Q} (|x| + |y|) \end{aligned}$$

which is indeed independent from the partition or the solution.

We consider the solution S for I induced by the solutions S_1 and S_2 and look at the leader lengths for sites $s = (x, y)$ in P_1, Q_1, P_2 and Q_2 separately. For $s \in Q_1$ and $s \in Q_2$, let s' be the site corresponding to s in the relevant axis-projected sub-instance of I.

- For $p \in P_1$, the orthogonal leader segment (a vertical line) has length $|y|$. The length of the parallel segment adds to $\text{length}_\text{p}(S_1)$ because s is a part of the sub-instance I_1.
- For $p \in P_2$, the orthogonal leader segment (a horizontal line) has length $|x|$. The length of the parallel segment adds to $\text{length}_\text{p}(S_2)$ because s is a part of the sub-instance I_2.
- For $p \in Q_1$, the orthogonal leader segment (a vertical line) has length $|y|$. The parallel segment can be split into a line between s and its projection s', which has length $|x|$, and the parallel segment of the leader of s' in the solution S_1, whose length adds to $\text{length}_\text{p}(S_1)$.

- For $p \in Q_2$, the orthogonal leader segment (a horizontal line) has length $|x|$. The parallel segment can be split into a line between s and its projection s', which has length $|y|$, and the parallel segment of the leader of s' in the solution S_2, whose length adds to $\text{length}_\text{p}(S_2)$.

Adding up the leader lengths yields the right-hand side of the above equation. To see that this is indeed equal to the total leader length of a minimum-length solution S', suppose for the sake of contradiction that we have $\text{length}(S) > \text{length}(S')$. But S' induces two solutions S'_1 and S'_2 for I_1 and I_2, and $\text{length}(S')$ can be split up in the same way as above. Thus the inequality becomes $\text{length}_\text{p}(S_1) + \text{length}_\text{p}(S_2) > \text{length}_\text{p}(S'_1) + \text{length}_\text{p}(S'_2)$, a contradiction because S_1 and S_2 were both minimum-length solutions of I_1 and I_2. □

We see that the total leader length depends only on the parallel leader segments in the solutions to the axis-projected sub-instances. These are the gray leader parts in Figure 3.3.

In our next step we assemble the previous lemmata to show that the minimum leader length depends only on the size of the partition of Q, not on the partition itself.

Lemma 3.3.8. *Let $T = (Q_1, Q_2)$ and $T' = (Q'_1, Q'_2)$ be two crossing-free partitions of a* Two-sided convex rectilinear boundary labeling *instance I with $|Q_1| = |Q'_1|$. Let S and S' be the minimum-length solutions of I partitioned by T and by T', respectively. Then* $\text{length}(S) = \text{length}(S')$.

Proof. Let S_1, S_2 and S'_1, S'_2 be minimum-length solutions of the axis-projected sub-instances of I partitioned by T and by T', respectively. It suffices to prove $\text{length}_\text{p}(S_i) = \text{length}_\text{p}(S'_i)$ for $i \in \{1, 2\}$. Then it follows from Lemma 3.3.7 that $\text{length}(S) = \text{length}(S')$.

We prove the equality $\text{length}_\text{p}(S_2) = \text{length}_\text{p}(S'_2)$, the other one is symmetric. S_2 and S'_2 are solutions for two One-sided boundary labeling instances that differ only in the sites projected from Q_2 and Q'_2, respectively. But we have $|Q_2| = |Q'_2|$ and these projected sites all have the same y coordinate 0, so the sites in both instances can be arranged in pairs of sites that share the same y coordinate. By Lemma 3.3.5, we have $\text{length}_\text{p}(S_2) = \text{length}_\text{p}(S'_2)$. □

This leads us to the main theorem of this section:

Theorem 3.3.9. *Let I be a* Length-minimal 2CRB labeling *instance. I can be solved in $\mathcal{O}((m - n + 1) \cdot \alpha)$ time, where α is the time to solve a partitioned instance.*

Proof. The number of differently-sized partitions of Q is $m-n+1$, as seen in Corollary 3.2.4. For each partition size s take an arbitrary partition of Q of size s and solve the partitioned instance in $\mathcal{O}(\alpha)$ time, then take the length-minimal one among all such partitions.

