A Fast Algorithm for Updating a Labeling to Avoid a Query Point

F. Rostamabadi and M. Ghodsi

[1] Computer Engineering Department, Sharif University of Technology,
[2] Computer Science Research Group, Institute of Fundamental Sciences, Tehran, Iran

Abstract. Given a set of labeled points forming a valid map labeling, we are interested in the fast update of the labels if a point shape object (query point) appears in the map. In this paper, the labels are assumed to be axis-parallel squared-shape with the same size, each attached to one point in the middle of one of its edges. The updated labeling should include all labels with no overlaps, avoid the query point, and use the largest size labels possible. The existing algorithm for this problem runs in $O(n \lg n)$. We present an algorithm that needs a preprocessing of $O(n \lg n)$ time, but can update the map, for any query point, in $O(\lg n + k)$, where $k \leq 2n$ is the minimum number of update operations needed.

1 Introduction

Automated label placement is an important problem in map generation, geographical information systems, and computer graphics. This problem, in its simple form, is to attach a label (regularly a text) to each point, line, curve, or a region in the map. Point-label placement has received good attention. In a valid labeling, labels should be pairwise disjoint, and each label should be attached to its feature point [1]. There are different variations of point-labeling that are discussed in [2–6].

In this paper, we are interested in a fast update of labels in a point-labeling map. For simplicity, we assume that our map is composed of a number of points (n) each labeled by a unit-size axis-parallel square label. The point of each label appears in the middle of one of its edges. Maps with more general labeling can also be considered in our algorithm.

The labeling should be updated when a point-shape object appears in the map. We call this object, a query point. The new labeling should be valid in a way that all points preserve their labels, but the labels may have to flip or resize to avoid the query point (Fig. 1(a)). The new labeling should be valid, have labels with sizes as close to one as possible, and use the fewest number of label flip operations.

Applications to this problem can be found in computer graphics, computer games, flight animation, and in other related fields.

In this paper, we first give a formal definition of the problem, and present some notations. We then show that given the original map, we can create several

data structures in a preprocessing phase that use $O(n)$ space and $O(n \lg n)$ time, so that for any query point, the updated labeling with the mentioned optimum property can be found in $O(\lg n + k)$. Here, $k \leq 2n$ is the smallest number of flip and resize operations needed.

The existing solution for this problem is based on 2-SAT algorithm [7]. This solution is independent of the existing valid labeling, and it basically solves the decision problem of whether there exits a valid labeling of size α (for all labels) for all points. Our problem can easily be modeled by an instance of a 2-SAT problem, by assigning variables x_i and $\bar{x_i}$ to two possible locations of labels for each point (see Fig. 1(b)). The whole labeling is then represented by several clauses each with two literals, which can be solved in $O(n)$. The problem is then to call this decision problem $O(\lg n)$ times as a binary search in the $O(n)$ possible values of α. This results in an $O(n \lg n)$ solution for each query point. We are unaware of any other result for this problem.

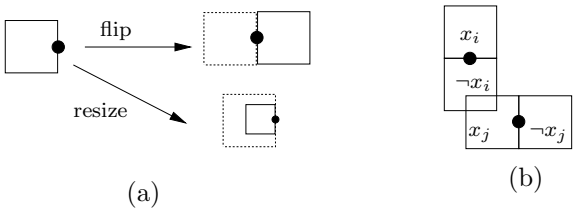

Fig. 1. (a) The flip and resize operations, (b) Intersection of $\neg x_i$ and x_j can be stated by a 2-SAT clause $\neg(\neg x_i \wedge x_j)$.

2 Problem Definition and the General Idea

We are given a valid labeling L composed of points $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ and unit-length axis-parallel square labels $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_n\}$, where ℓ_i is attached to p_i on the mid-point of one of its edges. For each query point q, the problem is to update L and obtain a new *q-avoiding* labeling L_q for all points in $\mathcal{P}$, such that q does not intersect with any label in L_q, while the new labels are as large as possible. We also want to create L_q with the minimum number of operations. Such a final re-labeling is denoted as *q-avoiding optimum labeling*.

