Combining Traditional Map Labeling with Boundary Labeling[⋆]

Michael A. Bekos[1], Michael Kaufmann[2], Dimitrios Papadopoulos[1], and Antonios Symvonis[1]

[1] School of Applied Mathematical & Physical Sciences,
National Technical University of Athens, Greece
{mikebekos,dpapadopoulos,symvonis}@math.ntua.gr

[2] University of Tübingen, Institute for Informatics, Germany
mk@informatik.uni-tuebingen.de

Abstract. The traditional map labeling problems are mostly $\mathcal{NP}$-hard. Hence, effective heuristics and approximations have been developed in the past. Recently, efficient algorithms for the so-called boundary labeling model have been introduced which assumes that the labels are placed on the boundary of the map and connected by polygonal leaders to their corresponding sites. Internal labels have been forbidden. In this paper, we allow both. Since clearly internal labels should be preferred, we consider several maximization problems for the number of internal labels and we show that they can be obtained efficiently or in quasi-polynomial time.

1 Motivation

In information visualization, geographic information systems, and cartography, map labeling constitutes an important task, which is concerned with the placement of extra information –in the form of textual labels– next to features of interest of an illustration. In order to ensure readability, it is suggested that the labels: i) do not overlap with each other, and ii) are close to (if possible, next to) the features they are associated with. Unfortunately, due to these constraints, the majority of the map labeling problems turns out to be computationally hard [9]. A detailed bibliography on map labeling can be found in [19].

A great amount of research on map labeling has been devoted to labeling site-features of a map. However, in the presence of large labels or dense point sets, producing a labeling with no overlaps is most of the times impossible. To address this problem, a commonly used approach is to place the labels next to the map, and to connect them to their sites by polylines, called *leaders*. This labeling approach is called *boundary labeling*; in the past, several papers have appeared that show that certain types of boundary labeling problems can be solved efficiently in theory and practice [2,3,4,5,6,12,15,16].

⋆ The work is funded by the National Technical University of Athens research program ΠEBE 2008.

© Springer-Verlag Berlin Heidelberg 2011

In this paper, we introduce and study a *mixed labeling* model, according to which we place the labels next to the sites and use boundary labeling in cases where this is not feasible due to overlaps.

As a first step towards solving this labeling problem, we investigate simple settings of the problem. Our motivation rests on the work of Fowler et al. [10], who proved that the number maximization problem (i.e., the problem of determining the maximum number of labels that can be placed on a set of sites without overlaps) is $\mathcal{NP}$-hard, even if the labels are of uniform size, each site has only one label candidate position and leaders are not permitted. In contrast to the negative result of Fowler et al., there are variants of the proposed model which admit an algorithm that maximizes the number of site-labels and guarantees the placement of all labels (either immediately next to the sites or on the boundary of the underlying drawing through leaders). Our variants, where we give quasi-polynomial time algorithms, suggest that using boundary labeling, the problem becomes easier as no $\mathcal{NP}$-hard problem is known which has a quasi-polynomial algorithm. More sophisticated variants are still to be considered. The proposed model is well suited when labeling technical or medical maps where it is common to explain certain features of the map with label on its boundary, since in the case where the map contains several features to be labeled, a great number of leaders will unavoidably occur, and subsequently clutter the illustration.

This rest of this paper is structured as follows: In Section 2, we formally define the mixed map labeling model. In Sections 3, 4 and 5, we study several variations of the mixed labeling problem. In Section 6, we provide an experimental evaluation of our algorithms. We conclude in Section 7 with open problems.

2 Problem Definition

Typically, a map labeling problem consists of a set $P = \{s_1, s_2, \dots s_n\}$ of n sites to be labeled. Each site $s_i = (x_i, y_i)$ is associated with an axis-parallel, rectangular label l_i of dimensions $w_i \times h_i$. We also assume that the site set is enclosed in an axis-parallel rectangle $R = [0, W] \times [0, H]$ (*enclosing rectangle*). According to our model, there exist two alternatives to label each site: i) either through a label "close" to the site (*internal labels*), or ii) through a label on the boundary of R and a leader which realizes the connection (*external labels*).

For each site which is to be labeled with an internal label, the input specifies a *model*, which indicates how the site's label can be placed w.r.t. the actual position of the site. The most important models studied in the literature are: i) the *fixed position model*, where each site has a finite set of *label candidates* (i.e., feasible label positions), and ii) the *slider model*, where each label can be placed at any position, so that it touches its site. Fig.1 depicts some commonly used variants of the both models. For a more formal definition refer to [13].