This solution has minimum leader length among all solutions: Let S be a solution with minimum leader length. S induces a partition (Q_1, Q_2) of Q with a certain partition size. The algorithm solves a (possibly different) partition of the same size, yielding a solution S'. According to Lemma 3.3.8, we have $\text{length}(S') = \text{length}(S)$, meaning the algorithm finds a solution of the same (and therefore minimum) length. □

Now we can begin to apply this to concrete problems. Our first result is that for the same number of sites and ports, the two-sided problem is not harder than the one-sided one:

Corollary 3.3.10. *Any* Length-minimal 2CRB labeling *instance with $m = n$ can be solved in $\mathcal{O}(n \log n)$ time.*

Proof. The number of differently-sized partitions is $m - n + 1 = 1$. The partitioned instances have the same number of sites and ports and thus can be solved in $\mathcal{O}(n \log n)$ time each. □

We prove one special case that may occur in real-world applications: Because the ports belong to labels, it is possible that there exists a label at the corner of the labeling area which has a port on r_1 and one on r_2. We cannot connect both ports, but instead have to decide from which side to connect to this label.

Corollary 3.3.11 (Length-minimal 2CRB labeling with connected pair of ports)**.** *Let I be a* Length-minimal 2CRB labeling *instance with $m = n+1$ and let (ℓ_1, ℓ_2) be a pair of ports with ℓ_1 on r_1 and ℓ_2 on r_2.*

Then a length-minimal solution in which only one of ℓ_1 and ℓ_2 is connected to a site can be found in $\mathcal{O}(n \log n)$ time.

Proof. The two instances I_1 and I_2 in which ℓ_1 and ℓ_2, respectively, is removed, are instances with $m = n$ and can be solved in $\mathcal{O}(n \log n)$ time. A length-minimal solution for I can be obtained by solving I_1 and I_2 and taking the shorter one of both solutions. □

Up until now, we have only considered the case where $m = n$. For $m > n$, the $\mathcal{O}(n \log n)$ solution described in [BHKN08] does not work. Alternatively, but with a much higher running time, we can solve such an instance using minimum-weight matching:

Theorem 3.3.12. *A* One-sided boundary labeling *instance with m ports and n sites ($m \geq n$) can be solved in $\mu(m, n) + \mathcal{O}(n^2)$ time, where $\mu(m, n)$ is the time needed to solve a minimum-weight matching problem on the complete bipartite graph $K_{m,n}$.*

Proof. Let P be the set of sites and L be the set of ports. We formulate the labeling problem as a weighted matching problem. We construct the complete bipartite graph $K_{m,n}$ from the node sets P and L, where the weight of each edge $\{p, \ell\}$ is the unique length of the po leader $\lambda(p, \ell)$.

We retrieve a minimum-weight matching of cardinality n in $\mu(m, n)$ time and create a leader $\lambda(p, \ell)$ for every edge (p, ℓ) in the matching. This solution may of course contain crossings, but these can be removed using $\mathcal{O}(n^2)$ crossing resolution steps that do not increase the total leader length, as described in [BHKN08].

This solution has minimal leader length because each solution to the labeling problem corresponds to a matching of cardinality n in $K_{m,n}$ whose weight is the total leader length of the solution. A solution of shorter leader length would therefore lead to a matching with less weight than the minimum-weight matching. □

Corollary 3.3.13. *Any* Length-minimal 2CRB labeling *instance can be solved in $\mathcal{O}((m-n+1) \cdot (\mu(m, n)+n^2))$ time, where $\beta(m, n)$ is the time needed to solve a minimum-weight matching problem on the complete bipartite graph $K_{m,n}$.*

Proof. Follows from Theorems 3.3.9 and 3.3.12. □

Some example values for $\mu(m, n)$ are $\mathcal{O}(v^3)$ (Gabow, 1974), $\mathcal{O}(ev \log v)$ (Gabow, 1985) or $\mathcal{O}(\sqrt{v\alpha(e, v) \log v}e \log(vN)$ (Gabow and Tarjan, 1989), where $e = n \cdot m$ is the number of edges, $v = m + n$ is the number of vertices and N is the maximum weight of an edge. These values are cited from [CR98].

4. External *po* labelings with rectangular labeling area

In the previous chapter we laid the groundwork for our next problem. This time we attempt to label a real "free space" problem: The labels lie inside a rectangle that is surrounded by the sites to be labeled.