We restrict ourselves with two operations on L, namely *flip* and *resize*, with which a label is flipped and/or resized. However, in order to maintain the structure of the original labeling, flipping a label ℓ_i can only be done on the edge where p_i locates, otherwise the new ℓ_i would detach from p_i. Also, resizing ℓ_i should be performed such that p_i remains in the middle of the same edge, and the label remains squared-shape.

We will show that this problem reduces to a decision problem of whether or not there exists a valid q-avoiding labeling of size α (i.e. all labels are of size α).

We then show that there are only $O(n)$ different values for α. If $\alpha_1 < \alpha_2 < \cdots$ are all these values, we later show that from the decision made on the problem for a size value of α_i, we can decide on the same problem with size value of α_{i+1} in $O(1)$ amortized time. It is also shown that the first decision (on α_1) can be made in $O(\lg n)$. This leads to an $O(\lg n + k)$ algorithm for the main problem, where k is the minimum number of operations.

So, we concentrate on the above decision problem. To solve this, we first model the original labeling with a graph, called *conflict graph,* that includes all possible flip and resize operations. The following section provides the necessary definitions in details.

3 Definitions

We first define the operations more precisely as follows. A label ℓ_i can be flipped over the edge containing its corresponding point p_i. The flipped label is denoted by $f(\ell_i)$. ℓ_i can also be resized to any length $\gamma \leq 1$ as long as p_i remains on the mid-point of its edge. The resized label is denoted by $r(\ell_i, \gamma)$.

Conflict graph $\mathcal{G} = (\mathcal{P} \cup \{p_0\}, \mathcal{E})$ is defined as a weighted directed multi-graph on the set of points $\mathcal{P}$ and a dummy vertex p_0, to model all possible flip and resize operations on $\mathcal{L}$. There are two different edges in this graph. One representing the normal flip operation, called *DominoEdges* to convey the domino effect that may be caused by a flip. We will show that a flipped label can not flip again. But, there may be cases where two flipped labels overlap, and we should resize one or both to avoid label overlaps. This operations are modeled by *BlockingEdges*. The dummy vertex p_0 is added to show the flip operations with no further intersections. These edges are precisely defined as,

$$\begin{aligned} &\mathit{DominoEdges} = \{(p_i, p_j) | f(\ell_i) \cap \ell_j \neq \emptyset\} \cup \{(p_i, p_0) |\ \forall p_i \in V - \{p_0\}\}, \text{and} \\ &\mathit{BlockingEdges} = \{(p_i, p_j) | f(\ell_i) \cap f(\ell_j) \neq \emptyset\}\ . \end{aligned}$$

The resize operation is needed when two (original or flipped) labels, say ℓ_a and ℓ_b, overlap. There are many resize values of γ_a and γ_b such that the resized labels $r(\ell_a, \gamma_a)$ and $r(\ell_b, \gamma_b)$ do not overlap. From the problem definition, we are interested in the values where $\min\{\gamma_a, \gamma_b\}$ is maximized. We define this value as the $g(\ell_a, \ell_b)$.

Based on the above g function, we define a weight function $w(e)$ for each edge $e = (p_i, p_j) \in \mathcal{E}$ as

$$w(e) = \begin{cases} g(f(\ell_i), \ell_j) & \text{if } (p_i, p_j) \in \mathit{DominoEdges}, \\ g(f(\ell_i), f(\ell_j)) & \text{if } (p_i, p_j) \in \mathit{BlockingEdges}, \\ 1 & \text{if } p_j = p_0\ . \end{cases}$$

A sample labeled map is shown in Fig. 2(a). The corresponding conflict graph is shown in Fig. 2(b), in which the domino edges are solid (domino edges ending at p_0 are not shown) and blocking edges are dashed arrows.