The external labels are usually attached to one, two or all four sides of R. Several types of leaders have been proposed, among them *straight-line* [5], *rectilinear* [5] and *octilinear* [6]. In this paper, we focus on rectilinear leaders of type-*opo*, that consist of three line segments, where the first and third ones are

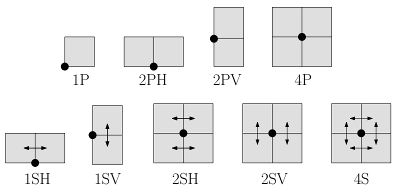

Fig. 1. Each model has an abbreviation of the form xMD, where $M \in \{P, S\}$ stands for fixed-position (P) or slider model (S), $x \in \{1, 2, 4\}$ refers to the number of fixed positions or sliding directions, and $D \in \{\emptyset, H, V\}$ indicates the horizontal or vertical direction in which fixed-position labels are arranged or labels slide

orthogonal (o) to the side of R containing the label, and the second one is parallel (p) to that side. We further assume that the sites are in *general position*, i.e., no two sites share the same y or x coordinate. For more details, refer to [5].

Conventionally, a mixed labeling model (m, k, t) is identified as a type-m traditional labeling model supported by a k-sided boundary labeling model with type-t leaders, where $m \in \{1P, 2PH, 2PV, 4P, 1SH, 1SV, 2SH, 2SV, 4S\}$, $k \in$ {Left-Sided, Right-Sided, Two-Sided, Four-Sided} and $t \in \{s, po, opo, do, od, pd\}$[1], in the case where the internal (external) labels are placed according to the rules of model m (k) and the leaders are of type t. Our intension is to obtain *legal labelings* in which no internal labels overlap and no leader intersect or overlap, and, simultaneously maximize the number of internal labels. We further assume that one of the rectangle's sides can always accommodate all labels.

3 The (1P, Right-Sided, opo) Mixed Labeling Model

We first consider the mixed model (1P, right-sided, *opo*) and present an algorithm which produces labelings with maximum number of internal labels. Our algorithm performs in three steps. Initially, it computes a legal labeling purely consisting of external labels. This is feasible since we have assumed that one of the enclosing rectangle's sides can accommodate all labels. As shown in [5], in a legal solution of the pure boundary labeling problem (i.e., no internal labels are allowed), the vertical order of the sites is identical to the vertical order of their corresponding labels and it can be computed in $O(n \log n)$ time. The leaders are considered to be preliminary and might either be removed or labeled permanent. In the second step, the algorithm identifies sites whose preliminary leaders cannot be replaced by internal labels. Crucial is the following. For two sites with overlapping internal labels, the lower one must be labeled through a leader (see the right part of Figure 2).

Based on this observation, we determine in a second step for each site s, if its inner label $l(s)$ is intersected by a label $l(s')$ such that s' lies above of s. If such a label $l(s')$ exists, we mark the leader of s to be permanent. In the final step, we perform a top-down plane sweep, where we stop at each leader that

[1] Abbreviations s, po, opo, do, od and pd refer to different types of leaders (see [2] and [5]).

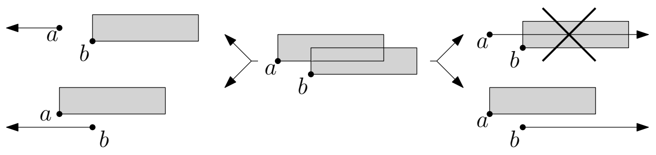

Fig. 2. The internal labels of sites a and b are involved in an overlap. In the case where the leaders are routed to the right only one feasible case exists. However, in the case where the leaders are routed to the left, two feasible solutions exist.

we reach. If the leader, say for site s, is already permanent and it intersects other internal labels $l(s')$, we mark the leaders for s' also to be permanent. If the leader of site s is still preliminary, we know that for site s there is no other intersecting internal label and also no intersecting leader. Hence we can replace it by an internal label.