4.1 Problem definition

Problem 4.1.1 (Rectangular area labeling)**.** *Let R be a rectangle, $\ell_1, \ldots, \ell_m$ be m ports that lie on the sides R (but not on the corner) and $s_1, \ldots, s_n$ be n ($n \leq m$) sites that lie outside R.*

Each two sites and/or ports must have a minimum vertical and horizontal distance of ε to ensure leader spacing.

The problem is to connect the sites $s_1, \ldots, s_n$ to the ports using crossing-free po leaders.

Problem 4.1.2 (Length-minimal Rectangular area labeling)**.** *The prerequisites are the same as in the* Rectangular area labeling *problem. The problem is to connect the sites $s_1, \ldots, s_n$ to the ports using crossing-free po leaders such that the total leader length* $\text{length}(S)$ *of the solution S is minimized.*

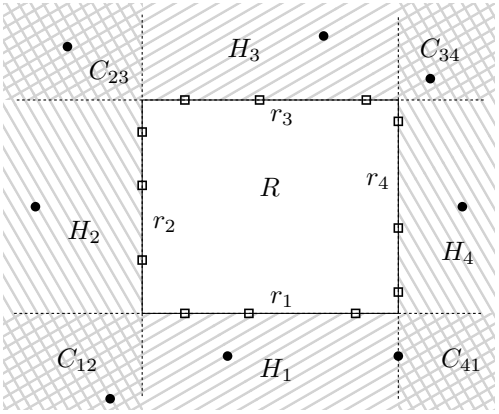
We divide the outside of the rectangle as before using half-planes: For $i \in \{1, \ldots, 4\}$, let H_i be the open half-plane defined by r_i that does not intersect R. For $(i, j) \in \{(1, 2), (2, 3), (3, 4), (4, 1)\}$, we set $C_{ij} := H_i \cap H_j$ (the "corners" of the labeling area). This is illustrated in Figure 4.1.

We partition the set of sites accordingly: Let Q_{ij} be the set of sites that lie in C_{ij}. For $i \in \{1, \ldots, 4\}$, let P_i be the sites that lie exclusively in H_i.

4.2 Solution

Our idea is to split a Rectangular area labeling instance I into two Two-sided convex rectilinear boundary labeling instances: one "bottom-left" instance I_{12} that contains all the ports from r_1 and r_2 and one "top-right" instance I_{34} that contains all the ports from r_3 and r_4. I_{12} contains at least the sites in P_1, P_2 and Q_{12} (since these

Figure 4.1: An instance of Rectangular area labeling with the area outside R partitioned.



have to be connected to ports on r_1 or r_2), similarly I_{34} contains at least the sites in P_3, P_4 and Q_{34}.

The sets Q_{23} and Q_{41} will have to be partitioned into the sets Q^2_{23}, Q^3_{23}, Q^4_{41} and Q^1_{41}. The instance I_{12} additionally contains the sites from Q^2_{23} and Q^1_{41}, and the instance I_{34} additionally contains the sites from Q^3_{23} and Q^4_{41}.

I_1 and I_2 are 2CRB instances if and only if the number of sites does not exceed the number of ports, which leads to the following conditions for the partition of Q_{23} and Q_{41}:

$$|Q^2_{23}| + |Q^1_{41}| \leq m_1 + m_2 - |P_1| - |P_2| - |Q_{12}| \tag{4.1}$$
$$|Q^3_{23}| + |Q^4_{41}| \leq m_3 + m_4 - |P_3| - |P_4| - |Q_{34}| \tag{4.2}$$

Additionally, Theorem 3.2.5 states that the 2CRB instance has a solution if and only if the number of sites that can only be connected to one side of the rectangle does not exceed the number of ports on that side. Thus we have the additional conditions:

$$0 \leq |Q^2_{23}| \leq m_2 - |P_2| \tag{4.3}$$
$$0 \leq |Q^3_{23}| \leq m_3 - |P_3| \tag{4.4}$$
$$0 \leq |Q^4_{41}| \leq m_4 - |P_4| \tag{4.5}$$
$$0 \leq |Q^1_{41}| \leq m_1 - |P_1| \tag{4.6}$$