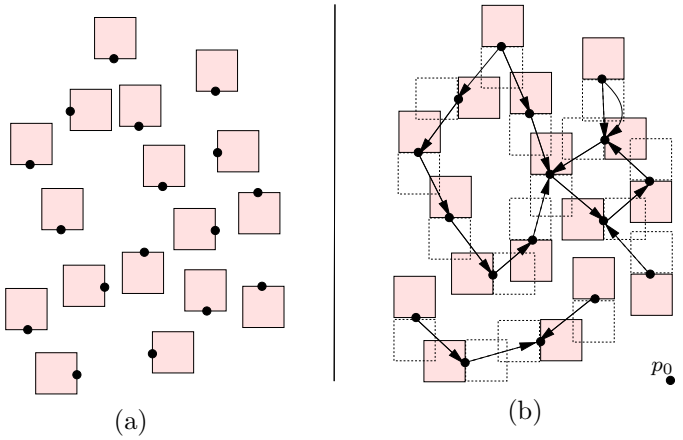

Fig. 2. (a) Initial labeled map. (b) The conflict graph: domino edges (solid) and blocking edges (dashed). Edges ending at p_0 are not shown.

Let L_q^α be a q-avoiding labeling with label size of α. We will define a subgraph G_q^α of $\mathcal{G}$ with minimum number of vertices and edges, and show that L_q^α exists if and only if there exists a valid (to be defined) G_q^α. We need the following definitions:

Definition 1. $(p_i, p_j) \in$ *DominoEdges is* domino-reachable *from any* p_k *in* $\mathcal{G}$*, if there is a directed simple path* $\pi : p_k \rightsquigarrow p_i$ *of domino edges.*

Definition 2. $(p_i, p_j) \in$ *DominoEdges is* α-terminating *from* p_k*, if* (p_i, p_j) *is domino-reachable from* p_k *with a path* π*, where for each edge* $e \in \pi$*,* $w(e) < \alpha$ *and* $w(p_i, p_j) \geq \alpha$*. The path* $p_k \rightsquigarrow p_j$ *is also defined as* α-terminating *path.*

Definition 3. $(p_i, p_j) \in$ *DominoEdges is* α-critical *from* p_k*, if* (p_i, p_j) *is* α*-terminating and* $w(p_i, p_j) = \alpha$*.*

Let ℓ_q be the label that contains the query point q. p_q is the corresponding point of ℓ_q. We define $G_q^\alpha = (P_q^\alpha, E_q^\alpha)$ as a subgraph of $\mathcal{G}$ containing all α-terminating paths from p_q. That is, P_q^α and E_q^α are the set of all vertices and edges on all α-terminating paths from p_q respectively (See Fig. 3). It is obvious that G_q^α is unique. Moreover, it is easy to see the following property.

Lemma 1. *If* $\alpha \leq \beta$ *then* $G_q^\alpha \subseteq G_q^\beta$*.*

Definition 4. *The internal nodes (not including the leaves) of* G_q^α *is denoted as* I_q^α*. Besides, the boundary edges (all edges with zero out-degree) of* G_q^α *is denoted as* B_q^α

We are only concerned with the valid G_q^α to be defined below.

Definition 5. G_q^α *is valid if for all* $p_i, p_j \in I_q^\alpha$ *and* $(p_i, p_j) \in$ *BlockingEdges, we have* $w(p_i, p_j) \geq \alpha$*.*

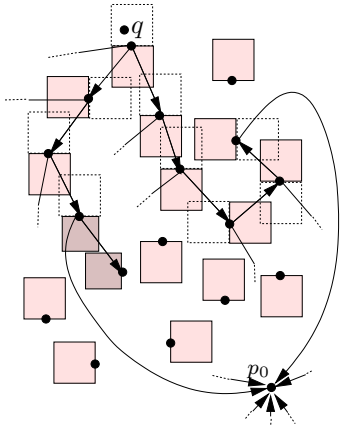

Fig. 3. G_q^α and the optimal generated L_q^α.

3.1 Properties of the Optimal Solution

The following is the main property of the optimal solution.

Theorem 1. *There exists a L_q^α if and and only if there exists a valid G_q^α.*

The proof is given in the following lemmas.

Lemma 2. *An L_q^α can be constructed from a valid G_q^α. Moreover, the minimum label size in L_q^α is α if at least one of the following conditions holds:*

1. *There is an α-critical edge in G_q^α.*
2. *There is a blocking edge with both ends in I_q^α and with weight of α.*

Proof. The operations required to construct L_q^α comes from the following steps:

1. Flip label ℓ_i for each domino edge $(p_i, p_j) \in E_q^\alpha$.
2. Resize one or both labels ℓ_i and ℓ_j to size $w(p_i, p_j)$ for each boundary edge $(p_i, p_j) \in B_q^\alpha$.
3. Resize one or both labels ℓ_i and ℓ_j to size $w(p_i, p_j)$ for each blocking edge (p_i, p_j) with both ends in I_q^α.