Theorem 1. *The (1P, right-sided, opo) problem can be solved in $O(n \log^2 n)$ time.*

Sketch of proof. As already stated, the first step of our algorithm needs $O(n \log n)$ time. By employing a dynamic data structure that maintains a set of rectangles (that changes under insertions and deletions) and given a query rectangle q reports whether q is involved in an overlap with other rectangle stored in the data structure in $O(\log^2 n)$ time [8], the second step of our algorithm can be implemented in a total of $O(n \log^2 n)$ time. The third step of our algorithm needs an extra $O(n \log n)$ time by employing a dynamic data-structure that maintains a set of points Q that changes under insertions, and supports visibility queries of the form in $O(\log n)$ time: "Given a threshold value x_0 and a query range (y_{bottom}, y_{top}), return the point of Q (if any) with the largest y-coordinate that is located within the rectangle $(0, x_0) \times (y_{bottom}, y_{top})$" [17, pp. 209]. □

4 The (1P, Left-Sided, opo) Mixed Labeling Model

In this section, we adopt the scenario of the previous section assuming that the external labels occupy the left side of R. Again, our goal is to obtain a labeling with maximum number of internal labels. In contrast to the case where the external labels are to the right of R, in this case when two labels overlap it is not necessary that the lower label is always external in a feasible solution and this makes the problem more complicated (see the left part of Figure 2).

4.1 A Quasi-polynomial Time Solution

We first present a quasi-polynomial time algorithm, assuming that the labels are of uniform height. Our recursive algorithm splits the problem into a particular number of half-sized subproblems by fixing an internal label each time, then recursively solves the subproblems and derives a solution of the initial problem by keeping the solution which implies the maximum number of internal labels.

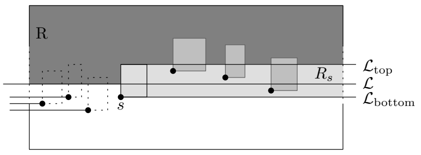

Fig. 3. An illustration of our recursive algorithm

Consider an arbitrary line $\mathcal{L}$ that is parallel to x-axis and intersects several internal labels and let l_s be any of these internal labels. Let also s be its corresponding site. In the final optimal solution, site s will be labeled either by an internal or by an external label. Consider a solution in which site s is the first site (from left to right) out of those intersected by line $\mathcal{L}$ to be labeled by an internal label (see Fig. 3). The sites to the left of s with internal labels intersecting $\mathcal{L}$ must all be routed through leaders. Let $\mathcal{L}_{\text{top}}$ ($\mathcal{L}_{\text{bottom}}$) be the horizontal half-line that emanates from the top-left (bottom-left) corner of l_s and coincides with the top (bottom) boundary edge of l_s. Also, let R_s be the rectangle defined by the left side of l_s, $\mathcal{L}_{\text{top}}$, $\mathcal{L}_{\text{bottom}}$ and the right side of R (see the light-gray rectangle of Fig. 3). Then, all sites inside R_s must be labeled by internal labels.

First, we have to check whether all sites inside R_s can be labeled by internal labels, which needs $O(n \log^2 n)$ time using range queries [17]. Next, we turn our attention to the solution of the internal label maximization problem restricted to instances where l_s is the first internal label (from left to right) out of those that intersect $\mathcal{L}$. The maximum number of internal labels can be determined by the solution of two independent problems, each consisting of a set of sites to be labeled and a set of already labeled sites through internal or external labels. For the first (top) subproblem, the set of sites consists of all unlabeled sites above line $\mathcal{L}$ and half-line $\mathcal{L}_{\text{top}}$ (within the dark-gray polygonal region of Fig.3), while the set of fixed labels consists of the internal labels for the sites in R_s, plus l_s (plus, any previously fixed label placements). For the second (bottom) subproblem, the set of sites consists of the remaining unlabeled sites, while the set of fixed sites consists of all sites in R_s, plus all sites to the left of s which are labeled by leaders, plus s (plus, any previously fixed label placements).

The global optimal solution can then be easily obtained by considering each of the internal labels crossed by $\mathcal{L}$ as the first internal label. If line $\mathcal{L}$ is carefully selected so that it has half of the unlabeled sites above and below it, then the time needed to calculate the optimal solution is given by the following formula:

$$T(k) = \begin{cases} 2kT(k/2) + n^2 \log^2 n, & k > 1 \\ O(1), & k = 1 \end{cases}$$

The $n^2 \log^2 n$ term is needed in order to check whether the labeling of the sites in each rectangle R_s is legal. The solution of this formula leads to a time complexity of $O(n^{\log n+3})$. So, we can state the following theorem.