Adding (4.3) to (4.4) and (4.5) to (4.6) yields the following equivalent system of inequations:

$$0 \leq |Q^2_{23}| \leq m_2 - |P_2| \tag{4.7}$$
$$0 \leq |Q_{23}| \leq m_2 + m_3 - |P_2| - |P_3| \tag{4.8}$$
$$0 \leq |Q^4_{41}| \leq m_4 - |P_4| \tag{4.9}$$
$$0 \leq |Q_{41}| \leq m_1 + m_4 - |P_1| - |P_4| \tag{4.10}$$

Now we can solve the basic problem of Rectangular area labeling:

Theorem 4.2.1. *A* Rectangular area labeling *instance I has a solution if and only if the inequations* (4.1), (4.2) *and* (4.7) *through* (4.10) *are satisfiable. I can be solved in $\mathcal{O}(m + n \log n + \gamma(m, n))$ time, where $\gamma(m, n)$ is the time needed to solve a 2CRB instance with m ports and n sites.*

Proof. We have to prove that I has a solution if and only if I can be partitioned into two solvable 2CRB instances I_{12} and I_{34} as described at the beginning of Section 4.2. This is then equivalent to the satisfiability of the equations, since I_{12} and I_{34} are 2CRB instances if and only if 4.1, 4.2 are satisfiable, and they have a solution if and only if 4.7 through 4.10 are satisfiable.

"⇒" Suppose we have a solution S for I. The leaders of sites in Q_{23} and Q_{41} induce partitions Q^2_{23}, Q^3_{23}, Q^4_{41} and Q^1_{41} of these two sets via $Q^k_{ij} = \{p \in Q_{ij} \mid \lambda(p, s) \in S, s \text{ lies on } r_k\}$ for $k \in \{i, j\}$. These partitions lead to two 2CRB instances I_{12} and I_{34}, and a solution for these two instances is induced by S.

"⇐" Suppose we can partition I into two solvable 2CRB instances I_{12} and I_{34} as described above. Let S_{12} and S_{34} be the respective solutions. Merging S_{12} and S_{34} yields a labeling S for I in which crossings may occur between leaders of sites in Q_{23} and Q_{41}. In other words, the partitions of Q_{23} and Q_{41} induced by S need not be crossing-free. We now partition Q_{23} and Q_{41} using crossing-free partitions of the same size as the partitions induced by S. This leads to two 2CRB instances I'_{12} and I'_{34} with the same number of sites and ports as I_{12} and I_{34}. By Theorem 3.2.5, the solvability of 2CRB instances depends only on the number of sites and ports, so I'_{12} and I'_{23} are solvable. Merging their solutions yields a solution for I which is now crossing-free because our partitions of Q_{23} and Q_{41} were crossing-free.

To prove the running time, first note that the satisfiability of (4.8) and (4.10) can be checked in $\mathcal{O}(1)$ since these depend only on I. We can check the satisfiability of the other inequations in $\mathcal{O}(m)$ by inserting every value X between 0 and $m_2 - |P_2|$ for $|Q^2_{23}|$. Those are the values for which inequation (4.7) is satisfied. We are then left with the following system of inequations for $y := |Q^4_{41}|$:

$$\begin{aligned} y &\geq 0 && \\ X + y &\leq m_1 + m_2 - |P_1| - |P_2| - |Q_{12}| && \text{from (4.1)} \\ (|Q_{23}| - X) + (|Q_{41}| - y) &\leq m_3 + m_4 - |P_3| - |P_4| - |Q_{34}| && \text{from (4.2)} \\ y &\leq m_4 - |P_4| && \text{from (4.9)} \end{aligned}$$

or equivalently

$$\begin{aligned} &y \geq 0 \\ &y \leq m_1 + m_2 - |P_1| - |P_2| - |Q_{12}| - X \\ &y \geq -m_3 - m_4 + |P_3| + |P_4| + |Q_{34}| + |Q_{23}| - X + |Q_{41}| \\ &y \leq m_4 - |P_4| \end{aligned}$$

whose satisfiability can be checked in $\mathcal{O}(1)$ time, including a satisfying value for y if one exists.