The generated labels are all larger than α since there is no resize operation to a length less than α. Besides, no two labels may intersect since otherwise, there must be a (domino or blocking) edge with weight less that α in $B_q^\alpha \cup I_q^\alpha$ which is in contradiction with the definition of valid G_q^α. It is easy to verify that, if there is an α-critical edge or a blocking edge with both ends in I_q^α with weight α, then a label with size equal to α is generated.

Lemma 3. *For any L_q^α, there exists a valid G_q^α.*

Proof. Define V as the set of points with flipped or resized labels, and $E = \{(p_i, p_j) | p_i, p_j \in V, (p_i, p_j) \in \textit{Domino Edges}\}$. Suppose π is an α-terminating path starting from q. This path should be in E since, otherwise, either there is a sequence of flips along this path not ending to a label with size at least α, or π is not an α-terminating path. So, E_q^α, which is the union of all α-terminating paths, is a subset of E, and hence $V_q^\alpha \subseteq V$. Since the initial labeling is valid, there is no blocking edge with both ends in internal nodes of V, and hence there no blocking edge with weight less that α in V_q^α. So, a valid G_q^α exists.

From Theorem 1, we can conclude that it is sufficient to check the existence of valid G_q^α. Lemma 4 proves that we only need to check this for at most $O(n)$ values of α's, and this can be searched more effectively from the fact given in Lemma 5.

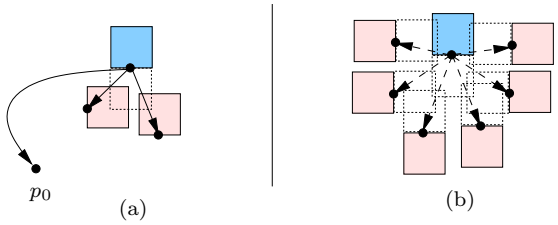

Fig. 4. Maximum number of domino and blocking edges from a given label (shown as darker label) in (a) and (b) respectively

Lemma 4. *Optimal label size value belongs to the set* $\{w(e)|e \in \mathcal{E}\}$*, which has at most* $O(n)$ *elements.*

Proof. For the first part, assume that the optimal label length α does not belong to $\{w(e)|e \in E\}$. So, there should be a label of length α, say ℓ_i, in the optimal labeling. This label is resized to α to resolve an intersection with some other label, say ℓ_j. Obviously, the best resizing for these labels is $w(p_i, p_j) > \alpha$ and hence the value of α is not optimal.

For the second part, we count the number of edges in the conflict graph $\mathcal{G}$. It is obvious that for each $p_i \in \mathcal{P}$, there is at most three edges in *DominoEdges* (Fig. 4(a)) and at most six edges in *BlockingEdges* (Fig. 4(b)) starting at p_i. Therefore, $|E| \leq 9n$ hence the set $\{w(e)|e \in E\}$ has $O(n)$ members.

It is easy to see the following.

Lemma 5. *If there is no valid* G_q^α *then there is no valid* G_q^β *for all* $\beta > \alpha$.

4 The Optimal Algorithm

The properties proved in the pervious section leads us to a rather simple non-optimal $O(n \lg n)$ algorithm. We first compute the set $A = \{w(e)|e \in E\}$ and sort its members in ascending order. We then use a binary search to find a valid G_q^α with maximum value of α. Since $G_q^{\alpha_i}$ can be constructed and checked for validity in $O(n)$, this algorithm requires $O(n \lg n)$ time. This complexity comes from the fact that this algorithm finds each $G_q^{\alpha_i}$ independently. We will show, however, that $G_q^{\alpha_{i+1}}$ can be efficiently constructed from $G_q^{\alpha_i}$ in $O(1)$ amortized time. This leads to our optimal algorithm.

In the optimal solution, we first construct several data structures in the preprocessing phase.