Theorem 2. *The* ($1P$, *left-sided, opo*) *problem with labels of uniform height can be solved in* $O(n^{\log n+3})$ *time.*

4.2 A 2-Approximation Algorithm

In the following, we present a more efficient, factor 2-approximation algorithm, which is based on a reduction to a variant of the 2-SAT problem. Reductions to the 2-SAT problem have been previously used in the map labeling literature [9,18]. In our approach, each site s_i in the labeling instance is identified by a boolean variable z_i, $i = 1, 2, \ldots, n$. The true or false value of each variable z_i specifies whether s_i is labeled through an external or an internal label, respectively. We proceed to construct a set of clauses based on the following cases:

1. **Avoidance of internal label overlaps:** Let $\overline{z_i}$ and $\overline{z_j}$ be two internal label candidates that overlap and assume that neither site s_i nor s_j is contained in $\overline{z_j}$ and $\overline{z_i}$, respectively. Then, $\overline{z_i}$ and $\overline{z_j}$ cannot simultaneously appear in a solution, which is ensured by clause: $z_i \vee z_j$.
2. **Avoidance of site and internal label overlaps:** Let $\overline{z_i}$ and $\overline{z_j}$ be two internal label candidates that overlap and assume that s_j is contained in $\overline{z_i}$. Then, $\overline{z_i}$ cannot appear in a solution, which is ensured by clause: z_i.
3. **Avoidance of leader and internal label intersections:** Let s_i and s_j be a pair of sites, such that the leader of s_j crosses the internal label of s_i. Then, z_j and $\overline{z_i}$ cannot simultaneously appear in a solution. This is ensured by clause: $z_i \vee \overline{z_j}$.

Having formulated our problem as a 2-SAT clausal problem, we need a satisfying truth assignment which simultaneously minimizes the number of true variables. Gusfield and Pitt [11] proved that given a satisfiable 2-SAT formula, it is $\mathcal{NP}$-hard to find a satisfying assignment that contains a minimum number of true variables and proposed an approximation that results in an assignment with at most twice as many true variables as necessary in $O(NM)$ time, where N and M are the number of variables and clauses, respectively. Hence, we can state the following theorem:

Theorem 3. *The (1P, left-sided, opo) problem admits a factor 2 approximation algorithm which needs $O(n^3)$ time.*

4.3 An ILP Formulation

In this subsection, we provide an alternative solution of the (1P, left-sided, *opo*) labeling problem, that is, a formulation as an integer linear program. ILP formulations have also been previously used in the map labeling literature [7,20]. In our formulation, each site s_i is associated with two variables z_i and w_i, each of which has value 0 or 1 indicating the absence or presence of an internal label and a leader, respectively. Then, one set of constraints expresses the requirement that each site should be labeled exactly once, either through an internal label or through a leader. Therefore, $z_i + w_i = 1$, for each $i = 1, 2, \ldots, n$. A second set of constraints expresses the requirement that no two internal label candidates overlap. More precisely, if s_i and s_j is a pair of sites, whose internal labels overlap then a constraint of the form $z_i + z_j \leq 1$ should be present. Finally, a last set of constraints is needed in order to avoid crossings among leaders and internal labels. So, the case where the leader emanating from a site s_i crosses the internal label of another site s_j, can be expressed by a constraint of the form $w_i + z_j \leq 1$. Since we seek to maximize the number of internal labels, the objective function of our integer linear program is $\sum_{i=1}^{n} z_i$. Clearly, the set of constraints can be constructed in $O(n^2)$. Therefore, we can state the following theorem.

Theorem 4. *An instance of the (1P, left-sided, opo) problem can be transformed into an integer linear program in $O(n^2)$ time.*

Note that the formulation given above with small modifications can be used to solve any type-*opo* mixed labeling problem that involves a fixed position model. However, the ILP presented above has an extra property; it contains two variables per constraint. Bar-Yehuda and Rawitz [1] proved that such integer programs can be approximated by a factor of 2 in $O(NMU)$ time, where N, M and U are the number of variables, constraints and the maximum variable range, respectively. This implies that our problem admits another 2-approximation algorithm that also needs $O(n^3)$ time.

5 The (1P, Two-Sided, *opo*) Mixed Labeling Model

In this section, we adopt the 1P point-labeling model supported by the two opposite-sided type-*opo* boundary labeling model as alternative, assuming that the labels are of uniform height and the external labels are allowed to be placed along the left and the right side of R. The algorithm we describe follows the lines of the recursive algorithm for the left sided case, presented in Section 4. However, due to the fact that for each site we have three possible labeling alternatives (i.e., leader to the left, internal label, leader to the right), the splitting into subproblems is slightly more complicated.