Computing the crossing-free partitions of Q_{34} and Q_{41} with the given partition sizes can be done in $\mathcal{O}(n \log n)$ time, and we are left with computing a solution for I_1 and I_2, which takes $2 \cdot \gamma(m, n)$ time. □

This also provides a solution for the length minimization variant of the problem:

Theorem 4.2.2. *Let I be a* Length-minimal Rectangular area labeling *instance. I can be solved in $\mathcal{O}(n \log n + m^2 \cdot \beta(m, n))$ time, where $\beta(m, n)$ is the time needed to solve a* Length-minimal 2CRB labeling *instance with m ports and n sites.*

Proof. The following algorithm called *LengthMinimalRectLabeling* computes a minimum-length solution: We conduct one step of the algorithm for each possible pair of partition sizes for Q_{23} and Q_{41}. As we saw in the proof of the previous theorem, there are $m_2-|P_2|+1$ possible values for $|Q^2_{23}|$. Because of inequation (4.9), there are $m_4-|P_4|+1$ possible values for $|Q^4_{41}|$. Together we have $\mathcal{O}(m^2)$ possible combinations of partition sizes.

In each step, we select crossing-free partitions $T_{23} = (Q^2_{23}, Q^3_{23})$ and $T_{41} = (Q^4_{41}, Q^1_{41})$ of Q_{23} and Q_{41} with the partition sizes given for this step. Partitioning takes $\mathcal{O}(1)$ time if the sites are sorted in advance (taking $\mathcal{O}(n \log n)$ time), e.g. by storing the sorted sites in an array and partitioning them by specifying two slices of the array. We solve the resulting LENGTH-MINIMAL 2CRB LABELING instances I_{12} and I_{34} in $\beta(m, n)$ time and select the solution with minimum total leader length for each instance. Merging the two solutions yields a crossing-free solution for I because the partitions of Q_{23} and Q_{41} are crossing-free.

We output a solution S with minimum total leader length among all the steps' solutions.

To see that S has minimum length, let S' be a minimum-length solution for I. S' induces crossing-free partitions $(Q^{1'}_{12}, Q^{2'}_{12})$ of Q_{12} and $(Q^{3'}_{34}, Q^{4'}_{34})$ of Q_{34}. This in turn induces two independent LENGTH-MINIMAL 2CRB LABELING instances I'_{23} (with the sites from $Q^{2'}_{12}$, Q_2, Q_3 and $Q^{3'}_{34}$ and the ports from r_2 and r_3) and I_{41} (with the rest of sites and ports), as well as minimum-length solutions S'_{23} and S'_{41} for these instances.

These solutions induce crossing-free partitions T'_{23} and T'_{41} of Q_{23} and Q_{41}. As *LengthMinimalRectLabeling* tries all possible partition sizes, it will in one step choose partitions T_{23} and T_{41} with sizes $|T'_{23}|$ and $|T'_{41}|$, respectively. Let $\tilde{S}$ be the solution for I computed in this step. We have $\text{length}(\tilde{S}) \geq \text{length}(S)$, as S has minimum length among all computed solutions.

From Lemma 3.3.8, we know that the minimum leader length of a LENGTH-MINIMAL 2CRB LABELING instance depends only on the partition size used and not on the partition itself. Using T_{23} instead of T'_{23} and T_{41} instead of T'_{41} to partition the instances I_{23} and I_{41} yields a solution S'' for I with $\text{length}(S'') = \text{length}(S')$.

As S'' now uses the same partitions for Q_{23} and Q_{41} as $\tilde{S}$, it also induces the same LENGTH-MINIMAL 2CRB LABELING instances I_{12} and I_{34}, and induces minimum-length solutions for these instances. Because *LengthMinimalRectLabeling* also computes a minimum-length solution for both of these instances, it follows that $\text{length}(S'') = \text{length}(\tilde{S})$. Together with $\text{length}(\tilde{S}) \geq \text{length}(S)$ and the minimality of S', it follows that S is a minimum-length solution for I. □

4.3 Outlook on labeling with more than one rectangle

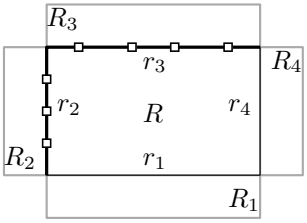
As seen for example in Figure 1.3, it may be useful to have not only one, but several rectangular labeling areas inside the map. We give a definition of this labeling problem:

Problem 4.3.1 (k-RECTANGLE BOUNDARY LABELING)**.** *Let $R_1, \ldots, R_k$ for $k \in \mathbb{N}$ be non-intersecting axis-parallel rectangles. Let $\ell_1, \ldots, \ell_m$ be m ports that lie on the sides of $R_1, \ldots, R_k$ and let $s_1, \ldots, s_n$ be n sites ($n \leq m$) that lie in the complement of $R_1 \cup \cdots \cup R_k$.*

The problem is to connect the sites $s_1, \ldots, s_n$ to the ports using crossing-free po leaders that do not intersect any of the rectangles.