4.1 Preprocessing Phase

This phase consists of the following steps:

1. Build the conflict graph $\mathcal{G}$,
2. Calculate the weight function $w(e)$ for each edge $e \in \mathcal{E}$,
3. Create a sorted list of all weight of $\alpha_i = w(e)$ and assume that $\alpha_1 < \alpha_2 < \ldots < 1$,
4. Construct a point location data structure on the initial labels [8].

Lemma 6. *The time required for preprocessing phase is $O(n \lg n)$.*

Proof. The conflict graph can be built using a simple vertical sweep line algorithm keeping track of all intersecting labels with the sweep line in $O(n \lg n)$ time. The value of $w(e)$ can also be computed in $O(1)$ time per each edge. Steps 3 and 4 also take $O(n \lg n)$ time. So, the overall time required in preprocessing phase is $O(n \lg n)$.

4.2 Updating a Labeling

The algorithm starts with locating the label ℓ_q containing the query point q. If ℓ_q exists, the subgraph $G_q^{\alpha_1}$ is built. Hereafter, the algorithm builds $G_q^{\alpha_{i+1}}$ by adding some vertices and edges (possibly nothing) to $G_q^{\alpha_i}$. The following properties is used in constructing $G_q^{\alpha_{i+1}}$:

1. $G_q^{\alpha_1} \subseteq G_q^{\alpha_2} \subseteq G_q^{\alpha_3} \subseteq \ldots$ (according to the definition of $G_q^{\alpha_i}$),
2. All edges of $G_q^{\alpha_{i+1}}$ connected to a leaf vertex of $G_q^{\alpha_{i+1}}$ has a weight not less than α_i (Def. 2),
3. Minimum weight of all *Blocking Edges* with both ends in $I_q^{\alpha_{i+1}}$, denoted as β_{i+1}, is an upper bound of optimal label size (Def. 5),
4. $G_q^{\alpha_{i+1}}$ is valid if and only if $\beta_{i+1} \geq \alpha_{i+1}$ (Def. 5),
5. Finally, the recursive definition of $G_q^{\alpha_{i+1}}$ (the definition of $G_q^{\alpha_{i+1}}$ and Def. 5):

$$G_q^{\alpha_{i+1}} = G_q^{\alpha_i} \cup \{G_v^{\alpha_{i+1}} | (u,v) \in B_q^{\alpha_i}, w(u,v) = \alpha_i\} \ .$$

The following is the optimal algorithm in detail for a given query point q:

The Optimal Algorithm

1. Locate ℓ_q containing q.
2. **if** no such label exists, **then** optimal solution is the initial labeling $\mathcal{L}$ and **terminate**.
3. $G_q^{\alpha_1} \leftarrow (V_q^{\alpha_1}, E_q^{\alpha_1})$ where:
   (a) $V_q^{\alpha_1} \leftarrow \{p_q\} \cup \{v | (p_q, v) \in \textit{Domino Edges}\}$, and
   (b) $E_q^{\alpha_1} \leftarrow \{(p_q, v) | v \in V_q^{\alpha_1}\}$.
4. $I_q^{\alpha_1} \leftarrow \{p_q\}$, $B_q^{\alpha_1} \leftarrow E_q^{\alpha_1}$, $\beta_1 \leftarrow 1$.
5. **for** $i = 1, 2, \ldots$ **do**
   (a) **if** there is no edge $e' \in B_i^{\alpha_i}$ with $w(e') < \alpha_{i+1}$ and $\alpha_{i+1} = \beta_i$, **then** $G_q^{\alpha_i}$ corresponds to the optimal labeling and **terminate**.
   (b) $V \leftarrow V_q^{\alpha_i}$, $E \leftarrow E_q^{\alpha_i}$, $I \leftarrow I_q^{\alpha_i}$, $B \leftarrow B_q^{\alpha_i}$ and $\beta \leftarrow \beta_i$.
   (c) **while** $\{e' | e' \in B, w(e') < \alpha_{i+1}\} \neq \emptyset$ **do**
      i. Select an edge $e = (u, v)$ from the above set.
      ii. $V' \leftarrow \{x | (v, x) \in \textit{Domino Edges} \wedge x \notin V\}$ and $E' \leftarrow \{(v, x) | x \in V'\}$.
      iii. $V \leftarrow V \cup V'$ and $E \leftarrow E \cup E'$.
      iv. $I \leftarrow I \cup \{u\}$ and $B \leftarrow B - \{(u, v)\} \cup E'$.
      v. $\beta = \min(\{\beta\} \cup \{w(y, u) | y \in I \wedge (y, u) \in \textit{Blocking Edges}\})$.
   (d) $V_q^{\alpha_{i+1}} \leftarrow V$, $E_q^{\alpha_{i+1}} \leftarrow E$, $I_q^{\alpha_{i+1}} \leftarrow I$, $B_q^{\alpha_{i+1}} \leftarrow B$ and $\beta_{i+1} \leftarrow \beta$.
   (e) **if** $\beta_{i+1} \leq \alpha_i$, **then** construct the optimal labeling from $G_q^{\alpha_i}$ and **terminate**.
   (f) **if** $\beta_{i+1} = \alpha_{i+1}$, **then** construct the optimal labeling from $G_q^{\alpha_{i+1}}$ and **terminate**.