Let h be the height of each label and λ be the maximum number of sites within any strip of height h. Consider some legal labeling and as before, consider an arbitrary line $\mathcal{L}$ that is parallel to the x-axis and intersects several internal labels. Let again l_s be the leftmost of these labels and s be its corresponding site. Define as before lines $\mathcal{L}_{\text{top}}$ and $\mathcal{L}_{\text{bottom}}$. Let t be the leftmost site out of those to the right of label l_s that is labeled by a leader towards the right side of R. Then, the following must hold:

- Let R_1 be a rectangle (of maximum height h) whose bottom-right corner coincides with site s, whereas its top-left corner coincides with the intersection of line $\mathcal{L}$ and the left side of the enclosing rectangle R (see Fig.4a). Then, all sites in R_1 are labeled through leaders to the left side of R.
- Let R_s be the rectangle that has the sites s and t on its left and right sides, respectively, and is defined by the half-lines $\mathcal{L}_{\text{top}}$ and $\mathcal{L}_{\text{bottom}}$ (see Fig.4a). Then, all sites in R_s must be labeled through internal labels.

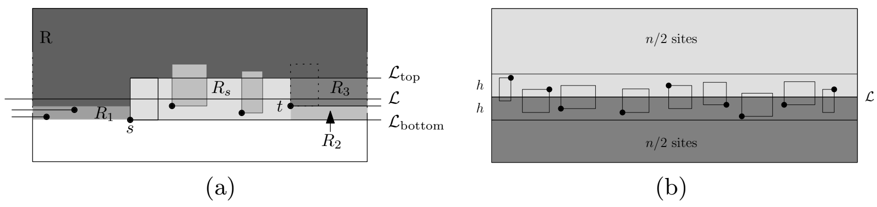

Fig. 4. Illustrations of our recursive algorithms

- Let R_2 be the rectangle having site t as its top-left corner and the intersection of $\mathcal{L}_{\text{bottom}}$ with the right side of R as its bottom-right corner. Then, all sites in R_2 must be labeled through leaders to the right side of R.
- Let R_3 be the rectangle with site t as its bottom-left corner and the intersection of $\mathcal{L}_{\text{top}}$ with the right side of R as its top-right corner. Then, all sites in R_3 must be labeled by internal labels or through leaders to the right side of R.

As in the previous case, the maximum number of internal labels can be determined by the solution of two independent problems, each consisting of a set of sites to be labeled and a set of already labeled sites through internal or external labels. For the first (top) subproblem, the set of sites consists of all unlabeled sites that lie above line $\mathcal{L}$, half-line $\mathcal{L}_{\text{top}}$ and in the interior of R_3 (within the dark-gray polygonal region of Fig.4a), while the set of fixed labeled sites consists of the sites in R_s (which are labeled by internal labels), plus, s and t (plus, any previously fixed label placements). For the second (bottom) subproblem, the set of sites consists of the remaining unlabeled sites, while the set of fixed labeled sites consists of all sites in $R_1 \cup R_2$ (which are labeled by leaders), plus, the sites in R_s (which are labeled by internal labels), plus, s and t (plus, any previously fixed label placements).

Attempting to do an analysis as in the previous case, we realize that there are at most $2\lambda^2$ subproblems, each defined by a pair of sites s and t (as defined above). If line $\mathcal{L}$ equally-splits the sites, observe that the top subproblem might have at most $n/2$ sites plus the sites of R_3 that are below line $\mathcal{L}$. Since each slice of height h contains at most λ sites, the top subproblem contains at most $n/2 + \lambda$ sites. Then, the bottom subproblem contains at least $n/2 - \lambda$ sites. So, in the i-th recursion-level, each subproblem has size at most $n/2^i + \sum_{k=0}^{i} \lambda/2^k$, which approximately equals $n/2^i + 2\lambda$. Note that parameter λ is not halfed at each recursion level, the subproblems however do.

We choose to stop the recursion when the size of a subproblem is less than 3λ to facilitate our analysis. Observe that a problem of size 3λ can be solved in $O(3^{3\lambda}\lambda \log^2 \lambda)$ time. For each site, we have three choices, i.e., leader to the left, internal label, leader to the right. Thus, the time complexity of our algorithm is given by the following formula.

$$T(n) \leq \begin{cases} \lambda^2(2T(n/2+\lambda) + n\log^2 n), & n > 3\lambda \\ 3^{3\lambda}\lambda \log^2 \lambda, & n \leq 3\lambda \end{cases}$$

The above formula, for $n > 3\lambda$ means that we have $\log(n/3\lambda)$ recursion levels, giving a total time of $O(n(n/(3\lambda))^{2\log\lambda}\log^2 n + 2\lambda(n/(3\lambda))^{2\log(2\lambda)}\log^2 n)$, while the second term gives for each subproblem of size λ an amount of at most $3^{3\lambda}\lambda\log^2\lambda$. On level i, we have $2^i\lambda^{2i}$ subproblems, hence in total the recursion ends with $(n/(3\lambda))^{2\log\lambda+1}$ subproblems. In total this gives a seemingly complicated formula for the time complexity: $O(n(n/(3\lambda))^{2\log\lambda}\log^2 n + 2\lambda(n/(3\lambda))^{2\log(2\lambda)} + (n/(3\lambda))^{2\log\lambda+1}\cdot 3^{3\lambda}\lambda\log^2\lambda)$. We conclude by the following theorem. If λ is constant, the following theorem gives a polynomial time bound.