Some work on labeling with two rectangular labeling areas has already been done, but had to be left out of this thesis due to time constraints. This work has however shown that labeling with two rectangles is likely to be solvable in polynomial time.

Figure 4.2: Transforming a k-sided boundary labeling instance into a k-RECTANGLE BOUNDARY LABELING instance



We show that the k-RECTANGLE BOUNDARY LABELING for $2 \leq k \leq 4$ is as least as hard as the k-sided boundary labeling problem studied in [KNR$^+$13]. In this problem, fixed ports lie on k adjacent sides of a rectangle R and all sites lie within R. The solutions obtained in the paper are:

- $n = 2$: solvable in $\mathcal{O}(n^2)$ time and $\mathcal{O}(n)$ space; the leader length minimization variant of the problem can be solved in $\mathcal{O}(n^8 \log n)$ time and $\mathcal{O}(n^6)$ space.
- $n = 3$: solvable in $\mathcal{O}(n^4)$ time and $\mathcal{O}(n)$ space.
- $n = 4$: solvable in $\mathcal{O}(n^{10})$ time and $\mathcal{O}(n)$ space.

These are not necessarily lower bounds for the k-sided problems, but give a general idea of how hard the problem might be.

Theorem 4.3.2. *Let $I = (R, P, S)$ with rectangle R, ports P and sites S be an instance of the k-sided boundary labeling problem defined in [KNR$^+$13] for $k \in \{2, 3, 4\}$. Then there exists a polynomial transformation from I into an instance I' of* k-RECTANGLE BOUNDARY LABELING *which has a solution if and only if I has a solution.*

Proof. We first create four non-intersecting rectangles $R_1, \ldots, R_4$ that each share one side with R, as shown in Figure 4.2. We create the k-RECTANGLE BOUNDARY LABELING instance I' by keeping the sites and ports from I. For each rectangle side r_i in I that contains ports, we add the rectangle R_i to I', so that each port lies on the side of a rectangle.

This is obviously a polynomial-time transformation. Each solution of I is a solution for I' because all leaders lie in R and none of the rectangles R_i intersects R, thus no leader intersects a rectangle. Conversely, a solution of I' is also a solution for I. □

5. Conclusion

In this thesis, we presented an introduction to the problem area of labeling maps with free spaces. We first presented an algorithm for external labeling with *opo* leaders and a rectilinear labeling area. The labels can be placed anywhere inside the labeling area and can even be placed next to each other vertically and horizontally. This allows optimal utilization of the available space at the boundary of a map.

We then solved the labeling problem for *po* leaders and fixed ports on two adjacent sides of a rectangle R, where the sites lie outside of R. We considered the variant with and without leader length minimization. A solution can be constructed by partitioning the set of sites using crossing-free partitions and solving the resulting instances using one-sided boundary labeling.

Lastly, we presented an algorithm for solving the labeling problem with fixed ports and a free space consisting of one rectangle. We again considered leader length minimization as an additional objective. We observed that the variants with as many sites as ports are solvable considerably faster than the variants with more ports than sites. We gave a short outlook on labeling problems with more than one free space.

The presented algorithms all run in polynomial time. The theoretical boundary of the running time can nevertheless be quite large, even if only corner cases are considered.

For real-world implementation purposes, these algorithms – apart from the most basic ones – are therefore most likely not directly suited. In this case, generic methods from Algorithm Engineering could be applied: Heuristics based on Genetic Algorithms, Simulated Annealing and the like could tackle the problems more easily.

This is especially the case because for some difficult instances, a solution conforming to a hard criterion like the absence of leader crossings might not even exist. With these kinds of algorithms, we could easily reduce the problem to minimizing the number of crossings instead of avoiding them. The same goes for the leader length: A viewer might not even be able to distinguish between a minimum length solution and one that is slightly longer.