Lemma 7. *The $G_q^{\alpha_i}$ for all values of α_i can be constructed in ascending order of α_i in overall $O(n)$ time.*

Proof. Any edge in the conflict graph is visited at most a constant number of times: A domino edge is visited at most once, and a blocking edge is visited at most twice (a blocking edge is checked whenever one of its ends is added to the internal vertex set). With an appropriate data structures for maintaining sets I and B, each visit to an edge can be implemented in $O(1)$.

Lemma 7 also implies that the construction of $G_q^{\alpha_{i+1}}$ from $G_q^{\alpha_i}$ takes $O(1)$ amortized time. Also, with an appropriate point location data structure to locate q in time $O(\lg n)$ [8], the following theorem holds:

Theorem 2. *Given a point q on a labeling L, the time required to generate an updated q-avoiding labeling is $O(\lg n + k)$ where k is the number of operation required to update L.*

5 Conclusions

In this paper, we introduced the problem of updating a squared axis-parallel labeled map to avoid a query point. We modeled the initial labeling and update

operations with a directed multi-graph with at most $O(n)$ edges and vertices called conflict graph. We also showed that given a query point q, the optimal q-avoiding labeling corresponds to a subgraph of the conflict graph that can be found in time $O(\lg n + k)$ where k is the number of update operations. This approach can be simply modified for the case of multi-point queries. The required time for finding the optimal labeling for m query points will be $O(m \lg n + k)$ where k is the number of operations needed to update the labeling.

The technique used in this paper can be extended to other labeling schemes, like axis parallel rectangular labels. The only difference is that the conflict graph may have $O(n^2)$ edges. The algorithm still produces the optimal labeling in $O(\lg n + k)$ time where k is $O(n^2)$.

References

1. Marks, J., Shieber, S.: The computational complexity of cartographic label placement. Technical Report TR-05-91, Harvard CS (1991)
2. Duncan, R., Qian, J., Vigneron, A., Zhu, B.: Polynomial time algorithms for three-label point labeling. Theoretical Computer Science **296** (2003) 75–87
3. Qin, Z., Wolff, A., Xu, Y., Zhu, B.: New algorithms for two-label point labeling. In Paterson, M., ed.: Proc. 8th Annu. Europ. Symp. on Algorithms (ESA'00). Volume 1879 of Lecture Notes in Computer Science., Saarbrücken, Springer-Verlag (2000) 368–379
4. Spriggs, M.J., Keil, J.M.: A new bound for map labeling with uniform circle pairs. Information Processing Letters **81** (2002) 47–53
5. Strijk, T., Wolff, A.: Labeling points with circles. International Journal of Computational Geometry and Applications **11** (2001) 181–195
6. Wolff, A., Thon, M., Xu, Y.: A simple factor-2/3 approximation algorithm for two-circle point labeling. International Journal of Computational Geometry and Applications **12** (2002) 269–281
7. Doddi, S., Marathe, M.V., Mirzaian, A., Moret, B.M., Zhu, B.: Map labeling and its generalizations. In: Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97), New Orleans, LA (1997) 148–157
8. Sarnak, N., Tarjan, R.E.: Planar point location using persistent search trees. Commun. ACM **29** (1986) 669–679