Theorem 5. *An instance of the (*$1P$*, 2-sided, opo) labeling problem can be solved in time* $(n/\lambda)^{O(\log\lambda)}\cdot 3^{O(\lambda)}$.

5.1 Generalizations

In the following, we present a general scheme for the mixed labeling model (m, k, opo), where $m \in \{2PH, 2PV, 4P\}$, $k \in \{\emptyset$, Left-Sided, Right-Sided, Two-Sided$\}$. The proposed scheme also supports labelings without external labels. It resembles to one of the first map labeling algorithms by Kucera et al [14].

Let κ and μ be two variables defined as follows. Variable κ equals to (a) zero, if $k = \emptyset$, (b) one, if $k \in \{$Left-Sided, Right-Sided$\}$, and, (c) two, if $k =$ Two-Sided. Similarly, variable μ equals to (a) two, if $m \in \{2PH, 2PV\}$, and, (b) four, if $m = 4P$. Let also $\mathcal{P}$ be a labeling problem with labels of uniform height h and λ-height restricted, i.e., in each horizontal strip of height h there are at most λ sites. Let $\mathcal{L}$ be a horizontal line bisecting the rectangle into two subproblems $\mathcal{P}_{top}$ and $\mathcal{P}_{bot}$, such that the subproblems have at most $n/2$ sites each. Because of the height restriction, at most λ sites above $\mathcal{L}$ might have a bottom-label which intersects $\mathcal{L}$, and analogously at most λ sites below $\mathcal{L}$ might have a top-label which intersects $\mathcal{L}$ (see Fig.4b). It is clear that any solution of $\mathcal{P}$ consists of a configuration $\mathcal{C}$ of labels intersecting $\mathcal{L}$ and the solution of $\mathcal{P}_{top}(\mathcal{C})$ and $\mathcal{P}_{bot}(\mathcal{C})$, where the two subproblems are modified according to the configuration $\mathcal{C}$. For each configuration $\mathcal{C}$, we solve the corresponding subproblems $\mathcal{P}_{top}(\mathcal{C})$ and $\mathcal{P}_{bot}(\mathcal{C})$, construct the solution $\mathcal{P}(\mathcal{C})$ and optimize over all configurations $\mathcal{C}$.

For each configuration $\mathcal{C}$, labels of the sites close to line $\mathcal{L}$ might intersect $\mathcal{L}$ or not. The sites above and below might intersect the strip only when they use their lower or upper internal labels, or they do not intersect at all, hence we have $\kappa/2+1$ possibilities for the upper sites and $\kappa/2+1$ possibilities for the sites below. Hence, we have at most $(\kappa/2+1)^\lambda\cdot(\kappa/2+1)^\lambda$ different configurations for the middle line $\mathcal{L}$. If we fix one of those configurations, then the two subproblems arising on top or below the strip are independent of each other, and can be solved recursively. They only depend on the fixed label configuration of the middle strip and they have size at most $n/2$ each. Each configuration can be checked, whether it is legal or not, in time $O(n\log^2 n)$, using the methods described in Section 4.

Note that the recursion stops when $n \leq \lambda$, after $\log(n/\lambda)$ recursion levels. Each time, we have to check the legality of the solution for the subproblem. This can be done in time $O(\lambda\log^2\lambda)$ by a series of $O(\lambda)$ range queries concerning at most $O(\lambda)$ rectangles. Then, $T(\lambda) \leq O((\kappa+\mu)^\lambda\lambda\log^2\lambda)$. Hence, we get a similar recursion as before, which can be summarized by the following theorem.