Other interesting open topics in this problem area would therefore include

- labeling with more than one rectangular free space (see Section 4.3)
- allowing other shapes as free spaces on the map,
- the combination of internal and boundary labeling (ex. [LN10], [BKPS11]),

- the combination of free spaces inside a map and boundary labeling outside the map,
- dynamic labeling, e. g. for zooming in and out of a map (ex. [NPS10]).

However, the results from this thesis and from other papers on related topics suggest that exact solutions of these problems, even if computable in polynomial time, might nevertheless be only of theoretical interest.

Therefore, the application of Algorithm Engineering methods on these problems, perhaps in the form of a generic labeling framework, would certainly be a valuable contribution. Experimental evaluation of the exact and heuristic methods might provide more insight into the hardness of these problems in general.

Bibliography

[BHKN08] M. Benkert, H. Haverkort, M. Kroll, and M. Nöllenburg, "Algorithms for multi-criteria one-sided boundary labeling," in *Graph Drawing*, ser. Lecture Notes in Computer Science, S.-H. Hong, T. Nishizeki, and W. Quan, Eds. Springer Berlin Heidelberg, 2008, vol. 4875, pp. 243–254. Available online: `http://dx.doi.org/10.1007/978-3-540-77537-9_25`

[BKPS11] M. Bekos, M. Kaufmann, D. Papadopoulos, and A. Symvonis, "Combining traditional map labeling with boundary labeling," in *SOFSEM 2011: Theory and Practice of Computer Science*, ser. Lecture Notes in Computer Science, I. Černá, T. Gyimóthy, J. Hromkovič, K. Jefferey, R. Královič, M. Vukolić, and S. Wolf, Eds. Springer Berlin Heidelberg, 2011, vol. 6543, pp. 111–122. Available online: `http://dx.doi.org/10.1007/978-3-642-18381-2_9`

[CR98] W. Cook and A. Rohe, "Computing minimum-weight perfect matchings," *INFORMS Journal on Computing*, vol. 11, pp. 138–148, 1998. Available online: `www.math.uwaterloo.ca/~bico/papers/match_ijoc.pdf`

[FW91] M. Formann and F. Wagner, "A packing problem with applications to lettering of maps," in *Proceedings of the seventh annual symposium on Computational geometry*, ser. SCG '91. New York, NY, USA: ACM, 1991, pp. 281–288. Available online: `http://doi.acm.org/10.1145/109648.109680`

[Imh75] E. Imhof, "Positioning names on maps," *The American Cartographer*, vol. 2, no. 2, pp. 128–144, 1975. Available online: `http://www.mapgraphics.net/downloads/Positioning_Names_on_Maps.pdf`

[Jam12] N. Jami, "Point labeling with leaders for convex boundaries," 2012, diploma thesis. Available online: `http://i11www.iti.uni-karlsruhe.de/_media/teaching/theses/da-jami-12.pdf`

[Kau09] M. Kaufmann, "On map labeling with leaders," in *Efficient Algorithms*. Springer, 2009, pp. 290–304.

[KNR+13] P. Kindermann, B. Niedermann, I. Rutter, M. Schaefer, A. Schulz, and A. Wolff, "Multi-sided boundary labeling," *CoRR*, vol. abs/1305.0750, 2013. Available online: `http://arxiv.org/pdf/1305.0750v1.pdf`

[LN10] M. Löffler and M. Nöllenburg, "Shooting bricks with orthogonal laser beams: A first step towards internal/external map labeling." in *CCCG*, 2010, pp. 203–206. Available online: `http://cccg.ca/proceedings/2010/paper54.pdf`

[Nie12] B. Niedermann, "Consistent labeling of dynamic maps using smooth trajectories," 2012, diploma thesis. Available online: `http://i11www.iti.uni-karlsruhe.de/_media/teaching/theses/da-jami-12.pdf`

[NPS10] M. Nöllenburg, V. Polishchuk, and M. Sysikaski, "Dynamic one-sided boundary labeling," in *Proceedings of the 18th SIGSPATIAL International*

Conference on Advances in Geographic Information Systems, ser. GIS '10. New York, NY, USA: ACM, 2010, pp. 310–319. Available online: http://doi.acm.org/10.1145/1869790.1869834

[WS09] A. Wolff and T. Strijk, "The map-labeling bibliography," 2009. Available online: http://i11www.iti.uni-karlsruhe.de/~awolff/map-labeling/bibliography/