Theorem 6. *A λ-height restricted instance of the (m, k, opo) problem, where $m \in \{2PH, 2PV, 4P\}$, $k \in \{\emptyset$, Left-Sided, Right-Sided, Two-Sided$\}$ can be solved in time $(n/\lambda)^{O(\lambda \log \kappa)} \cdot (\kappa + \mu)^{\lambda+1}$.*

More concretely, if $m \in \{2PH, 2PV\}$ and $k \in \{\text{Two-Sided}\}$, a λ-height restricted instance of this problem can be solved in $O((n/\lambda)^{1+2\lambda} \cdot (4^{\lambda}\lambda \log^2 \lambda + n \log^2 n)$. If $m \in \{4P\}$ and $k \in \{\text{Left-Sided, Right-Sided}\}$, we have 5 choices for each site (either through a leader or through an internal label utilizing one of the four available internal label candidates). For the sites directly above and below the strip, we have 3 choices, since two of the four available internal label candidates cannot be utilized. Hence, there are $3^{\lambda} \cdot 3^{\lambda}$ configurations for the middle line. Therefore, a λ-height-restricted instance of this model can be solved in $O((n/\lambda)^{1+\lambda \log 9} \cdot (5^{\lambda}\lambda \log^2 \lambda + n \log^2 n)$. When λ is constant or logarithmic in n, we obtain polynomial or quasi-polynomial algorithms, which do not contradict the $\mathcal{NP}$-hardness of the general problem if λ is large ($\lambda \geq n^{\epsilon}, \epsilon > 0$).

6 Experimental Results

In this section, we present the results of the experimental evaluation of the algorithms presented in Sections 3, 4.1 and 5. The experiment was performed on a Linux machine with 2.60 GHz CPU and 2GB RAM. A parameter that was taken into account in the experiment's setup was the *density* of each input point set, that is the ratio of the total area of all labels to the area of R. The labels had a predefined, fixed size. Hence, a specific density value together with a given value for the total number of point sites, also specifies the dimensions of the enclosing rectangle. The density values vary from 1/10 (low density) to 1/5 (high density), whereas the point sets were randomly generated with 5 up to 100 sites. Given a density value and a number of point sites, the experiment generated 100 random point sets, which were given as input to the three algorithms.

As expected the one-sided, polynomial time algorithm of Section 3 is slightly faster than the corresponding one of Section 4.1 that runs in quasi-polynomial time. Both algorithms need less than half a second to perform the labeling, regardless of the density of the labeling or the size of the input point set. On the other hand, the two-sided, quasi-polynomial time algorithm of Section 5 needs noticeably more time, especially in large and dense point sets. For input point sets consisting of less than 40 points the algorithm needs less than two and a half seconds. For input point sets with 50 up to 60 point sites less than a minute. However, for more dense point sets consisting of more than 70 points, there exist instances which need more than ten minutes. Note that in our experiment we didn't bound the parameter λ.

Regarding the quality of the produced labelings, which is measured in terms of the number internal labels, the two-sided, quasi-polynomial time algorithm of Section 5 produces labelings that have significantly more internal labels than the other two algorithms, especially in large and dense point sets. On the other hand, the one-sided, quasi-polynomial time algorithm of Section 4.1 and the one-sided, polynomial time algorithm of Section 3 have roughly the same performance.

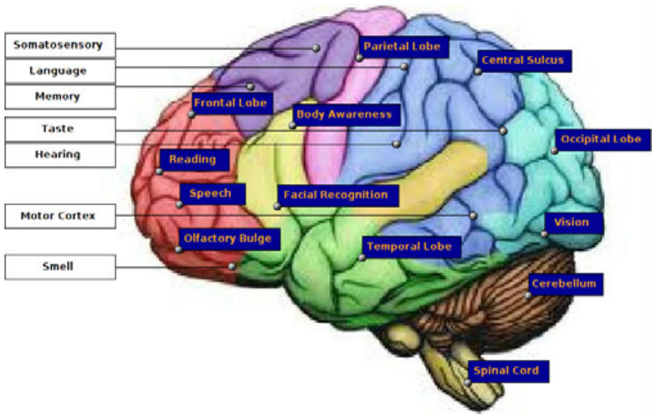

Fig. 5. The anatomy of a human brain produced by the (1P, left-sided, *opo*) model

Figure 5 depicts a medical map that describes the anatomy of a human brain. The labeling has been produced by our one-sided quasi-polynomial time algorithm of Section 4.1, in which we have also performed an additional post-process step, in order to minimize the number of leader-bends (using an algorithm given in [5]). Since the external labels are few, most of the leaders are eventually drawn as straight-lines, which drastically improves the quality of the final labeling.

7 Conclusions

Our experimental results indicate that our algorithms have practical impact, even if they are mostly subexponential (except from the first one). Of course, there is large space for improvements. The proposed model is not appropriate for every instance of map labeling problems. We evaluated our algorithms in terms of time complexity and number of internal labels. It should also be evaluated whether the criterion of maximizing the number of internal labels yields aesthetically valuable labelings. The extension from the type-*opo* boundary labeling model to the more appealing type-*po* (or even to octilinear models [2]) is still open and seems nontrivial. Other variants to be considered might be to use leaders only when they are bend-less, to incorporate length-restrictions on the leaders or more general penalty functions for intersections between internal labels and leaders.

References

1. Bar-Yehuda, R., Rawitz, D.: Efficient algorithms for integer programs with two variables per constraint (extended abstract). Algorithmica 29(44), 595–609 (2001)
2. Bekos, M.A., Kaufmann, M., Nöllenburg, M., Symvonis, A.: Boundary Labeling with Octilinear Leaders. Algorithmica 57(3), 436–461 (2009)

3. Bekos, M.A., Kaufmann, M., Potika, K., Symvonis, A.: Multi-Stack Boundary Labeling Problems. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 81–92. Springer, Heidelberg (2006)
4. Bekos, M.A., Kaufmann, M., Potika, K., Symvonis, A.: Polygons Labelling of Minimum Leader Length. In: Kazuo, M., Kozo, S., Jiro, T. (eds.) Proc. 5th Asia Pacific Symposium on Information Visualisation (APVIS206). CRPIT, vol. 60, pp. 15–21. ACS Inc. (2006)
5. Bekos, M.A., Kaufmann, M., Symvonis, A., Wolff, A.: Boundary Labeling: Models and Efficient Algorithms for Rectangular Maps. Computational Geometry: Theory and Applications 36, 215–236 (2007)
6. Benkert, M., Haverkort, H., Kroll, M., Nöllenburg, M.: Algorithms for multi-criteria one-sided boundary labeling. In: Hong, S.-H., Nishizeki, T., Quan, W. (eds.) GD 2007. LNCS, vol. 4875, pp. 243–254. Springer, Heidelberg (2008)
7. Cromley, R.G.: A spatial allocation analysis of the point annotation problem. In: Proc. 2nd International Symposium on Spatial Data Handling (SDH 1986), pp. 38–49 (1986)
8. Edelsbrunner, H.: Dynamic data structures for orthogonal intersection queries. Tech. Rep. F59, Tech. Univ. Graz, Institute für Informationsverarbeitung (1980)
9. Formann, M., Wagner, F.: A packing problem with applications to lettering of maps. In: Proc. 7th Annual ACM Symposium on Computational Geometry, pp. 281–288 (1991)
10. Fowler, R.J., Paterson, M., Tanimoto, S.L.: Optimal Packing and Covering in the Plane are NP-Complete. Information Processing Letters 12(3), 133–137 (1981)
11. Gusfield, D., Pitt, L.: A bounded approximation for the minimum cost 2-Sat problem. Algorithmica 8(2), 103–117 (1992)
12. Kao, H.-J., Lin, C.-C., Yen, H.-C.: Many-to-one boundary labeling. In: Hong, S.-H., Ma, K.-L. (eds.) Proc. 6th International Asia-Pacific Symposium on Visualization (APVIS 2007), pp. 65–72. IEEE, Los Alamitos (2007)
13. van Kreveld, M., Strijk, T., Wolff, A.: Point labeling with sliding labels. Computational Geomentry: Theory and Applications 13, 21–47 (1999)
14. Kucera, L., Mehlhorn, K., Preis, B., Schwarzenecker, E.: Exact algorithms for a geometric packing problem. In: Enjalbert, P., Wagner, K.W., Finkel, A. (eds.) STACS 1993. LNCS, vol. 665, pp. 317–322. Springer, Heidelberg (1993)
15. Lin, C.-C.: Crossing-free many-to-one boundary labeling with hyperleaders. In: 3rd IEEE Pacific Visualization Symposium (PacificVis 2010), pp. 185–192. IEEE Press, Los Alamitos (2010)
16. Lin, C.-C., Wu, H.-Y., Yen, H.-C.: Boundary labeling in text annotation. In: 13th International Conference on Information Visualisation (IV2009), pp. 110–115. IEEE, Los Alamitos (2009)
17. Mehlhorn, K.: Data structures and algorithms 3: multi-dimensional searching and computational geometry. Springer-Verlag New York, Inc., Heidelberg (1984)
18. Poon, C.K., Zhu, B., Chin, F.: A polynomial time solution for labeling a rectilinear map. In: Proc. 13th Annual ACM Symposium on Comp. Geom (SoCG 1997), pp. 451–453 (1997)
19. Wolff, A., Strijk, T.: The Map-Labeling Bibliography, maintained since (1996), `http://i11www.ira.uka.de/map-labeling/bibliography`
20. Zoraster, S.: Integer programming applied to the map label placement problem. Cartographica 23(3), 16–27 (1986)